

IPPM
Internet-Draft
Intended status: Informational
Expires: April 28, 2022

M. Cociglio
M. Nilo
F. Bulgarella
Telecom Italia - TIM
G. Fioccola
Huawei Technologies
October 25, 2021

User Devices Explicit Monitoring
draft-cnbf-ippm-user-devices-explicit-monitoring-03

Abstract

This document describes a methodology to monitor network performance exploiting user devices. This can be achieved using the Explicit Flow Measurement Techniques, protocol independent methods that employ few marking bits, inside the header of each packet, for loss and delay measurement. User devices and servers, marking the traffic, signal these metrics to intermediate network observers allowing them to measure connection performance, and to locate the network segment where impairments happen. In addition or in alternative to network observers, a probe can be installed on the user device with remarkable benefits in terms of hardware deployment and measurement scalability.

Status of This Memo

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

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

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

This Internet-Draft will expire on April 28, 2022.

Copyright Notice

Copyright (c) 2021 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. Notational Conventions	3
3. Explicit Performance Open Issues	3
4. Explicit Performance Probes on User Devices	3
5. Device Owner Activates Explicit Performance Measurements . .	4
6. Who Will Handle the Performance Data?	4
7. The Explicit Performance App	5
8. Improvements of Explicit Flow Measurement Techniques Using Probes on User Devices	5
8.1. Hidden Delay Bit Considerations	5
9. Security Considerations	6
10. Privacy Considerations	6
11. IANA Considerations	6
12. Change Log	6
13. Contributors	6
14. Acknowledgements	6
15. References	6
15.1. Normative References	6
15.2. Informative References	7
Authors' Addresses	7

1. Introduction

Explicit Performance Monitoring enables a passive observer (a probe) to measure delay and loss just watching the marking (a few header bits) of live traffic packets. It works on client-server protocols: e.g. QUIC [QUIC-TRANSPORT], TCP [TCP]. The different methods are described in [EXPLICIT-FLOW-MEASUREMENTS] and are inspired by [AltMark].

This document explains how to employ the methods described in [EXPLICIT-FLOW-MEASUREMENTS] by proposing the user device as a convenient place for the Explicit Performance Observer.

2. Notational 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].

3. Explicit Performance Open Issues

There are some open issues to consider for the deployment of [EXPLICIT-FLOW-MEASUREMENTS]:

- Who decides whether to mark traffic? Explicit measures only work if both the server and the client mark the production traffic.
- What about scalability? Could network probes monitor all the connections? If they cannot, which ones to choose?
- Which connections to monitor within the network? Network probes need an effective way to identify which connections really need to be monitored.
- How to monitor both traffic directions? Not always possible for network probes (asymmetric connections).

4. Explicit Performance Probes on User Devices

This document proposes the user device (e.g. mobile phones, PCs) as a convenient place where to put the Explicit Performance Observer.

The placement of the observer on the user device helps to mitigate the issues reported in the previous section, in particular:

- The device should decide whether to mark the traffic or not.
- Regarding the scalability issue, on the user device there are few connections to monitor so it becomes less relevant.
- Connections eligible for monitoring should be the impaired ones. User devices and network probes can cooperate to achieve this goal. It is possible to set alarm thresholds on the user device and to signal to the network probes only the sessions with impairments. This allows to segment the performance measurements and to locate the faults. In this way network probes, that could also be embedded into network nodes, have to monitor a limited number of connections.
- Monitoring both directions is always possible on the user device.

5. Device Owner Activates Explicit Performance Measurements

The decision whether to activate the marking (e.g. [SPIN-BIT], [ANRW19-PM-QUIC], [EXPLICIT-FLOW-MEASUREMENTS]) or not should be made by the device owner by properly configuring the applications (e.g. browsers) based on connection-oriented protocols that support explicit measurements (e.g. QUIC).

All applications should provide the activation or deactivation of packet marking, for example by providing an user interface or exposing API.

So, during the client-server handshake, the client will decide whether the marking is active or not within a session and notify its decision to the server.

An example of a simple explicit marking agreement of a protocol is the following. This works if the usage of each performance bit is unique and predefined. An endpoint set to 0 all the explicit performance measurement bits to indicate its intention not to mark. Then:

- the client set at least one of its marking bits to 1 notifying the server of its intention to use that/those marking bits; the server adapts according to the client's will;
- the server set at least one of its marking bits to 1; if the client does not start marking the same bit/bits, then the marking for that/those bits is aborted.

The best would be if both client and server started using the same marking bits from the beginning of the connection. In this case no alignment between endpoints would be required. This mechanism works best if, where possible, measurements start using 1 as the first marking value.

6. Who Will Handle the Performance Data?

Performance data are stored only on the user device or also sent to "external bodies" according to the will of the device owner.

The main recipient would be the Internet Service Provider. Indeed, as explained in the previous section, this enables user device and network probes coordination that permits an improved performance measurement approach.

Moreover these data could also be of interest for the national regulatory authorities or others authorized subjects.

7. The Explicit Performance App

This methodology could be implemented with an "Explicit Performance App" installed on the user device.

The App should perform the following tasks:

- collect user preferences;
- activate/deactivate marking on device Apps (e.g. browsers);
- implement the observer;
- show performances to the user;
- send data to the "Explicit Performance Management Center";
- set performance thresholds.

8. Improvements of Explicit Flow Measurement Techniques Using Probes on User Devices

- Spin bit and Delay bit: the observer-server RTT component measured on the user device is equivalent to the RTT, but without including the client-side application delay and therefore more precise.
- sSquare bit: would measure the End-to-End loss rate in the download direction instead of upstream loss rate.
- Loss event bit: would measure, as before, the End-to-End loss rate in both directions. Moreover, in the upload direction, the signal would be "clean" since it is captured at the origin and therefore not affected by losses.
- Reflection square bit: would measure the RT loss rate instead of three-quarters connection loss rate.

8.1. Hidden Delay Bit Considerations

The Explicit Flow Measurements draft introduces a new Delay Bit feature capable of masking the RTT of the connection to the observers on the network. To use this feature, the client must select an Additional Delay used to delay the client-side reflection of marked samples. Clearly, the introduction by the client of a reflection delay makes the client-observer component of the RTT inaccurate.

Using this feature on a user device probe has several advantages:

- A system-wide Additional Delay can be selected and periodically updated making it common to all applications installed on the device.
- The hidden Delay Bit produces the same metrics of the Delay Bit since the observer-server RTT, measured on the client, is equal to the end-to-end RTT of the connection.
- The user device can easily communicate the Additional Delay to network probes whenever an alarm threshold is triggered. In this way, the observer can compute the e2e RTT of the connection.

9. Security Considerations

TBD

10. Privacy Considerations

TBD

11. IANA Considerations

This document makes no request of IANA.

12. Change Log

TBD

13. Contributors

TBD

14. Acknowledgements

TBD

15. References

15.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [TCP] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <<https://www.rfc-editor.org/info/rfc793>>.

15.2. Informative References

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

[ANRW19-PM-QUIC] Bulgarella, F., Cociglio, M., Fioccola, G., Marchetto, G., and R. Sisto, "Performance measurements of QUIC communications", Proceedings of the Applied Networking Research Workshop, DOI 10.1145/3340301.3341127, July 2019.

[EXPLICIT-FLOW-MEASUREMENTS] Cociglio, M., Ferrieux, A., Fioccola, G., Lubashev, I., Bulgarella, F., Hamchaoui, I., Nilo, M., Sisto, R., and D. Tikhonov, "Explicit Flow Measurements Techniques", draft-ietf-ippm-explicit-flow-measurements-00 (work in progress), October 2021.

[QUIC-TRANSPORT] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <<https://www.rfc-editor.org/info/rfc9000>>.

[SPIN-BIT] Trammell, B., Vaere, P. D., Even, R., Fioccola, G., Fossati, T., Ihlar, M., Morton, A., and E. Stephan, "Adding Explicit Passive Measurability of Two-Way Latency to the QUIC Transport Protocol", draft-trammell-quick-spin-03 (work in progress), May 2018.

Authors' Addresses

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

EMail: mauro.cociglio@telecomitalia.it

Massimo Nilo
Telecom Italia - TIM
Via Reiss Romoli, 274
Torino 10148
Italy

EMail: massimo.nilo@telecomitalia.it

Fabio Bulgarella
Telecom Italia - TIM
Via Reiss Romoli, 274
Torino 10148
Italy

EMail: fabio.bulgarella@guest.telecomitalia.it

Giuseppe Fioccola
Huawei Technologies
Riesstrasse, 25
Munich 80992
Germany

EMail: giuseppe.fioccola@huawei.com

IP Performance Measurement
Internet-Draft
Intended status: Experimental
Expires: April 28, 2022

C. Paasch
R. Meyer
S. Cheshire
O. Shapira
Apple Inc.
October 25, 2021

Responsiveness under Working Conditions
draft-cpaasch-ippm-responsiveness-01

Abstract

For many years, a lack of responsiveness, variously called lag, latency, or bufferbloat, has been recognized as an unfortunate, but common symptom in today's networks. Even after a decade of work on standardizing technical solutions, it remains a common problem for the end users.

Everyone "knows" that it is "normal" for a video conference to have problems when somebody else at home is watching a 4K movie or uploading photos from their phone. However, there is no technical reason for this to be the case. In fact, various queue management solutions (fq_codel, cake, PIE) have solved the problem for tens of thousands of people.

Our networks remain unresponsive, not from a lack of technical solutions, but rather a lack of awareness of the problem. We believe that creating a tool whose measurement matches people's every day experience will create the necessary awareness, and result in a demand for products that solve the problem.

This document specifies the "RPM Test" for measuring responsiveness. It uses common protocols and mechanisms to measure user experience especially when the network is fully loaded ("responsiveness under working conditions".) The measurement is expressed as "Round-trips Per Minute" (RPM) and should be included with throughput (up and down) and idle latency as critical indicators of network quality.

Status of This Memo

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

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

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

This Internet-Draft will expire on April 28, 2022.

Copyright Notice

Copyright (c) 2021 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. Terminology	3
2. Design Constraints	4
3. Goals	4
4. Measuring Responsiveness Under Working Conditions	5
4.1. Working Conditions	5
4.1.1. From single-flow to multi-flow	6
4.1.2. Parallel vs Sequential Uplink and Downlink	6
4.1.3. Reaching saturation	6
4.1.4. Final "Working Conditions" Algorithm	7
4.2. Measuring Responsiveness	8
4.2.1. Aggregating the Measurements	8
4.2.2. Statistical Confidence	9
5. RPM Test Server API	9
6. Security Considerations	10
7. IANA Considerations	10
8. Acknowledgments	10
9. Informative References	10
Authors' Addresses	11

1. Introduction

For many years, a lack of responsiveness, variously called lag, latency, or bufferbloat, has been recognized as an unfortunate, but common symptom in today's networks [Bufferbloat]. Solutions like fq_codel [RFC8290] or PIE [RFC8033] have been standardized and are to some extent widely implemented. Nevertheless, people still suffer from bufferbloat.

Although significant, the impact on user experience can be transitory - that is, its effect is not always present. Whenever a network is actively being used at its full capacity, buffers can fill up and create latency for traffic. The duration of those full buffers may be brief: a medium-sized file transfer, like an email attachment or uploading photos, can create bursts of latency spikes. An example of this is lag occurring during a videoconference, where a connection is briefly shown as unstable.

These short-lived disruptions make it hard to narrow down the cause. We believe that it is necessary to create a standardized way to measure and express responsiveness.

Existing network measurement tools could incorporate a responsiveness measurement into their set of metrics. Doing so would also raise the awareness of the problem and make the standard "network quality measures" of throughput, idle latency, and responsiveness.

1.1. Terminology

A word about the term "bufferbloat" - the undesirable latency that comes from a router or other network equipment buffering too much data. This document uses the term as a general description of bad latency, using more precise wording where warranted.

"Latency" is a poor measure of responsiveness, since it can be hard for the general public to understand. The units are unfamiliar ("what is a millisecond?") and counterintuitive ("100 msec - that sounds good - it's only a tenth of a second!").

Instead, we create the term "Responsiveness under working conditions" to make it clear that we are measuring all, not just idle, conditions, and use "round-trips per minute" as the metric. The values range from 50 (poor) to 3,000 (excellent), with the added advantage that "bigger is better." Finally, we abbreviate the measurement to "RPM", a wink to the "revolutions per minute" that we use for cars.

This document defines an algorithm for the "RPM Test" that explicitly measures responsiveness under working conditions.

2. Design Constraints

There are many challenges around measurements on the Internet. They include the dynamic nature of the Internet, the diverse nature of the traffic, the large number of devices that affect traffic, and the difficulty of attaining appropriate measurement conditions.

Internet paths are changing all the time. Daily fluctuations in the demand make the bottlenecks ebb and flow. To minimize the variability of routing changes, it's best to keep the test duration relatively short.

TCP and UDP traffic, or traffic on ports 80 and 443, may take significantly different paths on the Internet and be subject to entirely different Quality of Service (QoS) treatment. A good test will use standard transport layer traffic - typical for people's use of the network - that is subject to the transport's congestion control that might reduce the traffic's rate and thus its buffering in the network.

Traditionally, one thinks of bufferbloat happening on the routers and switches of the Internet. However, the networking stacks of the clients and servers can have huge buffers. Data sitting in TCP sockets or waiting for the application to send or read causes artificial latency, and affects user experience the same way as "traditional" bufferbloat.

Finally, it is important to note that queueing only happens behind a slow "bottleneck" link in the network, and only occurs when sufficient traffic is present. The RPM Test must ensure that buffers are actually full for a sustained period, and only then make repeated latency measurements in this particular state.

3. Goals

The algorithm described here defines an RPM Test that serves as a good proxy for user experience. This means:

1. Today's Internet traffic primarily uses HTTP/2 over TLS. Thus, the algorithm should use that protocol.

As a side note: other types of traffic are gaining in popularity (HTTP/3) and/or are already being used widely (RTP). Traffic prioritization and QoS rules on the Internet may subject traffic to completely different paths:

these could also be measured separately.

2. The Internet is marked by the deployment of countless middleboxes like transparent TCP proxies or traffic prioritization for certain types of traffic. The RPM Test must take into account their effect on DNS-request [RFC1035], TCP-handshake [RFC0793], TLS-handshake, and request/response.
 3. The test result should be expressed in an intuitive, nontechnical form.
 4. Finally, to be useful to a wide audience, the measurement should finish within a short time frame. Our target is 20 seconds.
4. Measuring Responsiveness Under Working Conditions

To make an accurate measurement, the algorithm must reliably put the network in a state that represents those "working conditions". Once the network has reached that state, the algorithm can measure its responsiveness. The following explains how the former and the latter are achieved.

4.1. Working Conditions

For the purpose of this methodology, typical "working conditions" represent a state of the network in which the bottleneck node is experiencing ingress and egress flows similar to those created by humans in the typical day-to-day pattern.

While a single HTTP transaction might briefly put a network into working conditions, making reliable measurements requires maintaining the state over sufficient time.

The algorithm must also detect when the network is in a persistent working condition, also called "saturation".

Desired properties of "working condition":

- o Should not waste traffic, since the person may be paying for it
- o Should finish within a short time to avoid impacting other people on the same network, to avoid varying network conditions, and not try the person's patience.

4.1.1. From single-flow to multi-flow

A single TCP connection may not be sufficient to saturate a path. For example, the 4MB constraints on TCP window size constraints may not fill the pipe. Additionally, traditional loss-based TCP congestion control algorithms react aggressively to packet loss by reducing the congestion window. This reaction (intended by the protocol design) decreases the queueing within the network, making it hard to reach saturation.

The goal of the RPM Test is to keep the network as busy as possible in a sustained and persistent way. It uses multiple TCP connections and gradually adds more TCP flows until saturation is reached.

4.1.2. Parallel vs Sequential Uplink and Downlink

Poor responsiveness can be caused by queues in either (or both) the upstream and the downstream direction. Furthermore, both paths may differ significantly due to access link conditions (e.g., 5G downstream and LTE upstream) or the routing changes within the ISPs. To measure responsiveness under working conditions, the algorithm must saturate both directions.

Measuring in parallel achieves more data samples for a given duration. Given the desired test duration of 20 seconds, sequential uplink and downlink tests would only yield half the data. The RPM Test specifies parallel, concurrent measurements.

However, a number of caveats come with measuring in parallel:

- o Half-duplex links may not permit simultaneous uplink and downlink traffic. This means the test might not saturate both directions at once.
- o Debuggability of the results becomes harder: During parallel measurement it is impossible to differentiate whether the observed latency happens in the uplink or the downlink direction.
- o Consequently, the test should have an option for sequential testing.

4.1.3. Reaching saturation

The RPM Test gradually increases the number of TCP connections and measures "goodput" - the sum of actual data transferred across all connections in a unit of time. When the goodput stops increasing, it means that saturation has been reached.

Saturation has two criteria: a) the load bearing connections are utilizing all the capacity of the bottleneck, b) the buffers in the bottleneck are completely filled.

The algorithm notes that throughput gradually increases until TCP connections complete their TCP slow-start phase. At that point, throughput eventually stalls usually due to receive window limitations. The only means to further increase throughput is by adding more TCP connections to the pool of load bearing connections. If new connections leave the throughput the same, saturation has been reached and - more importantly - the working condition is stable.

Filling buffers at the bottleneck depends on the congestion control deployed on the sender side. Congestion control algorithms like BBR may reach high throughput without causing queueing because the bandwidth detection portion of BBR effectively seeks the bottleneck capacity.

RPM Test clients and servers should use loss-based congestion controls like Cubic to fill queues reliably.

The RPM Test detects saturation when the observed goodput is not increasing even as connections are being added, or it detects packet loss or ECN marks signaling congestion or a full buffer of the bottleneck link.

4.1.4. Final "Working Conditions" Algorithm

The following algorithm reaches working conditions (saturation) of a network by using HTTP/2 upload (POST) or download (GET) requests of infinitely large files. The algorithm is the same for upload and download and uses the same term "load bearing connection" for each.

The steps of the algorithm are:

- o Create 4 load bearing connections
- o At each 1 second interval:
 - * Compute "instantaneous aggregate" goodput which is the number of bytes transferred within the last second.
 - * Compute a moving average of the last 4 "instantaneous aggregate goodput" measurements
 - * If moving average > "previous" moving average + 5%:

- + Network did not yet reach saturation. If no flows added within the last 4 seconds, add 4 more flows
- * Else, network reached saturation for the current flow count.
 - + If new flows added and for 4 seconds the moving average throughput did not change: network reached stable saturation
 - + Else, add four more flows

Note: It is tempting to envision an initial base RTT measurement and adjust the intervals as a function of that RTT. However, experiments have shown that this makes the saturation detection extremely unstable in low RTT environments. In the situation where the "unloaded" RTT is in the single-digit millisecond range, yet the network's RTT increases under load to more than a hundred milliseconds, the intervals become much too low to accurately drive the algorithm.

4.2. Measuring Responsiveness

Once the network is in a consistent working conditions, the RPM Test must "probe" the network multiple times to measure its responsiveness.

Each RPM Test probe measures:

1. The responsiveness of the different steps to create a new connection, all during working conditions.

To do this, the test measures the time needed to make a DNS request, establish a TCP connection on port 443, establish a TLS context using TLS1.3 [RFC8446], and send and receive a one-byte object with a HTTP/2 GET request. It repeats these steps multiple times for accuracy.

2. The responsiveness of the network and the client/server networking stacks for the load bearing connections themselves.

To do this, the load bearing connections multiplex an HTTP/2 GET request for a one-byte object to get the end-to-end latency on the connections that are using the network at full speed.

4.2.1. Aggregating the Measurements

The algorithm produces sets of 5 times for each probe, namely: DNS handshake, TCP handshake, TLS handshake, HTTP/2 request/response on separate (idle) connections, HTTP/2 request/response on load bearing

connections. This fine-grained data is useful, but not necessary for creating a useful metric.

To create a single "Responsiveness" (e.g., RPM) number, this first iteration of the algorithm gives an equal weight to each of these values. That is, it sums the five time values for each probe, and divides by the total number of probes to compute an average probe duration. The reciprocal of this, normalized to 60 seconds, gives the Round-trips Per Minute (RPM).

4.2.2. Statistical Confidence

The number of probes necessary for statistical confidence is an open question. One could imagine a computation of the variance and confidence interval that would drive the number of measurements and balance the accuracy with the speed of the measurement itself.

5. RPM Test Server API

The RPM measurement uses standard protocols: no new protocol is defined.

Both the client and the server MUST support HTTP/2 over TLS 1.3. The client MUST be able to send a GET request and a POST. The server MUST be able to respond to both of these HTTP commands. Further, the server endpoint MUST be accessible through a hostname that can be resolved through DNS. The server MUST have the ability to provide content upon a GET request. Both client and server SHOULD use loss-based congestion controls like Cubic. The server MUST use a packet scheduling algorithm that minimizes internal queueing to avoid affecting the client's measurement.

The server MUST respond to 4 URLs:

1. A "small" URL/response: The server must respond with a status code of 200 and 1 byte in the body. The actual body content is irrelevant.
2. A "large" URL/response: The server must respond with a status code of 200 and a body size of at least 8GB. The body can be bigger, and may need to grow as network speeds increases over time. The actual body content is irrelevant. The client will probably never completely download the object, but will instead close the connection after reaching working condition and making its measurements.

3. An "upload" URL/response: The server must handle a POST request with an arbitrary body size. The server should discard the payload.
4. A configuration URL that returns a JSON [RFC8259] object with the information the client uses to run the test (sample below).
Sample JSON:

```
{
  "version": 1,
  "urls": {
    "small_https_download_url": "https://networkquality.example.com/api/v1/small"
  ,
    "large_https_download_url": "https://networkquality.example.com/api/v1/large"
  ,
    "https_upload_url": "https://networkquality.example.com/api/v1/upload"
  }
}
```

The client begins the responsiveness measurement by querying for the JSON configuration. This supplies the URLs for creating the load bearing connections in the upstream and downstream direction as well as the small object for the latency measurements.

6. Security Considerations

TBD

7. IANA Considerations

TBD

8. Acknowledgments

We would like to thank Rich Brown for his editorial pass over this I-D. We also thank Erik Auerswald for his constructive feedback on the I-D.

9. Informative References

[Bufferbloat]

Gettys, J. and K. Nichols, "Bufferbloat: Dark Buffers in the Internet", Communications of the ACM, Volume 55, Number 1 (2012) , n.d..

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <<https://www.rfc-editor.org/info/rfc793>>.

- [RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <<https://www.rfc-editor.org/info/rfc1035>>.
- [RFC8033] Pan, R., Natarajan, P., Baker, F., and G. White, "Proportional Integral Controller Enhanced (PIE): A Lightweight Control Scheme to Address the Bufferbloat Problem", RFC 8033, DOI 10.17487/RFC8033, February 2017, <<https://www.rfc-editor.org/info/rfc8033>>.
- [RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, <<https://www.rfc-editor.org/info/rfc8259>>.
- [RFC8290] Hoeiland-Joergensen, T., McKenney, P., Taht, D., Gettys, J., and E. Dumazet, "The Flow Queue CoDel Packet Scheduler and Active Queue Management Algorithm", RFC 8290, DOI 10.17487/RFC8290, January 2018, <<https://www.rfc-editor.org/info/rfc8290>>.
- [RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <<https://www.rfc-editor.org/info/rfc8446>>.

Authors' Addresses

Christoph Paasch
Apple Inc.
One Apple Park Way
Cupertino, California 95014
United States of America

Email: cpaasch@apple.com

Randall Meyer
Apple Inc.
One Apple Park Way
Cupertino, California 95014
United States of America

Email: rrm@apple.com

Stuart Cheshire
Apple Inc.
One Apple Park Way
Cupertino, California 95014
United States of America

Email: cheshire@apple.com

Omer Shapira
Apple Inc.
One Apple Park Way
Cupertino, California 95014
United States of America

Email: oesh@apple.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 23, 2022

A. Clemm
J. Strassner
Futurewei
J. Francois
Inria
October 20, 2021

High-Precision Service Metrics
draft-csfx-ippm-hipmetrics-00

Abstract

This document defines a set of metrics for high-precision networking services. These metrics can be used to assess the service levels that are being delivered for a networking flow. Specifically, they can be used to determine the degree of compliance with which service levels are being delivered relative to service level objectives that were defined for the flow. The metrics can be used as part of flow records and/or accounting records. They can also be used to continuously monitor the quality with which high-precision networking service are being delivered.

Status of This Memo

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

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

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

This Internet-Draft will expire on April 23, 2022.

Copyright Notice

Copyright (c) 2021 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. Key Words	3
3. Definitions and Acronyms	3
4. Metrics	3
5. Discussion Items	7
6. IANA Considerations	7
7. Security Considerations	7
8. Normative References	8
Authors' Addresses	9

1. Introduction

Many networking applications increasingly rely on high-precision networking services that have clearly defined service level objectives (SLOs), for example with regards to end-to-end latency. Applications requiring such services include industrial networks, for example cloud-based industrial controllers for precision machinery, vehicular applications, for example tele-driving in which a vehicle is remotely controlled by a human operators, or Augmented Reality / Virtual Reality (AR/VR) applications involving rendering of point clouds remotely. Many of those applications are not tolerant of degrading service levels. A slight miss in SLOs does not merely result in a slight deterioration of the Quality of Experience to end users, but may render the application inoperable. At the same time, many of those applications are mission critical, in which sudden failures can jeopardize safety or have other adverse consequences. However, clearly those applications represent significant business opportunity demanding dependable technical solutions.

Because of this, efforts such as Deterministic Networking (DetNet) [RFC8655] are attempting to create solutions in which clear bounds on parameters such as end-to-end latency and jitter can be defined in order to make service levels being delivered predictable and, ideally, deterministic. However, one area that has not kept pace concerns metrics that can account for service levels with which services are delivered, specifically the degree of precision for agreed-upon service level objectives. Such metrics, and the instrumentation to support them, are important for a number of purposes, including monitoring (to ensure that networking services

are performing according to their objectives) as well as accounting (to maintain a record of service levels actually delivered, important for monetization of such services as well as for triaging of problems).

The current state-of-the-art of such metrics includes (for example) interface metrics, useful to obtain data on traffic volume and behavior that can be observed at an interface [RFC2863] [RFC8343] but agnostic of actual end-to-end service levels and not specific to distinct flows. Flow records [RFC7011] [RFC7012] maintain statistics about flows, including flow volume and flow duration, but again contain very little information about end-to-end service levels, let alone whether the service levels delivered meet their targets, i.e. their associated SLOs.

This specification introduces a new set of metrics aimed at capturing end-to-end service levels for a flow, specifically the degree to which flows comply with the SLOs that are in effect.

It should be noted that at this point, the set of metrics proposed here is intended as a "starter set" that is intended to spark further discussion. Other metrics are certainly conceivable; we expect that the list of metrics will evolve over time as part of Working Group discussions.

2. Key Words

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. Definitions and Acronyms

MTBF: Mean Time Between Failures

SL: Service Level

SLA: Service Level Agreement

SLO: Service Level Objective

4. Metrics

The following section proposes a set of accounting metrics focus on end-to-end latency objectives. They indicate whether any violations of end-to-end latency occurred at the packet level. These metrics

are intended to be applied on a per-flow basis and are intended to assess the degree to which a flow's end-to-end service levels comply with the SLO in effect for that flow.

While the focus in this document concerns end-to-end latency objectives, analogous metrics could also be defined for other end-to-end service level parameters, such as loss (which is distinct from loss occurring at any one given interface) or delay variation.

- o Violated Packets. This indicates the number of packets for which a violation of a latency SLO occurred.
- o Violated Time Units (e.g. violated seconds, violated milliseconds). This indicates the number of time units during which one or more violations of SLOs were observed, regardless of how many violations took place during the same interval. This measure is useful in scenarios where bursts of violations might suddenly occur (e.g. due to temporary network congestion, during route convergence etc.) and the count of violated packets by itself might paint a misleading picture.

The following additional set of metrics may be useful in certain scenarios as well. However, their precise definition may be subject to policy and further discussion is needed:

- o Significantly Violated Packets. This indicates the number of packets for which a "significant" violation occurred, where "significant" implies an SLO that was not merely a near-miss but that missed the objective by a degree determined especially significant.
- o Significantly Violated Time Units (e.g. significantly violated seconds, significantly violated milliseconds). This indicates the number of time units during which any significant violation occurred.
- o Severely Violated Time Units (e.g. severely violated seconds, severely violated milliseconds). "Severe" here refers to the occurrence of multiple violations within the same time unit. The definition of "severe" may be subject to policy; it may also take into account the significance of the violations that occur.

Note that there is no definition of Severely Violated Packets. The term "severe" is used in conjunction with the occurrence of multiple violations related to multiple packets, not any one packet in isolation.

From these first-order metrics, second-order metrics can be defined that build on the first set of metrics. Some of these metrics are modeled after Mean Time Between Failure, or MTBF metrics - a "failure" in this context referring to a failure to deliver a packet according to its SLO.

- o Time since last violated time unit (i.e., since last violated ms, since last violated second). (This parameter is particularly useful for the monitoring of the current health.)
- o Packets since last violated packet. (This parameter is particularly useful for the monitoring of the current health.)
- o Mean time between violated time units (i.e. between violated milliseconds, between violated seconds). This refers to the arithmetic mean of time between violations such as violated time units.
- o Mean packets between violations. This refers to the arithmetic mean of the number of SLO-compliant packets between SLO violations. (Another variation of "MTBF" in a service setting.)

The same set of metrics can also be applied to significant violations, and to severe violations:

- o Time since last significantly violated time unit (i.e., since last significantly violated ms, since last significantly violated second).
- o Time since last severely violated time unit (i.e., since last severely violated ms, since last severely violated second).
- o Packets since last significantly violated packet.
- o Mean time between significantly violated time units (i.e. between significantly violated milliseconds, between significantly violated seconds).
- o Mean time between severely violated time units (i.e. between severely violated milliseconds, between severely violated seconds).
- o Mean packets between significant violations. This refers to the arithmetic mean of the number of SLO-compliant packets between significant SLO violations.

The next set of metrics puts the violations in relationship to non-violations. It is intended to provide an analogous measure to that

of availability, typically defined as the number of time units during which a system (or service) is unavailable divided by the total number of time units. In analogy, a time unit that is "violated" can be viewed as one in which a service is not available with the advertised precision:

- o Precision availability (of milliseconds, of seconds): the ratio between violated time units (seconds, milliseconds) and the total time units for the duration of the service.
- o Analogous metrics for precision availability re: severely violated time units, re: significantly violated time units.

It should be noted that certain Service Level Agreements may be statistical in nature, requiring the service levels of packets in a flow to adhere to certain distributions. For example, an SLA might state that any given SLO applies only to a certain percentage of packets, allowing for a certain amount of violations to take place. A "violated packet" in that case does not necessarily constitute an SLO violation. However, it is still useful to maintain those statistics, as the number of violated packets still matters when looked at in proportion to the total number of packets.

Along that vein, an SLA might establish an SLO of, say, end-to-end latency to not exceed 20ms for 99% of packets, to not exceed 25ms for 99.999% of packets, and to never exceed 30ms for anything beyond. In that case, any individual packet missing the 20 ms latency target cannot be considered an SLO violation in itself, but compliance with the SLO may need to be assessed after the fact.

To support statistical SLAs more directly, it is feasible to support additional metrics, such as metrics that represent histograms for service level parameters with buckets corresponding to individual service level objectives. For the example just given, a histogram for a given flow could be maintained with three buckets: one containing the count of packets within 20ms, a second with a count of packets between 20 and 25ms (or simply all within 25ms), a third with a count of packet between 25 and 30ms (or simply all packets within 30ms, and a fourth with a count of anything beyond (or simply a total count). Of course, the number of buckets and the boundaries between those buckets should correspond to the needs of the application respectively SLA, i.e. to the specific guarantees and SLOs that were provided. The definition of histogram metrics is for further study.

5. Discussion Items

The following is a list of items for which further discussion is needed as to whether they should be included in the scope of this specification:

- o A YANG data model
- o A set of IPFIX Information Elements
- o Statistical metrics: e.g. histograms/buckets
- o Policies regarding the definition of "significant" and "severe" violations
- o Additional second-order metrics, such as "longest disruption of service time" (measuring consecutive time units with violations)

6. IANA Considerations

TBD

7. Security Considerations

Instrumentation for metrics that are used to assess compliance with SLOs constitute an interesting target for an attacker. By interfering with the maintaining of such metrics, services could be falsely identified as being in compliance (when they are not), or vice-versa flagged as being non-compliant (when indeed they are). While this document does not specify how networks should be instrumented to maintain the identified metrics, such instrumentation needs to be properly secured to ensure accurate measurements and prohibit tampering with metrics being kept.

Where metrics are being defined relative to an SLO, the configuration of those SLOs needs to be properly secured. Likewise, where SLOs can be adjusted, it needs to be clear which particular SLO any given metrics instance refers to. The same service levels that constitute SLO violations for one flow, and that should be maintained as part of the "violated time units", "violated packets", and related metrics, may be perfectly compliant for another flow. Where it is not possible to properly tie together SLOs and violation metrics, it will be preferable to merely maintain statistics about service levels that were delivered (for example, overall histograms of end-to-end latency), without assessing which of these constitute violations.

By the same token, where the definition of what constitutes a "severe" violation or a "significant" violation depends on policy or

context, the configuration of such policy or context needs to be specially secured and the configuration of this policy be bound to the metrics being maintained. This way it will be clear which policy was in effect when those metrics were being assessed. An attacker that is able to tamper with such policies will render the corresponding metrics useless (in the best case) or misleading (in the worst case).

8. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC2863] McCloghrie, K. and F. Kastenholtz, "The Interfaces Group MIB", RFC 2863, DOI 10.17487/RFC2863, June 2000, <<https://www.rfc-editor.org/info/rfc2863>>.
- [RFC7011] Claise, B., Ed., Trammell, B., Ed., and P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information", STD 77, RFC 7011, DOI 10.17487/RFC7011, September 2013, <<https://www.rfc-editor.org/info/rfc7011>>.
- [RFC7012] Claise, B., Ed. and B. Trammell, Ed., "Information Model for IP Flow Information Export (IPFIX)", RFC 7012, DOI 10.17487/RFC7012, September 2013, <<https://www.rfc-editor.org/info/rfc7012>>.
- [RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <<https://www.rfc-editor.org/info/rfc7950>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8343] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 8343, DOI 10.17487/RFC8343, March 2018, <<https://www.rfc-editor.org/info/rfc8343>>.
- [RFC8655] Finn, N., Thubert, P., Varga, B., and J. Farkas, "Deterministic Networking Architecture", RFC 8655, DOI 10.17487/RFC8655, October 2019, <<https://www.rfc-editor.org/info/rfc8655>>.

Authors' Addresses

Alexander Clemm
Futurewei
2330 Central Expressway
Santa Clara CA 95050
USA

Email: ludwig@clemm.org

John Strassner
Futurewei
2330 Central Expressway
Santa Clara CA 95050
USA

Email: strazpdj@gmail.com

Jerome Francois
Inria
615 Rue du Jardin Botanique
Villers-les-Nancy 54600
France

Email: jerome.francois@inria.fr

Network Working Group
Internet-Draft
Intended status: Experimental
Expires: April 27, 2022

Z. Du
P. Liu
China Mobile
October 24, 2021

Self-Contained Alternate-Marking Mechanism for Performance Monitoring in
High-Quality Network
draft-du-ippm-self-contained-alt-mark-00

Abstract

This document introduces a self-contained method that can involve the client in based on some extensions to the alternate-marking (coloring) technique.

Requirements Language

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

Status of This Memo

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

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

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

This Internet-Draft will expire on April 27, 2022.

Copyright Notice

Copyright (c) 2021 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. Traditional Mechanism Description	3
3. Proposed Mechanism Description	4
4. Analysis of the Potential Problems	5
5. IANA Considerations	5
6. Security Considerations	5
7. Acknowledgements	5
8. References	5
8.1. Normative References	6
8.2. Informative References	6
Authors' Addresses	6

1. Introduction

The network operators are planning to provide network services with higher quality than the traditional BE (Best Effort) service, such as the DetNet service [RFC8655] and the Network Slicing service. In these practices, it is important to monitor the performance of the service, such as the packet loss, delay, and jitter of the flow with guaranteed quality.

In [RFC8321], an alternate-marking method for passive and hybrid performance monitoring is proposed. It marks the packet by using one or more bits in the packet headers, and collects the number of packets in a block sent on one end and the number of packets in the same block received on the other end. Finally, the two values are compared and accordingly, the packet loss of the flow are computed.

The alternate-marking method is potential applied to any kind of packet-based traffic, and easy to implement. However, a controller or NMS needs to collect the information from the coloring point and the monitoring point, and correlate the two pieces of information by using the same block ID. It is hard to make it an end-to-end solution because the client is not in the scope.

In this document, we propose a method that can involve the client in based on some extensions to the alternate-marking (coloring) technique. In this method, the block information is serialized and encoded in the packets of the block by the client. Then, the monitoring points can recover the information from the received

packets, such as the block ID, number of packets in the block, timestamps in the packet, and compute the target measurement values.

2. Traditional Mechanism Description

As described in [RFC8321], the alternate-marking method is based on the "block", which represents a measurable entity unambiguously recognizable by all network devices along the path.

In the alternate-marking (coloring) technique, the coloring point creates packet blocks, colors the packets in the block, and reports information including block ID to the controller or the NMS. The monitoring points recognize the coloring information, record some needed information and report it to the controller or the NMS.

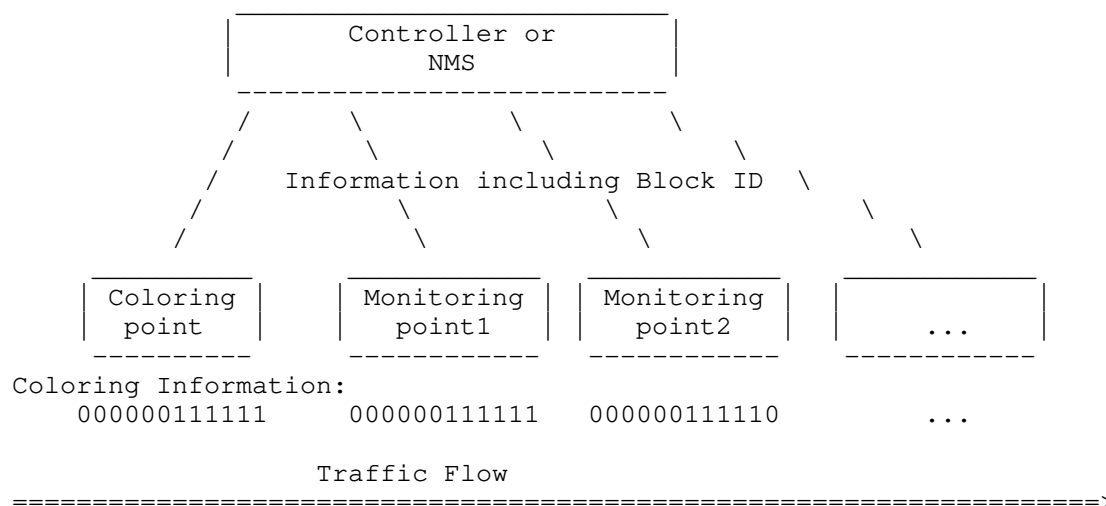


Figure 1: Mechanism in the traditional alternate-marking method

For example, if some packets are lost in the network, the packet numbers of the same block will be different between the coloring point and the monitoring point. If we need to compute the delay or jitter of the flow, the coloring point and the monitoring point can also report the timestamps of the packets in the block to the controller or NMS.

Traffic coloring can be implemented by setting a specific bit in the packet header and changing the value of that bit periodically. Thus, we only need two colors, and the packets belonging to the same block

have the same color, whilst consecutive blocks will have different colors.

When the color changes, the previous block terminates and the new one begins. Two mechanisms of switching color are introduced in [RFC8321]. The first one is to switch the color after a fixed number of packets. The second one is to switch according to a fixed timer. For example, the timer may be 5 minutes.

3. Proposed Mechanism Description

To make the block information self-contained in the block, we need to occupy another specific bit to encode the block information. Thus, the client in the proposed mechanism needs not to report anything to the controller or NMS, and the monitoring points can compute target measurement values themselves and report any problem if needed.

For example, we assume the fixed timer mechanism is used, and there are about 300 packets in a block. In the client, each packet carries one bit of the block information. Thus, if all the packets are received orderly, a monitoring point can recover the block information encoded in those 300 packets.

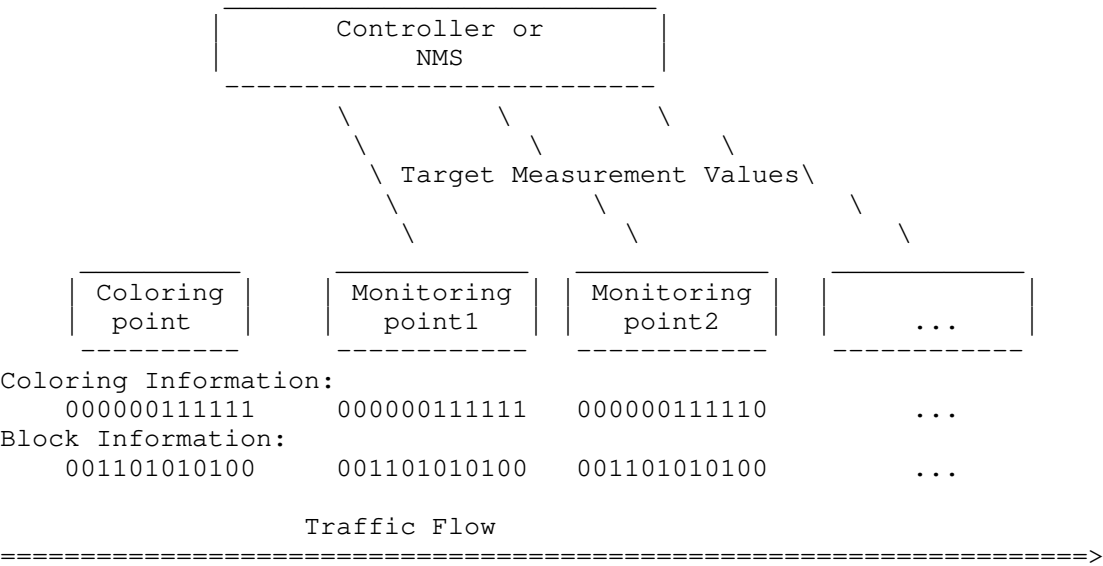


Figure 2: Mechanism in the self-contained alternate-marking method

The block information can include the block ID (32 bits), CRC (32bits), and some TLVs as described below.

- o TLV 1 may be the interval of the block (32bits).
- o TLV 2 may be the packet number of the last block (32bits).
- o TLV 3 may be the timestamp of the first packet in the block (32bits).

The encoding of the block information is done in the client, and the monitoring points need to understand the meaning of the encoding.

4. Analysis of the Potential Problems

As described in the last section, we assume that all the packets in a block are received in the monitoring point orderly. Normally, it is hard for the IP network with a relatively high packet loss rate. However, the situation may be much better in the DetNet service or the Network Slicing service, for which no or few packets would be lost. Meanwhile, an additional recovery block may also appear after several blocks, in which we will encode recovery information for the past several blocks, instead of the block information. Other fault tolerance mechanisms can also be considered.

Another problem is similar to the situation in [RFC8321]. It is whether we can find at least two reserved bits in the packet header to encode the coloring information and the block information. The detailed analysis can be found in that document.

5. IANA Considerations

TBD.

6. Security Considerations

TBD.

7. Acknowledgements

TBD.

8. References

8.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC8321] Fioccola, G., Ed., Capello, A., Cociglio, M., Castaldelli, L., Chen, M., Zheng, L., Mirsky, G., and T. Mizrahi, "Alternate-Marking Method for Passive and Hybrid Performance Monitoring", RFC 8321, DOI 10.17487/RFC8321, January 2018, <<https://www.rfc-editor.org/info/rfc8321>>.

8.2. Informative References

- [RFC8655] Finn, N., Thubert, P., Varga, B., and J. Farkas, "Deterministic Networking Architecture", RFC 8655, DOI 10.17487/RFC8655, October 2019, <<https://www.rfc-editor.org/info/rfc8655>>.

Authors' Addresses

Zongpeng Du
China Mobile
No.32 XuanWuMen West Street
Beijing 100053
China

Email: duzongpeng@foxmail.com

Peng Liu
China Mobile
No.32 XuanWuMen West Street
Beijing 100053
China

Email: liupengyjy@chinamobile.com

Internet Engineering Task Force
Internet-Draft
Intended status: Proposed Standard
Expires: 29 August 2022

N. Elkins
Inside Products, Inc.
M. Ackermann
BCBS Michigan
A. Deshpande
NITK Surathkal
T. Pecorella
A. Rashid
University of Florence
25 February 2022

IPv6 Performance and Diagnostic Metrics Version 2 (PDMv2) Destination
Option
draft-elkins-ippm-encrypted-pdmv2-02.txt

Abstract

RFC8250 describes an optional Destination Option (DO) header embedded in each packet to provide sequence numbers and timing information as a basis for measurements. As this data is sent in clear-text, this may create an opportunity for malicious actors to get information for subsequent attacks. This document defines PDMv2 which has a lightweight handshake (registration procedure) and encryption to secure this data. Additional performance metrics which may be of use are also defined.

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 29 August 2022.

Copyright Notice

Copyright (c) 2022 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. Current Performance and Diagnostic Metrics (PDM)	3
1.2. PDMv2 Introduction	3
2. Conventions used in this document	3
3. Terminology	4
4. Protocol Flow	4
4.1. Registration Phase	5
4.1.1. Rationale of Primary (Writer) and Secondary (Reader) Roles	5
4.1.2. Diagram of Registration Flow	5
4.2. Primary (Writer) Client - Primary (Writer) Server Negotiation Phase	5
4.3. Primary (Writer) Server / Client - Secondary (Reader) Server / Client Registration Phase	6
4.4. Secondary (Reader) Client - Secondary (Reader) Server communication	6
5. Security Goals	7
5.1. Security Goals for Confidentiality	7
5.2. Security Goals for Integrity	7
5.3. Security Goals for Authentication	7
5.4. Cryptographic Algorithm	8
6. PDMv2 Destination Options	8
6.1. Destinations Option Header	8
6.2. Metrics information in PDMv2	8
6.3. PDMv2 Layout	9
7. Security Considerations	12
8. Privacy Considerations	12
9. IANA Considerations	12
10. Contributors	12
11. References	12
11.1. References	12
11.2. Normative References	12
11.3. Informative References	13
Appendix A. Rationale for Primary (Writer) Server / Primary (Writer) Client	13
A.1. One Client / One Server	13
A.2. Multiple Clients / One Server	14

A.3. Multiple Clients / Multiple Servers	15
A.4. Primary (Writer) Client / Primary (Writer) Server	15
Appendix B. Sample Implementation of Registration	15
B.1. Overall summary	15
B.2. High level flow	15
B.3. Commands used	16
Appendix C. Change Log	16
Appendix D. Open Issues	16
Authors' Addresses	16

1. Introduction

1.1. Current Performance and Diagnostic Metrics (PDM)

The current PDM is an IPv6 Destination Options header which provides information based on the metrics like Round-trip delay and Server delay. This information helps to measure the Quality of Service (QoS) and to assist in diagnostics. However, there are potential risks involved transmitting PDM data during a diagnostics session.

PDM metrics can help an attacker understand about the type of machine and its processing capabilities. Inferring from the PDM data, the attack can launch a timing attack. For example, if a cryptographic protocol is used, a timing attack may be launched against the keying material to obtain the secret.

Along with this, PDM does not provide integrity. It is possible for a Man-In-The-Middle (MITM) node to modify PDM headers leading to incorrect conclusions. For example, during the debugging process using PDM header, it can mislead the person showing there are no unusual server delays.

1.2. PDMv2 Introduction

PDMv2 introduces confidential, integrity and authentication.

TBD

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14, RFC 2119 [RFC2119] .

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

3. Terminology

- * Primary (Writer) Client (WC): An authoritative node that creates cryptographic keys for multiple reader clients.
- * Primary (Writer) Server (WS): An authoritative node that creates cryptographic keys for multiple reader servers.
- * Secondary (Reader) Client (RC): An endpoint node which initiates a session with a listening port and sends PDM data. Connects to the Primary (Writer) Client to get cryptographic key material.
- * Secondary (Reader) Server (RS): An endpoint node which has a listening port and sends PDM data. Connects to the Primary (Writer) Server to get cryptographic key material.

Note: a client may act as a server (have listening ports).

- * Symmetric Key (K): A uniformly random bitstring as an input to the encryption algorithm, known only to Secondary (Reader) Clients and Secondary (Reader) Servers, to establish a secure communication.
- * Public and Private Keys: A pair of keys that is used in asymmetric cryptography. If one is used for encryption, the other is used for decryption. Private Keys are kept hidden by the source of the key pair generator, but Public Key is known to everyone. pkX (Public Key) and skX (Private Key). Where X can be, any client or any server.
- * Pre-shared Key (PSK): A symmetric key. Uniformly random bitstring, shared between any client or any server or a key shared between an entity that forms client-server relationship. This could happen through an out-of band mechanism: e.g., a physical meeting or use of another protocol.
- * Session Key: A temporary key which acts as a symmetric key for the whole session.

4. Protocol Flow

The protocol will proceed in 3 steps.

Step 1: Negotiation between Primary (Writer) Server and Primary (Writer) Client.

Step 2: Registration between Primary (Writer) Server / Client and Secondary (Reader) Server / Client

Step 3: PDM data flow between Secondary (Reader) Client and Secondary (Reader) Server

After-the-fact (or real-time) data analysis of PDM flow may occur by network diagnosticians or network devices. The definition of how this is done is out of scope for this document.

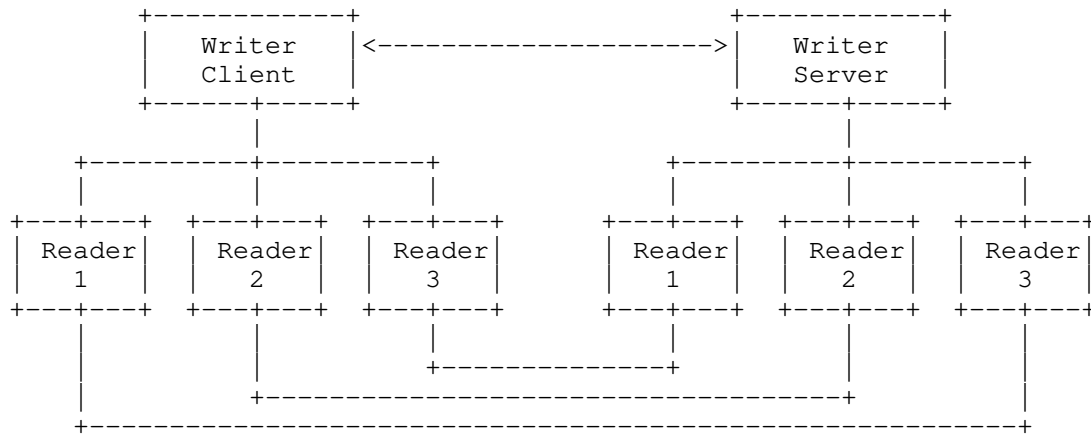
4.1. Registration Phase

4.1.1. Rationale of Primary (Writer) and Secondary (Reader) Roles

Enterprises have many servers and many clients. These clients and servers may be in multiple locations. It may be less overhead to have a secure location (ex. Shared database) for servers and clients to share keys. Otherwise, each client needs to keep track of the keys for each server.

Please view Appendix 1 for some sample topologies and further explanation.

4.1.2. Diagram of Registration Flow



4.2. Primary (Writer) Client - Primary (Writer) Server Negotiation Phase

The two entities exchange a set of data to ensure the respective identities.

They use HPKE KEM to negotiate a "SharedSecret".

4.3. Primary (Writer) Server / Client - Secondary (Reader) Server / Client Registration Phase

The "SharedSecret" is shared securely:

- * By the Primary (Writer) Client to all the Secondary (Reader) Clients under its control. How this is achieved is beyond the scope of the present specification.
- * By the Primary (Writer) Server to all the Secondary (Reader) Servers under its control. How this is achieved is beyond the scope of the present specification.

4.4. Secondary (Reader) Client - Secondary (Reader) Server communication

Each Client and Server derive a "SessionTemporaryKey" by using HPKE KDF, using the following inputs:

- * The "SharedSecret".
- * The 5-tuple (SrcIP, SrcPort, DstIP, DstPort, Protocol) of the communication.
- * A Key Rotation Index (Kri).

The Kri SHOULD be initialized to zero.

The server and client initialize (separately) a pseudo-random non-repeating sequence between 1 and $2^{15}-1$. How to generate this sequence is beyond the scope of this document, and does not affect the rest of the specification. When the sequence is used fully, or earlier if appropriate, the sender signals the other party that a key change is necessary. This is achieved by flipping the "F bit" and resetting the PRSEQ. The receiver increments the Kri of the sender, and derives another SessionTemporaryKey to be used for decryption.

It shall be stressed that the two SessionTemporaryKeys used in the communication are never the same, as the 5-tuple is reversed for the Server and Client. Moreover, the time evolution of the respective Kri can be different. As a consequence, each entity must maintain a table with (at least) the following informations:

- * Flow 5-tuple, Own Kri, Other Kri

An implementation might optimize this further by caching the OwnSessionTemporaryKey (used in Encryption) and OtherSessionTemporaryKey (used in Decryption).

5. Security Goals

As discussed in the introduction, PDM data can represent a serious data leakage in presence of a malicious actor.

In particular, the sequence numbers included in the PDM header allows correlating the traffic flows, and the timing data can highlight the operational limits of a server to a malicious actor. Moreover, forging PDM headers can lead to unnecessary, unwanted, or dangerous operational choices, e.g., to restore an apparently degraded Quality of Service (QoS).

Due to this, it is important that the confidentiality and integrity of the PDM headers is maintained. PDM headers can be encrypted and authenticated using the methods discussed in section [x], thus ensuring confidentiality and integrity. However, if PDM is used in a scenario where the integrity and confidentiality is already ensured by other means, they can be transmitted without encryption or authentication. This includes, but is not limited to, the following cases:

- a) PDM is used over an already encrypted medium (For example VPN tunnels).
- b) PDM is used in a link-local scenario.
- c) PDM is used in a corporate network where there are security measures strong enough to consider the presence of a malicious actor a negligible risk.

5.1. Security Goals for Confidentiality

PDM data must be kept confidential between the intended parties, which includes (but is not limited to) the two entities exchanging PDM data, and any legitimate party with the proper rights to access such data.

5.2. Security Goals for Integrity

PDM data must not be forged or modified by a malicious entity. In other terms, a malicious entity must not be able to generate a valid PDM header impersonating an endpoint, and must not be able to modify a valid PDM header.

5.3. Security Goals for Authentication

TBD

5.4. Cryptographic Algorithm

Symmetric key cryptography has performance benefits over asymmetric cryptography; asymmetric cryptography is better for key management. Encryption schemes that unite both have been specified in [RFC1421], and have been participating practically since the early days of public-key cryptography. The basic mechanism is to encrypt the symmetric key with the public key by joining both yields. Hybrid public-key encryption schemes (HPKE) [RFC9180] used a different approach that generates the symmetric key and its encapsulation with the public key of the receiver.

Our choice is to use the HPKE framework that incorporates key encapsulation mechanism (KEM), key derivation function (KDF) and authenticated encryption with associated data (AEAD). These multiple schemes are more robust and significantly efficient than the traditional schemes and thus lead to our choice of this framework.

6. PDMv2 Destination Options

6.1. Destinations Option Header

The IPv6 Destination Options extension header [RFC8200] is used to carry optional information that needs to be examined only by a packet's destination node(s). The Destination Options header is identified by a Next Header value of 60 in the immediately preceding header and is defined in RFC 8200 [RFC8200]. The IPv6 PDMv2 destination option is implemented as an IPv6 Option carried in the Destination Options header.

6.2. Metrics information in PDMv2

The IPv6 PDMv2 destination option contains the following base fields:

- SCALEDTLR: Scale for Delta Time Last Received
- SCALEDTLS: Scale for Delta Time Last Sent
- GLOBALPTR: Global Pointer
- PSNTP: Packet Sequence Number This Packet
- PSNLR: Packet Sequence Number Last Received
- DELTATLR: Delta Time Last Received
- DELTATLS: Delta Time Last Sent

PDMv2 adds a new metric to the existing PDM [RFC8250] called the Global Pointer. The existing PDM fields are identified with respect to the identifying information called a "5-tuple".

The 5-tuple consists of:

SADDR: IP address of the sender
 SPORT: Port for the sender
 DADDR: IP address of the destination
 DPORT: Port for the destination
 PROTC: Upper-layer protocol (TCP, UDP, ICMP, etc.)

Unlike PDM fields, Global Pointer (GLOBALPTR) field in PDMv2 is defined for the SADDR type. Following are the SADDR address types considered:

- a) Link-Local
- b) Global Unicast

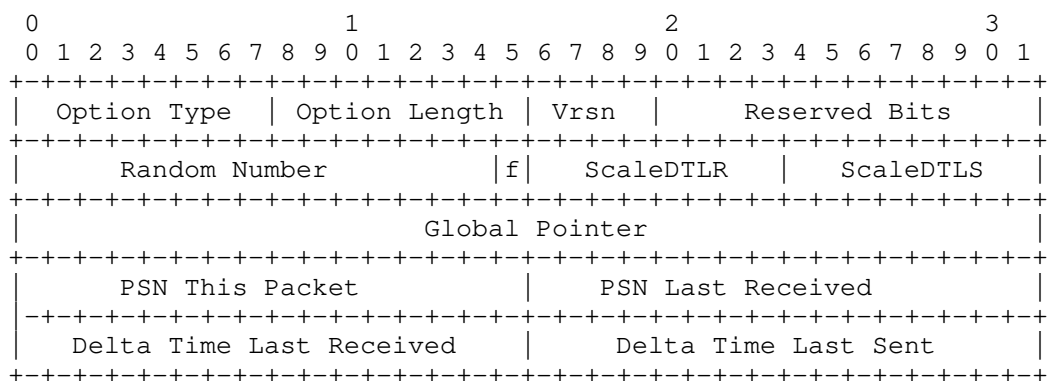
The Global Pointer is treated as a common entity over all the 5-tuples with the same SADDR type. It is initialised to the value 1 and increments for every packet sent. Global Pointer provides a measure of the amount of IPv6 traffic sent by the PDMv2 node.

When the SADDR type is Link-Local, the PDMv2 node sends Global Pointer defined for Link-Local addresses, and when the SADDR type is Global Unicast, it sends the one defined for Global Unicast addresses.

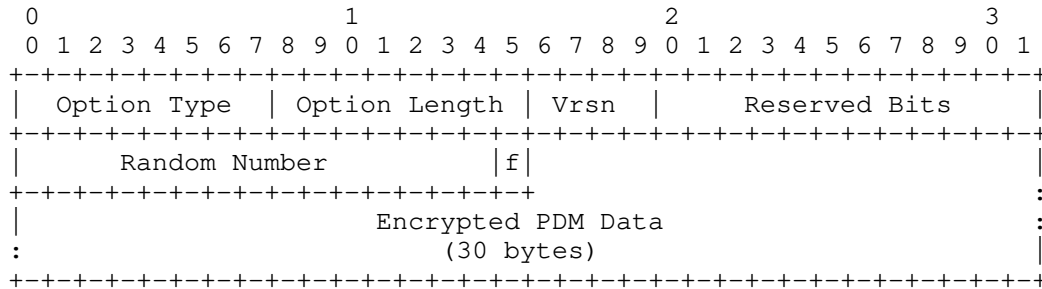
6.3. PDMv2 Layout

PDMv2 has two different header formats corresponding to whether the metric contents are encrypted or unencrypted. The difference between the two types of headers is determined from the Options Length value.

Following is the representation of the unencrypted PDMv2 header:



Following is the representation of the encrypted PDMv2 header:



Option Type

0x0F

8-bit unsigned integer. The Option Type is adopted from RFC 8250 [RFC8250].

Option Length

0x12: Unencrypted PDM

0x22: Encrypted PDM

8-bit unsigned integer. Length of the option, in octets, excluding the Option Type and Option Length fields. The options length is used for differentiating PDM [RFC8250], unencrypted PDMv2 and encrypted PDMv2.

Version Number

0x2

4-bit unsigned number.

Reserved Bits

12-bits.

Reserved bits for future use. They are initialised to 0 for PDMv2.

Random Number

15-bit unsigned number.

TBD

Flag Bit

1-bit field.

TBD

Scale Delta Time Last Received (SCALEDTLR)

8-bit unsigned number.

This is the scaling value for the Delta Time Last Sent (DELTATLS) field.

Scale Delta Time Last Sent (SCALEDTLS)

8-bit unsigned number.

This is the scaling value for the Delta Time Last Sent (DELTATLS) field.

Global Pointer

32-bit unsigned number.

Global Pointer is initialized to 1 for the different source address types and incremented monotonically for each packet with the corresponding source address type.

This field stores the Global Pointer type corresponding to the SADDR type of the packet.

Packet Sequence Number This Packet (PSNTP)

16-bit unsigned number.

This field is initialized at a random number and is incremented monotonically for each packet of the 5-tuple.

Packet Sequence Number Last Recieved (PSNLR)

16-bit unsigned number.

This field is the PSNTP of the last received packet on the 5-tuple.

Delta Time Last Received (DELTATLR)

16-bit unsigned integer.

The value is set according to the scale in SCALEDTLR.

Delta Time Last Received =
(send time packet n - receive time packet (n - 1))

Delta Time Last Sent (DELTATLS)

16-bit unsigned integer.

The value is set according to the scale in SCALEDTLS.

Delta Time Last Sent =
(receive time packet n - send time packet (n - 1))

7. Security Considerations

TBD

8. Privacy Considerations

TBD

9. IANA Considerations

This memo includes no request to IANA.

10. Contributors

TBD

11. References

11.1. References

11.2. Normative References

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

- [RFC8250] Elkins, N., Hamilton, R., and M. Ackermann, "IPv6 Performance and Diagnostic Metrics (PDM) Destination Option", RFC 8250, DOI 10.17487/RFC8250, September 2017, <<https://www.rfc-editor.org/info/rfc8250>>.
- [RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <<https://www.rfc-editor.org/info/rfc8200>>.

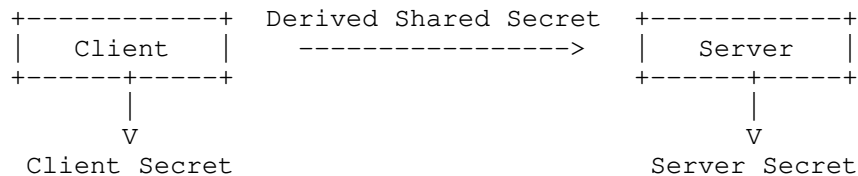
11.3. Informative References

- [RFC9180] Barnes, R., Bhargavan, K., Lipp, B., and C. Wood, "Hybrid Public Key Encryption", RFC 9180, DOI 10.17487/RFC9180, February 2022, <<https://www.rfc-editor.org/info/rfc9180>>.
- [RFC1421] Linn, J., "Privacy Enhancement for Internet Electronic Mail: Part I: Message Encryption and Authentication Procedures", RFC 1421, DOI 10.17487/RFC1421, February 1993, <<https://www.rfc-editor.org/info/rfc1421>>.

Appendix A. Rationale for Primary (Writer) Server / Primary (Writer) Client

A.1. One Client / One Server

Let's start with one client and one server.



The Client and Server create public / private keys and derive a shared secret. Let's not consider Authentication or Certificates at this point.

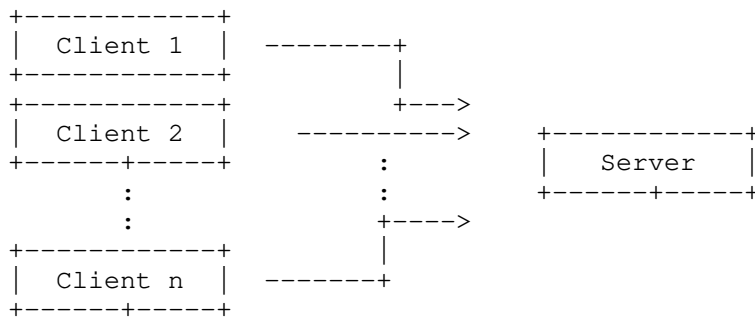
What is stored at the Client and Server to be able to encrypt and decrypt packets? The shared secret or private key.

Since we only have one Server and one Client, then we don't need to have any kind of identifier for which private key to use for which Server or Client because there is only one of each.

Of course, this is a ludicrous scenario since no real organization of interest has only one server and one client.

A.2. Multiple Clients / One Server

So, let's try with multiple clients and one Primary (Writer) server



The Clients and Server create public / private keys and derive a shared secret. Each Client has a unique private key.

What is stored at the Client and Server to be able to encrypt and decrypt packets?

Clients each store a private key. Server stores: Client Identifier and Private Key.

Since we only have one Server and multiple Clients, then the Clients don't need to have any kind of identifier for which private key to use for which Server but the Server needs to know which private key to use for which Client. So, the Server has to store an identifier as well as the Key.

But, this also is a ludicrous scenario since no real organization of interest has only one server.

A.3. Multiple Clients / Multiple Servers

When we have multiple clients and multiple servers, then each not only does the Server need to know which key to use for which Client, but the Client needs to know which private key to use for which Server.

A.4. Primary (Writer) Client / Primary (Writer) Server

Based on this rationale, we have chosen a Primary (Writer) Server / Primary (Writer) Client topology.

Appendix B. Sample Implementation of Registration

B.1. Overall summary

In the Registration phase, the objective is to generate a shared secret that will be used in encryption and decryption during the Data Transfer phase. We have adopted a Primary-Secondary architecture to represent the clients and servers (see Section 4.1.1). The primary server and primary client perform Key Encapsulation Mechanism (KEM) [RFC9180] to generate a primary shared secret. The primary server shares this secret with secondary servers, whereas the primary client performs Key Derivation Function (KDF) [RFC9180] to share client-specific secrets to corresponding secondary clients. During the Data Transfer phase, the secondary servers generate the client-specific secrets on the arrival of the first packet from the secondary client.

B.2. High level flow

The following steps describe the protocol flow:

1. Primary client initiates a request to the primary server. The request contains a list of available ciphersuites for KEM, KDF, and AEAD.
2. Primary server responds to the primary client with one of the available ciphersuites and shares its public key.
3. Primary client generates a secret and its encapsulation. The primary client sends the encapsulation and a salt to the primary server. The salt is required during KDF in the Data Transfer phase.
4. Primary Server generates the secret with the help of the encapsulation and responds with a status message.
5. Primary server shares this key with secondary servers over TLS.

6. Primary client generates the client-specific secrets with the help of KDF by using the info parameter as the Client IP address. The primary client shares these keys with the corresponding secondary clients over TLS.

B.3. Commands used

Two commands are used between the primary client and the primary server to denote the setup and KEM phases. Along with this, we have a "req / resp" to indicate whether it's a request or response.

Between primary and secondary entities, we have one command to denote the sharing of the secret keys.

Appendix C. Change Log

Note to RFC Editor: if this document does not obsolete an existing RFC, please remove this appendix before publication as an RFC.

Appendix D. Open Issues

Note to RFC Editor: please remove this appendix before publication as an RFC.

Authors' Addresses

Nalini Elkins
Inside Products, Inc.
36A Upper Circle
Carmel Valley, CA, 93924
United States of America

Phone: +1 831 234 4232
Email: nalini.elkins@insidethestack.com

Michael Ackermann
BCBS Michigan
P.O. Box 2888
Detroit, Michigan, 48231
United States of America

Phone: +1 248 703 3600
Email: mackermann@bcbsm.com
URI: <http://www.bcbsm.com>

Ameya Deshpande
NITK Surathkal
Pashan-Baner Link Road, Pashan
Pune, Maharashtra, 411021
India

Phone: +91 96893 26060
Email: ameyanrd@gmail.com
URI: <https://www.nitk.ac.in/>

Tommaso Pecorella
University of Florence
Dept. of Information Engineering, Via di Santa Marta, 3, 50139
Firenze
Italy

Phone: +39 055 2758540
Email: tommaso.pecorella@unifi.it
URI: <https://www.unifi.it/>

Adnan Rashid
University of Florence
Dept. of Information Engineering, Via di Santa Marta, 3, 50139
Firenze
Italy

Phone: +39 347 9821 467
Email: adnan.rashid@unifi.it
URI: <https://www.unifi.it/>

IPPM Working Group
Internet-Draft
Intended status: Standards Track
Expires: 8 August 2022

R. Gandhi, Ed.
C. Filsfils
Cisco Systems, Inc.
D. Voyer
Bell Canada
M. Chen
Huawei
B. Janssens
Colt
S. Salsano
Universita di Roma "Tor Vergata"
4 February 2022

Simple Two-Way Direct Loss Measurement Procedure
draft-gandhi-ippm-simple-direct-loss-02

Abstract

This document defines Simple Two-Way Direct Loss Measurement (DLM) procedure that can be used for Alternate-Marking Method for detecting accurate data packet loss in a network. Specifically, DLM probe packets are defined for both unauthenticated and authenticated modes and they are efficient for hardware-based implementation.

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 8 August 2022.

Copyright Notice

Copyright (c) 2022 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 Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction	2
2. Conventions Used in This Document	4
2.1. Requirements Language	4
2.2. Abbreviations	4
2.3. Reference Topology	5
3. Overview	5
4. Session-Sender Direct Loss Measurement Probe Packet	6
5. Session-Reflector Direct Loss Measurement Probe Packet	9
6. Data Loss Calculation	12
7. Optional Extensions	12
8. Integrity Protection and Confidentiality Protection	12
9. Operational Considerations	13
10. Security Considerations	13
11. IANA Considerations	13
12. References	13
12.1. Normative References	13
12.2. Informative References	14
Acknowledgments	15
Authors' Addresses	15

1. Introduction

Many Service Provider Service Level Agreements (SLAs) depend on the ability to measure performance loss metric experienced by the Customer data traffic flow. Accurate Customer data packet loss can be measured by using a Direct Loss Measurement (DLM) procedure. Currently there is no efficient active measurement procedure available for accurate data packet loss detection in IP networks. Note that an approach for conducting packet loss measurement in IP networks is documented in [RFC7680]. This approach requires clock synchronization between the measurement points and lacks support for accurate data packet loss measurement.

[ITU-Y1731] defines procedures for performance loss monitoring for Ethernet-based networks. Specifically, the Loss Measurement Message (LMM) defined in Section 9.12 of [ITU-Y1731] can be used for accurate frame loss measurement as described in Appendix II of that document. The procedure is specific to the Ethernet-based networks and does not apply to the IP networks.

The Simple Two-Way Active Measurement Protocol (STAMP) provides capabilities for the measurement of various performance metrics in IP networks [RFC8762] without the use of a control channel to pre-signal session parameters. The STAMP can be used for (synthetic or inferred) packet loss measurement based on the Sequence Number in the test packets, however, this method can only provide approximate packet loss metrics.

[RFC8972] defines optional extensions for STAMP. The STAMP test packet with the "Direct Measurement" TLV (Type 5) [RFC8972] can be used for combined timestamps and data packet counters collection. This method, however, has the following limitations when used for detecting data packet loss:

- * For only direct measurement, the STAMP "Direct Measurement" TLV in the test packet requires the hardware to support timestamps, in addition to data packet counters. One-way delay measurement also requires clock synchronization between the Session-Sender and Session-Reflector nodes.
- * The location of the transmit counter is not at the fixed location in the STAMP test packet with the "Direct Measurement" TLV. Also, the location of the transmit counter on the STAMP Session-Reflector reply test packet is not at the same location as the STAMP Session-Sender test packet using the "Direct Measurement" TLV. This makes it difficult to implement in hardware, e.g., for point-to-point links and circuits.
- * Furthermore, for hardware-based implementation, the optional "Direct Measurement" TLV adds unnecessary processing overhead on the Session-Reflector as not all STAMP Session-Sender test packets carry the "Direct Measurement" TLV and also there can be multiple TLV Types present.
- * The STAMP "Direct Measurement" TLV does not support 64-bit counters.
- * The STAMP "Direct Measurement" TLV does not support counters for bytes.

- * The STAMP "Direct Measurement" TLV does not support counters per traffic class.
- * The STAMP "Direct Measurement" TLV also does not identify the Block Number of the Direct Measurement, which is required for Alternate-Marking Method [RFC8321] for data packet loss measurement. The AMM also handles the case of out-of-order data packets.

This document defines Simple Two-Way Direct Loss Measurement (DLM) procedure that can be used for Alternate-Marking Method [RFC8321] for detecting accurate data packet loss in a network. Specifically, DLM probe packets are defined for both unauthenticated and authenticated modes and they are efficient for hardware-based implementation.

2. Conventions Used in This Document

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 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2.2. Abbreviations

AMM: Alternate-Marking Method.

DLM: Direct Loss Measurement.

HMAC: Hashed Message Authentication Code.

MBZ: Must be Zero.

PM: Performance Measurement.

SHA: Secure Hash Algorithm.

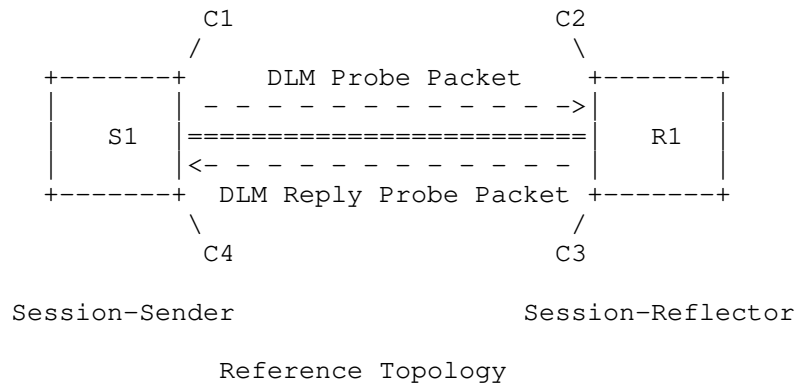
SSID: Sender Session Identifier.

STAMP: Simple Two-Way Active Measurement Protocol.

TTL: Time To Live.

2.3. Reference Topology

As shown in the reference topology, the Session-Sender S1 initiates a Direct Loss Measurement (DLM) probe packet over UDP transport. The Session-Reflector R1 receives the Session-Sender's DLM probe packet and acts according to the local configuration. The Session-Reflector R1 transmits a DLM reply probe packet to the Session-Sender S1.



3. Overview

For accurate data packet loss detection, the DLM probe packets are transmitted by the Session-Sender over UDP transport, and are used to collect the transmit and receive counters for the data traffic flow under measurement. The DLM reply probe packets are transmitted by the Session-Reflector to collect the transmit and receive counters for the data traffic flow under measurement in the reverse direction.

The DLM probe packets carry user-configured destination UDP port. The destination UDP port 862 is not used for the DLM probe packets. The user-configured destination UDP port follows the guidelines described in Section 4.1 of [RFC8762]. Different destination UDP port is used for DLM probe packets than the STAMP test packets defined in [RFC8762]. Hence, the Session-Sender and the Session-Reflector do not require backwards compatibility and support for STAMP.

A DLM session is identified by the 4-tuple (source and destination IP addresses, source and destination UDP port numbers). A DLM Session-Sender MAY generate a locally unique Sender Session Identifier (SSID). The SSID is a two-octet, non-zero unsigned integer. The SSID generation policy is implementation specific. An implementation MUST NOT assign the same identifier to different DLM sessions. A Session-Sender MAY use the SSID to identify a DLM session. If the SSID is used, it MUST be present in each probe packet of the given DLM session.

The DLM Session-Reflector operates in the Stateless mode. The DLM Session-Reflector does not maintain session state and will use the value in the Sequence Number field in the received probe packet as the value for the Sequence Number field in the reply probe packet. As a result, values in the Sequence Number and Session-Sender Sequence Number fields are the same in this mode.

In this document, the examples of DLM probe packets are shown with UDP header, however, the packets can be encapsulated with a different header based on the the transport protocol used in the network.

4. Session-Sender Direct Loss Measurement Probe Packet

In this document, base Session-Sender DLM probe packet formats are defined as shown in Figure 1 and Figure 2 for unauthenticated and authenticated modes, respectively. They are stand-alone DLM probe packet formats to carry the counters for the data traffic flow under measurement. The DLM probe packet formats are similar to the base STAMP test packet formats (for example the locations of the Counters and Timestamps).

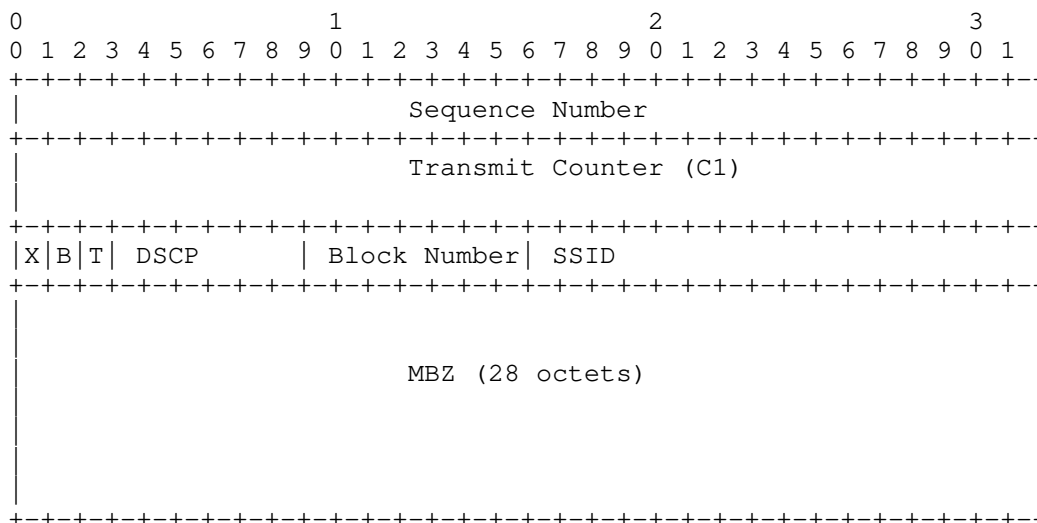


Figure 1: Session-Sender Direct Loss Measurement Probe Packet - Unauthenticated Mode

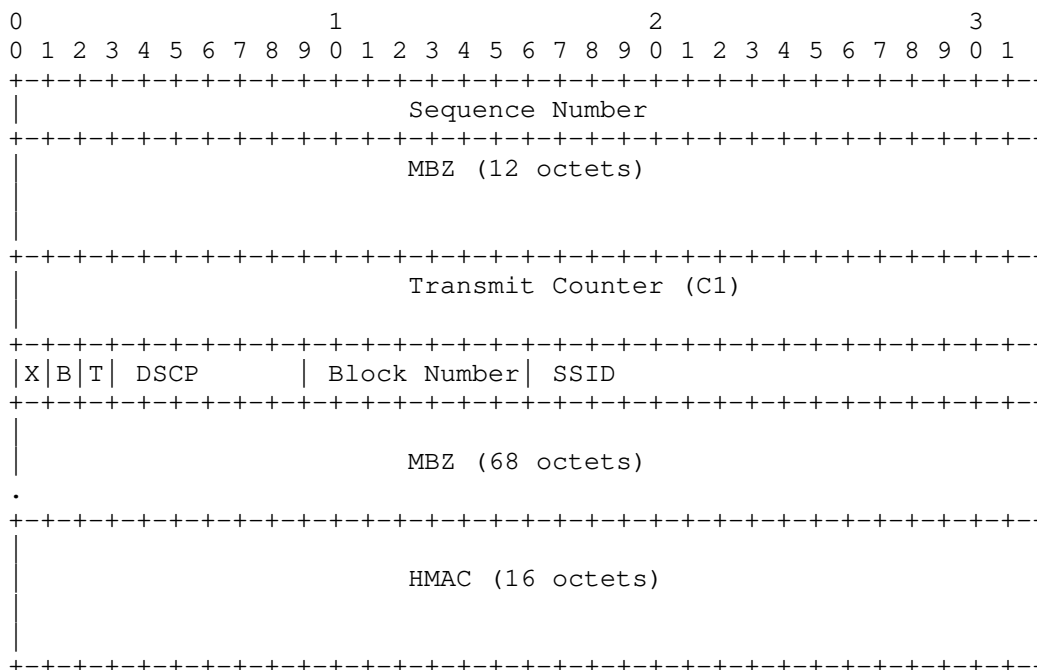


Figure 2: Session-Sender Direct Loss Measurement Probe Packet - Authenticated Mode

Fields are defined as the following:

Sequence Number (32-bit): For each new DLM session, its value starts at zero and is incremented by one with each transmitted DLM probe packet. The Sequence Number helps to check the DLM session state as active or not active, as well as detect probe packet drops.

Transmit Counter (64-bit): The number of packets or octets transmitted by the Session-Sender in the DLM probe packet. The counter is always written at the well-known fixed location in the DLM probe packet. This is an important property for hardware-based implementation, e.g., for point-to-point links and circuits. Counter is for the data traffic flow under measurement.

XBT Flags (3-bit): The meanings of the Flag bits are:

X: Extended counter format indicator. Indicates the use of extended (64-bit) counter values. Initialized to 1 upon creation (and prior to transmission) of a DLM probe packet. Set to 0 when the DLM probe packet is transmitted or received over an interface that writes 32-bit counter values.

B: Octet (byte) count. When set to 1, indicates that the Counter fields represent octet counts. The octet count applies to all packets within the DLM scope, and the octet count of a packet transmitted or received includes the total length of that packet (but excludes headers, labels, or framing of the channel itself). When set to 0, indicates that the Counter fields represent packet counts.

T: Traffic-class-specific measurement indicator. Set to 1 when the DLM session is scoped to data packets of a particular traffic class (DSCP value), and 0 otherwise. When set to 1, the DSCP field of the DLM probe packet indicates the measured traffic class.

DSCP (6-bit): DSCP of the data traffic flow being measured when T flag is set.

Block Number (7-bit): The Direct Loss Measurement using Alternate-Marking Method [RFC8321] requires to collect Block Number of the counters for the data traffic flow under measurement. To be able to correlate the transmit and receive counters of the matching Block Number, the Block Number of the counters carried in the DLM probe packets.

SSID (16-bit): DLM Sender Session Identifier.

HMAC: The use of the HMAC field is described in Section 4.4 of [RFC8762]. HMAC uses its own key and the mechanism to distribute the HMAC key is outside the scope of this document.

MBZ: Must be Zero. It MUST be all zeroed on the transmission and MUST be ignored on receipt.

5. Session-Reflector Direct Loss Measurement Probe Packet

The Session-Reflector receives the DLM Session-Sender probe packet and verifies it. If the DLM probe packet is validated, the Session-Reflector that supports this specification prepares and transmits the DLM reply probe packet. In this document, Session-Reflector DLM reply probe packet formats are defined as shown in Figure 3 and Figure 4, for unauthenticated and authenticated modes, respectively.

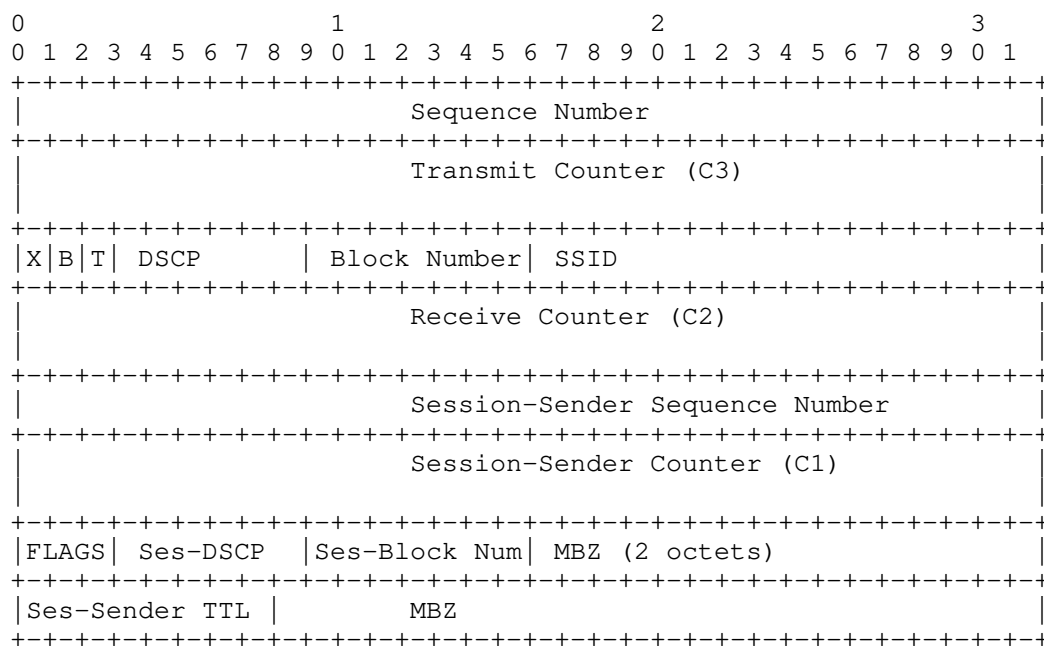


Figure 3: Session-Reflector Direct Loss Measurement Probe Packet - Unauthenticated Mode

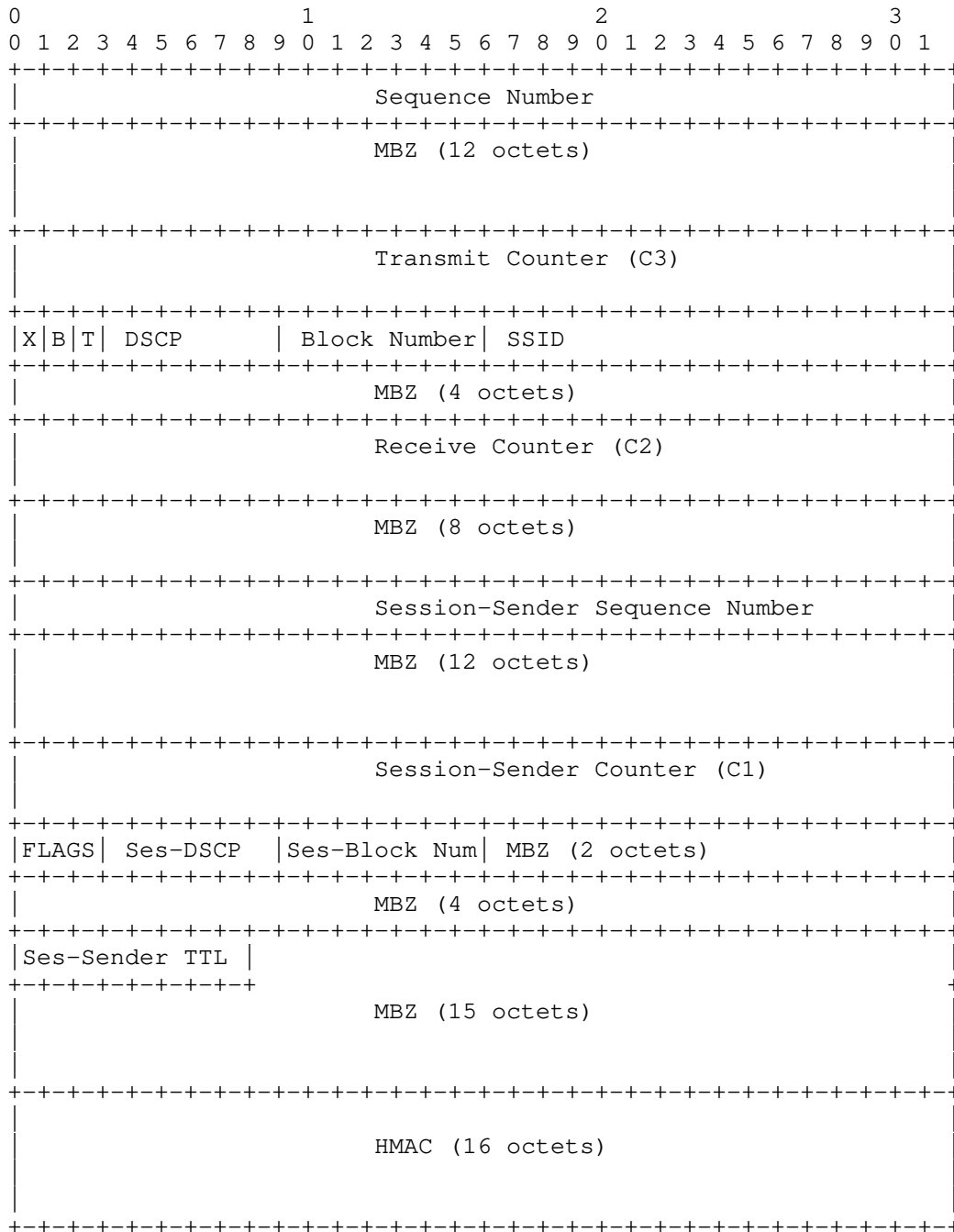


Figure 4: Session-Reflector Direct Loss Measurement Probe Packet -
Authenticated Mode

Fields are defined as the following:

Sequence Number (32-bit): This is the exact copy of the Sequence Number from the received Session-Sender DLM probe packet that allows Stateless mode of Session-Reflector.

Transmit Counter (64-bit): The number of packets or octets transmitted by the Session-Reflector in the DLM reply probe packet. Counter is for the reverse direction data traffic flow under measurement. The Session-Reflector writes the Transmit Counter at the same location in the DLM reply probe packet as the Session-Sender DLM probe packet. This is an important property for hardware-based implementation.

XBT Flags (3-bit): The XBT Flags for the reverse direction data traffic flow under measurement set using the same procedure defined for the Session-Sender DLM probe packet.

DSCP (6-bit): Set for the reverse direction data traffic flow under measurement using the same procedure defined for the Session-Sender DLM probe packet.

Block Number (7-bit): Set for the reverse direction data traffic flow under measurement using the same procedure defined for the Session-Sender DLM probe packet.

SSID: SSID is the exact copy of the SSID in the received Session-Sender DLM probe packet.

Receive Counter (64-bit): The number of packets or octets received at the Session-Reflector. It is written by the Session-Reflector in the DLM reply probe packet. Counter is for the data traffic flow under measurement.

Session-Sender Counter (64-bit): This is the exact copy of the Transmit Counter from the received Session-Sender DLM probe packet.

Session-Sender Sequence Number (32-bit): This is the exact copy of the Sequence Number from the received Session-Sender DLM probe packet.

Session-Sender Block Number: This is the exact copy of the Block Number from the received Session-Sender DLM probe packet.

Session-Sender FLAGS: This is the exact copy of the XBT Flags from the received Session-Sender DLM probe packet.

Session-Sender DSCP: This is the exact copy of the DSCP from the received Session-Sender DLM probe packet.

Session-Sender TTL: The Session-Sender TTL field is one octet long, and its value is the copy of the TTL field in IPv4 (or Hop Limit in IPv6) from the received Session-Sender DLM probe packet.

6. Data Loss Calculation

Using the Counters C1, C2, C3 and C4 as per reference topology, from the nth and (n-1)th DLM probe packets, packet loss and byte loss for the data traffic flow can be calculated as follows:

$$\text{Transmit Loss TxL}[n-1, n] = (C1[n] - C1[n-1]) - (C2[n] - C2[n-1])$$
$$\text{Receive Loss RxL}[n-1, n] = (C3[n] - C3[n-1]) - (C4[n] - C4[n-1])$$

The Total Transmit and Receive Loss are calculated as follows:

$$\text{Total Transmit Loss} = \text{TxL}[1, 2] + \text{TxL}[2, 3] + \dots$$
$$\text{Total Receive Loss} = \text{RxL}[1, 2] + \text{RxL}[2, 3] + \dots$$

These values are updated each time a DLM reply probe packet is received and processed at the Session-Sender, and they represent the Total Transmit and Total Receive Loss since the DLM session was initiated. When computing the values TxL[n-1,n] and RxL[n-1,n], the possibility of counter wrap must be taken into account.

When using Alternate-Marking Method, all Counters used for loss calculation belongs to the same Block Number, as described in Section 3.1 of [RFC8321].

7. Optional Extensions

There are currently no optional (TLV) extensions defined for the DLM probe packets.

8. Integrity Protection and Confidentiality Protection

The integrity protection and confidentiality protection specified in [RFC8762] also apply to the procedures defined in this document.

9. Operational Considerations

The operational considerations specified in [RFC8762] also apply to the procedures defined in this document.

10. Security Considerations

The DLM protocol is intended for deployment in limited domains [RFC8799]. As such, it assumes that a node involved in DLM protocol operation has previously verified the integrity of the path and the identity of the far-end Session-Reflector.

If desired, attacks can be mitigated by performing basic validation and sanity checks, at the Session-Sender, of the counter fields in received reply probe packets. The minimal state associated with these protocols also limits the extent of measurement disruption that can be caused by a corrupt or invalid probe packet to a single probe cycle.

The security considerations specified in [RFC8762] and [RFC8972] also apply to the protocol defined in this document. Specifically, the message integrity protection using HMAC, as defined in [RFC8762] Section 4.4, also apply to the procedure described in this document.

11. IANA Considerations

This document has no IANA actions.

12. References

12.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8321] Fioccola, G., Ed., Capello, A., Cociglio, M., Castaldelli, L., Chen, M., Zheng, L., Mirsky, G., and T. Mizrahi, "Alternate-Marking Method for Passive and Hybrid Performance Monitoring", RFC 8321, DOI 10.17487/RFC8321, January 2018, <<https://www.rfc-editor.org/info/rfc8321>>.

- [RFC8762] Mirsky, G., Jun, G., Nydell, H., and R. Foote, "Simple Two-Way Active Measurement Protocol", RFC 8762, DOI 10.17487/RFC8762, March 2020, <<https://www.rfc-editor.org/info/rfc8762>>.

12.2. Informative References

- [RFC7680] Almes, G., Kalidindi, S., Zekauskas, M., and A. Morton, Ed., "A One-Way Loss Metric for IP Performance Metrics (IPPM)", STD 82, RFC 7680, DOI 10.17487/RFC7680, January 2016, <<https://www.rfc-editor.org/info/rfc7680>>.
- [RFC8799] Carpenter, B. and B. Liu, "Limited Domains and Internet Protocols", RFC 8799, DOI 10.17487/RFC8799, July 2020, <<https://www.rfc-editor.org/info/rfc8799>>.
- [RFC8972] Mirsky, G., Min, X., Nydell, H., Foote, R., Masputra, A., and E. Ruffini, "Simple Two-Way Active Measurement Protocol Optional Extensions", RFC 8972, DOI 10.17487/RFC8972, January 2021, <<https://www.rfc-editor.org/info/rfc8972>>.
- [ITU-Y1731] Recommendation ITU-TG.8013/Y.1731: <https://www.itu.int/rec/T-REC-G.8013-201508-I/en>, "G.8013/Y.1731 : Operations, administration and maintenance (OAM) functions and mechanisms for Ethernet-based networks", August 2015.
- [SRV6-PM-TNSM] Loreti, P., Mayer, A., Lungaroni, P., Lombardo, F., Scarpitta, C., Sidoretta, G., Bracciale, L., Ferrari, M., Salsano, S., Abdelsalam, A., Gandhi, R., and C. Filsfils, IEEE Transactions on Network and Service Management, "SRv6-PM: Performance Monitoring of SRv6 Networks with a Cloud-Native Architecture: <https://arxiv.org/pdf/2007.08633.pdf>", February 2021.
- [SRV6-PM-IEEE] Loreti, P., Mayer, A., Lungaroni, P., Salsano, S., Gandhi, R., and C. Filsfils, IEEE International Conference on High Performance Switching and Routing, "Implementation of Accurate Per-Flow Packet Loss Monitoring in Segment Routing over IPv6 Networks: <https://arxiv.org/pdf/2004.11414.pdf>", May 2020.

Acknowledgments

The authors would like to thank Greg Mirsky, Tianran Zhou, Gyan Mishra, Zhenqiang Li, Reshad Rahman, Cheng Li, and Yali Wang for the comments on Direct Loss Measurement. The authors would like to thank Pierpaolo Loreti and the team for the Open Source implementation of SRv6-PM Loss Monitoring and its publications in [SRV6-PM-TNSM] and [SRV6-PM-IEEE]. The authors would like to acknowledge the earlier work on the loss measurement using TWAMP described in draft-xiao-ippm-twamp-ext-direct-loss.

Authors' Addresses

Rakesh Gandhi (editor)
Cisco Systems, Inc.
Canada

Email: rgandhi@cisco.com

Clarence Filsfils
Cisco Systems, Inc.

Email: cfilsfil@cisco.com

Daniel Voyer
Bell Canada

Email: daniel.voyer@bell.ca

Mach(Guoyi) Chen
Huawei

Email: mach.chen@huawei.com

Bart Janssens
Colt

Email: Bart.Janssens@colt.net

Stefano Salsano
Universita di Roma "Tor Vergata"
Italy

Internet-Draft Simple Direct Loss Measurement Procedure February 2022

Email: stefano.salsano@uniroma2.it

IPPM Working Group
Internet-Draft
Intended status: Informational
Expires: 28 April 2022

L. Han
M. Wang
China Mobile
F. Yang
J. Huang
Huawei Technologies
25 October 2021

Problem Statement and Requirement for Inband Flow Learning
draft-hwyh-ippm-ps-inband-flow-learning-01

Abstract

Alternate-Marking (coloring) provides a method to perform packet loss, delay, and jitter measurements on live traffic. At the same time, on-path telemetry techniques are used to enable the collection and correlation of performance information to further support autonomous network operations. However, network operators still face the challenge of inband flow identification in large scale deployment. This document addresses the problems by introducing the real network scenarios, and proposes the requirements of supporting inband flow learning of flow information telemetry.

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 28 April 2022.

Copyright Notice

Copyright (c) 2021 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. Terminology	3
3. Problem Statement	3
3.1. Frequent and Dynamic Change of Flows	3
3.1.1. Tidal Effect	4
3.1.2. UPF Expansion	4
3.2. Enterprise Service Demand	4
3.3. Large Scale Network Monitor Deployment and Maintenance	4
3.4. Service Flow Path Change	5
4. Requirement	5
4.1. Ingress Flow Learning	5
4.2. Egress Flow Learning	5
4.3. Hop-by-Hop Flow Learning	6
4.4. Auto Flow Aging	6
4.5. Flow Learning Policy	6
5. IANA Considerations	6
6. Security Considerations	6
7. References	6
7.1. Normative References	6
7.2. Informative References	7
Authors' Addresses	7

1. Introduction

Alternate-Marking (coloring) [RFC8321] provides a method to perform packet loss, delay, and jitter measurements on live traffic. [I-D.ietf-mpls-inband-pm-encapsulation] and [I-D.ietf-6man-ipv6-alt-mark] introduce the MPLS and IPv6 performance measurement applications of alternate marking method respectively, and specifies the encapsulations for MPLS and IPv6. On-path telemetry techniques are used to enable the collection and correlation of performance information from the network. By coloring

the real service flow and telemetry flow statistics, per-flow SLA compliance monitoring becomes available and scalable for network operators. When deployed in network, per-flow monitoring can be applied based on CLI configuration or via Netconf YANG.

However, even though Netconf YANG can provide feasibility to network administration, the characteristic of a flow (e.g. IP 5-tuple) can vary dynamically and mislead the service flow identification. Inband flow learning becomes a challenge in large scale deployment to network operators. This document addresses the problems by introducing the real network scenarios, and proposes the requirements of supporting inband flow learning of flow information telemetry.

2. Terminology

OAM: Operations, Administration, and Maintenance

SLA: Service Level Agreement

NFV: Network Function Virtualization

UNI: User-Network-Interface

CN: Core Network

3. Problem Statement

In an alternate marking application, it is usually to utilize the characteristic fields of packet to identify a service flow. For example, IP 5-tuple is usually to be used as the identifier of a service flow at source node. A concept of flow identifier, such as Flow-ID Label Indicator [I-D.ietf-mpls-inband-pm-encapsulation] or FlowMonID [I-D.ietf-6man-ipv6-alt-mark] is used to identify service flow at transit or egress nodes. The change of packet data fields would mislead the flow identification for flow monitoring and statistics telemetry in large scale.

3.1. Frequent and Dynamic Change of Flows

In 4G/5G mobile backhaul networks, IP address of one service can be changed based on location, time or even with business growth. The following scenarios describes the challenges which 4G/5G mobile service encounters.

3.1.1. Tidal Effect

A Tidal Effect phenomenon has been recognized as traffics between base station and Core Network (CN) show repetitive patterns with spatio-temporal variations. A typical example of Tidal phenomenon is the traffic difference happened in day and night time of a commercial and business area. In day time, eNodeB allocates more core network resources when a large number of user equipment accesses eNodeB, and less resources at night accordingly. The change of the number of UEs and the core network resources may affect the change on source and destination IP address of service flows.

Moreover, NFV used in core network makes the traffic change even worse as the IP address at CN cannot be manually configured or even predicted. In this case, it is impossible for operators to statically deploy flow monitoring and statistics telemetry.

3.1.2. UPF Expansion

In 5G deployment, the increase of number of subscribers triggers the expansion of UPF resources on data plane of 5G core network. After new UPF resource is added, eNodeB sets up a connection to the new UPF. Correspondingly, a new IP flow is created in mobile bearer network. In this scenario, if flow monitoring and statistics telemetry is deployed in a static mode, operators would need to manually add related configurations to mobile bearer network after the core network capacity is expanded, which is very difficult to deploy in practice.

3.2. Enterprise Service Demand

The enterprise services usually connect different private networks between Headquarter and Branches, Branches and Branches. Network operator has very limited or even no information about end users. Besides, information from one site could be changed from time to time. Unpredictable information on enterprise customer side makes impossible for network operators to set up real time flow monitoring, and to avoid the omission of flow monitoring.

3.3. Large Scale Network Monitor Deployment and Maintenance

In a large-scale mobile bearer network, a large number of base stations and corresponding access points may lead to a large number of IP addresses in core network. From network maintenance perspective, when flow monitoring and statistics telemetry is deployed in a static mode, network operator had to manually set up each monitoring instance between base station and core network, then separately delegate configurations to a large number of network

entities. It is difficult for network operators to find an effective way of monitoring creation and maintenance.

Note that traffic monitoring is comprised of uplink and downlink directions, which makes twice of workload on configurations.

3.4. Service Flow Path Change

When a hop-by-hop flow monitoring is required by critical traffic for deep SLA investigation, the actual forwarding path of service flow and the every forwarding nodes along the path are obtained. Network operator delegates different configurations to each node including ingress, transit, and egress nodes on the path.

Once the traffic forwarding path is changed because of service flow switching or route convergence, the monitoring instance on each node needs to be re-deployed on the new path. In this situation, a flexible and efficient deployment approach is required by network operators.

4. Requirement

To face the flow deployment challenges mentioned in preceding section, an approach of inband flow learning is required. It should simplify the deployment of flow monitoring and achieve an automatic mode of telemetry in large scale networks.

4.1. Ingress Flow Learning

On the UNI side of network node, ingress flow learning can help to capture the characteristic data fields of packet and create the monitoring instance when the flow is created from base station. Flexible policy based on access control list (ACL) can facilitate the identification of flow characteristic. For example, IP 2-tuple (DIP+SIP), DSCP value, etc.

4.2. Egress Flow Learning

Similar to the requirement on ingress node, traffic egress node should support the same capability of inband flow learning to create traffic monitoring instance for completing a monitor. When the egress node or egress port of a service flow is changed, the egress node or egress port of service flow can be triggered to re-learn and re-monitor the service flow.

4.3. Hop-by-Hop Flow Learning

When hop-by-hop flow monitoring and telemetry is required, the flow learning and monitor deployment should be created on all the ingress, transit, and egress nodes that service flows pass through. When the path of a service flow changes due to the service switching or network convergence, the service flow re-triggers the flow learning on the new path and starts the new monitoring of service flow.

4.4. Auto Flow Aging

In all the inband flow learning scenarios described above, when the path of a service flow changes, the flow learning on new path is triggered and new monitoring instances are created on devices. Regarding the monitoring instances that have been created before the path change, if there is no traffic detected within a certain period of time, automatic aging and resource recycle should be supported.

4.5. Flow Learning Policy

It is valuable to specify the flow learning policy on equipment when thousands or millions of flows are transmitted. Flow learning policy specifies the metrics and explicit rules executed on equipment, for example the flow is filtered based on a particular range of protocol number. Centralized controller specifies the flow learning policy via management and control plane to equipment, then data plane executes the policies to generate monitoring instance.

5. IANA Considerations

This document has no request to IANA

6. Security Considerations

TBD

7. References

7.1. Normative References

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

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

7.2. Informative References

- [I-D.ietf-6man-ipv6-alt-mark]
Fioccola, G., Zhou, T., Cociglio, M., Qin, F., and R. Pang, "IPv6 Application of the Alternate Marking Method", Work in Progress, Internet-Draft, draft-ietf-6man-ipv6-alt-mark-12, 22 October 2021, <<https://www.ietf.org/archive/id/draft-ietf-6man-ipv6-alt-mark-12.txt>>.
- [I-D.ietf-mpls-inband-pm-encapsulation]
Cheng, W., Min, X., Zhou, T., Dong, X., and Y. Peleg, "Encapsulation For MPLS Performance Measurement with Alternate Marking Method", Work in Progress, Internet-Draft, draft-ietf-mpls-inband-pm-encapsulation-02, 25 October 2021, <<https://www.ietf.org/archive/id/draft-ietf-mpls-inband-pm-encapsulation-02.txt>>.
- [RFC8321] Fioccola, G., Ed., Capello, A., Cociglio, M., Castaldelli, L., Chen, M., Zheng, L., Mirsky, G., and T. Mizrahi, "Alternate-Marking Method for Passive and Hybrid Performance Monitoring", RFC 8321, DOI 10.17487/RFC8321, January 2018, <<https://www.rfc-editor.org/info/rfc8321>>.

Authors' Addresses

Liuyan Han
China Mobile
Beijing
China

Email: hanliuyan@chinamobile.com

Minxue Wang
China Mobile
Beijing
China

Email: wangminxue@chinamobile.com

Fan Yang
Huawei Technologies
Beijing
China

Email: shirley.yangfan@huawei.com

Jinming Huang
Huawei Technologies
Dongguan
China

Email: zhangshengli4@huawei.com

ippm
Internet-Draft
Intended status: Experimental
Expires: 7 November 2022

R. Geib, Ed.
Deutsche Telekom
6 May 2022

A Connectivity Monitoring Metric for IPPM
draft-ietf-ippm-connectivity-monitoring-04

Abstract

Within a Segment Routing domain, segment routed measurement packets can be sent along pre-determined paths. This enables new kinds of measurements. Connectivity monitoring allows to supervise the state and performance of a connection or a (sub)path from one or a few central monitoring systems. This document specifies a suitable type-P connectivity monitoring metric.

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 7 November 2022.

Copyright Notice

Copyright (c) 2022 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 Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction	3
1.1. Requirements Language	5
2. A brief segment routing connectivity monitoring framework . .	5
3. Topology and measurement loop set up requirements	11
3.1. General network topology requirements	11
3.2. Sub-path Monitoring measurement loop routing requirements	11
3.3. Path	13
3.4. Sub-path Monitoring measurement loop packet spacing . . .	13
4. Generic Type-P-SR-Path-Periodic-* metric	13
4.1. Metric Name	14
4.2. Generic Metric Parameters	14
4.3. Metric Units	14
5. Singleton Definition for Type-P-SR-Path-Periodic-Delay . . .	14
5.1. Metric Name	14
5.2. Metric Parameters	14
5.3. Delay Metric Units	14
5.4. Definition	15
5.5. Discussion	15
5.6. Methodologies	15
5.7. Errors and Uncertainties	15
5.8. Reporting the metric	15
6. Singleton Definition for Type-P-SR-Path-Packet-Loss	15
6.1. Metric Name	15
6.2. Metric Parameters	15
6.3. Packet Loss Metric Units	16
6.4. Definition	16
6.5. Discussion	16
6.6. Methodologies	16
6.7. Errors and Uncertainties	16
6.8. Reporting the metric	16
7. Definition of Samples for Type-P-SR-Path-Periodic-Delay . . .	16
7.1. Generic Type-P-SR-Path-Periodic-Delay-* metric	16
7.1.1. Metric Name	17
7.1.2. Metric Parameters	17
7.1.3. Metric Units	17
7.1.4. Metric Defintion	17
7.1.5. Discussion	17
7.1.6. Errors and uncertainties	17
7.2. Definition of Type-P-SR-Path-Periodic-Delay-Stream . . .	17
7.2.1. Metric Name	17
7.3. Definition of Type-P-SR-Path-Periodic-Delay-Variation . .	18
7.3.1. Metric Name	18
7.3.2. Methodologies	18
7.3.3. Discussion of SRDV	18
7.3.4. Errors and uncertainties	18

7.4.	Definition of	
	Type-P-SR-Path-Periodic-Delay-Variation-Stream	18
7.4.1.	Metric Name	18
7.4.2.	Metric Defintion	18
8.	Statistic Definitions for SR-Path-Periodic-*-Stream	
	samples	19
8.1.	SR-Path-Periodic-*-Mean	19
8.2.	SR-Path-Periodic-*-Std	19
9.	Statistic Definitions for Type-P-SR-Path-Packet-Loss	19
9.1.	SR-Path-Packet-Loss-Ratio	19
10.	Sub-Path monitoring metrics derived from samples captured along	
	the measurement loops	20
10.1.	Baseline measurement	20
10.2.	Discussion of the baseline measurement	21
10.3.	Definition of SR-Path-Sub-Path-RTD-Estimate	22
10.4.	Definition of SR-Path-Sub-Path-*-Changepoint	22
10.5.	Discussion of SR-Path-Sub-Path-*-Changepoint	23
10.6.	Definition of SR-Path-Sub-Path-Congestion-Location	24
10.7.	Definition of SR-Path-Sub-Path-Disconnected	25
11.	Discussion of Temporal Resolution	27
12.	IANA Considerations	27
13.	Security Considerations	27
14.	References	27
14.1.	Normative References	27
14.2.	Informative References	29
	Author's Address	29

1. Introduction

Within a Segment Routing domain, measurement packets can be sent along pre-determined segment routed paths [RFC8402]. A segment routed path may consist of pre-determined sub paths, specific router-interfaces or a combination of both. A measurement path may also consist of sub paths spanning multiple routers, given that all segments to address a desired path are available and known at the SR domain edge interface.

A Path Monitoring System (PMS, see [RFC8403]) is a dedicated central Segment Routing (SR) domain monitoring device (as compared to a distributed monitoring approach based on router-data and -functions only). Monitoring individual sub-paths or point-to-point connections is executed for different purposes. IGP exchanges hello messages between neighbors to keep alive routing and swiftly adapt routing to topology changes. Network Operators may be interested in monitoring connectivity and congestion of interfaces or sub-paths at a timescale of seconds, minutes or hours. In both cases, the periodicity is significantly smaller than commodity interface monitoring based on router counters, which may be collected on a minute timescale to keep the processing- or monitoring data-load low.

The IPPM architecture was a first step to that direction [RFC2330]. Commodity IPPM solutions require dedicated measurement systems, a large number of measurement agents and synchronised clocks. Monitoring a domain from edge to edge by commodity IPPM solutions increases scalability of the monitoring system. But localising the site of a detected network behaviour change may then require suitable network tomography methods.

The IPPM Metrics for Measuring Connectivity offer generic connectivity metrics [RFC2678]. These metrics allow to measure connectivity between end nodes without making any assumption on the paths between them. The metric and the type-p packet specified by this document follow a different approach: they are designed to monitor connectivity and performance of a specific single link or a path segment. The underlying definition of connectivity is partially the same: a packet not reaching a destination indicates a loss of connectivity. An IGP re-route may indicate a loss of a link, while it might not cause loss of connectivity between end systems. The metric specified here detects a loss of connectivity, defined by a complete absence of a path between two nodes in both directions of communication (whereas a re-routing will briefly disturb a path, but connectivity is restored by the network after a short disturbance).

A Segment Routing PMS is part of an SR domain. The PMS is IGP topology aware, covering the IP and (if present) the MPLS layer topology [RFC8402]. This allows to steer PMS measurement packets along arbitrary pre-determined concatenated sub-paths, identified by suitable Segment IDs. Basically, the SR connectivity metric as specified by this document requires set up of a number of constrained, overlaid measurement loops (or measurement paths). The delay of the packets sent along each of these measurement loops is measured. A single congested interface along a monitored sub-path adds latency along a unique subset of several measurement loops. If a monitored sub-path no longer provides IP/MPLS connectivity between two nodes, another unique subset of measurement loops will drop all

traffic while connectivity is lost. The number of measurement loops required in total may be limited to one per sub-path (or connection) to be monitored, if a hub-and-spoke like sub-path topology as described below is monitored. In addition to information revealed by a commodity ICMP ping measurement, the metrics and methods specified here identify the location of a congested interface (or ingress of a congested sub-path, respectively). To do so, tomography assumptions and methods are combined to first plan the overlaid SR measurement loop set up and later on to evaluate the captured performance metrics.

There's another difference as compared with commodity ping: the measurement loop packets remain in the data plane of passed routers. These need to forward the measurement packets without any additional processing apart from that.

It is recommended to consider automated measurement loop set-up. The methods proposed here are error-prone if the topology and measurement loop design isn't followed properly. While details of an automated set-up are not within scope of this document, some formal definitions of constraints to be respected are given.

This document specifies type-p metrics determining properties of an SR path which allows to monitor connectivity and congestion of interfaces. The specified methods further allow to locate the path or interface which caused a change in the reported type-p metrics. This document is limited to the Segment Routing MPLS layer, but the methodology may be applied within SR domains or MPLS domains in general.

1.1. Requirements Language

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

2. A brief segment routing connectivity monitoring framework

The Segment Routing IGP topology information consists of the IP and (if present) the MPLS layer topology. The minimum SR topology information consists of Node-Segment-Identifiers (Node-SID), identifying an SR router. The IGP exchange of Adjacency-SIDs [RFC8667], which identify local interfaces to adjacent nodes, is optional. It is RECOMMENDED to distribute Adj-SIDs in a domain operating a PMS to monitor connectivity as specified below. If Adj-SIDs aren't available, [RFC8287] provides methods how to steer packets along desired paths by the proper choice of an MPLS Echo-request IP-destination address. A detailed description of [RFC8287]

methods as a replacement of Adj-SIDs is out of scope of this document. Monitoring interfaces connecting nodes requires Adj-SIDs, if re-converged IP/MPLS layer connectivity would result in re-routing packets (and re-establishment of IP/MPLS layer connectivity) using Node-SIDs.

An active round trip measurement between two adjacent nodes is a simple method to monitor connectivity of a connecting link. If multiple links are operational between two adjacent nodes and only a single one fails, a single plain round trip measurement may fail to notice that or identify which link has failed. A round trip measurement further fails to identify which interface is congested, even if only a single link connects two adjacent nodes.

Segment Routing enables the set-up of extended measurement loops. Several different measurement loops can be set up to form a partial overlay. If done properly, any network change impacts more than a single measurement loop's round trip delay or causes drops of packets of more than one loop. Randomly chosen measurement loop paths including the interfaces or paths to be monitored may fail to produce the desired unique result patterns, hence commodity network tomography methods aren't applicable [CommodityTomography]. The approach pursued here uses a pre-specified measurement loop overlay design to produce the desired results with a minimum effort.

A centralised monitoring approach doesn't require report collection and result correlation from two (or more) receivers. The metrics captured along different measurement loops however still need to be correlated.

An additional property of the measurement loop set-up specified below is that it allows to estimate the packet round trip delay of a monitored link or sub-path.

An example hub and spoke network, operated as SR domain, is shown below. The included PMS shown is supposed to monitor the connectivity of all the 6 links (a link is a simple and generic kind of sub-path) attaching the spoke-nodes L050, L060 and L070 to the hub-nodes L100 and L200. L300 only serves to connect the PMS to nodes L100 and L200.

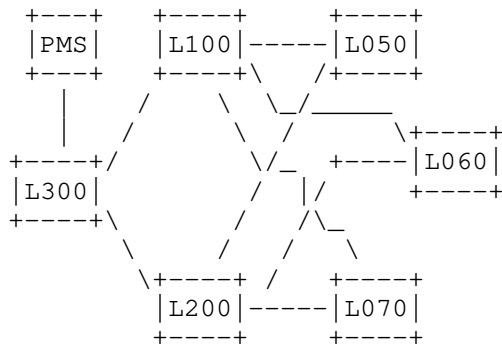


Figure 1

Example hub and spoke network allowing link connectivity verification with a PMS

The SID values are picked for convenient reading only. Node-SID: 100 identifies L100, Node-SID: 300 identifies L300 and so on. Adj-SID 10050: Adjacency L100 to L050, Adj-SID 10060: Adjacency L100 to L060, Adj-SID 60200: Adjacency L60 to L200 and so on (note that the Adj-SID are locally assigned per node interface, meaning two per link).

Monitoring the 6 links between hub nodes Ln00 (where n=1,2) and spoke nodes L0m0 (where m=5,6,7) requires 6 measurement loops, which have the following properties:

- * Each measurement loop follows a single round trip from one hub Ln00 to one spoke L0m0 (e.g., from L100 and L050 and back to L100).
- * Each measurement loop passes two more links: one between the same hub Ln00 and another spoke L0m0 and from there to the alternate hub Ln00 (e.g., from L100 to L060 and then from L060 to L200)
- * Every monitored link is passed by a single round trip measurement loop only once and further only once unidirectional by two other loops. These latter, unidirectional measurement loop sections forward packets in opposing direction along the monitored link. In the end, three measurement loops pass each single monitored link (sub-path). In figure 1, e.g. the link between L100 and L050 is passed by one measurement loop following a round trip L100 to L050 (the measured delay is M1, see below), a second loop passes in direction L100 to L050 only (delay M3) and a third loop passes in direction L050 to L100 only (delay M6).

Note that any 6 links connecting two to five nodes can be monitored that way too. Further note that the measurement loop overlay chosen is optimised for 6 links and a hub and spoke topology of two to five nodes. The 'one measurement loop per measured sub-path' paradigm only works under these conditions.

The above overlay scheme results in 6 measurement loops for the given example. The start and end of each measurement loop is PMS to L300 to L100 or L200 and a similar sub-path on the return leg. These parts of the measurement loops are omitted here for brevity (some discussion may be found below). The following delays are measured along the SR paths of each measurement loop:

1. M1 is the delay along L100 -> L050 -> L100 -> L060 -> L200
2. M2 is the delay along L100 -> L060 -> L100 -> L070 -> L200
3. M3 is the delay along L100 -> L070 -> L100 -> L050 -> L200
4. M4 is the delay along L200 -> L050 -> L200 -> L060 -> L100
5. M5 is the delay along L200 -> L060 -> L200 -> L070 -> L100
6. M6 is the delay along L200 -> L070 -> L200 -> L050 -> L100

For brevity, in the following delay M1 also identifies the corresponding measurement loop number 1 and so on.

An example for a stack of Adj-SID segments the loop resulting in M1 is (top to bottom): 100 | 10050 | 50100 | 10060 | 60200 | PMS. As can be seen, the Node-SIDs 100 and PMS are present at top and bottom of the segment stack. Their purpose is to transport the packet from the PMS to the start of the measurement loop at L100 and return it to the PMS from its end. When connectivity is lost, a path determined by Adj-SIDs behaves deterministic: packets forwarded to an Adj-SID without connectivity to the neighboring node are dropped.

An example for a stack of a loop consisting of Node-SID segments allowing to capture M1 is (top to bottom): 100 | 050 | 100 | 060 | 200 | PMS.

The evaluation of the measurement loop round trip delays M1 - M6 allows to detect the following state-changes of the monitored sub-paths:

- * If the loops are set up using Node-SIDs only, any single complete loss of connectivity caused by a failing single link between any Ln00 and any L0m0 node briefly disturbs three measurement loops

and changes the delay measured along them. The traffic to the Node-SIDs is re-routed (in the case of a single link loss, no node is completely disconnected in the example network). In that case, a suitable metric characterising re-routing coupled with the loss of that single link is required. The change in propagation delay might be an approach for such a metric (if there is any delay change, as that depends on the resulting alternate route delay). A delay based connectivity scheme may not work under all circumstances.

- * If the measurement loops are set up using Adj-SIDs only, a loss of connectivity caused by a failing single link between any Ln00 and any L0m0 node terminates the traffic along three measurement loops. The packets of all three loops will be dropped, until the link gets back into service. Traffic to Adj-SIDs is not rerouted. Note that Node-SIDs may be used to forward the measurement packets from the PMS to the hub node, where the first sub-path to be monitored begins and from the hub node receiving the measurement from the last monitored sub path to the PMS.
- * The simple example indicates superiority of Adj-SIDs over Node-SIDs only if links are monitored and the network architecture is similar to the one shown in the figure. The generic advice is, that unambiguous connectivity monitoring is best based on packet loss, rather than on delay changes.
- * A single congested interface between any Ln00 and any L0m0 node always only impacts the measured delay of two measurement loops.
- * As an example, the formula to calculate the (sub-path) Round Trip Delay (RTD) for link L100-L050 is given here

$$4 * RTD_L100-L050-L100 = 3 * M1 + M3 + M6 - M2 - M4 - M5.$$

This formula is reproducible for all other links: sum up 3*RTD measured along the loop passing the monitored link of interest in round trip fashion, and add the RTDs of the two measurement loops passing the evaluated monitored link only in a single direction. From this sum subtract the RTD captured for the measurement loops not passing the monitored link evaluated to get four times the RTD of the monitored link evaluated.

A closer look reveals that any single event of interest for the proposed metric, which are a single loss of connectivity or a single case of congestion, only impacts a unique set of measurement loops which can be determined a-priori. If, e.g., connectivity is lost between L200 and L050, measurement loops M3, M4 and M6 indicate packet loss (or a change of the measured delay, if a Node-SID based approach is preferred).

As a second example: if the interface L070 to L100 is congested, measurement loops M3 and M5 indicate a change in the measured delay. Without listing all events, it can be shown that all cases of single losses of connectivity or single events of congestion influence only delay measurements of a unique set of measurement loops.

The measurement loops are best set up while there's no congestion. In that case, the congestion free RTDs of all monitored links can be calculated as shown above which later allows to estimate the queue-depth under congestion. A single congestion event adds queuing delay to the RTD measured of two specific measurement loops. The two measurement loops impacted indicate the congested interface and enable estimation of the queue-depth (in terms of seconds based on comparing actual and prior delay measurements). The per link RTD can be calculated while the network is operating without congestion, say at interval T0. Then as an example, assume a queue of an average depth of 20 ms to build up at interface L200 to L070 at interval T1. The measurement loops M5 and M6 are the only ones passing the interface in that direction. Both indicate an added delay along M5 and M6 of + 20 ms during a measurement interval T1 with congestion on this interface, while M1-4 indicate unchanged delays. The location of the congested interface is determined by the combination of the two (and only two) measurement loops M5 and M6 showing a significant delay increase. The average queue depth [s] = (M5[T1] - M5[T0] + M6[T1] - M6[T0])/2.

As mentioned there's a constant delay added for each measurement loop, which is the delay of the path passed from PMS -> L100 + L200 -> PMS. Please note, that this added delay is appearing twice in the formula resulting in the monitored link delay estimate of the example network. Then it is the RTD PMS -> L100 + RTD L200 -> PMS. Both RTDs can be directly measured by two additional measurements Cor1 = RTD (PMS -> L100 -> PMS) and Cor2 = RTD (PMS -> L200 -> PMS). The monitored link RTD formula was $\text{linkRTDuncor} = 3 \cdot M_x + M_y + M_z - M_s - M_t - M_u$. The correct $4 \cdot \text{linkRTDx} = 4 \cdot \text{linkRTDxuncor} - \text{Cor1} - \text{Cor2}$.

If the interface between PMS and L100/L200 is congested, all measurement loops M1-M6 as well as Cor1 and Cor2 will see a change. A congested interface of a monitored link doesn't impact the RTDs captured by Cor1 and Cor2.

The measurement loops may also be set up between hub nodes L100 and L200, if that's preferred and supported by the nodes. In that case, the above formulas apply without correction.

3. Topology and measurement loop set up requirements

3.1. General network topology requirements

The metric and methods specified below can be applied to monitor networks or sub-paths forming a hub and spoke topology. A single sub-path status change of type loss of connectivity or congestion can be detected. The nodes don't have to act as hubs or spokes, this terminology is only chosen to describe a topology requirement. In detail, the topology to be monitored MUST meet the following constraints:

- * The SR domain sub-paths to be monitored create a hub and spoke topology with a PMS connected to all hub nodes. The PMS may reside in a hub.
- * Exactly 6 (six) sub-paths are monitored.
- * The monitored sub-paths connect at least two and no more than 5 nodes.
- * Every spoke node MUST have at least one path to every hub node.
- * Every spoke node MUST at least be connected to one (or more) hub node(s) by two monitored sub-paths.
- * Sub-paths between spokes can't be monitored and therefore are out of scope (the overlay measurement loops can't be set up as desired).

Shared resources, like a Shared Risk Link Group (e.g., a single fiber bundle) or a shared queue passed by several logical links need to be considered during set up. Shared resources may either be desired or to be avoided. As an example, if a set of logical links share one parental scheduler queue, it is sufficient to monitor a single logical connection to monitor the state of that parental scheduler.

3.2. Sub-path Monitoring measurement loop routing requirements

The methodologies specified by this document REQUIRE a measurement loop path overlay of all path delay measurement streams F_i , i in $[1, 2...6]$ as defined in this section. In the following, a path delay measurement stream F_i is called measurement (loop) F_i for brevity.

- * Define the segment routed Sub-paths SP_i , i in $[1, 2...6]$ to be monitored. The Sub-paths SP_i SHOULD not share resources, if the operator isn't aware of the impact of the shared resources on the measurement loops Fi and the methodologies defined below. The Sub-path SP_i topology SHOULD respect the general network topology requirements as specified above.
- * Set up $i = 1, 2...6$ measurement loops Fi thus that measurement Fi passes SP_i and only SP_i bidirectional (or by a round-trip) from Hub to Spoke and back. Note that the correspondance of SP_i and Fi isn't strictly required. Measurement Fi thus however appears in all methodologies calculating a metric related to SP_i .
- * Set up the SR path per measurement loops Fj and Fk thus that SP_i is passed by exactly one other measurement loop Fj unidirectional in direction Hub to Spoke and by exactly one other measurement loop Fk unidirectional in the opposite direction (Spoke to Hub). The measurement loop $Fi \neq Fj \neq Fk$. As a description, one measurement loop Fj pass SP_i in "downstream" direction from Hub to Spoke, whereas measurement loop Fk passes SP_i in "upstream" direction from Spoke to Hub.
- * Set up each segment routed measurement loop path Fi thus that it passes SP_i bidirectional as specified above, SP_j unidirectional from Hub to Spoke and SP_k unidirectional from Spoke to Hub. The monitored Sub-path SP_i MUST NOT be equal to SP_j and MUST NOT be equal to SP_k .
- * The measurement loop set up to monitor all Sub-paths SP_i is completed, if:
 - + Each Sub-path SP_i is passed by exactly three measurements loops Fi , Fj and Fk as specified above.
 - + Each segment routed measurement loop path Fi passes exactly three concatenated Sub-paths SP_i , SP_m and SP_n as specified above (indices m and n are chosen here only to avoid misconceptions which may result from picking indices j and k already appearing before - equality of j and k with either m and n is neither excluded nor required).

3.3. Path

This document specifies sub-path monitoring within a closed domain by a controlled and pre-designed measurement loop set-up. The path traversed by the packet SHOULD be reported, as detecting data plane forwarding in line with the desired measurement loop set-up is essential for the metric to enable and verify accurate evaluation. See [RFC8287] for SR MPLS OAM and [ID.draft-ietf-6man-spring-srv6-oam] for SRv6 OAM.

3.4. Sub-path Monitoring measurement loop packet spacing

Packets per measurement loop F_i are sent periodically by a temporal distance of $IncT$. For convenience, packets of the 6 measurement loops are assumed to be equally spaced at the sender too. Let's define the temporal distance $IncF$ between two consecutive packets sent along to different measurement loops F_i and F_j at a single sender to be

$$IncF = IncT / 6$$

Further it seems useful to suggest $IncF$ to be bigger than the largest measurement loop delay $\max(mi)$ under stable network operation (i.e., including some tolerance). Further assume the standard deviation of the measurement values mi to be much smaller than the delay mi , which is likely for a sub path being a regional or national link in many countries. Note that this definition isn't a strict requirement. Interpretation of results is however simplified by it. For the rest of the document assume

$$IncF > 2 * \max(mi), i \text{ in } [1...6], \text{ which results in}$$

$$IncT > 12 * \max(mi)$$

Discussion and reasoning for a reasonable smallest interval $IncF$ in relation to $\max(mi)$ follows below.

4. Generic Type-P-SR-Path-Periodic-* metric

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

4.1. Metric Name

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

4.2. Generic Metric Parameters

Refer to section 3.2. Metric Parameters: Type-P-* of [RFC6673]. The following parameters are added, enhanced or removed:

Dst SHOULD be a diagnostic IP address as specified by [RFC8287] and [RFC8029], if MPLS OAM is operated to capture the metric.

Fi, where i in [1, 2...6], a selection function defining unambiguously a packet of one particular stream i forming part of the monitoring overlay measurement loop set up.

L, a packet length in bits. The packets of all Type-P-SR-Path-Delay-Periodic-Streams Fi SHOULD all be of the same length.

MLAi, a stack of Segment IDs determining a monitoring loop Fi. The Segment-IDs MUST be chosen so that a singleton type-p packet of selection function Fi passes the sub-path i to be monitored.

No support: lambda (Poisson Streams remain ffs.)

4.3. Metric Units

Refer to section 3.4. Metric Units: Type-P-* of [RFC6673].

5. Singleton Definition for Type-P-SR-Path-Periodic-Delay

5.1. Metric Name

Type-P-SR-Path-Periodic-Delay

5.2. Metric Parameters

See section Section 4.2.

5.3. Delay Metric Units

A sequence of consecutive time values. The value of a Type-P-SR-Path-Periodic-Delay is either a real number or an undefined (informally, infinite) number of seconds per singleton of each stream Fi.

5.4. Definition

Section 3.4 of [RFC7679] applies per singleton of each stream Fi. The additional information related to singletons of section 4.2.4 of [RFC3432] applies too.

5.5. Discussion

See section 3.5 of [RFC7679]. One generalisation seems appropriate: a global satellite navigation system affords one way to achieve synchronization within usec.

5.6. Methodologies

Section 3.6 of [RFC7679] applies per stream Fi with one exception: at the Src host, select Src and Dst IP addresses, if IP-routing is applied, or select the proper functional IP-destination address if an [RFC8287] SR MPLS OAM packet format is applied. Further add the appropriate stack of Segment IDs MLAi determining the monitoring loop Fi and form a test packet of Type-P with these addresses and the segment stack.

5.7. Errors and Uncertainties

See section 3.7 of [RFC7679] and section 4.6 of [RFC3432].

5.8. Reporting the metric

See section 3.8 of [RFC7679].

6. Singleton Definition for Type-P-SR-Path-Packet-Loss

Editors note: To be added based on existing loss metrics. A delay based approach indicating loss of a physical interface by detecting delay changes caused by re-routing can't be assumed to reliably cause unique delay change patterns under all circumstances (consider a shortest path routed multi-hop MPLS sub-path to be monitored rather than a link or a scenario where a bundle of 6 equivalent links is monitored connecting a single hub and spoke).

6.1. Metric Name

Type-P-SR-Path-Packet-Loss

6.2. Metric Parameters

See section Section 4.2.

6.3. Packet Loss Metric Units

The value of a Type-P-SR-Path-Packet-Loss is either a zero (signifying successful transmission of the packet) or a one (signifying loss) per singleton of each stream Fi.

6.4. Definition

Section 2.4 of [RFC7680] applies per singleton of each stream Fi.

6.5. Discussion

See section 3.5 of [RFC7680].

6.6. Methodologies

Section 2.6 of [RFC7680] applies per stream Fi with one exception: at the Src host, select Src and Dst IP addresses, if IP-routing is applied, or select the proper functional IP-destination address if an [RFC8287] SR MPLS OAM packet format is applied. Further add the appropriate stack of Segment IDs MLAi determining the monitoring loop Fi and form a test packet of Type-P with these addresses and the segment stack.

6.7. Errors and Uncertainties

See section 2.7 of [RFC7680].

6.8. Reporting the metric

See section 2.8 of [RFC7680].

7. Definition of Samples for Type-P-SR-Path-Periodic-Delay

This sections defines metric samples and metrics derived from samples.

7.1. Generic Type-P-SR-Path-Periodic-Delay-* metric

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

7.1.1. Metric Name

Type-P-SR-Path-Periodic-Delay-*

7.1.2. Metric Parameters

Src, the IP address of a host

Dst, the IP address of a host

MLAi, a stack of Segment IDs

Ti0, a time

Tif, a time

incT, a time

7.1.3. Metric Units

See section Section 5.3.

7.1.4. Metric Defintion

Given Ti0 and Tif and nominal inter-packet interval incT, those time values greater than or equal to Ti0 and less than or equal to Tif are then selected. At each of the selected times in this process, we obtain one value of Type-P-SR-Path-Periodic-Delay. The value of the sample is the sequence made up of the resulting [time, delay] pairs. If there are no such pairs, the sequence is of length zero and the sample is said to be empty.

7.1.5. Discussion

See section 4.4 of [RFC3432].

7.1.6. Errors and uncertainties

See section 4.6 of [RFC3432].

7.2. Definition of Type-P-SR-Path-Periodic-Delay-Stream

The only definition required for this metric is a unique metric name.

7.2.1. Metric Name

Type-P-SR-Path-Periodic-Delay-Stream

7.3. Definition of Type-P-SR-Path-Periodic-Delay-Variation

The smallest sample Type-P-SR-Path-Periodic-Delay-Stream is one of two consecutively received values. These may be used to calculate a Segment Routed Path Delay-Variation (SRDV) singleton, defined below.

7.3.1. Metric Name

Type-P-SR-Path-Periodic-Delay-Variation

7.3.2. Methodologies

SRDV[i,j], for each sample of packets j and j-1 of stream Fi, j > 1, the delay variation between successive packets is calculated as:

$$\text{SRDV}[i,j] = \text{Delay}[i,j] - \text{Delay}[i,j-1],$$

j in [2,3...N] and N the total number of packets received at Dst. If one or more of the M packets sent by Src are lost, they are ignored for the metric, as no reasonable metric value is defined here. If N > 1, the metric is calculated for every valid packet received and the preceding one.

7.3.3. Discussion of SRDV

Evaluation statistics of differential SRDV metric samples may help to identify issues.

7.3.4. Errors and uncertainties

See section 2.7 of [RFC3393].

7.4. Definition of Type-P-SR-Path-Periodic-Delay-Variation-Stream

The only definition required for this metric is a unique metric name.

7.4.1. Metric Name

Type-P-SR-Path-Periodic-Delay-Variation-Stream

7.4.2. Metric Definition

Given Ti0 and Tif, those time values greater than or equal to Ti0 and less than or equal to Tif are then selected. At each of the selected times in this process, we obtain one value of Type-P-SR-Path-Periodic-Delay. The value of the sample is the sequence made up of the resulting [time, delay-variation] pairs with time being set to the Dst timestamp of the Delay-Variation singleton, for which a valid

singleton is calculated. If there are no such pairs, the sequence is of length zero and the sample is said to be empty. If N Delay singletons are captured and sampled N-1 Delay-Variation singletons are sampled during the same interval

8. Statistic Definitions for SR-Path-Periodic-*--Stream samples

Change point detection requires statistical definitions. These are provided below. The names of the statistics contain an "*" placeholder, which may be replaced by "Delay" or "Delay-Variation".

8.1. SR-Path-Periodic-*--Mean

For a type-p metric, the mean is specified by:

$SR\text{-}*\text{Mean} = (1/N) * \text{Sum}(\text{from } a=1 \text{ to } N, \text{ value}[a])$

* N sample size

* value sample value of a sampled [time, value] pair

8.2. SR-Path-Periodic-*--Std

For a type-p metric, the Standard-Deviation Std is specified by:

$SR\text{-}*\text{Std} = [1/(N-1)] * \text{Sum}(\text{from } a=1 \text{ to } N, [SR\text{-}*\text{Mean} - \text{value}[a]]^2)$

* N sample size

* value sample value of a sampled [time, value] pair

* SR-*Mean sample mean of the same metric as defined above

The definition as given above requires a two-pass calculation per sample. Algorithms estimating the standard-deviation by one-pass calculation have been published and might be preferable, if metric singletons and samples aren't buffered or calculations need to be fast.

9. Statistic Definitions for Type-P-SR-Path-Packet-Loss

The packet loss ratio is a useful metric to characterise congestion.

9.1. SR-Path-Packet-Loss-Ratio

See section 4.1 of [RFC7680]

10. Sub-Path monitoring metrics derived from samples captured along the measurement loops

To produce meaningful sub-path monitoring values, the measurement loop metrics are captured during a phase with stable networking conditions. In a backbone network domain, the absence of congestion often is a sufficient condition (frequent traffic shifts due to changes in routing and traffic engineering aren't expected). This may be different in a network based on a shared medium. It may be outright difficult in networks with frequently changing traffic management- and routing-policies.

In the following, the index CS indicates a statistic captured during a measurement interval with stable routing and no congestion.

10.1. Baseline measurement

Capture a sample of delay values Type-P-SR-Path-Periodic-Delay-Stream of sample size N for each measurement loop Fi. As a rule of thumb choose N in [30, 100].

For each measurement loop Fi, calculate the following metrics characterising the monitored Sub-Paths during stable and congestion free network conditions:

- * SR-Path-Delay-MeanCSi, the mean delay captured along measurement loop Fi
- * SR-Path-Delay-StdCSi, the standard-deviation of the delay captured along measurement loop Fi
- * SR-Path-Delay-Variation-MeanCSi, the mean delay variation captured along measurement loop Fi
- * SR-Path-Delay-Variation-StdCSi, the standard-deviation of the delay variation captured along measurement loop Fi

A stable and uncongested network should produce rather constant delays, resulting in low standard-deviation values and almost zero mean delay variation. [Editors note: Add text to select the median of a small set of stream mean captures, like 5 samples captured consecutively.]

Example data was captured in a lightly loaded Gigabit network. 11 routers are passed per measurement loop. The sample size is 30 packets, more than 200 samples were captured per measurement loop. The loops are set up for a different purpose than specified here, they are picked due to a high number of passed routers. Note that SR-DV-Mean here refers to an abs(SR-DV-Mean) sample, thus small, positive, non-zero means result. The time unit is microseconds.

Metric	Quantile	SR-D-Mean	SR-D-Std	SR-DV-Mean	SR-DV-Std
Loop1	95%	34507	62	41	84
Loop2	95%	35104	45	34	49
Loop1	50%	34496	19	19	17
Loop2	50%	35088	15	14	12
Loop1	5%	34491	14	20	12
Loop2	5%	35080	13	12	9

Figure 2

Example baseline metrics for an 11 hop measurement loop (quantiles refer to SR-D-Mean)

10.2. Discussion of the baseline measurement

Delay outliers may occur at any time in any communication network, and the measurement system packet processing itself may also produce some. It is fair to expect only single outliers in a stable, not congested network. It may be worth to capture several consecutive SR-Path-Periodic-*Stream samples and compare their statistics, before picking reasonable baseline metric values. Samples showing higher standard deviations (compare the 95% quantile values in the above figure to the 50% quantile values) may benefit from removing the maximum singleton value from the sample. This will smooth the mean and standard-deviation, and if the result then is closer to those of the majority of the samples, foster confidence in determining the baseline metrics. Depending on the preferred method of data-processing and storing, this may require capturing the sample maximum as a separate metric.

10.3. Definition of SR-Path-Sub-Path-RTD-Estimate

Within a single evaluation interval of identical Time T_0 and T_f , SR-Path-Delay-MeanCSi (from now on DMeanCSi) is the mean delay of the measurement loop passing the monitored Sub-Path S_{Pi} by a round trip. Let's keep the index i applied above, then F_j and F_k with captured mean delays DMeanCSj and DMeanCSk pass S_{Pi} unidirectional. Further, 3 measurement loops F_x , F_y and F_z don't pass Sub-Path S_{Pi} at all. The corresponding mean delays are DMeanCSs, DMeanCS_t and DMeanCS_u.

The the SR-Path-Sub-Path-RTD-Estimate of the Round Trip Delay along the monitored Sub-Path F_i , RTD_{Fi} , is

$$RTD_{Fi} = (3 * DMeanCSi + DMeanCSj + DMeanCSk - DMeanCSx - DMeanCSy - DMeanCSz) / 4$$

10.4. Definition of SR-Path-Sub-Path-*--Changepoint

The asterisk stands for "Interface" as well as "Connectivity". If connectivity is lost and no path is available between two nodes, any packets to be transmitted will be dropped. A change in sub-path routes with a change in measurement loop delay indicates a re-routing event (a temporal loss in connectivity), not a long lasting loss of connectivity. Hence a change in measurement loop delays caused by a re-routed monitored sub isn't useful to derive a metric indicating connectivity loss on a monitored sub path (a sub-path-route-change metric might be of interest, but isn't within scope of this document).

Network changes like congestion or re-routing are often characterised by a change in the mean delay of a monitoring measurement. CUSUM (cumulative sum) charts have been shown to be efficient in detecting shifts in the mean of a process [NIST]. The upper bound CUSUM is defined as:

$$Sup(t) - Fi - Delay = \max(0, Sup(t-1) + x_t - SR-Path-* - MeanCSi - k_i)$$

with $Sup(0) = 0$, $k_i = \Delta * SR-Path-* - StdCSi$ (Δ is a dimensionless integer number), $x_t = Type-P-SR-Path-Periodic-*$ singleton for measurement loop F_i at time t .

The actual SR-Path-Delay-Mean of Measurement Loop F_i is decided to be significantly above $SR-Path-* - MeanCSi$, if:

$$Sup(t) - Fi - Delay > h_{SP}, \text{ with } h_{SP} = d * k_i \text{ (d is a dimensionless integer number)}.$$

An analogous CUSUM controls changes to a lower mean delay (which may be caused by a re-routing event):

$$\text{Slo}(t)\text{-Fi-Delay} = \max(0, \text{Slo}(t-1) + \text{SR-Path-}^*\text{-MeanCSi} - x_j - k)$$

The actual SR-Path-Delay-Mean of Fi is decided to be significantly below SR-Path-^{*}-MeanCSi, if:

$$\text{Slo}(t)\text{-Fi-Delay} > h_{\text{SP}}$$

10.5. Discussion of SR-Path-Sub-Path-^{*}-Changepoint

CUSUM chart based changepoint detection is sensible even to small changes in the mean. CUSUM charts offer a limited protection against single, isolated outliers. A cumulated sum only grows, if the controlled process consistently changes its mean (or standard deviation, respectively). Assuming constant physical minimum delays to characterise wireline communication networks, a change in standard deviation not affecting the mean delay doesn't seem to be caused by a change in networking conditions.

The measured delays will change once a Sub-Path route has changed, or once persistent congestion starts to fill a queue. Both indicate changes in the network. As the Sub-Pathes SPi form an overlay with designed properties, every network change affecting a sub-path creates correlated SR-Path-^{*} metric changes. As the correspondance of network changes to Sub-Path metrics is known a-priori, detecting correlated SR-Path-^{*} metric changes allows to locate the change.

In the absence of packet re-routing, packet loss is characterising a loss of connectivity. Packet loss requires a time threshold when to decide that an active measurement packet was lost, and consecutive loss requires receiver awareness, that packets have been sent (this argues for the sender to be the receiver, unless both communicate fast and reliable out of band).

The preferred CUSUM parametrisation will depend on the kind of events to detected and on the outlier characteristics.

$k_i = \Delta * \text{SR-Path-}^*\text{-StdCSi}$ may be set to a value relevant high enough to exclude single outliers to trigger an alert, but low enough to indicate persistent changes in delay. The same holds for the to be picked for d.

A broader discussion on CUSUM parametrisation may be found in literature. Networking skills are required to parametrise CUSUM, as well as to interpret the results (notably to differ re-routing from congestion).

10.6. Definition of SR-Path-Sub-Path-Congestion-Location

An interface along a single monitored Sub-Path SP_i whose queue is persistently filled adds latency to measurement loop F_i and one of the two unidirectional measurement loops F_j and F_k passing Sub-Path SP_i . F_j has been defined to pass SP_i from Hub to Spoke and F_k pass SP_i in opposite direction. Then SR-Path-Sub-Path-Congestion-Location metric for the traffic directed from "Hub to Spoke" along Sub-Path SP_i is:

$$SP_i_ConLoc_ij = Sup(t)_SP_i_Periodic-Delay + Sup(t)_SP_j_Periodic-Delay$$

And for the opposite traffic direction, from "Spoke to Hub":

$$SP_i_ConLoc_ik = Sup(t)_SP_i_Periodic-Delay + Sup(t)_SP_k_Periodic-Delay$$

Note that another 10 SR-Path-Sub-Path-Congestion-Location metrics are calculated, one per monitored Sub Path and traffic direction. The evaluation can be simplified as follows:

```
IF  $SP_i\_ConLoc\_ij > h\_SP$ 
AND  $h\_SP > Sup(t)\_SP_k\_Periodic-Delay$ 
AND  $h\_SP > Sup(t)\_SP_x\_Periodic-Delay$ 
AND  $h\_SP > Sup(t)\_SP_y\_Periodic-Delay$ 
AND  $h\_SP > Sup(t)\_SP_z\_Periodic-Delay$ 
```

Then Sub-Path SP_i faces congestion in direction "Hub to Spoke".

```
IF  $SP_i\_ConLoc\_ik > h\_SP$ 
AND  $h\_SP > Sup(t)\_SP_j\_Periodic-Delay$ 
AND  $h\_SP > Sup(t)\_SP_x\_Periodic-Delay$ 
AND  $h\_SP > Sup(t)\_SP_y\_Periodic-Delay$ 
AND  $h\_SP > Sup(t)\_SP_z\_Periodic-Delay$ 
```

Then Sub-Path SP_i faces congestion in direction "Spoke to Hub".

Here, h_{SP} is a universal threshold in unit time to indicate a filling queue or a significant change in delay due to a Sub-Path reroute or another persistent change in topology (like e.g. automated Layer 1 / Layer 2 topology changes). Packets following SP_x , SP_y and SP_z don't pass the congested interface of Sub-Path SP_i .

10.7. Definition of SR-Path-Sub-Path-Disconnected

The idea of this document is to monitor a set of sub-paths for a single case of congestion or a single loss of connectivity. If a single sub-path SP_i loses connectivity, i.e., all packets are dropped in both sub-path forwarding directions, then three measurement loops mi , mj and mk fail to receive any traffic. A single interface congestion will add latency to mi and one of mj or mk , respectively. Still, if it is congestion of a single sub-path SP_i interface causing additional latency, either mj or mk face no congestion and the one measured delay mj or mk should be within the expected range of values. Rather than basing a loss of connectivity metric on a "reliable" indication SR-Path-Packet-Loss on each measurement loop mi , mj and mk by waiting for T_{max} to receive any of the missed packets, this allows for a reaction independent of a conservative packet loss threshold like T_{max} . The idea is to judge on disconnectivity if no packet is received on all three measurement loops mi , mj and mk after the time interval the last single packet was expected to be received, if there was no prior indication of congestion.

If the spacing of packets along consecutive measurement loops Fi is $IncF$ as defined within section Section 3.4, then under stable network conditions every measurement packet sent along measurement loop Fi is received, before the next measurement packet is sent along measurement loop Fj . If a measurement interval starts at $T1$ and none of the three measurement loops Fi , Fj and Fk received a packet within $T1 + incT = T1 + 6 * incF$, monitored Sub-Path i is disconnected. It doesn't matter, along which of the three measurement loops the first not received packet was sent (there's no order here).

$$incF > \max (SR\text{-}Path\text{-}Delay\text{-}MeanCS_i + d * \Delta * SR\text{-}Path\text{-}Delay\text{-}StdCS_i), i \text{ in } [1...6]$$

With d and Δ being integer numbers as specified in section Section 10.4. If Fi and $Fi+1$ are measurement loops along which measurement packets are sent in consecutive order, this definition of $incF$ ensures that the measurement packet sent along measurement loop Fi is received prior to sending the next measurement packet along measurement loop $Fi+1$ (under stable network conditions). The product $d * \Delta * SR\text{-}Path\text{-}Delay\text{-}StdCS_i$ allows to set the preferred tolerance for outliers. It impacts the tradeoff between speed of

detection and false positive ratio. With this parameterisation, the metric indicating a loss of bidirectional connectivity along Sub-Path i is defined as

either zero or one (or some logical equivalent), where $LofCi=1$ indicates loss of continuity along monitored Sub-Path Fi and $LofCi=0$ indicates successful arrival of at least one packet sent along measurement-loop Fi , Fj or Fk within $incT$.

Under conditions of section Section 3.4, if at any sliding interval $incT$ no singleton was received along measurement-loops Fi , Fj and Fk , no more packets are forwarded in any direction of monitored sub-path SPi .

Faster detection of disconnectivity is likely possible by a different metric definition, which likely will depend on the measurement-loop delay Mi , Mj and Mk . The metric chosen above allows for a simple parametrisation. Metrics allowing for a faster determination of disconnection are not within scope of this document.

The sub-path SPi is judged to be disconnected from the earliest time, when a packet was sent but not received on any of the three sub-paths Fi , Fj or Fk . The sub-path SPi is judged to be connected, whenever a measurement packet sent along one or more of the measurement-loops Fi , Fj and Fk is received again.

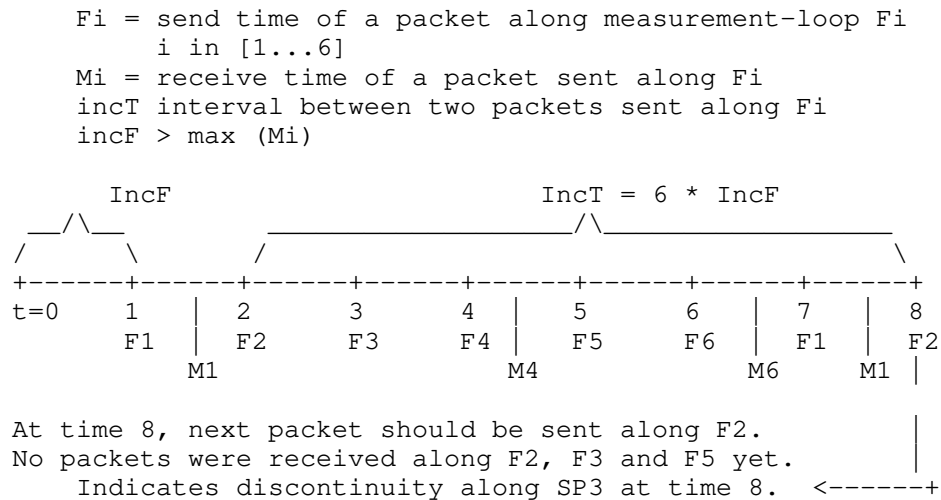


Figure 3

Illustration of the sub-path disconnectivity metric; sub-path SP3 is link L100 <-> L070 of the example network Figure 1.

Note, if F2 sent at time 2 was received at time 2 + M2, but no more packet passing SP3 afterwards, discontinuity of SP3 is indicated at time 9, when F3 is to send the next packet. Also note that discontinuity of SP3 could be indicated as early as time 6 in the example. That requires a different metric. Basing the metric definition on incT however covers all potential intervals between relevant Fi, Fj and Fk.

11. Discussion of Temporal Resolution

A loss of connectivity is detected after a temporal distance of IncT, the time period between two packets being sent along the same measurement-loop Fi. IncT is specified as 6*IncF, where IncF is 2 times the largest measurement-loop delay in the absence of congestion. Hence a loss of connectivity is indicated after 12 * the largest measurement-loop delay.

Reliable indications of lost connectivity may be possible also at smaller timescales. The specification chosen seems to be simple as well as reliable and thus defines a starting point for advanced designs offering faster reaction.

12. IANA Considerations

If standardised, the metric will require an entry in the IPPM metric registry.

13. Security Considerations

This draft specifies how to use methods specified or described within [RFC8402] and [RFC8403]. It does not introduce new or additional SR features. The security considerations of both references apply here too.

14. References

14.1. Normative References

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

- [RFC2678] Mahdavi, J. and V. Paxson, "IPPM Metrics for Measuring Connectivity", RFC 2678, DOI 10.17487/RFC2678, September 1999, <<https://www.rfc-editor.org/info/rfc2678>>.
- [RFC3393] Demichelis, C. and P. Chimento, "IP Packet Delay Variation Metric for IP Performance Metrics (IPPM)", RFC 3393, DOI 10.17487/RFC3393, November 2002, <<https://www.rfc-editor.org/info/rfc3393>>.
- [RFC3432] Raisanen, V., Grotefeld, G., and A. Morton, "Network performance measurement with periodic streams", RFC 3432, DOI 10.17487/RFC3432, November 2002, <<https://www.rfc-editor.org/info/rfc3432>>.
- [RFC6673] Morton, A., "Round-Trip Packet Loss Metrics", RFC 6673, DOI 10.17487/RFC6673, August 2012, <<https://www.rfc-editor.org/info/rfc6673>>.
- [RFC7679] Almes, G., Kalidindi, S., Zekauskas, M., and A. Morton, Ed., "A One-Way Delay Metric for IP Performance Metrics (IPPM)", STD 81, RFC 7679, DOI 10.17487/RFC7679, January 2016, <<https://www.rfc-editor.org/info/rfc7679>>.
- [RFC7680] Almes, G., Kalidindi, S., Zekauskas, M., and A. Morton, Ed., "A One-Way Loss Metric for IP Performance Metrics (IPPM)", STD 82, RFC 7680, DOI 10.17487/RFC7680, January 2016, <<https://www.rfc-editor.org/info/rfc7680>>.
- [RFC8029] Kompella, K., Swallow, G., Pignataro, C., Ed., Kumar, N., Aldrin, S., and M. Chen, "Detecting Multiprotocol Label Switched (MPLS) Data-Plane Failures", RFC 8029, DOI 10.17487/RFC8029, March 2017, <<https://www.rfc-editor.org/info/rfc8029>>.
- [RFC8287] Kumar, N., Ed., Pignataro, C., Ed., Swallow, G., Akiya, N., Kini, S., and M. Chen, "Label Switched Path (LSP) Ping/Traceroute for Segment Routing (SR) IGP-Prefix and IGP-Adjacency Segment Identifiers (SIDs) with MPLS Data Planes", RFC 8287, DOI 10.17487/RFC8287, December 2017, <<https://www.rfc-editor.org/info/rfc8287>>.
- [RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <<https://www.rfc-editor.org/info/rfc8402>>.

- [RFC8667] Previdi, S., Ed., Ginsberg, L., Ed., Filsfils, C., Bashandy, A., Gredler, H., and B. Decraene, "IS-IS Extensions for Segment Routing", RFC 8667, DOI 10.17487/RFC8667, December 2019, <<https://www.rfc-editor.org/info/rfc8667>>.

14.2. Informative References

- [CommodityTomography]
Lakhina, A., Papagiannaki, K., Crovella, M., Diot, C., Kolaczyk, ED., and N. Taft, "Structural analysis of network traffic flows", 2004, <https://www.cc.gatech.edu/classes/AY2007/cs7260_spring/papers/odflows-sigm04.pdf>.
- [ID.draft-ietf-6man-spring-srv6-oam]
Zafar, A., Filsfils, C., Matsushima, S., Voyer, D., and M. Chen, "Operations, Administration, and Maintenance (OAM) in Segment Routing Networks with IPv6 Data plane (SRv6)", 2021.
- [NIST] NIST, "NIST/SEMATECH e-Handbook of Statistical Methods, section CUSUM Control Charts", 2021, <<http://www.itl.nist.gov/div898/handbook/>>.
- [RFC2330] Paxson, V., Almes, G., Mahdavi, J., and M. Mathis, "Framework for IP Performance Metrics", RFC 2330, DOI 10.17487/RFC2330, May 1998, <<https://www.rfc-editor.org/info/rfc2330>>.
- [RFC8403] Geib, R., Ed., Filsfils, C., Pignataro, C., Ed., and N. Kumar, "A Scalable and Topology-Aware MPLS Data-Plane Monitoring System", RFC 8403, DOI 10.17487/RFC8403, July 2018, <<https://www.rfc-editor.org/info/rfc8403>>.

Author's Address

Ruediger Geib (editor)
Deutsche Telekom
Heinrich Hertz Str. 3-7
64295 Darmstadt
Germany
Phone: +49 6151 5812747
Email: Ruediger.Geib@telekom.de

IPPM
Internet-Draft
Intended status: Informational
Expires: November 6, 2022

M. Cociglio
Telecom Italia - TIM
A. Ferrieux
Orange Labs
G. Fioccola
Huawei Technologies
I. Lubashev
Akamai Technologies
F. Bulgarella
Telecom Italia - TIM
I. Hamchaoui
Orange Labs
M. Nilo
Telecom Italia - TIM
R. Sisto
Politecnico di Torino
D. Tikhonov
LiteSpeed Technologies
May 5, 2022

Explicit Flow Measurements Techniques
draft-ietf-ippm-explicit-flow-measurements-01

Abstract

This document describes protocol independent methods called Explicit Flow Measurement Techniques that employ few marking bits, inside the header of each packet, for loss and delay measurement. The endpoints, marking the traffic, signal these metrics to intermediate observers allowing them to measure connection performance, and to locate the network segment where impairments happen. Different alternatives are considered within this document. These signaling methods apply to all protocols but they are especially valuable when applied to protocols that encrypt transport header and do not allow traditional methods for delay and loss detection.

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 November 6, 2022.

Copyright Notice

Copyright (c) 2022 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
2. Latency Bits	5
2.1. Spin Bit	5
2.2. Delay Bit	6
2.2.1. Generation Phase	8
2.2.2. Reflection Phase	8
2.2.3. T_Max Selection	9
2.2.4. Delay Measurement Using Delay Bit	10
2.2.4.1. RTT Measurement	10
2.2.4.2. Half-RTT Measurement	10
2.2.4.3. Intra-Domain RTT Measurement	11
2.2.5. Observer's Algorithm	12
2.2.6. Two Bits Delay Measurement: Spin Bit + Delay Bit . .	13
2.2.7. Hidden Delay Bit - Delay Bit with Privacy Protection	13
3. Loss Bits	13
3.1. T Bit - Round Trip Loss Bit	14
3.1.1. Round Trip Loss	15
3.1.2. Setting the Round Trip Loss Bit on Outgoing Packets .	16
3.1.3. Observer's Logic for Round Trip Loss Signal	18
3.1.4. Loss Coverage and Signal Timing	18
3.2. Q Bit - sSquare Bit	19
3.2.1. Q Block Length Selection	19
3.2.2. Upstream Loss	20
3.2.3. Identifying Q Block Boundaries	20

3.2.3.1. Improved Resilience to Burst Losses	21
3.3. L Bit - Loss Event Bit	21
3.3.1. End-To-End Loss	22
3.3.1.1. Loss Profile Characterization	22
3.3.2. L+Q Bits - Loss Measurement Using L and Q Bits	22
3.3.2.1. Correlating End-to-End and Upstream Loss	23
3.3.2.2. Downstream Loss	23
3.3.2.3. Observer Loss	24
3.4. R Bit - Reflection Square Bit	24
3.4.1. Enhancement of R Block Length Computation	25
3.4.2. Improved Resilience to Packet Reordering	26
3.4.2.1. Improved Resilience to Burst Losses	26
3.4.3. R+Q Bits - Loss Measurement Using R and Q Bits	26
3.4.3.1. Three-Quarters Connection Loss	27
3.4.3.2. End-To-End Loss in the Opposite Direction	27
3.4.3.3. Half Round-Trip Loss	28
3.4.3.4. Downstream Loss	29
3.5. E Bit - ECN-Echo Event Bit	29
3.5.1. Setting the ECN-Echo Event Bit on Outgoing Packets .	29
3.5.2. Using E Bit for Passive ECN-Reported Congestion	
Measurement	30
4. Summary of Delay and Loss Marking Methods	30
5. Protocol Ossification Considerations	32
6. Examples of Application	33
6.1. QUIC	33
6.2. TCP	34
7. Security Considerations	34
7.1. Optimistic ACK Attack	35
8. Privacy Considerations	35
9. IANA Considerations	36
10. Contributors	36
11. Acknowledgements	36
12. References	36
12.1. Normative References	36
12.2. Informative References	37
Authors' Addresses	39

1. Introduction

Packet loss and delay are hard and pervasive problems of day-to-day network operation. Proactively detecting, measuring, and locating them is crucial to maintaining high QoS and timely resolution of crippling end-to-end throughput issues. To this effect, in a TCP-dominated world, network operators have been heavily relying on information present in the clear in TCP headers: sequence and acknowledgment numbers and SACKs when enabled (see [RFC8517]). These allow for quantitative estimation of packet loss and delay by passive

on-path observation. Additionally, the problem can be quickly identified in the network path by moving the passive observer around.

With encrypted protocols, the equivalent transport headers are encrypted and passive packet loss and delay observations are not possible, as described in [RFC9065].

Measuring TCP loss and delay between similar endpoints cannot be relied upon to evaluate encrypted protocol loss and delay. Different protocols could be routed by the network differently, and the fraction of Internet traffic delivered using protocols other than TCP is increasing every year. It is imperative to measure packet loss and delay experienced by encrypted protocol users directly.

This document defines Explicit Flow Measurement Techniques. These hybrid measurement path signals (see [IPM-Methods]) are to be embedded into a transport layer protocol and are explicitly intended for exposing RTT and loss rate information to on-path measurement devices. They are designed to facilitate network operations and management and are "beneficial" for maintaining the quality of service (see [RFC9065]). These measurement mechanisms are applicable to any transport-layer protocol, and, as an example, the document describes QUIC and TCP bindings.

The Explicit Flow Measurement Techniques described in this document can be used alone or in combination with other Explicit Flow Measurement Techniques. Each technique uses a small number of bits and exposes a specific measurement.

Following the recommendation in [RFC8558] of making path signals explicit, this document proposes adding a small number of dedicated measurement bits to the clear portion of the protocol headers. These bits can be added to an unencrypted portion of a header belonging to any protocol layer, e.g. IP (see [IP]) and IPv6 (see [IPv6]) headers or extensions, such as [IPv6AltMark], UDP surplus space (see [UDP-OPTIONS] and [UDP-SURPLUS]), reserved bits in a QUIC v1 header, as already done with the latency Spin bit (see [QUIC-TRANSPORT]).

The measurements are not designed for use in automated control of the network in environments where signal bits are set by untrusted hosts. Instead, the signal is to be used for troubleshooting individual flows as well as for monitoring the network by aggregating information from multiple flows and raising operator alarms if aggregate statistics indicate a potential problem.

The Spin bit, Delay bit and loss bits explained in this document are inspired by [AltMark], [SPIN-BIT], [I-D.trammell-tsvwg-spin] and [I-D.trammell-ippm-spin].

Additional details about the Performance Measurements for QUIC are described in the paper [ANRW19-PM-QUIC].

2. Latency Bits

This section introduces bits that can be used for round trip latency measurements. Whenever this section of the specification refers to packets, it is referring only to packets with protocol headers that include the latency bits.

[QUIC-TRANSPORT] introduces an explicit per-flow transport-layer signal for hybrid measurement of RTT. This signal consists of a Spin bit that toggles once per RTT. [SPIN-BIT] discusses an additional two-bit Valid Edge Counter (VEC) to compensate for loss and reordering of the Spin bit and increase fidelity of the signal in less than ideal network conditions.

This document introduces a stand-alone single-bit delay signal that can be used by passive observers to measure the RTT of a network flow, avoiding the Spin bit ambiguities that arise as soon as network conditions deteriorate.

2.1. Spin Bit

This section is a small recap of the Spin bit working mechanism. For a comprehensive explanation of the algorithm, please see [SPIN-BIT].

The Spin bit is an alternate marking [AltMark] generated signal, where the size of the alternation changes with the flight size each RTT.

The latency Spin bit is a single bit signal that toggles once per RTT, enabling latency monitoring of a connection-oriented communication from intermediate observation points.

A "spin period" is a set of packets with the same Spin bit value sent during one RTT time interval. A "spin period value" is the value of the Spin bit shared by all packets in a spin period.

The client and server maintain an internal per-connection spin value (i.e. 0 or 1) used to set the Spin bit on outgoing packets. Both endpoints initialize the spin value to 0 when a new connection starts. Then:

- when the client receives a packet with the packet number larger than any number seen so far, it sets the connection spin value to the opposite value contained in the received packet;

- when the server receives a packet with the packet number larger than any number seen so far, it sets the connection spin value to the same value contained in the received packet.

The computed spin value is used by the endpoints for setting the spin bit on outgoing packets. This mechanism allows the endpoints to generate a square wave such that, by measuring the distance in time between pairs of consecutive edges observed in the same direction, a passive on-path observer can compute the round trip delay of that network flow.

Spin bit enables round trip latency measurement by observing a single direction of the traffic flow.

Note that packet reordering can cause spurious edges that require heuristics to correct. The Spin bit performance deteriorates as soon as network impairments arise as explained in Section 2.2.

2.2. Delay Bit

The Delay bit has been designed to overcome accuracy limitations experienced by the Spin bit under difficult network conditions:

- packet reordering leads to generation of spurious edges and errors in delay estimation;
- loss of edges causes wrong estimation of spin periods and therefore wrong RTT measurements;
- application-limited senders cause the Spin bit to measure the application delays instead of network delays.

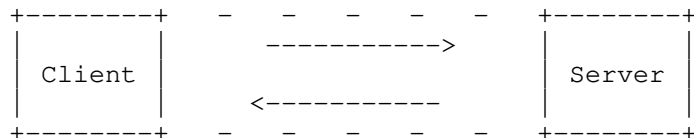
Unlike the Spin bit, which is set in every packet transmitted on the network, the Delay bit is set only once per round trip.

When the Delay bit is used, a single packet with a marked bit (the Delay bit) bounces between a client and a server during the entire connection lifetime. This single packet is called "delay sample".

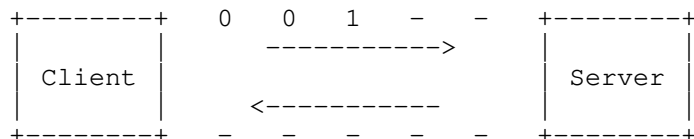
An observer placed at an intermediate point, observing a single direction of traffic, tracking the delay sample and the relative timestamp, can measure the round trip delay of the connection.

The delay sample lifetime is comprised of two phases: initialization and reflection. The initialization is the generation of the delay sample, while the reflection realizes the bounce behavior of this single packet between the two endpoints.

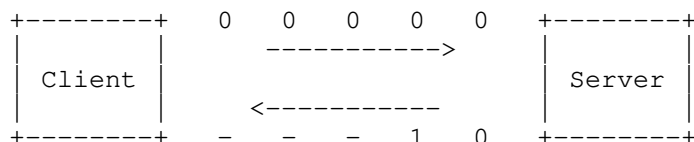
The next figure describes the elementary Delay bit mechanism.



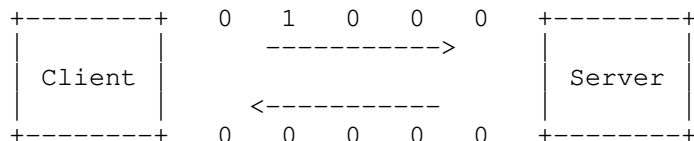
(a) No traffic at beginning.



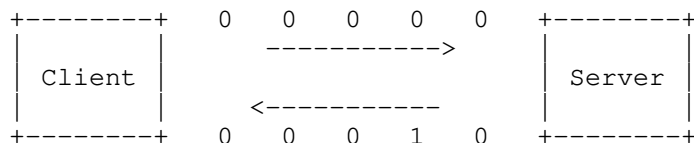
(b) The Client starts sending data and sets the first packet as Delay Sample.



(c) The Server starts sending data and reflects the Delay Sample.



(d) The Client reflects the Delay Sample.



(e) The Server reflects the Delay Sample and so on.

Delay bit mechanism

2.2.1. Generation Phase

Only client is actively involved in the generation phase. It maintains an internal per-flow timestamp variable ("ds_time") updated every time a delay sample is transmitted.

When connection starts, the client generates a new delay sample initializing the Delay bit of the first outgoing packet to 1. Then it updates the "ds_time" variable with the timestamp of its transmission.

The server initializes the Delay bit to 0 at the beginning of the connection, and its only task during the connection is described in Section 2.2.2.

In absence of network impairments, the delay sample should bounce between client and server continuously, for the entire duration of the connection. That is highly unlikely for two reasons:

1. the packet carrying the Delay bit might be lost;
2. an endpoint could stop or delay sending packets because the application is limiting the amount of traffic transmitted.

To deal with these problems, the client generates a new delay sample if more than a predetermined time ("T_Max") has elapsed since the last delay sample transmission (including reflections). Note that "T_Max" should be greater than the max measurable RTT on the network. See Section 2.2.3 for details.

2.2.2. Reflection Phase

Reflection is the process that enables the bouncing of the delay sample between a client and a server. The behavior of the two endpoints is almost the same.

- Server side reflection: when a delay sample arrives, the server marks the first packet in the opposite direction as the delay sample.
- Client side reflection: when a delay sample arrives, the client marks the first packet in the opposite direction as the delay sample. It also updates the "ds_time" variable when the outgoing delay sample is actually forwarded.

In both cases, if the outgoing delay sample is being transmitted with a delay greater than a predetermined threshold after the reception of

the incoming delay sample (1ms by default), the delay sample is not reflected, and the outgoing Delay bit is kept at 0.

By doing so, the algorithm can reject measurements that would overestimate the delay due to lack of traffic on the endpoints. Hence, the maximum estimation error would amount to twice the threshold (e.g. 2ms) per measurement.

2.2.3. T_Max Selection

The internal "ds_time" variable allows a client to identify delay sample losses. Considering that a lost delay sample is regenerated at the end of an explicit time ("T_Max") since the last generation, this same value can be used by an observer to reject a measure and start a new one.

In other words, if the difference in time between two delay samples is greater or equal than "T_Max", then these cannot be used to produce a delay measure. Therefore the value of "T_Max" must also be known to the on-path network probes.

There are two alternatives to select the "T_Max" value so that both client and observers know it. The first one requires that "T_Max" is known a priori ("T_Max_p") and therefore set within the protocol specifications that implements the marking mechanism (e.g. 1 second which usually is greater than the max expectable RTT). The second alternative requires a dynamic mechanism able to adapt the duration of the "T_Max" to the delay of the connection ("T_Max_c").

For instance, client and observers could use the connection RTT as a basis for calculating an effective "T_Max". They should use a predetermined initial value so that "T_Max = T_Max_p" (e.g. 1 second) and then, when a valid RTT is measured, change "T_Max" accordingly so that "T_Max = T_Max_c". In any case, the selected "T_Max" should be large enough to absorb any possible variations in the connection delay.

"T_Max_c" could be computed as two times the measured "RTT" plus a fixed amount of time ("100ms") to prevent low "T_Max" values in case of very small RTTs. The resulting formula is: "T_Max_c = 2RTT + 100ms". If "T_Max_c" is greater than "T_Max_p" then "T_Max_c" is forced to "T_Max_p" value.

Note that the observer's "T_Max" should always be less than or equal to the client's "T_Max" to avoid considering as a valid measurement what is actually the client's "T_Max". To obtain this result, the client waits for two consecutive incoming samples and computes the two related RTTs. Then it takes the largest of them as the basis of

the "T_Max_c" formula. At this point, observers have already measured a valid RTT and then computed their "T_Max_c".

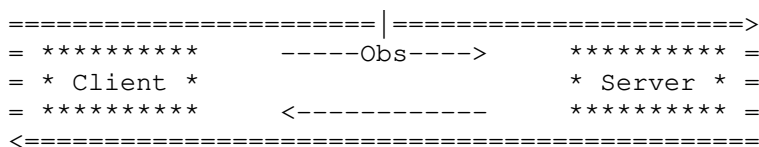
2.2.4. Delay Measurement Using Delay Bit

When the Delay bit is used, a passive observer can use delay samples directly and avoid inherent ambiguities in the calculation of the RTT as can be seen in Spin bit analysis.

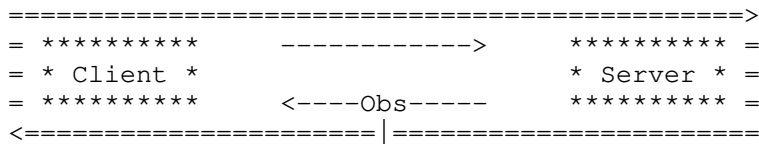
2.2.4.1. RTT Measurement

The delay sample generation process ensures that only one packet marked with the Delay bit set to 1 runs back and forth between two endpoints per round trip time. To determine the RTT measurement of a flow, an on-path passive observer computes the time difference between two delay samples observed in a single direction.

To ensure a valid measurement, the observer must verify that the distance in time between the two samples taken into account is less than "T_Max".



(a) client-server RTT



(b) server-client RTT

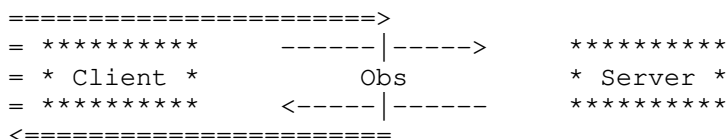
Round-trip time (both direction)

2.2.4.2. Half-RTT Measurement

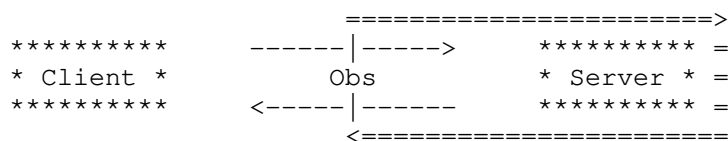
An observer that is able to observe both forward and return traffic directions can use the delay samples to measure "upstream" and "downstream" RTT components, also known as the half-RTT measurements. It does this by measuring the time between a delay sample observed in one direction and the delay sample previously observed in the opposite direction.

As with RTT measurement, the observer must verify that the distance in time between the two samples taken into account is less than "T_Max".

Note that upstream and downstream sections of paths between the endpoints and the observer, i.e. observer-to-client vs client-to-observer and observer-to-server vs server-to-observer, may have different delay characteristics due to the difference in network congestion and other factors.



(a) client-observer half-RTT

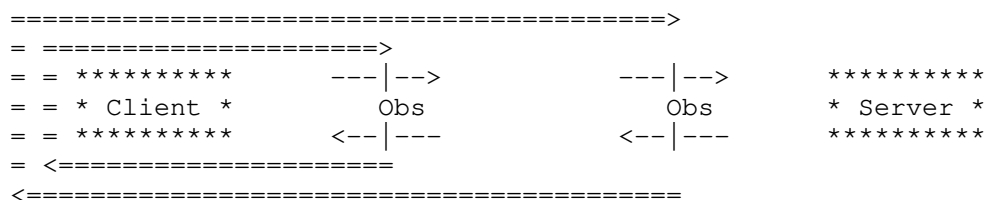


(b) observer-server half-RTT

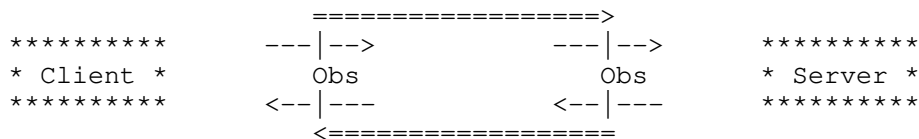
Half Round-trip time (both direction)

2.2.4.3. Intra-Domain RTT Measurement

Intra-domain RTT is the portion of the entire RTT used by a flow to traverse the network of a provider. To measure intra-domain RTT, two observers capable of observing traffic in both directions must be employed simultaneously at ingress and egress of the network to be measured. Intra-domain RTT is difference between the two computed upstream (or downstream) RTT components.



(a) client-observer RTT components (half-RTTs)



(b) the intra-domain RTT resulting from the subtraction of the above RTT components

Intra-domain Round-trip time (client-observer: upstream)

2.2.5. Observer's Algorithm

An on-path observer maintains an internal per-flow variable to keep track of time at which the last delay sample has been observed.

A unidirectional observer, upon detecting a delay sample:

- if a delay sample was also detected previously in the same direction and the distance in time between them is less than " $T_{Max} - K$ ", then the two delay samples can be used to calculate RTT measurement. " K " is a protection threshold to absorb differences in " T_{Max} " computation and delay variations between two consecutive delay samples (e.g. " $K = 10\% T_{Max}$ ").

If the observer can observe both forward and return traffic flows, and it is able to determine which direction contains the client and the server (e.g. by observing the connection handshake), upon detecting a delay sample:

- if a delay sample was also detected in the opposite direction and the distance in time between them is less than " $T_{Max} - K$ ", then the two delay samples can be used to measure the observer-client half-RTT or the observer-server half-RTT, according to the direction of the last delay sample observed.

2.2.6. Two Bits Delay Measurement: Spin Bit + Delay Bit

Spin and Delay bit algorithms work independently. If both marking methods are used in the same connection, observers can choose the best measurement between the two available:

- when a precise measurement can be produced using the Delay bit, observers choose it;
- when a Delay bit measurement is not available, observers choose the approximate Spin bit one.

2.2.7. Hidden Delay Bit - Delay Bit with Privacy Protection

Theoretically, delay measurements can be used to roughly evaluate the distance of the client from the server (using the RTT) or from any intermediate observer (using the client-observer half-RTT). To protect users' privacy, the Delay bit algorithm can be slightly modified to mask the RTT of the connection to an intermediate observer. This result can be achieved by, for example, delaying the client-side reflection of the delay sample by a fixed randomly chosen time value. This would lead an intermediate observer to measure a delay greater than the real one.

This Additional Delay should be randomly selected by the client and kept constant for a certain amount of time across multiple connections. This ensures that the client-server jitter remains the same as if no Additional Delay had been inserted. For example, a new Additional Delay value could be generated whenever the client's IP address changes.

Using this technique, despite the Additional Delay introduced, it is still possible to correctly measure the right component of RTT (observer-server) and all the intra-domain measurements used to distribute the delay in the network. Furthermore, differently from the Delay bit, the hidden Delay bit makes the use of the client reflection threshold (1ms) redundant. Removing this threshold leads to the further advantage of increasing the number of valid measurements produced by the algorithm.

3. Loss Bits

This section introduces bits that can be used for loss measurements. Whenever this section of the specification refers to packets, it is referring only to packets with protocol headers that include the loss bits - the only packets whose loss can be measured.

- T: the "round Trip loss" bit is used in combination with the Spin bit to measure round-trip loss. See Section 3.1.
- Q: the "square signal" bit is used to measure upstream loss. See Section 3.2.
- L: the "Loss event" bit is used to measure end-to-end loss. See Section 3.3.
- R: the "Reflection square signal" bit is used in combination with Q bit to measure end-to-end loss. See Section 3.1.

Loss measurements enabled by T, Q, and L bits can be implemented by those loss bits alone (T bit requires a working Spin bit). Two-bit combinations Q+L and Q+R enable additional measurement opportunities discussed below.

Each endpoint maintains appropriate counters independently and separately for each separately identifiable flow (each sub-flow for multipath connections).

Since loss is reported independently for each flow, all bits (except for L bit) require a certain minimum number of packets to be exchanged per flow before any signal can be measured. Therefore, loss measurements work best for flows that transfer more than a minimal amount of data.

3.1. T Bit - Round Trip Loss Bit

The round Trip loss bit is used to mark a variable number of packets exchanged twice between the endpoints realizing a two round-trip reflection. A passive on-path observer, observing either direction, can count and compare the number of marked packets seen during the two reflections, estimating the loss rate experienced by the connection. The overall exchange comprises:

- the client selects, generates and consequently transmits a first train of packets, by setting the T bit to 1;
- the server, upon receiving each packet included in the first train, reflects to the client a respective second train of packets of the same size as the first train received, by setting the T bit to 1;
- the client, upon receiving each packet included in the second train, reflects to the server a respective third train of packets of the same size as the second train received, by setting the T bit to 1;

- the server, upon receiving each packet included in the third train, finally reflects to the client a respective fourth train of packets of the same size as the third train received, by setting the T bit to 1.

Packets belonging to the first round trip (first and second train) represent the Generation Phase, while those belonging to the second round trip (third and fourth train) represent the Reflection Phase.

A passive on-path observer can count and compare the number of marked packets seen during the two round trips (i.e. the first and third or the second and the fourth trains of packets, depending on which direction is observed) and estimate the loss rate experienced by the connection. This process is repeated continuously to obtain more measurements as long as the endpoints exchange traffic. These measurements can be called Round Trip losses.

Since packet rates in two directions may be different, the number of marked packets in the train is determined by the direction with the lowest packet rate. See Section 3.1.2 for details on packet generation.

3.1.1. Round Trip Loss

Since the measurements are performed on a portion of the traffic exchanged between the client and the server, the observer calculates the end-to-end Round Trip Packet Loss (RTPL) that, statistically, will correspond to the loss rate experienced by the connection along the entire network path.

```

=====|=====>
= *****      -----Obs----->      ***** =
= * Client *                                * Server * =
= *****      <-----              ***** =
<=====

```

(a) client-server RTPL

```

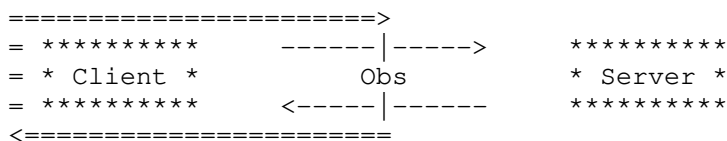
=====|=====>
= *****      ----->      ***** =
= * Client *                                * Server * =
= *****      <----Obs-----      ***** =
<=====|=====

```

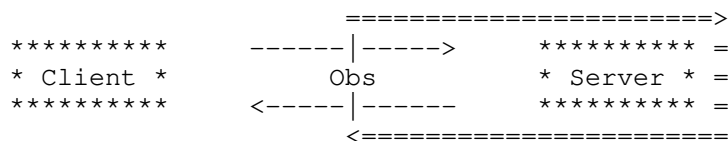
(b) server-client RTPL

Round-trip packet loss (both direction)

This methodology also allows the Half-RTPL measurement and the Intra-domain RTPL measurement in a way similar to RTT measurement.

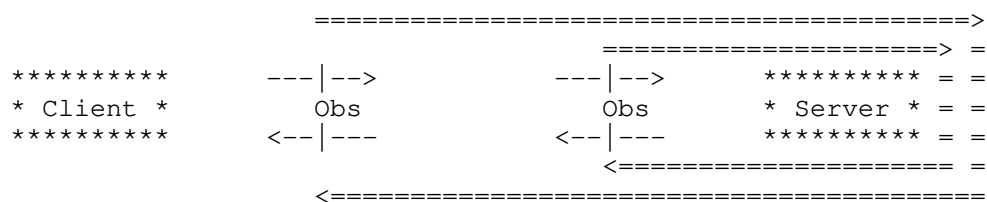


(a) client-observer half-RTPL

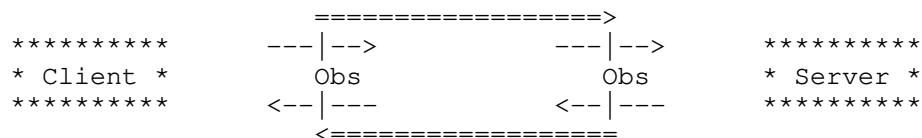


(b) observer-server half-RTPL

Half Round-trip packet loss (both direction)



(a) observer-server RTPL components (half-RTPLs)



(b) the intra-domain RTPL resulting from the subtraction of the above RTPL components

Intra-domain Round-trip packet loss (observer-server)

3.1.2. Setting the Round Trip Loss Bit on Outgoing Packets

The round Trip loss signal requires a working Spin-bit signal to separate trains of marked packets (packets with T bit set to 1). A "pause" of at least one empty spin-bit period between each phase of the algorithm serves as such separator for the on-path observer.

The client maintains a "generation token" count that is set to zero at the beginning of the session and is incremented every time a packet is received (marked or unmarked). The client also maintains a "reflection counter" that starts at zero at the beginning of the session.

The client is in charge of launching trains of marked packets and does so according to the algorithm:

1. Generation Phase. The client starts generating marked packets for two consecutive spin-bit periods; when the client transmits a packet and a "generation token" is available, the client marks the packet and retires a "generation token". If no token is available, the outgoing packet is transmitted unmarked. At the end of the first spin-bit period spent in generation, the reflection counter is unlocked to start counting incoming marked packets that will be reflected later;
2. Pause Phase. When the generation is completed, the client pauses till it has observed one entire Spin bit period with no marked packets. That Spin bit period is used by the observer as a separator between generated and reflected packets. During this marking pause, all the outgoing packets are transmitted with T bit set to 0. The reflection counter is still incremented every time a marked packet arrives;
3. Reflection Phase. The client starts transmitting marked packets, decrementing the reflection counter for each transmitted marked packet until the reflection counter reached zero. The "generation token" method from the generation phase is used during this phase as well. At the end of the first spin-period spent in reflection, the reflection counter is locked to avoid incoming reflected packets incrementing it;
4. Pause Phase 2. The pause phase is repeated after the reflection phase and serves as a separator between the reflected packet train and a new packet train.

The generation token counter should be capped to limit the effects of a subsequent sudden reduction in the other endpoint's packet rate that could prevent that endpoint from reflecting collected packets. The most conservative cap value is "1".

A server maintains a "marking counter" that starts at zero and is incremented every time a marked packet arrives. When the server transmits a packet and the "marking counter" is positive, the server marks the packet and decrements the "marking counter". If the

"marking counter" is zero, the outgoing packet is transmitted unmarked.

Note that a choice of 2-RTT (two spin periods) for the generation phase is a tradeoff between the percentage of marked packets (i.e. the percentage of traffic monitored) and the measurement delay. Using this value the algorithm produces a measurement approximately every 6-RTT ("2" generation, "2" reflection, "2" pauses), marking " $\sim 1/3$ " of packets exchanged in the slower direction (see Section 3.1.4). Choosing a generation phase of 1-RTT, we would produce measurements every 4-RTT, monitoring just " $\sim 1/4$ " of packets in the slower direction.

3.1.3. Observer's Logic for Round Trip Loss Signal

The on-path observer counts marked packets and separates different trains by detecting spin-bit periods (at least one) with no marked packets. The Round Trip Packet Loss (RTPL) is the difference between the size of the Generation train and the Reflection train.

In the following example, packets are represented by two bits (first one is the Spin bit, second one is the round Trip loss bit):

Generation	Pause	Reflection	Pause
01 01 00 01 11 10 11	00 00 10 10 10	01 00 01 01 10 11 10	00 00 10

Round Trip Loss signal example

Note that 5 marked packets have been generated of which 4 have been reflected.

3.1.4. Loss Coverage and Signal Timing

A cycle of the round Trip loss signaling algorithm contains 2 RTTs of Generation phase, 2 RTTs of Reflection phase, and two Pause phases at least 1 RTT in duration each. Hence, the loss signal is delayed by about 6 RTTs since the loss events.

The observer can only detect loss of marked packets that occurs after its initial observation of the Generation phase and before its subsequent observation of the Reflection phase. Hence, if the loss occurs on the path that sends packets at a lower rate (typically ACKs in such asymmetric scenarios), " $2/6$ " (" $1/3$ ") of the packets will be sampled for loss detection.

If the loss occurs on the path that sends packets at a higher rate, " $\text{lowPacketRate}/(3*\text{highPacketRate})$ " of the packets will be sampled for loss detection. For protocols that use ACKs, the portion of packets sampled for loss in the higher rate direction during unidirectional data transfer is " $1/(3*\text{packetsPerAck})$ ", where the value of " packetsPerAck " can vary by protocol, by implementation, and by network conditions.

3.2. Q Bit - sQuare Bit

The sQuare bit (Q bit) takes its name from the square wave generated by its signal. Every outgoing packet contains the Q bit value, which is initialized to the 0 and inverted after sending N packets (a sQuare Block or simply Q Block). Hence, Q Period is $2*N$. The Q bit represents "packet color" as defined by [AltMark].

Observation points can estimate upstream losses by watching a single direction of the traffic flow and counting the number of packets in each observed Q Block, as described in Section 3.2.2.

3.2.1. Q Block Length Selection

The length of the block must be known to the on-path network probes. There are two alternatives to selecting the Q Block length. The first one requires that the length is known a priori and therefore set within the protocol specifications that implements the marking mechanism. The second requires the sender to select it.

In this latter scenario, the sender is expected to choose N (Q Block length) based on the expected amount of loss and reordering on the path. The choice of N strikes a compromise - the observation could become too unreliable in case of packet reordering and/or severe loss if N is too small, while short flows may not yield a useful upstream loss measurement if N is too large (see Section 3.2.2).

The value of N should be at least 64 and be a power of 2. This requirement allows an Observer to infer the Q Block length by observing one period of the square signal. It also allows the Observer to identify flows that set the loss bits to arbitrary values (see Section 5).

If the sender does not have sufficient information to make an informed decision about Q Block length, the sender should use $N=64$, since this value has been extensively tried in large-scale field tests and yielded good results. Alternatively, the sender may also choose a random power-of-2 N for each flow, increasing the chances of using a Q Block length that gives the best signal for some flows.

The sender must keep the value of N constant for a given flow.

3.2.2. Upstream Loss

Blocks of N (Q Block length) consecutive packets are sent with the same value of the Q bit, followed by another block of N packets with an inverted value of the Q bit. Hence, knowing the value of N , an on-path observer can estimate the amount of upstream loss after observing at least N packets. The upstream loss rate ("uloss") is one minus the average number of packets in a block of packets with the same Q value (" p ") divided by N (" $uloss=1-avg(p)/N$ ").

The observer needs to be able to tolerate packet reordering that can blur the edges of the square signal, as explained in Section 3.2.3.

```

=====>
*****      -----Obs----->      *****
* Client *
*****      <-----          *****

```

(a) in client-server channel (uloss_up)

```

*****      ----->      *****
* Client *
*****      <----Obs----- *****
               <=====

```

(b) in server-client channel (uloss_down)

Upstream loss

3.2.3. Identifying Q Block Boundaries

Packet reordering can produce spurious edges in the square signal. To address this, the observer should look for packets with the current Q bit value up to X packets past the first packet with a reverse Q bit value. The value of X , a "Marking Block Threshold", should be less than " $N/2$ ".

The choice of X represents a trade-off between resiliency to reordering and resiliency to loss. A very large Marking Block Threshold will be able to reconstruct Q Blocks despite a significant amount of reordring, but it may erroneously coalesce packets from multiple Q Blocks into fewer Q Blocks, if loss exceeds 50% for some Q Blocks.

3.2.3.1. Improved Resilience to Burst Losses

Burst losses can affect Q measurements accuracy. Generally, burst losses can be absorbed and correctly measured if smaller than the established Q Block length. If entire Q Block length of packets get lost in a burst, however, the observer may be left completely unaware of the loss.

To improve burst loss resilience, an observer may consider a received Q Block larger than the selected Q Block length as an indication of a burst loss event. The observer would then compute the loss as three times Q Block length minus the measured block length. By doing so, the observer can detect burst losses of less than two blocks (e.g., less than 128 packets for Q Block length of 64 packets). A burst loss of two or more consecutive periods would still remain unnoticed by the observer (or underestimated if a period longer than Q Block length were formed).

3.3. L Bit - Loss Event Bit

The Loss Event bit uses an Unreported Loss counter maintained by the protocol that implements the marking mechanism. To use the Loss Event bit, the protocol must allow the sender to identify lost packets. This is true of protocols such as QUIC, partially true for TCP and SCTP (losses of pure ACKs are not detected) and is not true of protocols such as UDP and IP/IPv6.

The Unreported Loss counter is initialized to 0, and L bit of every outgoing packet indicates whether the Unreported Loss counter is positive (L=1 if the counter is positive, and L=0 otherwise).

The value of the Unreported Loss counter is decremented every time a packet with L=1 is sent.

The value of the Unreported Loss counter is incremented for every packet that the protocol declares lost, using whatever loss detection machinery the protocol employs. If the protocol is able to rescind the loss determination later, a positive Unreported Loss counter may be decremented due to the rescission, but it should not become negative due to the rescission.

This loss signaling is similar to loss signaling in [ConEx], except the Loss Event bit is reporting the exact number of lost packets, whereas Echo Loss bit in [ConEx] is reporting an approximate number of lost bytes.

For protocols, such as TCP ([TCP]), that allow network devices to change data segmentation, it is possible that only a part of the

packet is lost. In these cases, the sender must increment Unreported Loss counter by the fraction of the packet data lost (so Unreported Loss counter may become negative when a packet with $L=1$ is sent after a partial packet has been lost).

Observation points can estimate the end-to-end loss, as determined by the upstream endpoint, by counting packets in this direction with the L bit equal to 1, as described in Section 3.3.1.

3.3.1. End-To-End Loss

The Loss Event bit allows an observer to estimate the end-to-end loss rate by counting packets with L bit value of 0 and 1 for a given flow. The end-to-end loss rate is the fraction of packets with $L=1$.

The assumption here is that upstream loss affects packets with $L=0$ and $L=1$ equally. If some loss is caused by tail-drop in a network device, this may be a simplification. If the sender's congestion controller reduces the packet send rate after loss, there may be a sufficient delay before sending packets with $L=1$ that they have a greater chance of arriving at the observer.

3.3.1.1. Loss Profile Characterization

The Loss Event bit allows an observer to characterize loss profile, since the distribution of observed packets with L bit set to 1 roughly corresponds to the distribution of packets lost between 1 RTT and 1 RTO before (see Section 3.3.2.1). Hence, observing random single instances of L bit set to 1 indicates random single packet loss, while observing blocks of packets with L bit set to 1 indicates loss affecting entire blocks of packets.

3.3.2. $L+Q$ Bits - Loss Measurement Using L and Q Bits

Combining L and Q bits allows a passive observer watching a single direction of traffic to accurately measure:

- upstream loss: sender-to-observer loss (see Section 3.2.2)
- downstream loss: observer-to-receiver loss (see Section 3.3.2.2)
- end-to-end loss: sender-to-receiver loss on the observed path (see Section 3.3.1) with loss profile characterization (see Section 3.3.1.1)

3.3.2.1. Correlating End-to-End and Upstream Loss

Upstream loss is calculated by observing packets that did not suffer the upstream loss (Section 3.2.2). End-to-end loss, however, is calculated by observing subsequent packets after the sender's protocol detected the loss. Hence, end-to-end loss is generally observed with a delay of between 1 RTT (loss declared due to multiple duplicate acknowledgments) and 1 RTO (loss declared due to a timeout) relative to the upstream loss.

The flow RTT can sometimes be estimated by timing protocol handshake messages. This RTT estimate can be greatly improved by observing a dedicated protocol mechanism for conveying RTT information, such as the Spin bit (see Section 2.1) or Delay bit (see Section 2.2).

Whenever the observer needs to perform a computation that uses both upstream and end-to-end loss rate measurements, it should use upstream loss rate leading the end-to-end loss rate by approximately 1 RTT. If the observer is unable to estimate RTT of the flow, it should accumulate loss measurements over time periods of at least 4 times the typical RTT for the observed flows.

If the calculated upstream loss rate exceeds the end-to-end loss rate calculated in Section 3.3.1, then either the Q Period is too short for the amount of packet reordering or there is observer loss, described in Section 3.3.2.3. If this happens, the observer should adjust the calculated upstream loss rate to match end-to-end loss rate, unless the following applies.

In case of a protocol, such as TCP or SCTP, that does not track losses of pure ACK packets, observing a direction of traffic dominated by pure ACK packets could result in measured upstream loss that is higher than measured end-to-end loss, if said pure ACK packets are lost upstream. Hence, if the measurement is applied to such protocols, and the observer can confirm that pure ACK packets dominate the observed traffic direction, the observer should adjust the calculated end-to-end loss rate to match upstream loss rate.

3.3.2.2. Downstream Loss

Because downstream loss affects only those packets that did not suffer upstream loss, the end-to-end loss rate ("eloss") relates to the upstream loss rate ("uloss") and downstream loss rate ("dloss") as $(1-uloss)(1-dloss)=1-eloss$. Hence, $dloss=(eloss-uloss)/(1-uloss)$.

3.3.2.3. Observer Loss

A typical deployment of a passive observation system includes a network tap device that mirrors network packets of interest to a device that performs analysis and measurement on the mirrored packets. The observer loss is the loss that occurs on the mirror path.

Observer loss affects upstream loss rate measurement, since it causes the observer to account for fewer packets in a block of identical Q bit values (see Section 3.2.2). The end-to-end loss rate measurement, however, is unaffected by the observer loss, since it is a measurement of the fraction of packets with the L bit value of 1, and the observer loss would affect all packets equally (see Section 3.3.1).

The need to adjust the upstream loss rate down to match end-to-end loss rate as described in Section 3.3.2.1 is an indication of the observer loss, whose magnitude is between the amount of such adjustment and the entirety of the upstream loss measured in Section 3.2.2. Alternatively, a high apparent upstream loss rate could be an indication of significant packet reordering, possibly due to packets belonging to a single flow being multiplexed over several upstream paths with different latency characteristics.

3.4. R Bit - Reflection Square Bit

R bit requires a deployment alongside Q bit. Unlike the square signal for which packets are transmitted in blocks of fixed size, the number of packets in Reflection square signal blocks (also an alternate marking signal) varies according to these rules:

- when the transmission of a new block starts, its size is set equal to the size of the last Q Block whose reception has been completed;
- if, before transmission of the block is terminated, the reception of at least one further Q Block is completed, the size of the block is updated to be the average size of the further received Q Blocks.

The Reflection square value is initialized to 0 and is applied to the R bit of every outgoing packet. The Reflection square value is toggled for the first time when the completion of a Q Block is detected in the incoming square signal (produced by the other endpoint using the Q bit). The number of packets detected within this first Q Block ("p"), is used to generate a reflection square signal that toggles every "M=p" packets (at first). This new signal

produces blocks of M packets (marked using the R bit) and each of them is called "Reflection Block" (R Block).

The M value is then updated every time a completed Q Block in the incoming square signal is received, following this formula:
"M=round(avg(p))".

The parameter "avg(p)", the average number of packets in a marking period, is computed based on all the Q Blocks received since the beginning of the current R Block.

The transmission of an R Block is considered complete (and the signal toggled) when the number of packets transmitted in that block is at least the latest computed M value.

To ensure a proper computation of the M value, endpoints implementing the R bit must identify the boundaries of incoming Q Blocks. The same approach described in Section 3.2.3 should be used.

Looking at the R bit, unidirectional observation points have an indication of loss experienced by the entire unobserved channel plus the loss on the path from the sender to them.

Since the Q Block is sent in one direction, and the corresponding reflected R Block is sent in the opposite direction, the reflected R signal is transmitted with the packet rate of the slowest direction. Namely, if the observed direction is the slowest, there can be multiple Q Blocks transmitted in the unobserved direction before a complete R Block is transmitted in the observed direction. If the unobserved direction is the slowest, the observed direction can be sending R Blocks of the same size repeatedly before it can update the signal to account for a newly-completed Q Block.

3.4.1. Enhancement of R Block Length Computation

The use of the rounding function used in the M computation introduces errors that can be minimized by storing the rounding applied each time M is computed, and using it during the computation of the M value in the following R Block.

This can be achieved introducing the new "r_avg" parameter in the computation of M. The new formula is "Mr=avg(p)+r_avg; M=round(Mr); r_avg=Mr-M" where the initial value of "r_avg" is equal to 0.

3.4.2. Improved Resilience to Packet Reordering

When a protocol implementing the marking mechanism is able to detect when packets are received out of order, it can improve resilience to packet reordering beyond what is possible using methods described in Section 3.2.3.

This can be achieved by updating the size of the current R Block while it is being transmitted. The reflection block size is then updated every time an incoming reordered packet of the previous Q Block is detected. This can be done if and only if the transmission of the current reflection block is in progress and no packets of the following Q Block have been received.

3.4.2.1. Improved Resilience to Burst Losses

Burst losses can affect R measurements accuracy similarly to how they affect Q measurements accuracy. Therefore, recommendations in section Section 3.2.3.1 apply equally to improving burst loss resilience for R measurements.

3.4.3. R+Q Bits - Loss Measurement Using R and Q Bits

Since both sSquare and Reflection square bits are toggled at most every N packets (except for the first transition of the R bit as explained before), an on-path observer can count the number of packets of each marking block and, knowing the value of N, can estimate the amount of loss experienced by the connection. An observer can calculate different measurements depending on whether it is able to observe a single direction of the traffic or both directions.

Single directional observer:

- upstream loss in the observed direction: the loss between the sender and the observation point (see Section 3.2.2)
- "three-quarters" connection loss: the loss between the receiver and the sender in the unobserved direction plus the loss between the sender and the observation point in the observed direction
- end-to-end loss in the unobserved direction: the loss between the receiver and the sender in the opposite direction

Two directions observer (same metrics seen previously applied to both direction, plus):

- client-observer half round-trip loss: the loss between the client and the observation point in both directions
- observer-server half round-trip loss: the loss between the observation point and the server in both directions
- downstream loss: the loss between the observation point and the receiver (applicable to both directions)

3.4.3.1. Three-Quarters Connection Loss

Except for the very first block in which there is nothing to reflect (a complete Q Block has not been yet received), packets are continuously R-bit marked into alternate blocks of size lower or equal than N. Knowing the value of N, an on-path observer can estimate the amount of loss occurred in the whole opposite channel plus the loss from the sender up to it in the observation channel. As for the previous metric, the "three-quarters" connection loss rate ("tqloss") is one minus the average number of packets in a block of packets with the same R value ("t") divided by "N" ("tqloss=1-avg(t)/N").

```

=====>
= *****      -----Obs----->      *****
= * Client *
= *****      <-----              *****
<=====

```

(a) in client-server channel (tqloss_up)

```

=====>
*****      ----->      ***** =
* Client *
*****      <----Obs----- ***** =
<=====

```

(b) in server-client channel (tqloss_down)

Three-quarters connection loss

The following metrics derive from this last metric and the upstream loss produced by the Q bit.

3.4.3.2. End-To-End Loss in the Opposite Direction

End-to-end loss in the unobserved direction ("eloss_unobserved") relates to the "three-quarters" connection loss ("tqloss") and upstream loss in the observed direction ("uloss") as

"(1-eloss_unobserved) (1-uloss)=1-tqloss". Hence,
 "eloss_unobserved=(tqloss-uloss)/(1-uloss)".

```

*****      -----Obs----->      *****
* Client *                               * Server *
*****      <-----              *****
<=====

```

(a) in client-server channel (eloss_down)

```

=====>
*****      ----->      *****
* Client *                               * Server *
*****      <----Obs----- *****

```

(b) in server-client channel (eloss_up)

End-To-End loss in the opposite direction

3.4.3.3. Half Round-Trip Loss

If the observer is able to observe both directions of traffic, it is able to calculate two "half round-trip" loss measurements - loss from the observer to the receiver (in a given direction) and then back to the observer in the opposite direction. For both directions, "half round-trip" loss ("hrtloss") relates to "three-quarters" connection loss ("tqloss_opposite") measured in the opposite direction and the upstream loss ("uloss") measured in the given direction as
 "(1-uloss) (1-hrtloss)=1-tqloss_opposite". Hence,
 "hrtloss=(tqloss_opposite-uloss)/(1-uloss)".

```

=====>
= *****      -----|----->      *****
= * Client *           Obs          * Server *
= *****      <-----|-----      *****
<=====

```

(a) client-observer half round-trip loss (hrtloss_co)

```

=====>
*****      -----|----->      ***** =
* Client *           Obs          * Server * =
*****      <-----|-----      ***** =
<=====

```

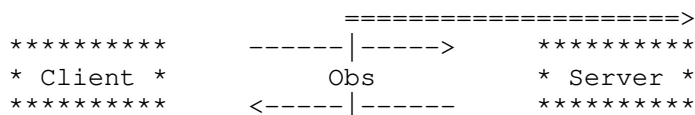
(b) observer-server half round-trip loss (hrtloss_os)

Half Round-trip loss (both direction)

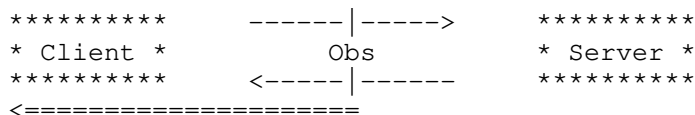
3.4.3.4. Downstream Loss

If the observer is able to observe both directions of traffic, it is able to calculate two downstream loss measurements using either end-to-end loss and upstream loss, similar to the calculation in Section 3.3.2.2 or using "half round-trip" loss and upstream loss in the opposite direction.

For the latter, "dloss=(hrtloss-uloss_opposite)/(1-uloss_opposite)".



(a) in client-server channel (dloss_up)



(b) in server-client channel (dloss_down)

Downstream loss

3.5. E Bit - ECN-Echo Event Bit

While the primary focus of this draft is on exposing packet loss and delay, modern networks can report congestion before they are forced to drop packets, as described in [ECN]. When transport protocols keep ECN-Echo feedback under encryption, this signal cannot be observed by the network operators. When tasked with diagnosing network performance problems, knowledge of a congestion downstream of an observation point can be instrumental.

If downstream congestion information is desired, this information can be signaled with an additional bit.

- E: The "ECN-Echo Event" bit is set to 0 or 1 according to the Unreported ECN Echo counter, as explained below in Section 3.5.1.

3.5.1. Setting the ECN-Echo Event Bit on Outgoing Packets

The Unreported ECN-Echo counter operates identically to Unreported Loss counter (Section 3.3), except it counts packets delivered by the network with CE markings, according to the ECN-Echo feedback from the receiver.

This ECN-Echo signaling is similar to ECN signaling in [ConEx]. ECN-Echo mechanism in QUIC provides the number of packets received with CE marks. For protocols like TCP, the method described in [ConEx-TCP] can be employed. As stated in [ConEx-TCP], such feedback can be further improved using a method described in [ACCURATE].

3.5.2. Using E Bit for Passive ECN-Reported Congestion Measurement

A network observer can count packets with CE codepoint and determine the upstream CE-marking rate directly.

Observation points can also estimate ECN-reported end-to-end congestion by counting packets in this direction with a E bit equal to 1.

The upstream CE-marking rate and end-to-end ECN-reported congestion can provide information about downstream CE-marking rate. Presence of E bits along with L bits, however, can somewhat confound precise estimates of upstream and downstream CE-markings in case the flow contains packets that are not ECN-capable.

4. Summary of Delay and Loss Marking Methods

This section summarizes the marking methods described in this draft.

For the Delay measurement, it is possible to use the Spin bit and/or the delay bit. A unidirectional or bidirectional observer can be used.

Method	# of bits	Available Delay Metrics		Impairments Resiliency	# of meas.
		UniDir Observer	BiDir Observer		
S: Spin Bit	1	RTT	x2 Half RTT	low	very high
D: Delay Bit	1	RTT	x2 Half RTT	high	medium
D [^] : Hidden Delay Bit	1	RTT [^]	x2 Left Half [^] Right Half	high	high
SD: Spin Bit & Delay Bit *	2	RTT	x2 Half RTT	high	very high

x2 Same metric for both directions

* Both bits work independently; an observer could use less accurate Spin bit measurements when Delay bit ones are unavailable

[^] Masked metric (real value can be calculated only by those who know the Additional Delay)

Figure 1: Delay Comparison

For the Loss measurement, each row in the table of Figure 2 represents a loss marking method. For each method the table specifies the number of bits required in the header, the available metrics using an unidirectional or bidirectional observer, applicable protocols, measurement fidelity and delay.

Method	Bits	Available Loss Metrics		Protocols	Measurement Aspects	
		UniDir Observer	BiDir Observer		Fidelity	Delay
T: Round Trip Loss Bit	\$ 1	RT	x2 Half RT	*	Rate by sampling 1/3 to 1/(3*ppa) of pkts over 2 RTT	~6 RTT
Q: sQuare Bit	1	Upstream	x2	*	Rate over N pkts (e.g. 64)	N pkts (e.g. 64)
L: Loss Event Bit	1	E2E	x2	#	Loss shape (and rate)	Min: RTT Max: RTO
QL: sQuare + Loss Ev. Bits	2	Upstream Downstream E2E	x2 x2 x2	#	-> see Q -> see Q L -> see L	Up: see Q Others: see L
QR: sQuare + Ref. Sq. Bits	2	Upstream 3/4 RT !E2E	x2 x2 E2E Downstream Half RT	*	Rate over N*ppa pkts (see Q bit for N)	Up: see Q Others: N*ppa pk (see Q for N)

* All protocols

Protocols employing loss detection (with or without pure ACK loss detection)

\$ Require a working Spin bit

! Metric relative to the opposite channel

x2 Same metric for both directions

ppa Packets-Per-Ack

Q|L See Q if Upstream loss is significant; L otherwise

Figure 2: Loss Comparison

5. Protocol Ossification Considerations

Accurate loss and delay information is not critical to the operation of any protocol, though its presence for a sufficient number of flows is important for the operation of networks.

The delay and loss bits are amenable to "greasing" described in [RFC8701], if the protocol designers are not ready to dedicate (and

ossify) bits used for loss reporting to this function. The greasing could be accomplished similarly to the Latency Spin bit greasing in [QUIC-TRANSPORT]. Namely, implementations could decide that a fraction of flows should not encode loss and delay information and, instead, the bits would be set to arbitrary values. The observers would need to be ready to ignore flows with delay and loss information more resembling noise than the expected signal.

6. Examples of Application

6.1. QUIC

The binding of a delay signal to QUIC is partially described in [QUIC-TRANSPORT], which adds the Spin bit to the first byte of the short packet header, leaving two reserved bits for future use.

To implement the additional signals discussed in this document, the first byte of the short packet header can be modified as follows:

- the Delay bit (D) can be placed in the first reserved bit (i.e. the fourth most significant bit `_0x10_`) while the round trip loss bit (T) in the second reserved bit (i.e. the fifth most significant bit `_0x08_`); the proposed scheme is:

```

  0 1 2 3 4 5 6 7
+---+---+---+---+
|0|1|S|D|T|K|P|P|
+---+---+---+---+
```

Scheme 1

- alternatively, a two bits loss signal (QL or QR) can be placed in both reserved bits; the proposed schemes, in this case, are:

```

  0 1 2 3 4 5 6 7
+---+---+---+---+
|0|1|S|Q|L|K|P|P|
+---+---+---+---+
```

Scheme 2A

```

  0 1 2 3 4 5 6 7
+---+---+---+---+
|0|1|S|Q|R|K|P|P|
+---+---+---+---+
```

Scheme 2B

A further option would be to substitute the Spin bit with the Delay bit (or hidden Delay bit), leaving the two reserved bits for loss detection. The proposed schemes are:

0 1 2 3 4 5 6 7		0 1 2 3 4 5 6 7
+--+--+--+--+--+--+		+--+--+--+--+--+--+
0 1 D Q L K P P	OR	0 1 D^ Q L K P P
+--+--+--+--+--+--+		+--+--+--+--+--+--+

Scheme 3A

0 1 2 3 4 5 6 7		0 1 2 3 4 5 6 7
+--+--+--+--+--+--+		+--+--+--+--+--+--+
0 1 D Q R K P P	OR	0 1 D^ Q R K P P
+--+--+--+--+--+--+		+--+--+--+--+--+--+

Scheme 3B

6.2. TCP

The signals can be added to TCP by defining bit 4 of byte 13 of the TCP header to carry the Spin bit or the Delay bit, and possibly bits 5 and 6 to carry additional information, like the Delay bit and the round-trip loss bit (DT), or a two bits loss signal (QL or QR).

7. Security Considerations

Passive loss and delay observations have been a part of the network operations for a long time, so exposing loss and delay information to the network does not add new security concerns for protocols that are currently observable.

In the absence of packet loss, Q and R bits signals do not provide any information that cannot be observed by simply counting packets transiting a network path. In the presence of packet loss, Q and R bits will disclose the loss, but this is information about the environment and not the endpoint state. The L bit signal discloses internal state of the protocol's loss detection machinery, but this state can often be gleamed by timing packets and observing congestion controller response.

Hence, loss bits do not provide a viable new mechanism to attack data integrity and secrecy.

The described techniques can generally apply to different communication protocols operating in different security environments. An implementation of these techniques for a particular protocol must consider specifics of the protocol and its expected operating

environment. For example, security considerations for QUIC, discussed in [QUIC-TRANSPORT] and [QUIC-TLS], consider a possibility of active and passive attackers in the network as well as attacks on specific QUIC mechanisms.

7.1. Optimistic ACK Attack

A defense against an Optimistic ACK Attack, described in [QUIC-TRANSPORT], involves a sender randomly skipping packet numbers to detect a receiver acknowledging packet numbers that have never been received. The Q bit signal may inform the attacker which packet numbers were skipped on purpose and which had been actually lost (and are, therefore, safe for the attacker to acknowledge). To use the Q bit for this purpose, the attacker must first receive at least an entire Q Block of packets, which renders the attack ineffective against a delay-sensitive congestion controller.

A protocol that is more susceptible to an Optimistic ACK Attack with the loss signal provided by Q bit and uses a loss-based congestion controller, should shorten the current Q Block by the number of skipped packets numbers. For example, skipping a single packet number will invert the square signal one outgoing packet sooner.

Similar considerations apply to the R bit, although a shortened R Block along with a matching skip in packet numbers does not necessarily imply a lost packet, since it could be due to a lost packet on the reverse path along with a deliberately skipped packet by the sender.

8. Privacy Considerations

To minimize unintentional exposure of information, loss bits provide an explicit loss signal - a preferred way to share information per [RFC8558].

New protocols commonly have specific privacy goals, and loss reporting must ensure that loss information does not compromise those privacy goals. For example, [QUIC-TRANSPORT] allows changing Connection IDs in the middle of a connection to reduce the likelihood of a passive observer linking old and new sub-flows to the same device. A QUIC implementation would need to reset all counters when it changes the destination (IP address or UDP port) or the Connection ID used for outgoing packets. It would also need to avoid incrementing Unreported Loss counter for loss of packets sent to a different destination or with a different Connection ID.

9. IANA Considerations

This document makes no request of IANA.

10. Contributors

The following people provided valuable contributions to this document:

- Marcus Ihlar, Ericsson, marcus.ihlar@ericsson.com
- Jari Arkko, Ericsson, jari.arkko@ericsson.com
- Emile Stephan, Orange, emile.stephan@orange.com

11. Acknowledgements

The authors would like to thank the QUIC WG for their contributions, Christian Huitema for implementing Q and L bits in his picoquic stack, and Ike Kunze for providing constructive reviews and helpful suggestions.

12. References

12.1. Normative References

- [ConEx] Mathis, M. and B. Briscoe, "Congestion Exposure (ConEx) Concepts, Abstract Mechanism, and Requirements", RFC 7713, DOI 10.17487/RFC7713, December 2015, <<https://www.rfc-editor.org/info/rfc7713>>.
- [ConEx-TCP] Kuehlewind, M., Ed. and R. Scheffenegger, "TCP Modifications for Congestion Exposure (ConEx)", RFC 7786, DOI 10.17487/RFC7786, May 2016, <<https://www.rfc-editor.org/info/rfc7786>>.
- [ECN] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, DOI 10.17487/RFC3168, September 2001, <<https://www.rfc-editor.org/info/rfc3168>>.
- [IP] Postel, J., "Internet Protocol", STD 5, RFC 791, DOI 10.17487/RFC0791, September 1981, <<https://www.rfc-editor.org/info/rfc791>>.

[IPM-Methods]

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

[IPv6]

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

[RFC8558]

Hardie, T., Ed., "Transport Protocol Path Signals", RFC 8558, DOI 10.17487/RFC8558, April 2019, <<https://www.rfc-editor.org/info/rfc8558>>.

[TCP]

Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <<https://www.rfc-editor.org/info/rfc793>>.

12.2. Informative References

[ACCURATE]

Briscoe, B., Kuehlewind, M., and R. Scheffenegger, "More Accurate ECN Feedback in TCP", draft-ietf-tcpm-accurate-ecn-18 (work in progress), March 2022.

[AltMark]

Fioccola, G., Cociglio, M., Mirsky, G., Mizrahi, T., and T. Zhou, "Alternate-Marking Method", draft-ietf-ippm-rfc8321bis-01 (work in progress), April 2022.

[ANRW19-PM-QUIC]

Bulgarella, F., Cociglio, M., Fioccola, G., Marchetto, G., and R. Sisto, "Performance measurements of QUIC communications", Proceedings of the Applied Networking Research Workshop, DOI 10.1145/3340301.3341127, July 2019.

[I-D.trammell-ippm-spin]

Trammell, B., "An Explicit Transport-Layer Signal for Hybrid RTT Measurement", draft-trammell-ippm-spin-00 (work in progress), January 2019.

[I-D.trammell-tsvwg-spin]

Trammell, B., "A Transport-Independent Explicit Signal for Hybrid RTT Measurement", draft-trammell-tsvwg-spin-00 (work in progress), July 2018.

[IPv6AltMark]

Fioccola, G., Zhou, T., Cociglio, M., Qin, F., and R. Pang, "IPv6 Application of the Alternate Marking Method", draft-ietf-6man-ipv6-alt-mark-14 (work in progress), April 2022.

[QUIC-TLS]

Thomson, M., Ed. and S. Turner, Ed., "Using TLS to Secure QUIC", RFC 9001, DOI 10.17487/RFC9001, May 2021, <<https://www.rfc-editor.org/info/rfc9001>>.

[QUIC-TRANSPORT]

Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <<https://www.rfc-editor.org/info/rfc9000>>.

[RFC8517] Dolson, D., Ed., Snellman, J., Boucadair, M., Ed., and C. Jacquenet, "An Inventory of Transport-Centric Functions Provided by Middleboxes: An Operator Perspective", RFC 8517, DOI 10.17487/RFC8517, February 2019, <<https://www.rfc-editor.org/info/rfc8517>>.

[RFC8701] Benjamin, D., "Applying Generate Random Extensions And Sustain Extensibility (GREASE) to TLS Extensibility", RFC 8701, DOI 10.17487/RFC8701, January 2020, <<https://www.rfc-editor.org/info/rfc8701>>.

[RFC9065] Fairhurst, G. and C. Perkins, "Considerations around Transport Header Confidentiality, Network Operations, and the Evolution of Internet Transport Protocols", RFC 9065, DOI 10.17487/RFC9065, July 2021, <<https://www.rfc-editor.org/info/rfc9065>>.

[SPIN-BIT]

Kuehlewind, M. and B. Trammell, "Manageability of the QUIC Transport Protocol", draft-ietf-quic-manageability-16 (work in progress), April 2022.

[UDP-OPTIONS]

Touch, J., "Transport Options for UDP", draft-ietf-tsvwg-udp-options-18 (work in progress), March 2022.

[UDP-SURPLUS]

Herbert, T., "UDP Surplus Header", draft-herbert-udp-space-hdr-01 (work in progress), July 2019.

Authors' Addresses

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

EMail: mauro.cociglio@telecomitalia.it

Alexandre Ferrieux
Orange Labs

EMail: alexandre.ferrieux@orange.com

Giuseppe Fioccola
Huawei Technologies
Riesstrasse, 25
Munich 80992
Germany

EMail: giuseppe.fioccola@huawei.com

Igor Lubashev
Akamai Technologies

EMail: ilubashe@akamai.com

Fabio Bulgarella
Telecom Italia - TIM
Via Reiss Romoli, 274
Torino 10148
Italy

EMail: fabio.bulgarella@guest.telecomitalia.it

Isabelle Hamchaoui
Orange Labs

EMail: isabelle.hamchaoui@orange.com

Massimo Nilo
Telecom Italia - TIM
Via Reiss Romoli, 274
Torino 10148
Italy

Email: massimo.nilo@telecomitalia.it

Riccardo Sisto
Politecnico di Torino

Email: riccardo.sisto@polito.it

Dmitri Tikhonov
LiteSpeed Technologies

Email: dtikhonov@litespeedtech.com

IPPM Working Group
Internet-Draft
Intended status: Standards Track
Expires: 30 July 2022

X. Min
ZTE Corp.
G. Mirsky
Ericsson
L. Bo
China Telecom
26 January 2022

Echo Request/Reply for Enabled In-situ OAM Capabilities
draft-ietf-ippm-ioam-conf-state-03

Abstract

This document describes an extension to the echo request/reply mechanisms used in IPv6 (including SRv6), MPLS (including SR-MPLS), SFC and BIER environments, which can be used within the IOAM domain, allowing the IOAM encapsulating node to discover the enabled IOAM capabilities of each IOAM transit and IOAM decapsulating node.

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 30 July 2022.

Copyright Notice

Copyright (c) 2022 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 Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction	2
2. Conventions	4
2.1. Requirements Language	4
2.2. Abbreviations	5
3. IOAM Capabilities Formats	5
3.1. IOAM Capabilities Query Container	6
3.2. IOAM Capabilities Response Container	6
3.2.1. IOAM Pre-allocated Tracing Capabilities Object	7
3.2.2. IOAM Incremental Tracing Capabilities Object	8
3.2.3. IOAM Proof-of-Transit Capabilities Object	9
3.2.4. IOAM Edge-to-Edge Capabilities Object	10
3.2.5. IOAM DEX Capabilities Object	11
3.2.6. IOAM End-of-Domain Object	12
4. Operational Guide	13
5. IANA Considerations	13
5.1. IOAM SoP Capability Registry	14
5.2. IOAM TSF Capability Registry	14
6. Security Considerations	15
7. Acknowledgements	15
8. References	15
8.1. Normative References	15
8.2. Informative References	16
Authors' Addresses	17

1. Introduction

In-situ OAM (IOAM) ([I-D.ietf-ippm-ioam-data] [I-D.ietf-ippm-ioam-direct-export]) defines data fields that record OAM information within the packet while the packet traverses a particular network domain, called an IOAM domain. IOAM can be used to complement OAM mechanisms based on, e.g., ICMP or other types of probe packets, and IOAM mechanisms can be leveraged where mechanisms using, e.g., ICMP, do not apply or do not offer the desired results.

As specified in [I-D.ietf-ippm-ioam-data], within the IOAM domain, the IOAM data may be updated by network nodes that the packet traverses. The device which adds an IOAM header to the packet is

called an "IOAM encapsulating node". In contrast, the device which removes an IOAM header is referred to as an "IOAM decapsulating node". Nodes within the domain that are aware of IOAM data and read and/or write and/or process IOAM data are called "IOAM transit nodes". IOAM encapsulating or decapsulating nodes can also serve as IOAM transit nodes at the same time. IOAM encapsulating or decapsulating nodes are also referred to as IOAM domain edge devices, which can be hosts or network devices.

As specified in [I-D.ietf-ippm-ioam-data], IOAM is focused on "limited domains" as defined in [RFC8799]. In a limited domain, a control entity that has control over every IOAM device may be deployed. If that's the case, the control entity can provision both the explicit transport path and the IOAM header applied to data packet at every IOAM encapsulating node.

In a case when a control entity that has control over every IOAM device is not deployed in the IOAM domain, the IOAM encapsulating node needs to discover the enabled IOAM capabilities at the IOAM transit and decapsulating nodes. For example, what types of IOAM tracing data can be added by the transit nodes along the transport path of the data packet IOAM is applied to. The IOAM encapsulating node can then add the correct IOAM header to the data packet according to the discovered IOAM capabilities. Specifically, the IOAM encapsulating node first identifies the types and lengths of IOAM options included in the IOAM data according to the discovered IOAM capabilities. Then the IOAM encapsulating node can add the IOAM header to the data packet based on the identified types and lengths of IOAM options included in the IOAM data. The IOAM encapsulating node may use NETCONF/YANG or IGP to discover these IOAM capabilities. However, NETCONF/YANG or IGP has some limitations:

- * When NETCONF/YANG is used in this scenario, each IOAM encapsulating node (including the host when it takes the role of an IOAM encapsulating node) needs to implement a NETCONF Client, each IOAM transit and IOAM decapsulating node (including the host when it takes the role of an IOAM decapsulating node) needs to implement a NETCONF Server, the complexity can be an issue. Furthermore, each IOAM encapsulating node needs to establish NETCONF Connection with each IOAM transit and IOAM decapsulating node, the scalability can be an issue.

- * When IGP is used in this scenario, the IGP and IOAM domains don't always have the same coverage. For example, when the IOAM encapsulating node or the IOAM decapsulating node is a host, the availability can be an issue. Furthermore, it might be too challenging to reflect enabled IOAM capabilities at the IOAM transit and IOAM decapsulating node if these are controlled by a local policy depending on the identity of the IOAM encapsulating node.

This document describes an extension to the echo request/reply mechanisms used in IPv6 (including SRv6), MPLS (including SR-MPLS), SFC and BIER environments, which can be used within the IOAM domain, allowing the IOAM encapsulating node to discover the enabled IOAM capabilities of each IOAM transit and IOAM decapsulating node.

The following documents contain references to the echo request/reply mechanisms used in IPv6 (including SRv6), MPLS (including SR-MPLS), SFC and BIER environments:

- * [RFC4443] ("Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification"), [RFC4884] ("Extended ICMP to Support Multi-Part Messages") and [RFC8335] ("PROBE: A Utility for Probing Interfaces")
- * [RFC8029] ("Detecting Multiprotocol Label Switched (MPLS) Data-Plane Failures")
- * [I-D.ietf-sfc-multi-layer-oam] ("Active OAM for Service Function Chains in Networks")
- * [I-D.ietf-bier-ping] ("BIER Ping and Trace")

The precondition for the feature described in this document to work is that the echo request reaches each IOAM transit node as the data packet traverses, so the feature is applied to explicit path (strict or loose), or there is only one path between the IOAM encapsulating node and the IOAM decapsulating node, or the echo request can experience the same ECMP processing as the data packet.

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

BIER: Bit Index Explicit Replication

BGP: Border Gateway Protocol

ECMP: Equal-Cost Multipath

E2E: Edge to Edge

ICMP: Internet Control Message Protocol

IGP: Interior Gateway Protocol

IOAM: In-situ Operations, Administration, and Maintenance

LSP: Label Switched Path

MPLS: Multi-Protocol Label Switching

MBZ: Must Be Zero

MTU: Maximum Transmission Unit

NTP: Network Time Protocol

OAM: Operations, Administration, and Maintenance

PCEP: Path Computation Element (PCE) Communication Protocol

POSIX: Portable Operating System Interface

POT: Proof of Transit

PTP: Precision Time Protocol

SR-MPLS: Segment Routing with MPLS data plane

SRv6: Segment Routing with IPv6 data plane

SFC: Service Function Chain

TTL: Time to Live

3. IOAM Capabilities Formats

3.1. IOAM Capabilities Query Container

For echo request, IOAM Capabilities Query uses container which has the following format:

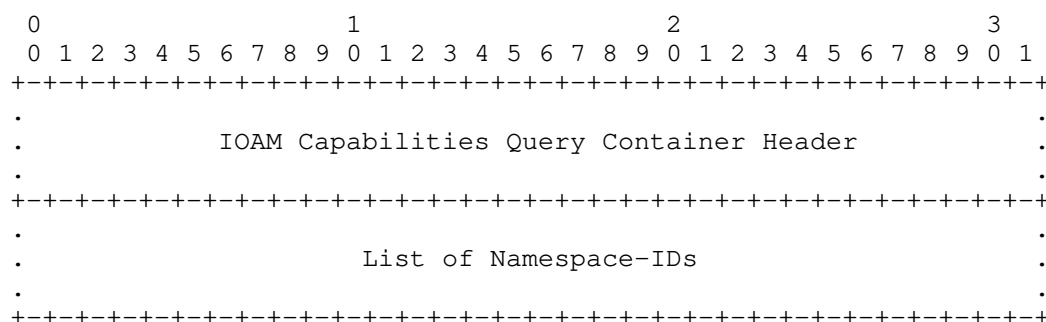


Figure 1: IOAM Capabilities Query Container of Echo Request

When this container is present in or equal to the echo request sent by an IOAM encapsulating node, that means the IOAM encapsulating node requests the receiving node to reply with its enabled IOAM capabilities. If there is no IOAM capability to be reported by the receiving node, then this container SHOULD be ignored by the receiving node, which means the receiving node SHOULD send an echo reply without IOAM capabilities or no echo reply, in the light of whether the echo request includes other containers than the IOAM Capabilities Query Container. A list of Namespace-IDs (one or more Namespace-IDs) MUST be included in this container in the echo request. The IOAM encapsulating node requests only the enabled IOAM capabilities that match one of the Namespace-IDs. The Namespace-ID has the same definition as what's specified in Section 5.3 of [I-D.ietf-ippm-ioam-data].

The IOAM Capabilities Query Container has a container header that is used to identify the type and optionally length of the container payload, and the container payload (List of Namespace-IDs) is zero-padded to align to a 4-octet boundary.

The length, structure, and definition of the IOAM Capabilities Query Container Header depends on the specific environment it is applied at.

3.2. IOAM Capabilities Response Container

For echo reply, IOAM Capabilities Response uses container which has the following format:

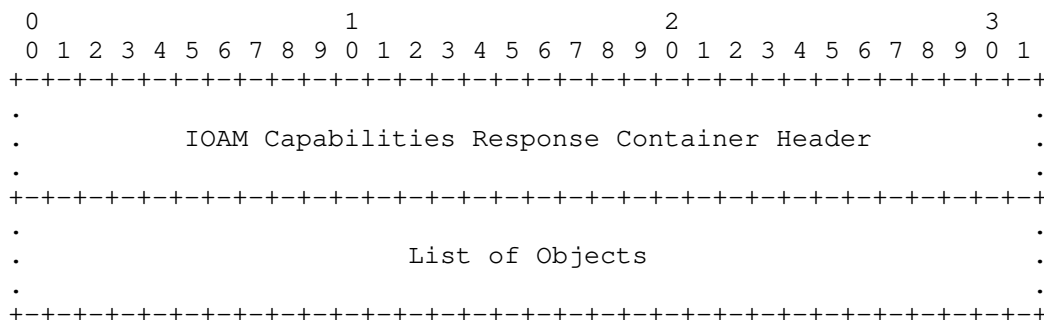


Figure 2: IOAM Capabilities Response Container of Echo Reply

When this container is present in or equal to the echo reply sent by an IOAM transit node or IOAM decapsulating node, that means the IOAM function is enabled at this node, and this container contains the enabled IOAM capabilities of the sender. A list of objects (one or more objects) which contains the enabled IOAM capabilities SHOULD be included in this container of echo reply.

The IOAM Capabilities Response Container has a container header that is used to identify the type and optionally length of the container payload, and the container payload (List of Objects) is zero-padded to align to a 4-octet boundary.

The length, structure, and definition of the IOAM Capabilities Response Container Header depends on the specific environment it is applied at.

Based on the IOAM data fields defined in [I-D.ietf-ippm-ioam-data] and [I-D.ietf-ippm-ioam-direct-export], six types of objects are defined in this document. The same type of object MAY be present in the IOAM Capabilities Response Container more than once, only if with a different Namespace-ID.

Similar to the container, each object has an object header that is used to identify the type and length of the object payload, and the object payload is zero-padded to align to a 4-octet boundary.

The length, structure, and definition of Object Header depends on the specific environment it is applied at.

3.2.1. IOAM Pre-allocated Tracing Capabilities Object

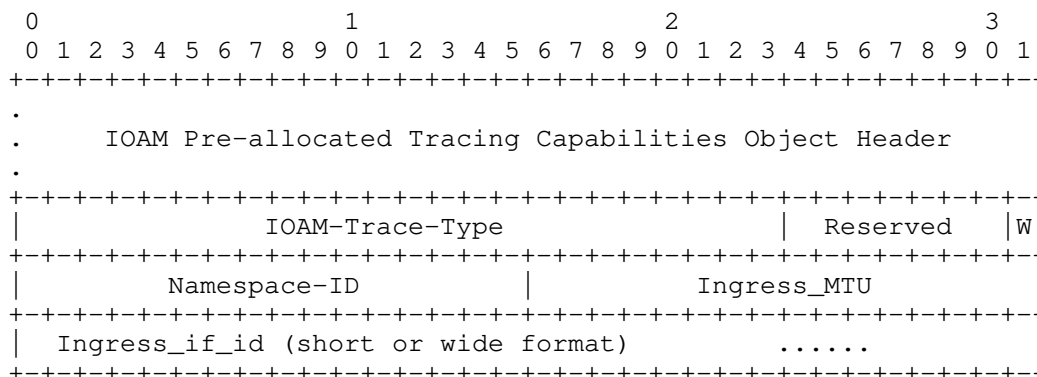


Figure 3: IOAM Pre-allocated Tracing Capabilities Object

When this Object is present in the IOAM Capabilities Response Container, that means the sending node is an IOAM transit node and the IOAM pre-allocated tracing function is enabled at this IOAM transit node.

IOAM-Trace-Type field has the same definition as what's specified in Section 5.4 of [I-D.ietf-ippm-ioam-data].

Reserved field is reserved for future use and MUST be set to zero.

W flag indicates whether Ingress_if_id is in short or wide format. The W-bit is set if the Ingress_if_id is in wide format. The W-bit is clear if the Ingress_if_id is in short format.

Namespace-ID field has the same definition as what's specified in Section 5.3 of [I-D.ietf-ippm-ioam-data], it should be one of the Namespace-IDs listed in the IOAM Capabilities Query Object of the echo request.

Ingress_MTU field has 16 bits and specifies the MTU (in octets) of the ingress interface from which the sending node received echo request.

Ingress_if_id field has 16 bits (in short format) or 32 bits (in wide format) and specifies the identifier of the ingress interface from which the sending node received echo request. If the W-bit is cleared that indicates Ingress_if_id field has 16 bits, then the 16 bits following the Ingress_if_id field are reserved for future use and MUST be set to zero.

3.2.2. IOAM Incremental Tracing Capabilities Object

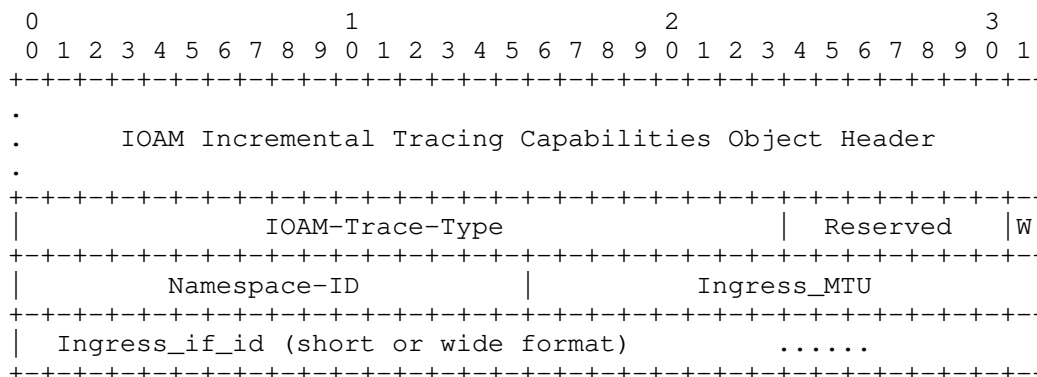


Figure 4: IOAM Incremental Tracing Capabilities Object

When this Object is present in the IOAM Capabilities Response Container, that means the sending node is an IOAM transit node and the IOAM incremental tracing function is enabled at this IOAM transit node.

IOAM-Trace-Type field has the same definition as what's specified in Section 5.4 of [I-D.ietf-ippm-ioam-data].

Reserved field is reserved for future use and MUST be set to zero.

W flag indicates whether Ingress_if_id is in short or wide format. The W-bit is set if the Ingress_if_id is in wide format. The W-bit is clear if the Ingress_if_id is in short format.

Namespace-ID field has the same definition as what's specified in Section 5.3 of [I-D.ietf-ippm-ioam-data], it should be one of the Namespace-IDs listed in the IOAM Capabilities Query Object of the echo request.

Ingress_MTU field has 16 bits and specifies the MTU (in octets) of the ingress interface from which the sending node received echo request.

Ingress_if_id field has 16 bits (in short format) or 32 bits (in wide format) and specifies the identifier of the ingress interface from which the sending node received echo request. If the W-bit is cleared that indicates Ingress_if_id field has 16 bits, then the 16 bits following the Ingress_if_id field are reserved for future use and MUST be set to zero.

3.2.3. IOAM Proof-of-Transit Capabilities Object

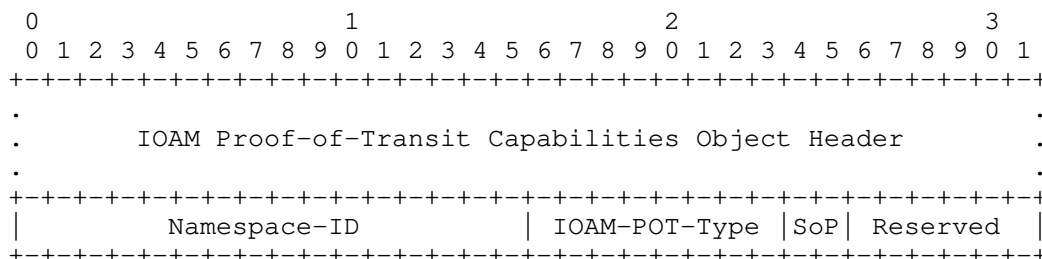


Figure 5: IOAM Proof-of-Transit Capabilities Object

When this Object is present in the IOAM Capabilities Response Container, that means the sending node is an IOAM transit node and the IOAM Proof of Transit function is enabled at this IOAM transit node.

Namespace-ID field has the same definition as what's specified in Section 5.3 of [I-D.ietf-ippm-ioam-data], it should be one of the Namespace-IDs listed in the IOAM Capabilities Query Object of the echo request.

IOAM-POT-Type field has the same definition as what's specified in Section 5.5 of [I-D.ietf-ippm-ioam-data].

SoP field has two bits, which means the size of "PktID" and "Cumulative" data that are specified in Section 5.5 of [I-D.ietf-ippm-ioam-data]. This document defines SoP as follow:

0b00 means 64-bit "PktID" and 64-bit "Cumulative" data.

0b01~0b11: Reserved for future standardization

Reserved field is reserved for future use and MUST be set to zero.

3.2.4. IOAM Edge-to-Edge Capabilities Object

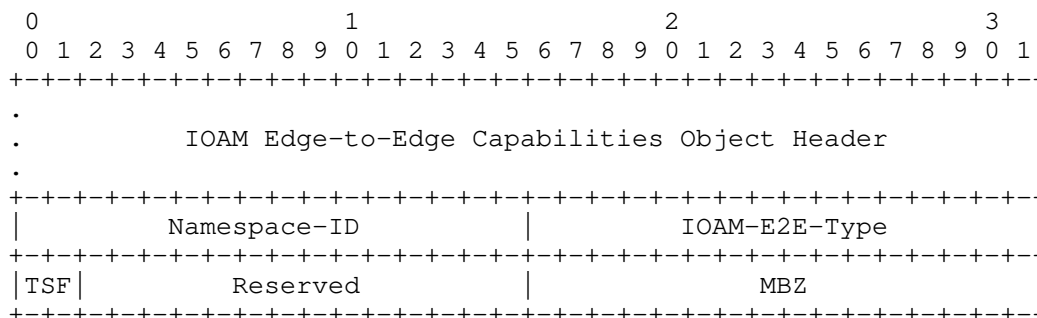


Figure 6: IOAM Edge-to-Edge Capabilities Object

When this Object is present in the IOAM Capabilities Response Container, that means the sending node is an IOAM decapsulating node and IOAM edge-to-edge function is enabled at this IOAM decapsulating node.

Namespace-ID field has the same definition as what's specified in Section 5.3 of [I-D.ietf-ippm-ioam-data], it should be one of the Namespace-IDs listed in the IOAM Capabilities Query Object of the echo request.

IOAM-E2E-Type field has the same definition as what's specified in Section 5.6 of [I-D.ietf-ippm-ioam-data].

TSF field specifies the timestamp format used by the sending node. Aligned with three possible timestamp formats specified in Section 6 of [I-D.ietf-ippm-ioam-data], this document defines TSF as follows:

0b00: PTP truncated timestamp format

0b01: NTP 64-bit timestamp format

0b10: POSIX-based timestamp format

0b11: Reserved for future standardization

Reserved field is reserved for future use and MUST be set to zero.

3.2.5. IOAM DEX Capabilities Object

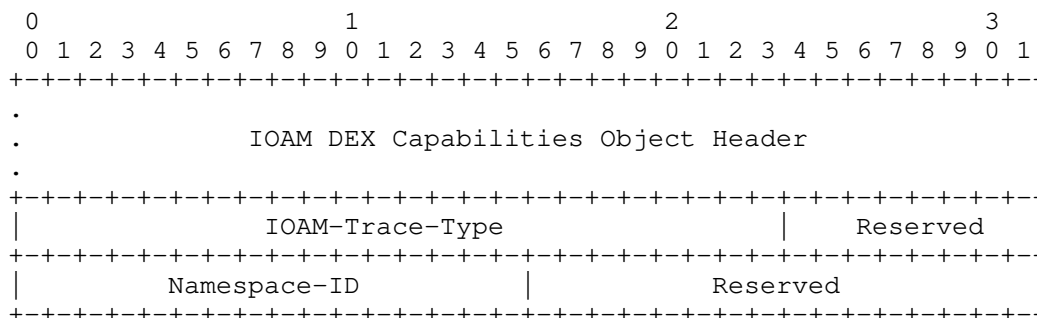


Figure 7: IOAM DEX Capabilities Object

When this Object is present in the IOAM Capabilities Response Container, that means the sending node is an IOAM transit node and the IOAM direct exporting function is enabled at this IOAM transit node.

IOAM-Trace-Type field has the same definition as what's specified in Section 3.2 of [I-D.ietf-ippm-ioam-direct-export].

Namespace-ID field has the same definition as what's specified in Section 5.3 of [I-D.ietf-ippm-ioam-data], it should be one of the Namespace-IDs listed in the IOAM Capabilities Query Object of the echo request.

Reserved field is reserved for future use and MUST be set to zero.

3.2.6. IOAM End-of-Domain Object

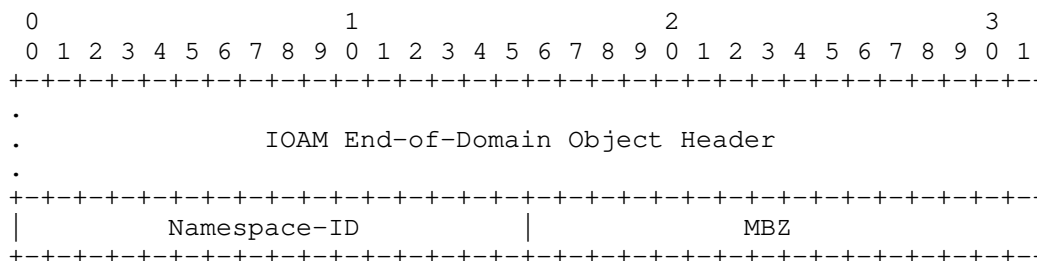


Figure 8: IOAM End-of-Domain Object

When this Object is present in the IOAM Capabilities Response Container, that means the sending node is an IOAM decapsulating node. Unless the IOAM Edge-to-Edge Capabilities Object is present, which also indicates that the sending node is an IOAM decapsulating node,

the End-of-Domain Object MUST be present in the IOAM Capabilities Response Container sent by an IOAM decapsulating node. When the IOAM edge-to-edge function is enabled at the IOAM decapsulating node, it's RECOMMENDED to include only the IOAM Edge-to-Edge Capabilities Object but not the IOAM End-of-Domain Object.

Namespace-ID field has the same definition as what's specified in Section 5.3 of [I-D.ietf-ippm-ioam-data], it SHOULD be one of the Namespace-IDs listed in the IOAM Capabilities Query Container.

4. Operational Guide

Once the IOAM encapsulating node is triggered to discover the enabled IOAM capabilities of each IOAM transit and IOAM decapsulating node, the IOAM encapsulating node will send echo requests that include the IOAM Capabilities Query Container. First, with TTL equal to 1 to reach the closest node, which may be an IOAM transit node or not. Then with TTL equal to 2 to reach the second nearest node, which also may be an IOAM transit node or not. And further, increasing by 1 the TTL every time the IOAM encapsulating node sends a new echo request, until the IOAM encapsulating node receives an echo reply sent by the IOAM decapsulating node, which should contain the IOAM Capabilities Response Container including the IOAM Edge-to-Edge Capabilities Object or the IOAM End-of-Domain Object. Alternatively, if the IOAM encapsulating node knows precisely all the IOAM transit and IOAM decapsulating nodes beforehand, once the IOAM encapsulating node is triggered to discover the enabled IOAM capabilities, it can send an echo request to each IOAM transit and IOAM decapsulating node directly, without TTL expiration.

The IOAM encapsulating node may be triggered by the device administrator, the network management system, the network controller, or data traffic. The specific triggering mechanisms are outside the scope of this document.

Each IOAM transit and IOAM decapsulating node that receives an echo request containing the IOAM Capabilities Query Container will send an echo reply to the IOAM encapsulating node. For the echo reply, there should be an IOAM Capabilities Response Container containing one or more Objects. The IOAM Capabilities Query Container of the echo request would be ignored by the receiving node unaware of IOAM.

5. IANA Considerations

This document requests the following IANA Actions.

IANA is requested to create a registry group named "In-Situ OAM (IOAM) Capabilities Parameters".

This group will include the following registries:

- * IOAM SoP Capability
- * IOAM TSF Capability

New registries in this group can be created via RFC Required process as per [RFC8126].

The subsequent sub-sections detail the registries herein contained.

Considering the Containers/Objects defined in this document would be carried in different types of Echo Request/Reply messages, such as ICMPv6 or LSP Ping, it is intended that the registries for Container/Object Type would be requested in subsequent documents.

5.1. IOAM SoP Capability Registry

This registry defines 4 code points for the IOAM SoP Capability field for identifying the size of "PktID" and "Cumulative" data as explained in Section 5.5 of [I-D.ietf-ippm-ioam-data]. The following code points are defined in this document:

SoP ----	Description -----
0b00	64-bit "PktID" and 64-bit "Cumulative" data

0b01 - 0b11 are available for assignment via RFC Required process as per [RFC8126].

5.2. IOAM TSF Capability Registry

This registry defines 3 code points for the IOAM TSF Capability field for identifying the timestamp format as explained in Section 6 of [I-D.ietf-ippm-ioam-data]. The following code points are defined in this document:

TSF ----	Description -----
0b00	PTP Truncated Timestamp Format
0b01	NTP 64-bit Timestamp Format
0b10	POSIX-based Timestamp Format

0b11 is available for assignment via RFC Required process as per [RFC8126].

6. Security Considerations

Queries and responses about the state of an IOAM domain should be processed only from a trusted source. An unauthorized query **MUST** be discarded by an implementation that supports this specification. Similarly, an unsolicited echo response with the IOAM Capabilities Container **MUST** be discarded. Authentication of echo request/reply that includes the IOAM Capabilities Container is one of the integrity protection methods. Implementations could also provide a means of filtering based on the source address of the received echo request/reply. The integrity protection for enabled IOAM capabilities information collection can also be achieved using mechanisms in the underlay data plane. For example, if the underlay is an IPv6 network, IP Authentication Header [RFC4302] or IP Encapsulating Security Payload Header [RFC4303] can be used to provide integrity protection, the specific requirements on integrity protection for enabled IOAM capabilities in IPv6 networks are discussed in [I-D.xiao-6man-icmpv6-ioam-conf-state].

Information about the state of the IOAM domain collected in the IOAM Capabilities Container is confidential. An implementation can use secure transport to provide privacy protection. For example, if the underlay is an IPv6 network, confidentiality can be achieved using the IP Encapsulating Security Payload Header [RFC4303], the specific requirements on privacy protection for enabled IOAM capabilities in IPv6 networks are discussed in [I-D.xiao-6man-icmpv6-ioam-conf-state].

7. Acknowledgements

The authors would like to acknowledge Tianran Zhou, Dhruv Dhody, Frank Brockners, Cheng Li and Gyan Mishra for their careful review and helpful comments.

The authors appreciate the f2f discussion with Frank Brockners on this document.

The authors would like to acknowledge Tommy Pauly and Ian Swett for their good suggestion and guidance.

8. References

8.1. Normative References

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

Brockners, F., Bhandari, S., and T. Mizrahi, "Data Fields for In-situ OAM", Work in Progress, Internet-Draft, draft-ietf-ippm-ioam-data-17, 13 December 2021, <<https://www.ietf.org/archive/id/draft-ietf-ippm-ioam-data-17.txt>>.

[I-D.ietf-ippm-ioam-direct-export]

Song, H., Gafni, B., Zhou, T., Li, Z., Brockners, F., Bhandari, S., Sivakolundu, R., and T. Mizrahi, "In-situ OAM Direct Exporting", Work in Progress, Internet-Draft, draft-ietf-ippm-ioam-direct-export-07, 13 October 2021, <<https://www.ietf.org/archive/id/draft-ietf-ippm-ioam-direct-export-07.txt>>.

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

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.

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

8.2. Informative References

[I-D.ietf-bier-ping]

Kumar, N., Pignataro, C., Akiya, N., Zheng, L., Chen, M., and G. Mirsky, "BIER Ping and Trace", Work in Progress, Internet-Draft, draft-ietf-bier-ping-07, 11 May 2020, <<https://www.ietf.org/archive/id/draft-ietf-bier-ping-07.txt>>.

[I-D.ietf-sfc-multi-layer-oam]

Mirsky, G., Meng, W., Khasnabish, B., Ao, T., Leung, K., and G. Mishra, "Active OAM for Service Function Chaining", Work in Progress, Internet-Draft, draft-ietf-sfc-multi-layer-oam-18, 20 December 2021, <<https://www.ietf.org/archive/id/draft-ietf-sfc-multi-layer-oam-18.txt>>.

- [I-D.xiao-6man-icmpv6-ioam-conf-state]
Min, X. and G. Mirsky, "ICMPv6 Echo Request/Reply for Enabled In-situ OAM Capabilities", Work in Progress, Internet-Draft, draft-xiao-6man-icmpv6-ioam-conf-state-00, 24 October 2021, <<https://www.ietf.org/archive/id/draft-xiao-6man-icmpv6-ioam-conf-state-00.txt>>.
- [RFC4302] Kent, S., "IP Authentication Header", RFC 4302, DOI 10.17487/RFC4302, December 2005, <<https://www.rfc-editor.org/info/rfc4302>>.
- [RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, DOI 10.17487/RFC4303, December 2005, <<https://www.rfc-editor.org/info/rfc4303>>.
- [RFC4443] Conta, A., Deering, S., and M. Gupta, Ed., "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", STD 89, RFC 4443, DOI 10.17487/RFC4443, March 2006, <<https://www.rfc-editor.org/info/rfc4443>>.
- [RFC4884] Bonica, R., Gan, D., Tappan, D., and C. Pignataro, "Extended ICMP to Support Multi-Part Messages", RFC 4884, DOI 10.17487/RFC4884, April 2007, <<https://www.rfc-editor.org/info/rfc4884>>.
- [RFC8029] Kompella, K., Swallow, G., Pignataro, C., Ed., Kumar, N., Aldrin, S., and M. Chen, "Detecting Multiprotocol Label Switched (MPLS) Data-Plane Failures", RFC 8029, DOI 10.17487/RFC8029, March 2017, <<https://www.rfc-editor.org/info/rfc8029>>.
- [RFC8335] Bonica, R., Thomas, R., Linkova, J., Lenart, C., and M. Boucadair, "PROBE: A Utility for Probing Interfaces", RFC 8335, DOI 10.17487/RFC8335, February 2018, <<https://www.rfc-editor.org/info/rfc8335>>.
- [RFC8799] Carpenter, B. and B. Liu, "Limited Domains and Internet Protocols", RFC 8799, DOI 10.17487/RFC8799, July 2020, <<https://www.rfc-editor.org/info/rfc8799>>.

Authors' Addresses

Xiao Min
ZTE Corp.
Nanjing
China

Phone: +86 25 88013062
Email: xiao.min2@zte.com.cn

Greg Mirsky
Ericsson
United States of America

Email: gregimirsky@gmail.com

Lei Bo
China Telecom
Beijing
China

Phone: +86 10 50902903
Email: leibo@chinatelecom.cn

ippm
Internet-Draft
Intended status: Standards Track
Expires: September 3, 2022

F. Brockners
Cisco
S. Bhandari
Thoughtspot
T. Mizrahi
Huawei
J. Iurman
ULiege
March 2, 2022

Integrity of In-situ OAM Data Fields
draft-ietf-ippm-ioam-data-integrity-01

Abstract

In-situ Operations, Administration, and Maintenance (IOAM) records operational and telemetry information in the packet while the packet traverses a path in the network. IETF protocols require features to ensure their security. This document describes the integrity protection of IOAM-Data-Fields.

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 3, 2022.

Copyright Notice

Copyright (c) 2022 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	3
3. Threat Analysis	4
3.1. Modification: IOAM-Data-Fields	4
3.2. Modification: IOAM Option-Type Headers	5
3.3. Injection: IOAM-Data-Fields	5
3.4. Injection: IOAM Option-Type Headers	6
3.5. Management and Exporting	6
3.6. Delay	6
3.7. Threat Summary	7
4. Integrity Protected Option-Types	7
4.1. Integrity Protected Trace Option-Types	8
4.2. Integrity Protected POT Option-Type	9
4.3. Integrity Protected E2E Option-Type	9
5. Methods for space optimized integrity protection	10
5.1. Symmetric key based signature	11
5.2. Asymmetric key based signature	11
6. IANA Considerations	11
6.1. IOAM Option-Type Registry	11
6.2. IOAM Integrity Protection Algorithm Suite Registry	12
7. Security Considerations	12
7.1. Replay protection	12
8. Acknowledgements	13
9. References	13
9.1. Normative References	13
9.2. Informative References	13
Authors' Addresses	14

1. Introduction

"In-situ" Operations, Administration, and Maintenance (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 OAM data is added to the data packets rather than being sent within packets specifically dedicated to OAM. IOAM is to complement mechanisms such as Ping or Traceroute. In terms of "active" or "passive" OAM, "in-situ" OAM can be considered a hybrid OAM type. "In-situ" mechanisms do not require extra packets to be sent. IOAM adds information to the already available data packets and therefore cannot be considered passive. In terms of the classification given

in [RFC7799], IOAM could be portrayed as Hybrid Type I. 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 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.

The current [I-D.ietf-ippm-ioam-data] assumes that IOAM is deployed in limited domains, where an operator has means to select, monitor, and control the access to all the networking devices, making the domain a trusted network. As such, IOAM-Data-Fields are carried in clear within packets and there are no protections against any node or middlebox tampering with the data. IOAM-Data-Fields collected in an untrusted or semi-trusted environment require integrity protection to support critical operational decisions.

The following considerations and requirements are to be taken into account in addition to addressing the problem of detectability of any integrity breach of the IOAM-Data-Fields collected:

1. IOAM data is processed by the data plane, hence viability of any method to prove integrity of the IOAM-Data-Fields must be feasible at data plane processing/forwarding rates (IOAM might be applied to all traffic a router forwards).
2. IOAM data is carried within packets. Additional space required to prove integrity of the IOAM-Data-Fields needs to be optimal, i.e. should not exceed the MTU or have adverse effect on packet processing.
3. Replay protection of older IOAM data should be possible. Without replay protection, a rogue node can present the old IOAM data, masking any ongoing network issues/activity and making the IOAM-Data-Fields collection useless.

This document defines the methods to protect the integrity of IOAM-Data-Fields, using the IOAM Option-Types specified in [I-D.ietf-ippm-ioam-data] as an example. The methods similarly apply to other IOAM Option-Types which contain IOAM-Data-Fields.

2. Conventions

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].

Abbreviations used in this document:

IOAM: In-situ Operations, Administration, and Maintenance

MTU: Maximum Transmit Unit

OAM: Operations, Administration, and Maintenance

POT: Proof of Transit

E2E: Edge to Edge

3. Threat Analysis

This section presents a threat analysis of integrity-related threats in the context of IOAM. The threats that are discussed are assumed to be independent of the lower layer protocols; it is assumed that threats at other layers are handled by security mechanisms that are deployed at these layers.

This document is focused on integrity protection for IOAM-Data-Fields. Thus the threat analysis includes threats that are related to or result from compromising the integrity of IOAM-Data-Fields. Other security aspects such as confidentiality are not within the scope of this document.

Throughout the analysis there is a distinction between on-path and off-path attackers. As discussed in [I-D.ietf-detnet-security], on-path attackers are located in a position that allows interception and modification of in-flight protocol packets, whereas off-path attackers can only attack by generating protocol packets.

The analysis also includes the impact of each of the threats. Generally speaking, the impact of a successful attack on an OAM protocol [RFC7276] is a false illusion of nonexistent failures or preventing the detection of actual ones; in both cases, the attack may result in denial of service (DoS). Furthermore, creating the false illusion of a nonexistent issue may trigger unnecessary processing in some of the IOAM nodes along the path, and may cause more IOAM-related data to be exported to the management plane than is conventionally necessary. Beyond these general impacts, threat-specific impacts are discussed in each of the subsections below.

3.1. Modification: IOAM-Data-Fields

Threat

An attacker can maliciously modify the IOAM-Data-Fields of in-transit packets. The modification can either be applied to all packets or selectively applied to a subset of the en route packets. This threat is applicable to on-path attackers.

Impact

By systematically modifying the IOAM-Data-Fields of some or all of the in-transit packets, an attacker can create a false picture of the paths in the network, the existence of faulty nodes and their location, and the network performance.

3.2. Modification: IOAM Option-Type Headers

Threat

An on-path attacker can modify the header in IOAM Option-Types in order to change or disrupt the behavior of nodes processing IOAM-Data-Fields along the path. This threat is not within the scope of this document.

Impact

Changing the header of IOAM Option-Types may have several implications. An attacker can maliciously increase the processing overhead in nodes that process IOAM-Data-Fields and increase the on-the-wire overhead of IOAM-Data-Fields, for example by modifying the IOAM-Trace-Type field in the IOAM Trace Option-Type header. An attacker can also prevent some of the nodes that process IOAM-Data-Fields from incorporating IOAM-Data-Fields, by modifying the RemainingLen field in the IOAM Trace Option-Type header.

3.3. Injection: IOAM-Data-Fields

Threat

An attacker can inject packets with IOAM Option-Types and IOAM-Data-Fields. This threat is applicable to both on-path and off-path attackers.

Impact

This attack and its impacts are similar to Section 3.1.

3.4. Injection: IOAM Option-Type Headers

Threat

An attacker can inject packets with IOAM Option-Type headers, thus manipulating other nodes that process IOAM-Data-Fields in the network. This threat is applicable to both on-path and off-path attackers. This threat is not within the scope of this document.

Impact

This attack and its impacts are similar to Section 3.2.

3.5. Management and Exporting

Threat

Attacks that compromise the integrity of IOAM-Data-Fields can be applied at the management plane, e.g., by manipulating network management packets. Furthermore, the integrity of IOAM-Data-Fields that are exported to a receiving entity can also be compromised. Management plane attacks are not within the scope of this document; the network management protocol is expected to include inherent security capabilities. The integrity of exported data is also not within the scope of this document. It is expected that the specification of the export format will discuss the relevant security aspects.

Impact

Malicious manipulation of the management protocol can cause nodes that process IOAM-Data-Fields to malfunction, to be overloaded, or to incorporate unnecessary IOAM-Data-Fields into user packets. The impact of compromising the integrity of exported IOAM-Data-Fields is similar to the impacts of previous threats that were described in this section.

3.6. Delay

Threat

An on-path attacker may delay some or all of the in-transit packets that include IOAM-Data-Fields in order to create the false illusion of congestion. Delay attacks are well known in the context of deterministic networks [I-D.ietf-detnet-security] and synchronization [RFC7384], and may be somewhat mitigated in these environments by using redundant paths in a way that is resilient to an attack along one of the paths. This approach does not

address the threat in the context of IOAM, as it does not meet the requirement to measure a specific path or to detect a problem along the path. It is noted that this threat is not within the scope of the threats that are mitigated in this document.

Impact

Since IOAM can be applied to a fraction of the traffic, an attacker can detect and delay only the packets that include IOAM-Data-Fields, thus preventing the authenticity of delay and load measurements.

3.7. Threat Summary

Threat	In scope	Out of scope
Modification: IOAM-Data-Fields	+	
Modification: IOAM Option-Type Headers		+
Injection: IOAM-Data-Fields	+	
Injection: IOAM Option-Type Headers		+
Management and Exporting		+
Delay		+

Figure 1: Threat Analysis Summary

4. Integrity Protected Option-Types

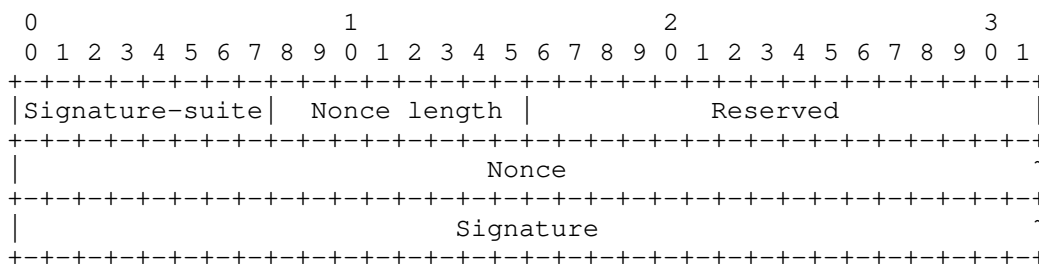
This section defines new IOAM Option-Types to be allocated in the IOAM Option-Type Registry. Their purpose is to carry IOAM-Data-Fields with integrity protection. Each of the IOAM Option-Types defined in [I-D.ietf-ippm-ioam-data] is extended as follows:

- 64 IOAM Pre-allocated Trace Integrity Protected Option-Type:
corresponds to the IOAM Pre-allocated Trace Option-Type with integrity protection.
- 65 IOAM Incremental Trace Integrity Protected Option-Type:
corresponds to the IOAM Incremental Trace Option-Type with integrity protection.

66 IOAM POT Integrity Protected Option-Type: corresponds to the IOAM POT Option-Type with integrity protection.

67 IOAM E2E Integrity Protected Option-Type: corresponds to the IOAM E2E Option-Type with integrity protection.

The Integrity Protection subheader follows the IOAM Option-Type header when the IOAM Option-Type is an Integrity Protected Option-Type. It is defined as follows:



Signature-suite: 8-bit unsigned integer. This field defines the algorithms used to compute the digest and the signature over the IOAM-Data-Fields.

Nonce length: 8-bit unsigned integer. This field specifies the length of the Nonce in octets.

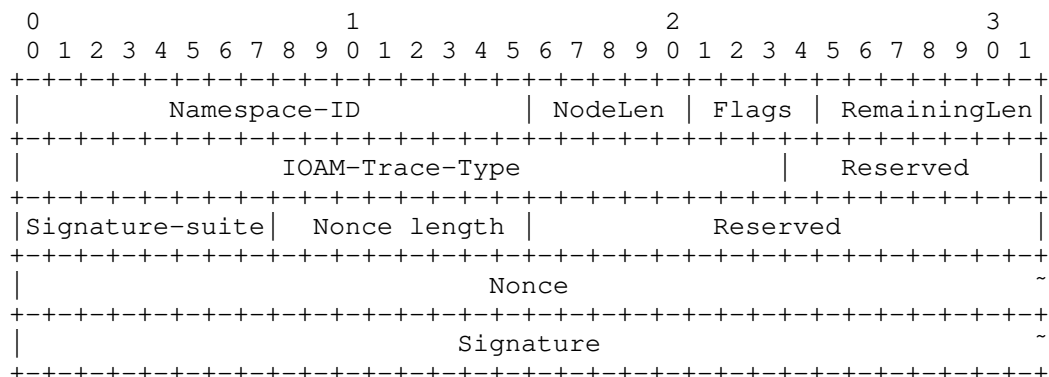
Reserved: 16-bit Reserved field. MUST be set to zero upon transmission and ignored upon receipt.

Nonce: Variable length field with length specified in Nonce length.

Signature: Digital signature value generated by the method and algorithm specified by Signature-suite.

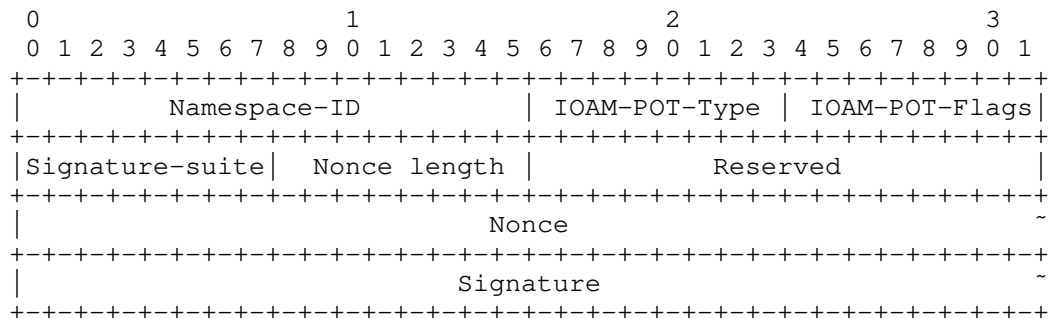
4.1. Integrity Protected Trace Option-Types

Both the IOAM Pre-allocated Trace Option-Type header and the IOAM Incremental Trace Option-Type header, as defined in [I-D.ietf-ippm-ioam-data], are followed by the Integrity Protection subheader when the IOAM Option-Type is respectively set to the IOAM Pre-allocated Trace Integrity Protected Option-Type or the IOAM Incremental Trace Integrity Protected Option-Type:



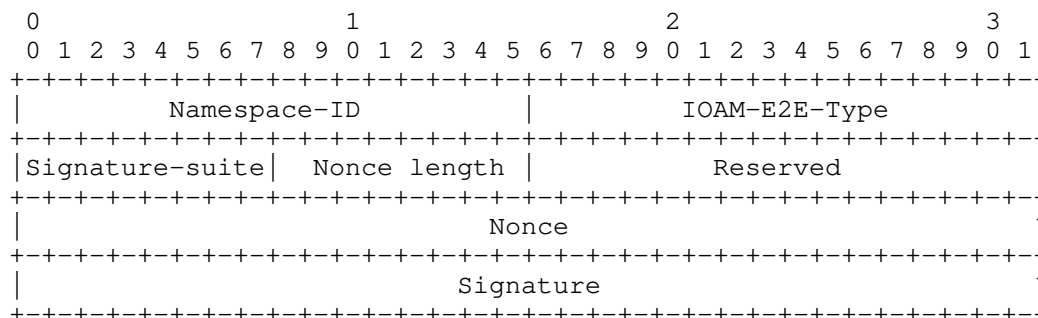
4.2. Integrity Protected POT Option-Type

The IOAM POT Option-Type header, as defined in [I-D.ietf-ippm-ioam-data], is followed by the Integrity Protection subheader when the IOAM Option-Type is set to the IOAM POT Integrity Protected Option-Type:



4.3. Integrity Protected E2E Option-Type

The IOAM E2E Option-Type header, as defined in [I-D.ietf-ippm-ioam-data], is followed by the Integrity Protection subheader when the IOAM Option-Type is set to the IOAM E2E Integrity Protected Option-Type:



5. Methods for space optimized integrity protection

Methods for space optimized integrity protection can leverage symmetric or asymmetric key based signatures, as described in the subsections below. The Signature consumes 32 octets and is carried only once for the entire packet. In case of performance concerns, such method can be applied to a subset of the traffic by using sampling of data to enable IOAM with integrity protection. Both symmetric and asymmetric signature methods work as follows:

1. The encapsulating node creates a nonce and stores it in the Nonce field of the Integrity Protection subheader. The signature is generated over the Nonce field and the hash of IOAM-Data-Fields it has inserted, i.e., $\text{sign}(\text{Nonce} || \text{hash}(\text{IOAM-Data-Fields}))$. IOAM-Data-Fields that will be updated in-place MUST be excluded from the signature (e.g., the POT Cumulative field). The signature is stored in the Signature field of the Integrity Protection subheader. Important note: if all the inserted IOAM-Data-Fields are to be updated in-place, or if there is no IOAM-Data-Field at all, the encapsulating node MUST NOT use an Integrity Protected Option-Type.
2. A transit node generates a signature over the Signature field and the hash of IOAM-Data-Fields it has inserted, i.e., $\text{sign}(\text{Signature} || \text{hash}(\text{IOAM-Data-Fields}))$. IOAM-Data-Fields that are updated in-place MUST be excluded from the signature (e.g., the POT Cumulative field). The signature is stored in the Signature field of the Integrity Protection subheader. Important note: if all the IOAM-Data-Fields involved are updated in-place, or if there is no IOAM-Data-Field involved, the transit node MUST NOT generate a signature and MUST NOT update the Signature field.
3. The decapsulating node behaves exactly the same as a transit node. The only difference is that the signature MAY NOT be

required when the decapsulating node is also the Validator, for obvious performance reasons.

4. The Validator recreates the signature over IOAM-Data-Fields collected and checks the integrity against the Signature field. In order to recompute the signature, the Validator iteratively follows the same procedure as for the encapsulating, transit and decapsulating nodes, in that order. It is trivial in some cases (e.g., POT Type-0 or E2E), where only the encapsulating node generates a signature. For other cases where transit nodes also generate a signature (e.g., Trace Option-Types), node-ids MUST be recorded. Details on how the mapping between node-ids and keys is implemented are outside the scope of this document.

5.1. Symmetric key based signature

This method assumes that symmetric keys have been distributed to the respective nodes as well as the Validator (the Validator receives all the keys). The details of the mechanisms responsible for key distribution are outside the scope of this document.

This method MUST use an algorithm pair defined in Section 6.2 and the approach MUST be symmetric.

5.2. Asymmetric key based signature

This method assumes that asymmetric keys have been generated per IOAM node and the respective nodes can access their keys (the Validator receives all the public keys). The details of the mechanisms responsible for key distribution are outside the scope of this document.

This method MUST use an algorithm pair defined in Section 6.2 and the approach MUST be asymmetric.

6. IANA Considerations

6.1. IOAM Option-Type Registry

This draft defines the following new code points in the IOAM Option-Type Registry:

64 IOAM Pre-allocated Trace Integrity Protected Option-Type

65 IOAM Incremental Trace Integrity Protected Option-Type

66 IOAM POT Integrity Protected Option-Type

67 IOAM E2E Integrity Protected Option-Type

6.2. IOAM Integrity Protection Algorithm Suite Registry

"IOAM Integrity Protection Algorithm Suite Registry" in the "In-Situ OAM (IOAM) Protocol Parameters" group. The one-octet "IOAM Integrity Protection Algorithm Suite Registry" identifiers assigned by IANA identify the digest algorithm and signature algorithm used in the Signature Suite Identifier field. IANA has registered the following algorithm suite identifiers for the digest algorithm and for the signature algorithm.

Algorithm Suite Identifier	Digest Algorithm	Signature Algorithm	Specification Pointer	Approach
0x00	Reserved	Reserved	This document	None
0x01	SHA-256	ECDSA P-256	[SHS] [DSS] [RFC6090] This document	Asymmetric
0x02	SHA-256	AES-256	[AES] [NIST.800-38D] This document	Symmetric
0x03-0xFF	Unassigned	Unassigned		

IOAM Integrity Protection Algorithm Suite Registry

Future assignments are to be made using the Standards Action process defined in [RFC8126]. Assignments consist of the one-octet algorithm suite identifier value and the associated digest algorithm name and signature algorithm name.

7. Security Considerations

This section discusses additional security aspects.

7.1. Replay protection

The nonce makes a signature chain unique but does not necessarily prevent replay attacks. To enable replay protection, both the encaps node and the validator MUST synchronize on a unique nonce, e.g., based on a timestamp only valid for a short period of time.

8. Acknowledgements

The authors would like to thank Santhosh N, Rakesh Kandula, Saiprasad Muchala, Al Morton, Greg Mirsky, Benjamin Kaduk and Martin Duke for their comments and advice.

9. References

9.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

9.2. Informative References

- [AES] National Institute of Standards and Technology, "Advanced Encryption Standard (AES)", FIPS PUB 197, 2001, <<http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf>>.
- [DSS] National Institute of Standards and Technology, "Digital Signature Standard (DSS)", NIST FIPS Publication 186-4, DOI 10.6028/NIST.FIPS.186-4, 2013, <<http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf>>.
- [I-D.ietf-detnet-security] Grossman, E., Mizrahi, T., and A. J. Hacker, "Deterministic Networking (DetNet) Security Considerations", draft-ietf-detnet-security-16 (work in progress), March 2021.
- [I-D.ietf-ippm-ioam-data] Brockners, F., Bhandari, S., and T. Mizrahi, "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data-17 (work in progress), December 2021.

- [NIST.800-38D] National Institute of Standards and Technology, "Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC", NIST Special Publication 800-38D, 2001, <<http://csrc.nist.gov/publications/nistpubs/800-38D/SP-800-38D.pdf>>.
- [RFC6090] McGrew, D., Igoe, K., and M. Salter, "Fundamental Elliptic Curve Cryptography Algorithms", RFC 6090, DOI 10.17487/RFC6090, February 2011, <<https://www.rfc-editor.org/info/rfc6090>>.
- [RFC7276] Mizrahi, T., Sprecher, N., Bellagamba, E., and Y. Weingarten, "An Overview of Operations, Administration, and Maintenance (OAM) Tools", RFC 7276, DOI 10.17487/RFC7276, June 2014, <<https://www.rfc-editor.org/info/rfc7276>>.
- [RFC7384] Mizrahi, T., "Security Requirements of Time Protocols in Packet Switched Networks", RFC 7384, DOI 10.17487/RFC7384, October 2014, <<https://www.rfc-editor.org/info/rfc7384>>.
- [RFC7799] Morton, A., "Active and Passive Metrics and Methods (with Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799, May 2016, <<https://www.rfc-editor.org/info/rfc7799>>.
- [SHS] National Institute of Standards and Technology, "Secure Hash Standard (SHS)", NIST FIPS Publication 180-4, DOI 10.6028/NIST.FIPS.180-4, 2015, <<http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf>>.

Authors' Addresses

Frank Brockners
Cisco Systems, Inc.
Hansaallee 249, 3rd Floor
DUESSELDORF, NORDRHEIN-WESTFALEN 40549
Germany

Email: fbrockne@cisco.com

Shwetha Bhandari
Thoughtspot
3rd Floor, Indiqube Orion, 24th Main Rd, Garden Layout, HSR Layout
Bangalore, KARNATAKA 560 102
India

Email: shwetha.bhandari@thoughtspot.com

Tal Mizrahi
Huawei
8-2 Matam
Haifa 3190501
Israel

Email: tal.mizrahi.phd@gmail.com

Justin Iurman
Universite de Liege
10, Allee de la decouverte (B28)
Sart-Tilman, LIEGE 4000
Belgium

Email: justin.iurman@uliege.be

ippm
Internet-Draft
Intended status: Best Current Practice
Expires: October 13, 2022

F. Brockners, Ed.
Cisco
S. Bhandari, Ed.
Thoughtspot
D. Bernier
Bell Canada
T. Mizrahi, Ed.
Huawei
April 11, 2022

In-situ OAM Deployment
draft-ietf-ippm-ioam-deployment-01

Abstract

In-situ Operations, Administration, and Maintenance (IOAM) collects operational and telemetry information in the packet while the packet traverses a path between two points in the network. This document provides a framework for IOAM deployment and provides best current practices.

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 13, 2022.

Copyright Notice

Copyright (c) 2022 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	3
3. IOAM Deployment: Domains And Nodes	4
4. Types of IOAM	5
4.1. Per-hop Tracing IOAM	7
4.2. Proof of Transit IOAM	8
4.3. Edge to Edge IOAM	8
4.4. Direct Export IOAM	8
5. IOAM Encapsulations	9
5.1. IPv6	9
5.2. NSH	9
5.3. BIER	9
5.4. GRE	9
5.5. Geneve	10
5.6. Segment Routing	10
5.7. Segment Routing for IPv6	10
5.8. VXLAN-GPE	10
6. IOAM Data Export	10
7. IOAM Deployment Considerations	12
7.1. IOAM Namespaces	12
7.2. IOAM Layering	14
7.3. IOAM Trace Option Types	15
7.4. Traffic-sets That IOAM Is Applied To	17
7.5. IOAM Loopback Mode	17
7.6. IOAM Active Mode	17
7.7. Brown Field Deployments: IOAM Unaware Nodes	18
8. IOAM Manageability	18
9. IANA Considerations	19
10. Security Considerations	19
11. Acknowledgements	20
12. References	20
12.1. Normative References	20
12.2. Informative References	21
Authors' Addresses	24

1. Introduction

"In-situ" Operations, Administration, and Maintenance (IOAM) collects 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, Traceroute, or other active probing mechanisms. In terms of "active" or "passive" OAM, "in-situ" OAM can be considered a hybrid OAM type. "In-situ" mechanisms do not require extra packets to be sent. IOAM adds information to the already available data packets and therefore cannot be considered passive. In terms of the classification given in [RFC7799] IOAM could be portrayed as Hybrid Type I. 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", "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.

Abbreviations used in this document:

BIER:	Bit Index Explicit Replication
E2E:	Edge to Edge
Geneve:	Generic Network Virtualization Encapsulation [I-D.ietf-nvo3-geneve]
GRE:	Generic Routing Encapsulation
IOAM:	In-situ Operations, Administration, and Maintenance
MTU:	Maximum Transmit Unit
NSH:	Network Service Header [RFC8300]
OAM:	Operations, Administration, and Maintenance
POT:	Proof of Transit
SFC:	Service Function Chain

SID: Segment Identifier

SR: Segment Routing

VXLAN-GPE: Virtual eXtensible Local Area Network, Generic Protocol Extension [I-D.ietf-nvo3-vxlan-gpe]

3. IOAM Deployment: Domains And Nodes

IOAM is focused on "limited domains" as defined in [RFC8799]. IOAM is not targeted for a deployment on the global Internet. The part of the network which employs IOAM is referred to as the "IOAM-Domain". For example, an IOAM-domain can include an enterprise campus using physical connections between devices or an overlay network using virtual connections / tunnels for connectivity between said devices. An IOAM-domain is defined by its perimeter or edge. The operator of an IOAM-domain is expected to put provisions in place to ensure that packets which contain IOAM data fields do not leak beyond the edge of an IOAM domain, e.g. using for example packet filtering methods. The operator should consider the potential operational impact of IOAM to mechanisms such as ECMP load-balancing schemes (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.

An IOAM-Domain consists of "IOAM encapsulating nodes", "IOAM decapsulating nodes" and "IOAM transit nodes". The role of a node (i.e., encapsulating, transit, decapsulating) is defined within an IOAM-Namespace (see below). A node can have different roles in different IOAM-Namespaces.

An "IOAM encapsulating node" incorporates one or more IOAM-Option-Types into packets that IOAM is enabled for. If IOAM is enabled for a selected subset of the traffic, the IOAM encapsulating node is responsible for applying the IOAM functionality to the selected subset.

An "IOAM transit node" updates one or more of the IOAM-Data-Fields. If both the Pre-allocated and the Incremental Trace Option-Types are present in the packet, each IOAM transit node will update at most one of these Option-Types. A transit node does not add new IOAM-Option-Types to a packet, and does not change the IOAM-Data-Fields of an IOAM Edge-to-Edge Option-Type.

An "IOAM decapsulating node" removes IOAM-Option-Type(s) from packets.

The role of an IOAM-encapsulating, IOAM-transit or IOAM-decapsulating node is always performed within a specific IOAM-Namespace. This means that an IOAM node which is e.g., an IOAM-decapsulating node for IOAM-Namespace "A" but not for IOAM-Namespace "B" will only remove the IOAM-Option-Types for IOAM-Namespace "A" from the packet. An IOAM decapsulating node situated at the edge of an IOAM domain removes all IOAM-Option-Types and associated encapsulation headers for all IOAM-Namespace from the packet.

IOAM-Namespace allow for a namespace-specific definition and interpretation of IOAM-Data-Fields. An interface-id could for example 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). Please refer to Section 7.1 for a discussion of IOAM-Namespace.

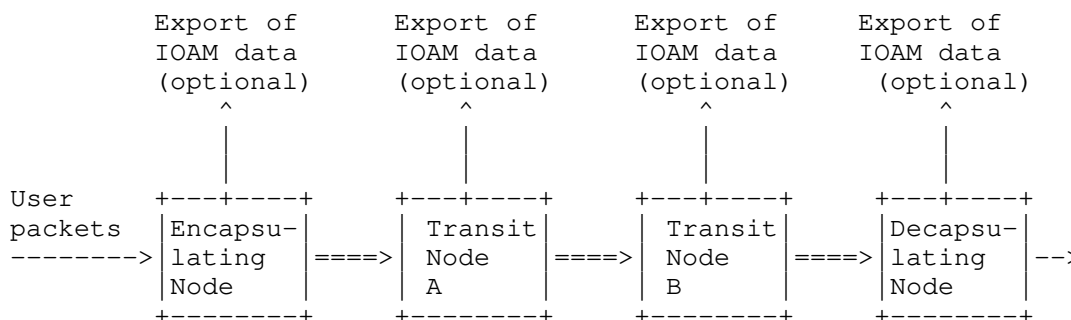


Figure 1: Roles of IOAM nodes

IOAM nodes which add or remove the IOAM-Data-Fields can also update the IOAM-Data-Fields at the same time. Or in other words, IOAM encapsulating 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 deployment might require that packets traverse a set of firewalls which support IOAM. In that case, only the set of firewall nodes would be IOAM transit nodes rather than all nodes.

4. Types of IOAM

IOAM supports different modes of operation, which are differentiated by the type of IOAM data fields being carried in the packet, the data being collected, the type of nodes which collect or update data as well as whether and how nodes export IOAM information.

- o Per-hop tracing: OAM information about each IOAM node a packet traverses is collected and stored within the user data packet as the packet progresses through the IOAM domain. Potential uses of IOAM per-hop tracing include:
 - * Understand the different paths different packets traverse between an IOAM encapsulating and an IOAM decapsulating node in a network that uses load balancing such as Equal Cost Multi-Path (ECMP). This information could be used to tune the algorithm for ECMP for optimized network resource usage.
 - * Operations/Troubleshooting: Understand which path a particular packet or set of packets take as well as what amount of jitter and delay different IOAM nodes in the path contribute to the overall delay and jitter between the IOAM encapsulating node and the IOAM decapsulating node.
- o Proof-of-transit: Information that a verifier node can use to verify whether a packet has traversed all nodes that is supposed to traverse is stored within the user data packet. Proof-of-transit could for example be used to verify that a packet indeed passes through all services of a service function chain (e.g., verify whether a packet indeed traversed the set of firewalls that it is expected to traverse), or whether a packet indeed took the expected path.
- o Edge-to-edge: OAM information which is specific to the edges of an IOAM domain is collected and stored within the user data packet. Edge-to-Edge OAM could be used to gather operational information about a particular network domain, such as the delay and jitter incurred by that network domain or the traffic matrix of the network domain.
- o Direct export: OAM information about each IOAM node a packet traverses is collected and immediately exported to a collector. Direct export could be used in situations where per-hop tracing information is desired, but gathering the information within the packet - as with per-hop tracing - is not feasible. Rather than automatically correlating the per-hop tracing information, as done with per-hop tracing, direct export requires a collector to correlate the information from the individual nodes. In addition, all nodes enabled for direct export need to be capable to export the IOAM information to the collector.

4.1. Per-hop Tracing IOAM

"IOAM tracing data" is expected to be collected at every IOAM transit 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 IOAM-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 (i.e., node data, see below) 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.

IOAM offers two different trace Option-Types, the "incremental" Option-Type as well as the "pre-allocated" Option-Type. For a discussion which of the two option types is the most suitable for an implementation and/or deployment, see Section 7.3.

Every node data entry holds information for a particular IOAM transit node that is traversed by a packet. The IOAM decapsulating node removes the IOAM-Option-Type(s) and processes and/or exports the associated data. All IOAM-Data-Fields are defined in the context of an IOAM-Namespace.

IOAM tracing can collect the following types of information:

- 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 as well as the transit delay. 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 IOAM-Namespace, 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 Information to detect whether IOAM trace data was added at every hop or whether certain hops in the domain weren't IOAM transit nodes.
- o Data that relates to how the packet traversed a node (transit delay, buffer occupancy in case the packet was buffered, queue depth in case the packet was queued)

The Option-Types of incremental tracing and pre-allocated tracing are defined in [I-D.ietf-ippm-ioam-data].

4.2. Proof of Transit IOAM

IOAM Proof of Transit Option-Type is to support 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).

The IOAM Proof of Transit Option-Type consist of a fixed size "IOAM proof of transit option header" and "IOAM proof of transit option data fields". For details see [I-D.ietf-ippm-ioam-data].

4.3. Edge to Edge IOAM

The IOAM Edge-to-Edge Option-Type 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 but must not modify it.

The IOAM Edge-to-Edge Option-Type consist of a fixed size "IOAM Edge-to-Edge Option-Type header" and "IOAM Edge-to-Edge Option-Type data fields". For details see [I-D.ietf-ippm-ioam-data].

4.4. Direct Export IOAM

Direct Export is an IOAM mode of operation within which IOAM data to be directly exported to a collector rather than being collected within the data packets. The IOAM Direct Export Option-Type consist of a fixed size "IOAM direct export option header". Direct Export for IOAM is defined in [I-D.ietf-ippm-ioam-direct-export].

5. IOAM Encapsulations

IOAM data fields and associated data types for in-situ OAM are defined in [I-D.ietf-ippm-ioam-data]. The in-situ OAM data field can be transported by a variety of transport protocols, including NSH, Segment Routing, Geneve, BIER, IPv6, etc.

5.1. IPv6

IOAM encapsulation for IPv6 is defined in [I-D.ietf-ippm-ioam-ipv6-options], which also discussed IOAM deployment considerations for IPv6 networks

5.2. NSH

IOAM encapsulation for NSH is defined in [I-D.ietf-sfc-ioam-nsh].

5.3. BIER

IOAM encapsulation for BIER is defined in [I-D.xzlnp-bier-ioam].

5.4. GRE

IOAM encapsulation for GRE is outlined as part of the "EtherType Protocol Identification of In-situ OAM Data" in [I-D.weis-ippm-ioam-eth], though no example protocol header stacks are provided in the document. When used with GRE, the IOAM-Option-Types (the below diagram uses "IOAM" as shorthand for IOAM-Option-Types) are sequenced in behind the GRE header that follows the "outer" IP header. Figure 2 shows two example protocol header stacks that use GRE along with IOAM.

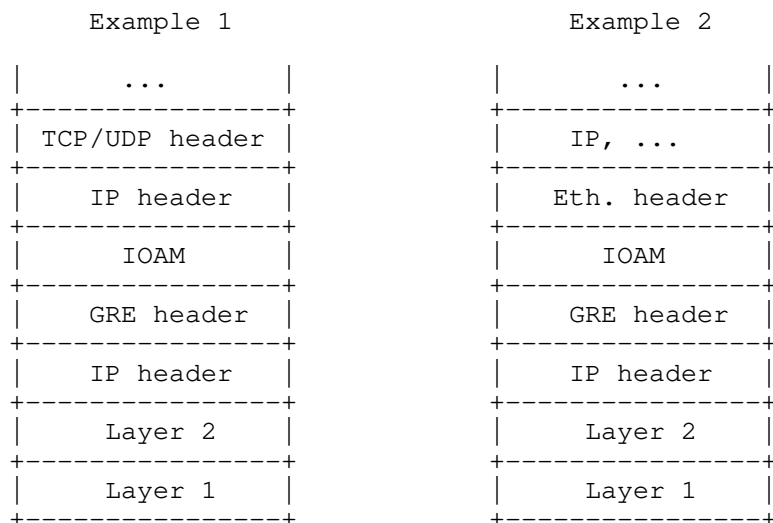


Figure 2: GRE with IOAM examples

5.5. Geneve

IOAM encapsulation for Geneve is defined in [I-D.brockners-ippm-ioam-geneve].

5.6. Segment Routing

IOAM encapsulation for Segment Routing is defined in [I-D.gandhi-spring-ioam-sr-mpls].

5.7. Segment Routing for IPv6

IOAM encapsulation for Segment Routing over IPv6 is defined in [I-D.ali-spring-ioam-srv6].

5.8. VXLAN-GPE

IOAM encapsulation for VXLAN-GPE is defined in [I-D.brockners-ippm-ioam-vxlan-gpe].

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]. "Raw export of IOAM data" refers to a mode of operation where a node exports the IOAM data as it is received in the packet. The exporting node neither interprets, aggregates nor reformats the IOAM data before it is exported. Raw export of IOAM data is to support an operational model where the processing and interpretation of IOAM data is decoupled from the operation of encapsulating/updating/decapsulating IOAM data, which is also referred to as IOAM data-plane operation. The figure below shows the separation of concerns for IOAM export: Exporting IOAM data is performed by the "IOAM node" which performs IOAM data-plane operation, whereas the interpretation of IOAM data is performed by one or several IOAM data processing systems. The separation of concerns is to off-load interpretation, aggregation and formatting of IOAM data from the node which performs data-plane operations. In other words, a node which is focused on data-plane operations, i.e. forwarding of packets and handling IOAM data will not be tasked to also interpret the IOAM data, but can leave this task to another system or a set of systems. For scalability reasons, a single IOAM node could choose to export IOAM data to several IOAM data processing systems. Similarly, there several monitoring systems or analytics systems can be used to further process the data received from the IOAM preprocessing systems. Figure 3 shows an overview of IOAM export, including IOAM data processing systems and monitoring/analytics systems.

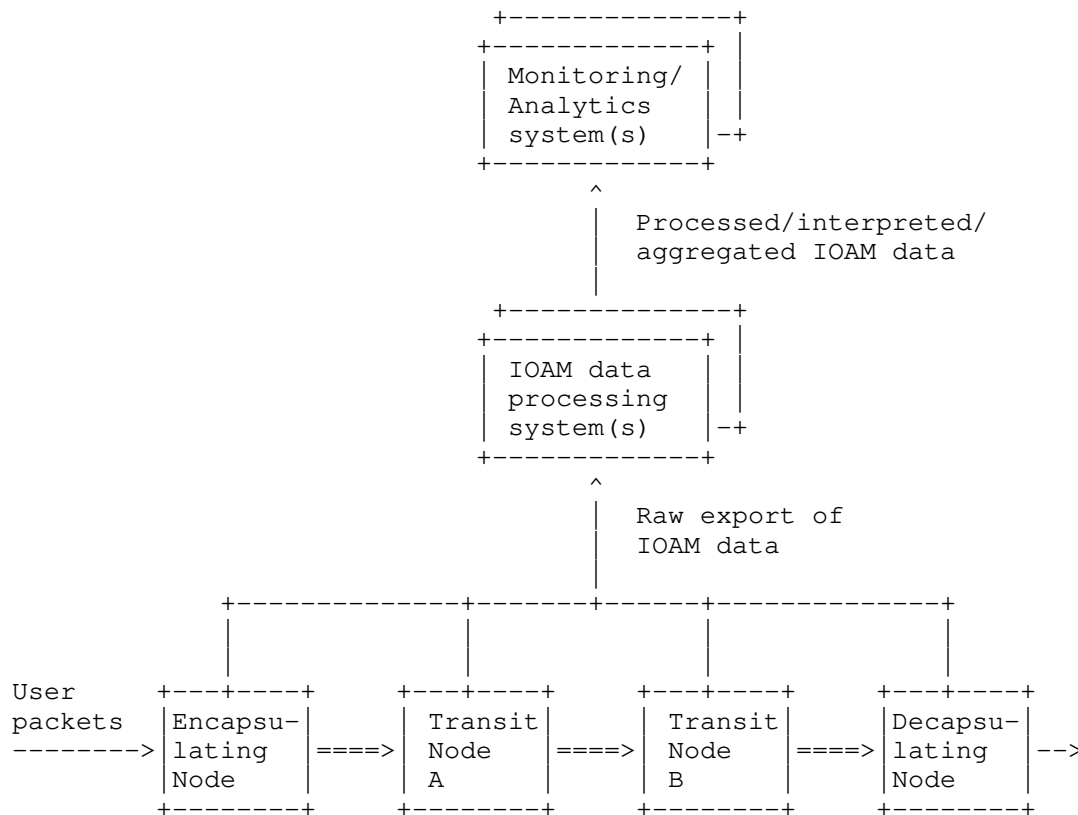


Figure 3: IOAM framework with data export

7. IOAM Deployment Considerations

This section discusses several aspects of an IOAM deployment, including IOAM Namespaces, IOAM Layering, traffic-sets that IOAM is applied to and IOAM loopback mode.

7.1. IOAM Namespaces

IOAM-Namespaces add further context to IOAM-Option-Types and associated IOAM-Data-Fields. IOAM-Namespaces support several different uses:

- o IOAM-Namespaces can be used by an operator to distinguish different operational domains. Devices at domain edges can filter on Namespace-IDs to provide for proper IOAM-Domain isolation.

- o IOAM-Namespaces provide additional context for IOAM-Data-Fields and thus ensure that IOAM-Data-Fields are unique and can be interpreted properly by management stations or network controllers. While, for example, the node identifier field does not need to be unique in a deployment (e.g., an operator may wish to use different node identifiers for different IOAM layers, even within the same device; or node identifiers might not be unique for other organizational reasons, such as after a merger of two formerly separated organizations), the combination of node_id and Namespace-ID should always be unique. Similarly, IOAM-Namespaces can be used to define how certain IOAM-Data-Fields are interpreted: IOAM offers three different timestamp format options. The Namespace-ID can be used to determine the timestamp format. IOAM-Data-Fields (e.g., buffer occupancy) which do not have a unit associated are to be interpreted within the context of a IOAM-Namespace.
- o IOAM-Namespaces can be used to identify different sets of devices (e.g., different types of devices) in a deployment: If an operator desires to insert different IOAM-Data-Fields based on the device, the devices could be grouped into multiple IOAM-Namespaces. This could be due to the fact that the IOAM feature set differs between different sets of devices, or it could be for reasons of optimized space usage in the packet header. It could also stem from hardware or operational limitations on the size of the trace data that can be added and processed, preventing collection of a full trace for a flow.
- * Assigning different IOAM Namespace-IDs to different sets of nodes or network partitions and using the Namespace-ID as a selector at the IOAM encapsulating node, a full trace for a flow could be collected and constructed via partial traces in different packets of the same flow. Example: An operator could choose to group the devices of a domain into two IOAM-Namespaces, in a way that at average, only every second hop would be recorded by any device. To retrieve a full view of the deployment, the captured IOAM-Data-Fields of the two IOAM-Namespaces need to be correlated.
- * Assigning different IOAM Namespace-IDs to different sets of nodes or network partitions and using a separate instance of an IOAM-Option-Type for each Namespace-ID, a full trace for a flow could be collected and constructed via partial traces from each IOAM-Option-Type in each of the packets in the flow. Example: An operator could choose to group the devices of a domain into two IOAM-Namespaces, in a way that each IOAM-Namespace is represented by one of two IOAM-Option-Types in the packet. Each node would record data only for the IOAM-Namespace that it

belongs to, ignoring the other IOAM-Option-Type with a IOAM-Namespace to which it doesn't belong. To retrieve a full view of the deployment, the captured IOAM-Data-Fields of the two IOAM-Namespace need to be correlated.

7.2. IOAM Layering

If several encapsulation protocols (e.g., in case of tunneling) are stacked on top of each other, IOAM-Data-Fields could be present in different protocol fields at different layers. Layering allows operators to instrument the protocol layer they want to measure. The behavior follows the ships-in-the-night model, i.e. IOAM-Data-Fields in one layer are independent from IOAM-Data-Fields in another layer. Or in other words: Even though the term "layering" often implies some form of hierarchy and relationship, in IOAM, layers are independent from each other and don't assume any relationship among them. The different layers could, but do not have to share the same IOAM encapsulation mechanisms. Similarly, the semantics of the IOAM-Data-Fields, can, do not have to be associated to across different layers. For example, a node which inserts node-id information into two different layers could use "node-id=10" for one layer and "node-id=1000" for the second layer.

Figure 4 shows an example of IOAM layering. The figure shows a Geneve tunnel carried over IPv6 which starts at node A and ends at node D. IOAM information is encapsulated in IPv6 as well as in Geneve. At the IPv6 layer, node A is IOAM encapsulating node (into IPv6), node D is the IOAM decapsulating node and node B and node C are IOAM transit nodes. At the Geneve layer, node A is IOAM encapsulating node (into Geneve) and node D is IOAM decapsulating node (from Geneve). The use of IOAM at both layers as shown in the example here could be used to reveal which nodes of an underlay (here the IPv6 network) are traversed by tunneled packet in an overlay (here the Geneve network) - which assumes that the IOAM information encapsulated by nodes A and D into Geneve and IPv6 is associated to each other.

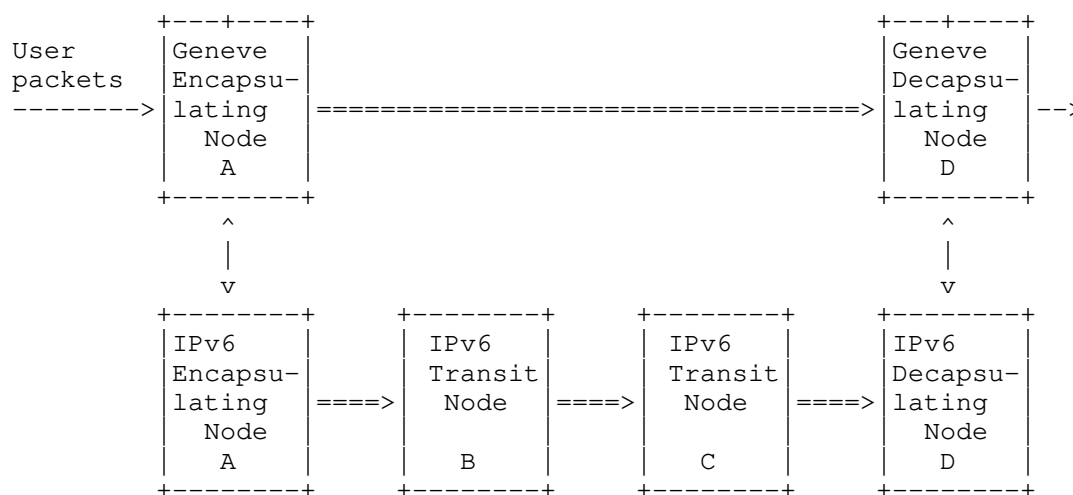


Figure 4: IOAM layering example

7.3. IOAM Trace Option Types

IOAM offers two different IOAM Option-Types for tracing:

"Incremental" Trace-Option-Type and "Pre-allocated" Trace-Option-Type. "Incremental" refers to a mode of operation where the packet is expanded at every IOAM node that adds IOAM-Data-Fields. "Pre-Allocated" describes a mode of operation where the IOAM encapsulating node allocates room for all IOAM-Data-Fields in the entire IOAM domain. More specifically:

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 implementations where it is efficient to allocate the space once and index into the array to populate the data during transit (e.g., software forwarders often fall into this class).

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.

A deployment can choose to configure and support one or both of the IOAM Trace-Option-Types. The operator decides by means of configuration which Trace-Option-Type(s) will be used for a particular domain. Deployments can mix devices which include either the Incremental Trace-Option-Type or the Pre-allocated Trace-Option-Type, e.g., in case different types of packet forwarders and

associated different types of IOAM implementations exist in a deployment. As a result, both Option-Types can be present in a packet. IOAM decapsulating nodes remove both types of Trace-Option-Types from the packet.

The two different Option-Types cater to different packet forwarding infrastructures and are to allow an optimized implementation of IOAM tracing:

Pre-allocated Trace-Option: For some implementations of packet forwarders it is efficient to allocate the space for the maximum number of nodes that IOAM Trace Data-Fields should be collected from and insert/update information in the packet at flexible locations, based on a pointer retrieved from a field in the packet. The IOAM encapsulating node allocates an array of the size of the maximum number of nodes that IOAM Trace Data-Fields should be collected from. IOAM transit nodes index into the array to populate the data during transit. Software forwarders often fall into this class of packet forwarder implementations. The maximum number of nodes that IOAM information could be collected from is configured by the operator on the IOAM encapsulating node. The operator has to ensure that the packet with the pre-allocated array that carries the IOAM Data-Fields does not exceed the MTU of any of the links in the IOAM domain.

Incremental Trace-Option: Looking up a pointer contained in the packet and inserting/updating information at a flexible location in the packet as a result of the pointer lookup is costly for some forwarding infrastructures. Hardware-based packet forwarding infrastructures often fall into this category. Consequently, hardware-based packet forwarders could choose to support the incremental IOAM-Trace-Option-Type. The incremental IOAM-Trace-Option-Type eliminates the need for the IOAM transit nodes to read the full array in the Trace-Option-Type and allows packets to grow to the size of the MTU of the IOAM domain. IOAM transit nodes will expand the packet and insert the IOAM-Data-Fields as long as there is space available in the packet, i.e. as long as the size of the packet stays within the bounds of the MTU of any of the links in the IOAM domain. There is no need for the operator to configure the IOAM encapsulation node with the maximum number of nodes that IOAM information could be collected from. The operator has to ensure that the minimum MTU of any of the links in the IOAM domain is known to all IOAM transit nodes.

7.4. Traffic-sets That IOAM Is Applied To

IOAM can be deployed on all or only on subsets of the live user traffic, e.g., per interface, based on an access control list or flow specification defining a specific set of traffic, etc.

7.5. IOAM Loopback Mode

IOAM Loopback is used to trigger each transit device along the path of a packet to send a copy of the data packet back to the source. Loopback allows an IOAM encapsulating node to trace the path to a given destination, and to receive per-hop data about both the forward and the return path. Loopback is enabled by the encapsulating node setting the loopback flag. Looped-back packets use the source address of the original packet as destination address and the address of the node which performs the loopback operation as source address. Nodes which loop back a packet clear the loopback flag before sending the copy back towards the source. Loopback applies to IOAM deployments where the encapsulating node is either a host or the start of a tunnel: For details on IOAM loopback, please refer to [I-D.ietf-ippm-ioam-flags].

7.6. IOAM Active Mode

The IOAM active mode flag indicates that a packet is an active OAM packet as opposed to regular user data traffic. Active mode is expected to be used for active measurement using IOAM. For details on IOAM active mode, please refer to [I-D.ietf-ippm-ioam-flags].

Example use-cases for IOAM Active Mode include:

- o Endpoint detailed active measurement: Synthetic probe packets are sent between the source and destination. These probe packets include a Trace Option-Type (i.e., either incremental or pre-allocated). Since the probe packets are sent between the endpoints, these packets are treated as data packets by the IOAM domain, and do not require special treatment at the IOAM layer. The source, which is also the IOAM encapsulating node can choose to set the Active flag, providing an explicit indication that these probe packets are meant for telemetry collection.
- o IOAM active measurement using probe packets: Probe packets are generated and transmitted by an IOAM encapsulating node towards a destination which is also the IOAM decapsulating node. Probe packets include a Trace Option-Type (i.e., either incremental or pre-allocated) which has its Active flag set.

- o IOAM active measurement using replicated data packets: Probe packets are created by an IOAM encapsulating node by selecting some or all of the en route data packets and replicating them. A selected data packet that is replicated, and its (possibly truncated) copy is forwarded with one or more IOAM option, while the original packet is forwarded normally, without IOAM options. To the extent possible, the original data packet and its replica are forwarded through the same path. The replica includes a Trace Option-Type that has its Active flag set, indicating that the IOAM decapsulating node should terminate it. In this case the IOAM Active flag ensures that the replicated traffic is not forwarded beyond the IOAM domain.

7.7. Brown Field Deployments: IOAM Unaware Nodes

A network can consist of a mix of IOAM aware and IOAM unaware nodes. The encapsulation of IOAM-Data-Fields into different protocols (see also Section 5) are defined such that data packets that include IOAM-Data-Fields do not get dropped by IOAM unaware nodes. For example, packets which contain the IOAM-Trace-Option-Types in IPv6 Hop by Hop extension headers are defined with bits to indicate "00 - skip over this option and continue processing the header". This will ensure that when a node that is IOAM unaware receives a packet with IOAM-Data-Fields included, does not drop the packet.

Deployments which leverage the IOAM-Trace-Option-Type(s) could benefit from the ability to detect the presence of IOAM unaware nodes, i.e. nodes which forward the packet but do not update/add IOAM-Data-Fields in IOAM-Trace-Option-Type(s). The node data that is defined as part of the IOAM-Trace-Option-Type(s) includes a Hop_Lim field associated to the node identifier to detect missed nodes, i.e. "holes" in the trace. Monitoring/Analytics system(s) could utilize this information to account for the presence of IOAM unaware nodes in the network.

8. IOAM Manageability

The YANG model for configuring IOAM in network nodes which support IOAM is defined in [I-D.zhou-ippm-ioam-yang].

A deployment can leverage IOAM profiles is to limit the scope of IOAM features, allowing simpler implementation, verification, and interoperability testing in the context of specific use cases that do not require the full functionality of IOAM. An IOAM profile defines a use case or a set of use cases for IOAM, and an associated set of rules that restrict the scope and features of the IOAM specification, thereby limiting it to a subset of the full functionality. IOAM profiles are defined in [I-D.mizrahi-ippm-ioam-profile].

9. IANA Considerations

This document does not request any IANA actions.

10. 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-Type (Section 4.2) is used for verifying the path of data packets. The security considerations of POT are further discussed in [I-D.ietf-sfc-proof-of-transit].

Security considerations related to the use of IOAM flags, in particular the loopback flag are found in [I-D.ietf-ippm-ioam-flags].

IOAM data can be subject to eavesdropping. Although the confidentiality of the user data is not at risk in this context, the IOAM data elements can be used for network reconnaissance, allowing attackers to collect information about network paths, performance, queue states, buffer occupancy and other information. Recon is an improbable security threat in an IOAM deployment that is within a confined physical domain. However, in deployments that are not confined to a single LAN, but span multiple inter-connected sites (for example, using an overlay network), the inter-site links can be secured (e.g., by IPsec) in order to avoid external eavesdropping. Another possible mitigation approach is to use the "direct exporting" mode [I-D.ietf-ippm-ioam-direct-export]. In this case the IOAM related trace information would not be available in the customer data packets, but would trigger exporting of (secured) packet-related IOAM information at every node. IOAM data export and securing IOAM data export is outside the scope of this document.

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 or modify an IOAM header in en route 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 headers associated with IOAM Option-Types into data packets, causing these packets to be increased beyond the MTU size, resulting in fragmentation or in packet drops. Such DoS attacks can be mitigated by deploying IOAM in confined administrative domains, and by defining performance limits on IOAM encapsulation and IOAM exporting. By limiting the rate and/or percentage of packets that

are subject to IOAM encapsulation and the rate of exported traffic, it is possible to confine the impact of such attacks.

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. Synchronization attacks can be mitigated by combining a secured time distribution scheme, e.g., [RFC8915], and by using redundant clock sources [RFC5905] and/or redundant network paths for the time distribution protocol [RFC8039].

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. Note that the Immediate Export mode (reference to be added in a future revision) can mitigate the potential threat of IOAM data leaking through data packets.

11. Acknowledgements

The authors would like to thank Tal Mizrahi, Eric Vyncke, Nalini Elkins, Srihari Raghavan, Ranganathan T S, Barak Gafni, Karthik Babu Harichandra Babu, Akshaya Nadahalli, LJ Wobker, Erik Nordmark, Vengada Prasad Govindan, Andrew Yourtchenko, Aviv Kfir, Tianran Zhou, Zhenbin (Robin), Joe Clarke, Al Morton, Tom Herbet, Haoyu song, and Mickey Spiegel for the comments and advice on IOAM.

12. References

12.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

12.2. Informative References

[I-D.ali-spring-ioam-srv6]

Ali, Z., Gandhi, R., Filsfils, C., Brockners, F., Nainar, N., Pignataro, C., Li, C., Chen, M., and G. Dawra, "Segment Routing Header encapsulation for In-situ OAM Data", draft-ali-spring-ioam-srv6-05 (work in progress), January 2022.

[I-D.brockners-ippm-ioam-geneve]

Brockners, F., Bhandari, S., Govindan, V. P., Pignataro, C., Nainar, N. K., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Lapukhov, P., Gafni, B., Kfir, A., and M. Spiegel, "Geneve encapsulation for In-situ OAM Data", draft-brockners-ippm-ioam-geneve-05 (work in progress), November 2020.

[I-D.brockners-ippm-ioam-vxlan-gpe]

Brockners, F., Bhandari, S., Govindan, V. P., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Kfir, A., Gafni, B., Lapukhov, P., and M. Spiegel, "VXLAN-GPE Encapsulation for In-situ OAM Data", draft-brockners-ippm-ioam-vxlan-gpe-03 (work in progress), November 2019.

[I-D.gandhi-spring-ioam-sr-mpls]

Gandhi, R., Ali, Z., Filsfils, C., Brockners, F., Wen, B., and V. Kozak, "Segment Routing with MPLS Data Plane Encapsulation for In-situ OAM Data", draft-gandhi-spring-ioam-sr-mpls-02 (work in progress), August 2019.

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

Brockners, F., Bhandari, S., and T. Mizrahi, "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data-17 (work in progress), December 2021.

[I-D.ietf-ippm-ioam-direct-export]

Song, H., Gafni, B., Zhou, T., Li, Z., Brockners, F., Bhandari, S., Sivakolundu, R., and T. Mizrahi, "In-situ OAM Direct Exporting", draft-ietf-ippm-ioam-direct-export-07 (work in progress), October 2021.

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

Mizrahi, T., Brockners, F., Bhandari, S., Sivakolundu, R., Pignataro, C., Kfir, A., Gafni, B., Spiegel, M., and J. Lemon, "In-situ OAM Loopback and Active Flags", draft-ietf-ippm-ioam-flags-07 (work in progress), October 2021.

- [I-D.ietf-ippm-ioam-ipv6-options]
Bhandari, S. and F. Brockners, "In-situ OAM IPv6 Options", draft-ietf-ippm-ioam-ipv6-options-07 (work in progress), February 2022.
- [I-D.ietf-nvo3-geneve]
Gross, J., Ganga, I., and T. Sridhar, "Geneve: Generic Network Virtualization Encapsulation", draft-ietf-nvo3-geneve-16 (work in progress), March 2020.
- [I-D.ietf-nvo3-vxlan-gpe]
(Editor), F. M., (editor), L. K., and U. E. (editor), "Generic Protocol Extension for VXLAN (VXLAN-GPE)", draft-ietf-nvo3-vxlan-gpe-12 (work in progress), September 2021.
- [I-D.ietf-sfc-ioam-nsh]
Brockners, F. and S. Bhandari, "Network Service Header (NSH) Encapsulation for In-situ OAM (IOAM) Data", draft-ietf-sfc-ioam-nsh-08 (work in progress), April 2022.
- [I-D.ietf-sfc-proof-of-transit]
Brockners, F., Bhandari, S., Mizrahi, T., Dara, S., and S. Youell, "Proof of Transit", draft-ietf-sfc-proof-of-transit-08 (work in progress), November 2020.
- [I-D.mizrahi-ippm-ioam-profile]
Mizrahi, T., Brockners, F., Bhandari, S., Sivakolundu, R., Pignataro, C., Kfir, A., Gafni, B., Spiegel, M., and T. Zhou, "In Situ OAM Profiles", draft-mizrahi-ippm-ioam-profile-06 (work in progress), February 2022.
- [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-06 (work in progress), February 2022.
- [I-D.weis-ippm-ioam-eth]
Weis, B., Brockners, F., Hill, C., Bhandari, S., Govindan, V. P., Pignataro, C., Nainar, N. K., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Kfir, A., Gafni, B., Lapukhov, P., and M. Spiegel, "EtherType Protocol Identification of In-situ OAM Data", draft-weis-ippm-ioam-eth-05 (work in progress), February 2022.

- [I-D.xzlnp-bier-ioam]
Min, X., Zhang, Z., Liu, Y., Nainar, N. K., and C. Pignataro, "Bit Index Explicit Replication (BIER) Encapsulation for In-situ OAM (IOAM) Data", draft-xzlnp-bier-ioam-03 (work in progress), January 2022.
- [I-D.zhou-ippm-ioam-yang]
Zhou, T., Guichard, J., Brockners, F., and S. Raghavan, "A YANG Data Model for In-Situ OAM", draft-zhou-ippm-ioam-yang-08 (work in progress), July 2020.
- [RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <<https://www.rfc-editor.org/info/rfc5905>>.
- [RFC7276] Mizrahi, T., Sprecher, N., Bellagamba, E., and Y. Weingarten, "An Overview of Operations, Administration, and Maintenance (OAM) Tools", RFC 7276, DOI 10.17487/RFC7276, June 2014, <<https://www.rfc-editor.org/info/rfc7276>>.
- [RFC7384] Mizrahi, T., "Security Requirements of Time Protocols in Packet Switched Networks", RFC 7384, DOI 10.17487/RFC7384, October 2014, <<https://www.rfc-editor.org/info/rfc7384>>.
- [RFC7665] Halpern, J., Ed. and C. Pignataro, Ed., "Service Function Chaining (SFC) Architecture", RFC 7665, DOI 10.17487/RFC7665, October 2015, <<https://www.rfc-editor.org/info/rfc7665>>.
- [RFC7799] Morton, A., "Active and Passive Metrics and Methods (with Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799, May 2016, <<https://www.rfc-editor.org/info/rfc7799>>.
- [RFC8039] Shpiner, A., Tse, R., Schelp, C., and T. Mizrahi, "Multipath Time Synchronization", RFC 8039, DOI 10.17487/RFC8039, December 2016, <<https://www.rfc-editor.org/info/rfc8039>>.
- [RFC8300] Quinn, P., Ed., Elzur, U., Ed., and C. Pignataro, Ed., "Network Service Header (NSH)", RFC 8300, DOI 10.17487/RFC8300, January 2018, <<https://www.rfc-editor.org/info/rfc8300>>.
- [RFC8799] Carpenter, B. and B. Liu, "Limited Domains and Internet Protocols", RFC 8799, DOI 10.17487/RFC8799, July 2020, <<https://www.rfc-editor.org/info/rfc8799>>.

[RFC8915] Franke, D., Sibold, D., Teichel, K., Dansarie, M., and R. Sundblad, "Network Time Security for the Network Time Protocol", RFC 8915, DOI 10.17487/RFC8915, September 2020, <<https://www.rfc-editor.org/info/rfc8915>>.

Authors' Addresses

Frank Brockners (editor)
Cisco Systems, Inc.
Hansaallee 249, 3rd Floor
DUESSELDORF, NORDRHEIN-WESTFALEN 40549
Germany

Email: fbrockne@cisco.com

Shwetha Bhandari (editor)
Thoughtspot
3rd Floor, Indiqube Orion, 24th Main Rd, Garden Layout, HSR Layout
Bangalore, KARNATAKA 560 102
India

Email: shwetha.bhandari@thoughtspot.com

Daniel Bernier
Bell Canada
Canada

Email: daniel.bernier@bell.ca

Tal Mizrahi (editor)
Huawei
8-2 Matam
Haifa 3190501
Israel

Email: tal.mizrahi.phd@gmail.com

IPPM Working Group
Internet-Draft
Intended status: Standards Track
Expires: 6 August 2022

R. Gandhi, Ed.
C. Filsfils
Cisco Systems, Inc.
D. Voyer
Bell Canada
M. Chen
Huawei
B. Janssens
Colt
R. Foote
Nokia
2 February 2022

Simple TWAMP (STAMP) Extensions for Segment Routing Networks
draft-ietf-ippm-stamp-srpm-03

Abstract

Segment Routing (SR) leverages the source routing paradigm. SR is applicable to both Multiprotocol Label Switching (SR-MPLS) and IPv6 (SRv6) forwarding planes. This document specifies RFC 8762 (Simple Two-Way Active Measurement Protocol (STAMP)) extensions for SR networks, for both SR-MPLS and SRv6 forwarding planes by augmenting the optional extensions defined in RFC 8972.

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 6 August 2022.

Copyright Notice

Copyright (c) 2022 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 Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction	2
2. Conventions Used in This Document	3
2.1. Requirements Language	3
2.2. Abbreviations	3
2.3. Reference Topology	4
3. Destination Node Address TLV	4
4. Return Path TLV	6
4.1. Return Path Sub-TLVs	7
4.1.1. Return Path Control Code Sub-TLV	7
4.1.2. Return Address Sub-TLV	8
4.1.3. Return Segment List Sub-TLVs	8
5. Security Considerations	11
6. IANA Considerations	12
7. References	13
7.1. Normative References	13
7.2. Informative References	14
Acknowledgments	15
Authors' Addresses	15

1. Introduction

Segment Routing (SR) leverages the source routing paradigm for Software Defined Networks (SDNs). SR is applicable to both Multiprotocol Label Switching (SR-MPLS) and IPv6 (SRv6) forwarding planes [RFC8402]. SR Policies as defined in [I-D.ietf-spring-segment-routing-policy] are used to steer traffic through a specific, user-defined paths using a stack of Segments. A comprehensive SR Performance Measurement (PM) toolset is one of the essential requirements to measure network performance to provide Service Level Agreements (SLAs).

The Simple Two-Way Active Measurement Protocol (STAMP) provides capabilities for the measurement of various performance metrics in IP networks [RFC8762] without the use of a control channel to pre-signal session parameters. [RFC8972] defines optional extensions, in the form of TLVs, for STAMP. Note that the YANG data model defined in [I-D.ietf-ippm-stamp-yang] can be used to provision the STAMP Session-Sender and STAMP Session-Reflector.

The STAMP test packets are transmitted along an IP path between a Session-Sender and a Session-Reflector to measure performance delay and packet loss along that IP path. It may be desired in SR networks that the same path (same set of links and nodes) between the Session-Sender and Session-Reflector is used for the STAMP test packets in both directions. This is achieved by using the STAMP [RFC8762] extensions for SR-MPLS and SRv6 networks specified in this document by augmenting the optional extensions defined in [RFC8972].

2. Conventions Used in This Document

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 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2.2. Abbreviations

MPLS: Multiprotocol Label Switching.

PM: Performance Measurement.

SID: Segment ID.

SL: Segment List.

SR: Segment Routing.

SR-MPLS: Segment Routing with MPLS forwarding plane.

SRv6: Segment Routing with IPv6 forwarding plane.

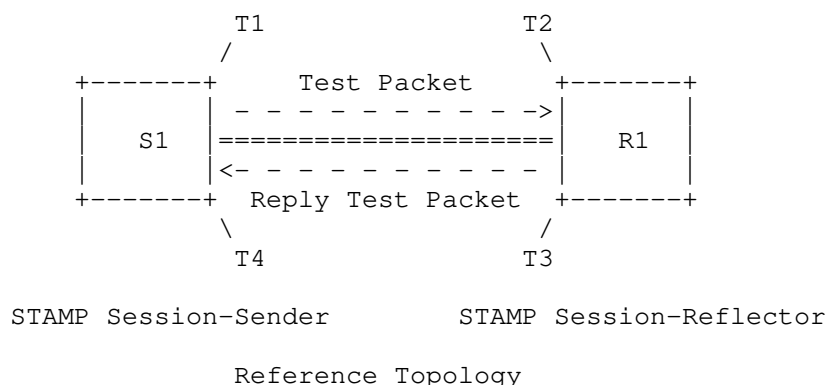
SSID: STAMP Session Identifier.

STAMP: Simple Two-Way Active Measurement Protocol.

2.3. Reference Topology

In the reference topology shown below, the STAMP Session-Sender S1 initiates a STAMP test packet and the STAMP Session-Reflector R1 transmits a reply STAMP test packet. The reply test packet may be transmitted to the Session-Sender S1 on the same path (same set of links and nodes) or a different path in the reverse direction from the path taken towards the Session-Reflector R1.

The nodes S1 and R1 may be connected via a link or an SR path [RFC8402]. The link may be a physical interface, virtual link, or Link Aggregation Group (LAG) [IEEE802.1AX], or LAG member link. The SR path may be an SR Policy [I-D.ietf-spring-segment-routing-policy] on node S1 (called head-end) with destination to node R1 (called tail-end).



3. Destination Node Address TLV

The Session-Sender may need to transmit test packets to the Session-Reflector with a different destination address not matching an address on the Session-Reflector e.g. when the STAMP test packet is encapsulated by a tunneling protocol or an MPLS Segment List with IPv4 address from 127/8 range or Segment Routing Header (SRH) with IPv6 address `::1/128`. For testing ECMPs, the Session-Sender may select different IPv4 addresses from 127/8 range or select different Flow Label values for IPv6. When using IPv4 destination address from 127/8 range, the STAMP test packet may not reach the intended Session-Reflector in an error condition, an un-intended node may transmit reply test packets resulting in reporting of invalid measurement metrics.

4. Return Path TLV

For end-to-end SR paths, the Session-Reflector may need to transmit the reply test packet on a specific return path. The Session-Sender can request this in the test packet to the Session-Reflector using a Return Path TLV. With this TLV carried in the Session-Sender test packet, signaling and maintaining dynamic SR network state for the STAMP sessions on the Session-Reflector are avoided.

For links, the Session-Reflector may need to transmit the reply test packet on the same incoming link in the reverse direction. The Session-Sender can request this in the test packet to the Session-Reflector using a Return Path TLV.

[RFC8972] defines STAMP test packets that can include one or more optional TLVs. In this document, the TLV Type (value TBA2) is defined for the Return Path TLV that carries the return path for the Session-Sender test packet. The format of the Return Path TLV is shown in Figure 2:

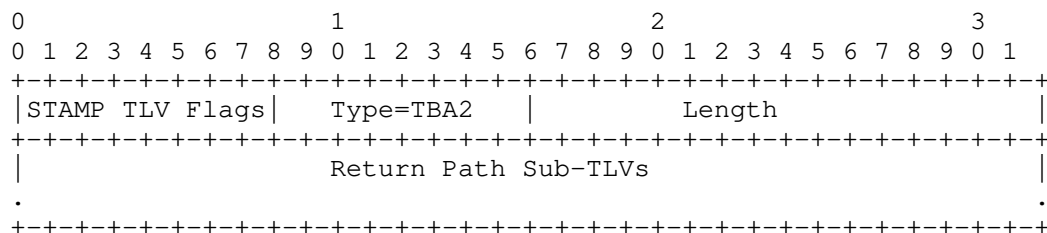


Figure 2: Return Path TLV

The STAMP TLV Flags are set using the procedures described in [RFC8972].

The Return Path TLV is optional. The Session-Sender MUST only insert one Return Path TLV in the STAMP test packet. The Session-Reflector that supports this TLV, MUST only process the first Return Path TLV in the test packet and ignore other Return Path TLVs if present, and it MUST NOT add Return Path TLV in the reply test packet. The Session-Reflector that supports this TLV MUST reply using the Return Path received in the Session-Sender test packet. Otherwise, the procedure defined in [RFC8762] is followed by the Session-Reflector.

Session-Reflector may locally stream performance metrics via telemetry using the information from the received test packet. All other Return Path Sub-TLVs MUST be ignored in this case.

When Control Code flag is set to 0x1 in the Session-Sender test packet, the Session-Reflector transmits the reply test packet over the same incoming link where the test packet is received in the reverse direction towards the Session-Sender. All other Return Path Sub-TLVs MUST be ignored in this case.

4.1.2. Return Address Sub-TLV

The STAMP reply test packet may be transmitted to the Session-Sender to a different destination address on the Session-Sender using Return Path TLV. For this, the Session-Sender can specify in the test packet the receiving destination node address for the Session-Reflector reply test packet. When transmitting the STAMP test packet to a different destination address, the Session-Sender MUST follow the procedure defined in Section 4.3 of [RFC8762].

The format of the Return Address Sub-TLV is shown in Figure 4. The Address Family field indicates the type of the address, and it SHALL be set to one of the assigned values in the "IANA Address Family Numbers" registry. The Type of the Return Address Sub-TLV is defined as following:

- * Type (value 2): Return Address. Destination node address of the Session-Reflector reply test packet different than the Source Address in the Session-Sender test packet.

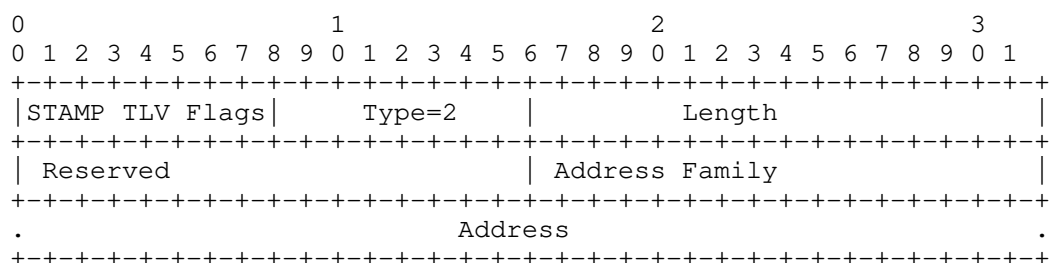


Figure 4: Return Address Sub-TLV in Return Path TLV

4.1.3. Return Segment List Sub-TLVs

The format of the Segment List Sub-TLVs in the Return Path TLV is shown in Figures 5, 6, and 7. The segment entries MUST be in network order. The Segment List Sub-TLV can be one of the following Types:

- * Type (value 3): SR-MPLS Label Stack of the Return Path
- * Type (value 4): SRv6 Segment List of the Return Path
- * Type (value 5): Structured SRv6 Segment List of the Return Path

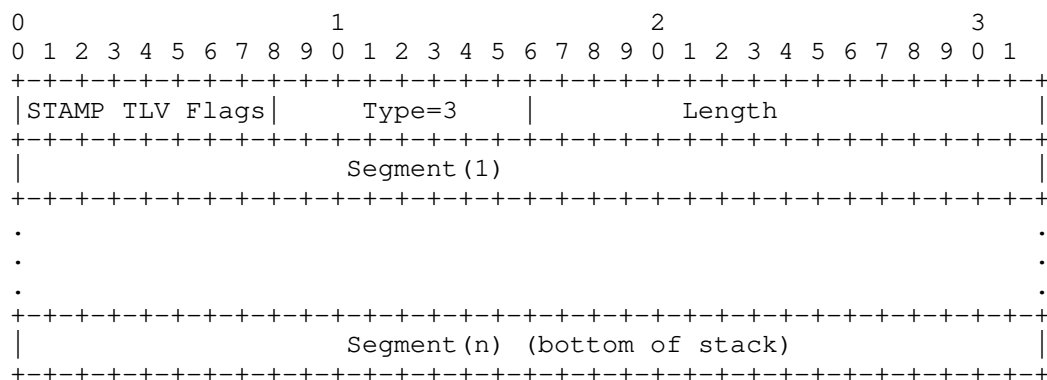


Figure 5: SR-MPLS Segment List Sub-TLV in Return Path TLV

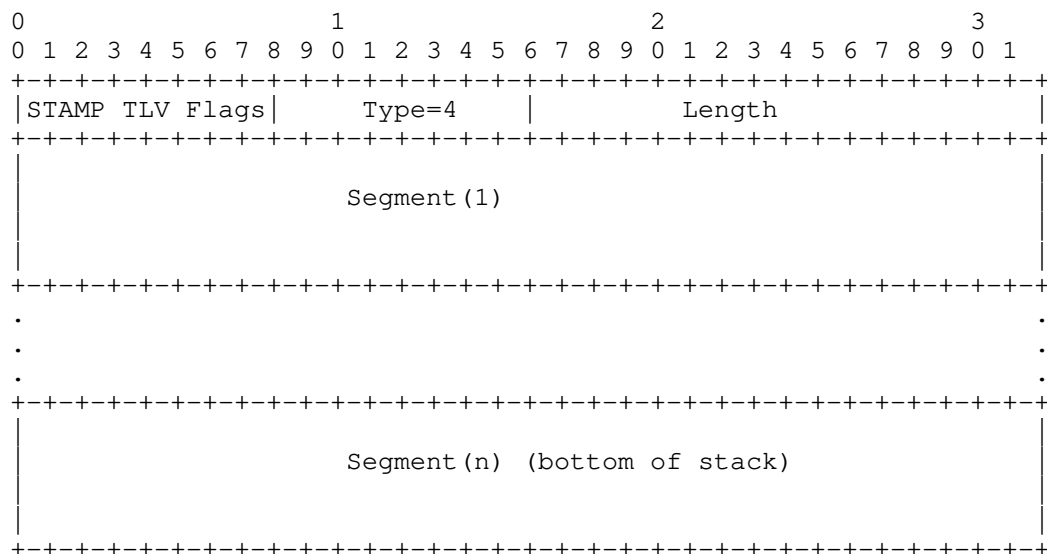


Figure 6: SRv6 Segment List Sub-TLV in Return Path TLV

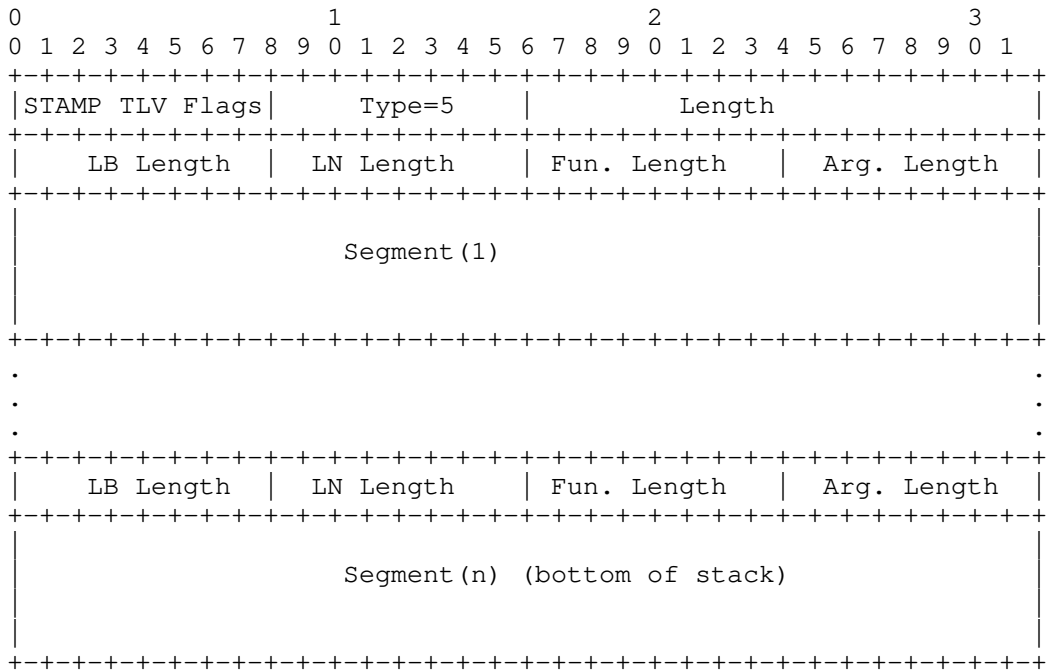


Figure 7: Structured SRv6 Segment List Sub-TLV in Return Path TLV

An SR-MPLS Label Stack Sub-TLV may carry only Binding SID [I-D.ietf-pce-binding-label-sid] of the Return SR-MPLS Policy.

An SRv6 Segment List Sub-TLV and Structured SRv6 Segment List Sub-TLV may carry only Binding SID [I-D.ietf-pce-binding-label-sid] of the Return SRv6 Policy.

A Structured SRv6 Segment List Sub-TLV is used carry the structure and behavior for SRv6 SIDs [RFC8986] used in the Return SRv6 path as shown in Figure 7. The structure is intended for informational use by the control and management planes. The fields in the structure of the Sub-TLV are defined as follows [RFC8986]:

- * LB Length: 1 octet. SRv6 SID Locator Block (LB) length in bits.
- * LN Length: 1 octet. SRv6 SID Locator Node (LN) length in bits.
- * Fun. Length: 1 octet. SRv6 SID Function length in bits.
- * Arg. Length: 1 octet. SRv6 SID Arguments length in bits.

In Structured SRv6 Segment List Sub-TLV, the sum of all four sizes MUST be less than or equal to 128 bits. If the sum of all four sizes is larger than 128 bits, the Sub-TLV MUST be ignored by the Session-Reflector.

The Session-Sender MUST only insert one Segment List Return Path Sub-TLV in the test packet. The Session-Reflector MUST only process the first Segment List Return Path Sub-TLV in the test packet and ignore other Segment List Return Path Sub-TLVs if present.

Note that in addition to P2P SR paths, the Return Segment List Sub-TLV is also applicable to P2MP SR paths. For example, for P2MP SR paths, it may only carry the Node Segment Identifier of the Session-Sender in order for the reply test packet to follow an SR path to the Session-Sender.

5. Security Considerations

The usage of STAMP protocol is intended for deployment in limited domains [RFC8799]. As such, it assumes that a node involved in STAMP protocol operation has previously verified the integrity of the path and the identity of the far-end Session-Reflector.

If desired, attacks can be mitigated by performing basic validation and sanity checks, at the Session-Sender, of the timestamp fields in received reply test packets. The minimal state associated with these protocols also limits the extent of measurement disruption that can be caused by a corrupt or invalid test packet to a single test cycle.

The security considerations specified in [RFC8762] and [RFC8972] also apply to the extensions defined in this document. Specifically, the message integrity protection using HMAC, as defined in [RFC8762] Section 4.4, also apply to the procedure described in this document.

STAMP uses the well-known UDP port number that could become a target of denial of service (DoS) or could be used to aid man-in-the-middle (MITM) attacks. Thus, the security considerations and measures to mitigate the risk of the attack documented in Section 6 of [RFC8545] equally apply to the STAMP extensions in this document.

The STAMP extensions defined in this document may be used for potential "proxying" attacks. For example, a Session-Sender may specify a return path that has a destination different from that of the Session-Sender. But normally, such attacks will not happen in an SR domain where the Session-Senders and Session-Reflectors belong to the same domain. In order to prevent using the extension defined in this document for proxying any possible attacks, the return path has destination to the same node where the forward path is from. The

Session-Reflector may drop the Session-Sender test packet when it cannot determine whether the Return Path has the destination to the Session-Sender. That means, the Session-Sender should choose a proper source address according to the specified Return Path to help the Session-Reflector to make that decision.

6. IANA Considerations

IANA has created the "STAMP TLV Types" registry for [RFC8972]. IANA is requested to allocate a value for the Destination Address TLV Type and a value for the Return Path TLV Type from the IETF Review TLV range of the same registry.

Value	Description	Reference
TBA1	Destination Node Address TLV	This document
TBA2	Return Path TLV	This document

Table 1: STAMP TLV Types

IANA is requested to create a sub-registry for "Return Path Sub-TLV Type". All code points in the range 1 through 175 in this registry shall be allocated according to the "IETF Review" procedure as specified in [RFC8126]. Code points in the range 176 through 239 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 Table 2:

Value	Description	Reference
1 - 175	IETF Review	This document
176 - 239	First Come First Served	This document
240 - 251	Experimental Use	This document
252 - 254	Private Use	This document

Table 2: Return Path Sub-TLV Type Registry

IANA is requested to allocate the values for the following Sub-TLV Types from this registry.

Type	Description	Reference
0	Reserved	This document
1	Return Path Control Code	This document
2	Return Address	This document
3	SR-MPLS Label Stack of the Return Path	This document
4	SRv6 Segment List of the Return Path	This document
5	Structured SRv6 Segment List of the Return Path	This document
255	Reserved	This document

Table 3: Return Path Sub-TLV Types

IANA has created the "STAMP TLV Flags" subregistry. IANA is requested to allocate the following bit position in the "STAMP TLV Flags" subregistry.

Bit Position	Symbol	Description	Reference
TBA3	D	Wrong Destination	This document

Table 4: STAMP TLV Flags

7. References

7.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

- [RFC8762] Mirsky, G., Jun, G., Nydell, H., and R. Foote, "Simple Two-Way Active Measurement Protocol", RFC 8762, DOI 10.17487/RFC8762, March 2020, <<https://www.rfc-editor.org/info/rfc8762>>.
- [RFC8972] Mirsky, G., Min, X., Nydell, H., Foote, R., Masputra, A., and E. Ruffini, "Simple Two-Way Active Measurement Protocol Optional Extensions", RFC 8972, DOI 10.17487/RFC8972, January 2021, <<https://www.rfc-editor.org/info/rfc8972>>.
- [RFC8986] Filsfils, C., Ed., Camarillo, P., Ed., Leddy, J., Voyer, D., Matsushima, S., and Z. Li, "Segment Routing over IPv6 (SRv6) Network Programming", RFC 8986, DOI 10.17487/RFC8986, February 2021, <<https://www.rfc-editor.org/info/rfc8986>>.

7.2. Informative References

- [RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <<https://www.rfc-editor.org/info/rfc8402>>.
- [RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.
- [RFC8545] Morton, A., Ed. and G. Mirsky, Ed., "Well-Known Port Assignments for the One-Way Active Measurement Protocol (OWAMP) and the Two-Way Active Measurement Protocol (TWAMP)", RFC 8545, DOI 10.17487/RFC8545, March 2019, <<https://www.rfc-editor.org/info/rfc8545>>.
- [RFC8799] Carpenter, B. and B. Liu, "Limited Domains and Internet Protocols", RFC 8799, DOI 10.17487/RFC8799, July 2020, <<https://www.rfc-editor.org/info/rfc8799>>.
- [I-D.ietf-spring-segment-routing-policy] Filsfils, C., Talaulikar, K., Voyer, D., Bogdanov, A., and P. Mattes, "Segment Routing Policy Architecture", Work in Progress, Internet-Draft, draft-ietf-spring-segment-routing-policy-14, 25 October 2021, <<https://www.ietf.org/archive/id/draft-ietf-spring-segment-routing-policy-14.txt>>.

[I-D.ietf-pce-binding-label-sid]
Sivabalan, S., Filsfils, C., Tantsura, J., Previdi, S.,
and C. L. (editor), "Carrying Binding Label/Segment
Identifier in PCE-based Networks.", Work in Progress,
Internet-Draft, draft-ietf-pce-binding-label-sid-12, 24
January 2022, <<https://www.ietf.org/archive/id/draft-ietf-pce-binding-label-sid-12.txt>>.

[I-D.ietf-ippm-stamp-yang]
Mirsky, G., Min, X., and W. S. Luo, "Simple Two-way Active
Measurement Protocol (STAMP) Data Model", Work in
Progress, Internet-Draft, draft-ietf-ippm-stamp-yang-09,
12 July 2021, <<https://www.ietf.org/archive/id/draft-ietf-ippm-stamp-yang-09.txt>>.

[IEEE802.1AX]
IEEE Std. 802.1AX, "IEEE Standard for Local and
metropolitan area networks - Link Aggregation", November
2008.

Acknowledgments

The authors would like to thank Thierry Couture for the discussions on the use-cases for Performance Measurement in Segment Routing. The authors would also like to thank Greg Mirsky, Mike Koldychev, Gyan Mishra, Tianran Zhou, Al Mortons, Reshad Rahman, Zhenqiang Li, Frank Brockners, and Cheng Li for providing comments and suggestions.

Authors' Addresses

Rakesh Gandhi (editor)
Cisco Systems, Inc.
Canada

Email: rgandhi@cisco.com

Clarence Filsfils
Cisco Systems, Inc.

Email: cfilsfil@cisco.com

Daniel Voyer
Bell Canada

Email: daniel.voyer@bell.ca

Mach (Guoyi) Chen
Huawei

Email: mach.chen@huawei.com

Bart Janssens
Colt

Email: Bart.Janssens@colt.net

Richard Foote
Nokia

Email: footer.foote@nokia.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: 14 August 2022

Y. Li
H. Yang
T. Sun
China Mobile
10 February 2022

One-way Delay Measurement Based on Deterministic Networking
draft-li-ippm-deterministic-owd-measurement-01

Abstract

One-way delay is a key indicator to measure network quality. Some applications are one-way transmission in the network, such as some high-definition video services, and are very sensitive to one-way delay. Excessive delay will affect user experience greatly. To some extent, the network can't even be used, so it is very important to accurately measure the network transmission delay. The current one-way delay measurement method has problems such as high complexity and low measurement accuracy. In order to solve the problem of high-precision one-way delay measurement, a one-way delay measurement method based on deterministic networking is proposed in this document. The method takes advantage of the delay characteristics of the deterministic networking and does not depend on precise time synchronization. The method realizes the one-way delay measurement of any service flow between any network elements. Its technical advantages are: the network does not need to send measurement packets, can test all traffic types, does not change network status, does not change the format of traffic packets, and does not require network elements to support time synchronization protocols.

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 14 August 2022.

Copyright Notice

Copyright (c) 2022 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 Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction	2
2. Conventions Used in This Document	3
2.1. Terminology	3
2.2. Requirements Language	4
3. One-way Delay Measurement Method Based on Deterministic Networking	4
4. Procedures of the One-way Delay Measurement Method	7
5. Security Considerations	8
6. IANA Considerations	8
7. Normative References	8
Authors' Addresses	9

1. Introduction

One-way transmission delay is a key indicator to measure network quality. Some applications are based on one-way transmission in the network, such as some high-definition video services, and are very sensitive to one-way delay. Excessive one-way delay will affect user experience dramatically, so it is very important to accurately measure the one-way transmission delay of the network.

There are several kinds of methods to measure one-way delay. The first kind of methods is active measurement. A sender will send measurement protocol messages, such as Two-Way Active Measurement Protocol (TWAMP) [RFC8186] messages, to the network to measure the one-way delay of the sender and receiver. The advantage of active measurement is that it is flexible in application. The disadvantage is that the measurement messages cannot measure the delay of real services, and the measurement of one-way delay requires sender and receiver to support time synchronization protocol, such as NTP [RFC5905] and PTP [IEEE.1588.2008]. The first kind of methods is passive measurement. The passive measurement devices will calculate

network delay by collecting actual business traffic. The advantage of passive measurement is that it can measure the one-way delay of real services. The disadvantage is that two passive measurement devices need to be deployed, and the two devices require time synchronization, which is difficult to implement. The third kind of methods is hybrid measurement. Hybrid measurement is a combination of active and passive measurements, that is, inserting some fields or flags in the service message to realize the delay measurement of the actual service. The disadvantage is that the message format of the actual service is changed, which will affect the forwarding behavior of the service and have observer effect. The network element needs to be able to recognize and forward the modified service message, and time synchronization of the network element is also required.

The above-mentioned one-way delay measurement methods have the following shortcomings. Firstly, if the measurement message is injected into actual network, it will occupy network bandwidth resources and interfere with the actual service flow, so the measured delay is not the delay of the actual service. Secondly, the measurement equipment or network elements need to support time synchronization protocols, which is difficult to implement and costly.

To address the following shortcomings of existing methods, this document presents the following technical solution. A high-precision one-way delay measurement method is proposed, which can be used to measure the one-way delay of actual service packets, without sending measurement messages, without changing the actual network status, without changing service messages, and without the need for network elements to support time synchronization protocols.

2. Conventions Used in This Document

2.1. Terminology

NTP Network Time Protocol

PTP Precision Time Protocol

TWAMP Two-Way Active Measurement Protocol

SLA Service Level Agreement

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. One-way Delay Measurement Method Based on Deterministic Networking

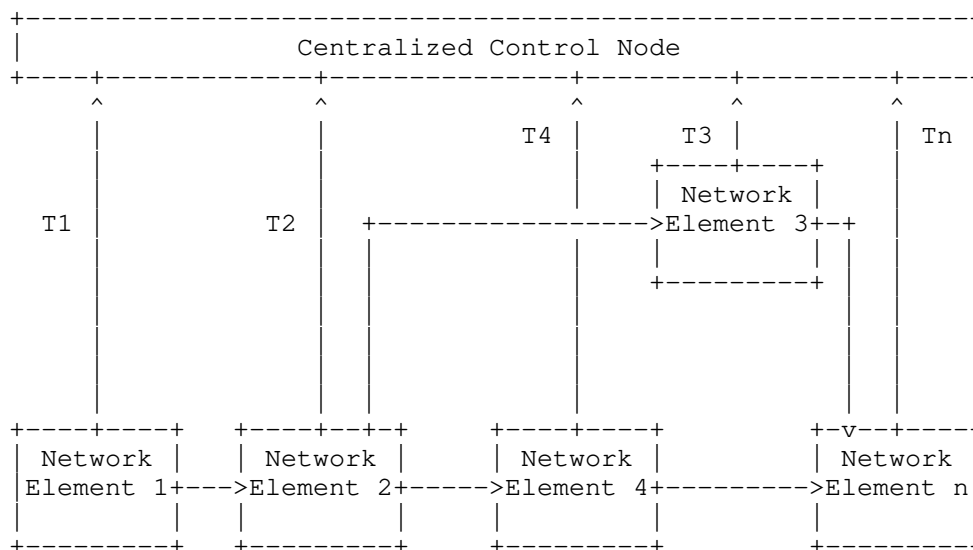


Figure 1: Figure 1: A schematic diagram of the network topology structure

A schematic diagram of the network topology structure to describe the proposed method is shown in Figure 1. The network may be a SDN (Software Defined Network) or a traditional network. Whether it is SDN or traditional network, there is a centralized control node (or called a centralized management unit) for collecting network information sent by network elements and sending control information to the network. Taking SDN as an example, the centralized control node can be a SDN controller. For traditional networks, the centralized control node can be a network management system. The information from the network element to the centralized control node generally passes through the management network. In our solution, the management network from each network element to the centralized control node is required to use a delay deterministic network. As an example, the delay deterministic network may be a time sensitive network (TSN) or a deterministic Internet (Deterministic Internet

Network, DIP) [RFC8655], etc. Through the delay deterministic network, the transmission delay of the network element information from the network element to the centralized control node can be guaranteed to be fixed. $T_1 \sim T_n$ in Figure 1 represent the network element information delay from the network element to the centralized control node of network element 1 to n respectively.

As shown in Figure 1, suppose network traffic of a real service flow passes through network element 1, network element 2, ..., network element n in turn, and the time when network traffic passes through the network element is recorded as t_1, t_2, \dots, t_n . The timestamp maybe the ingress timestamp of network traffic entering the network element or the egress timestamp of network traffic flowing out of the network element after the forwarding is completed. Each network element transmits the flow information to the centralized control node through the delay deterministic network when real traffic passes, and the transmission delays of each network element to transmit the flow information to the centralized control node through the delay deterministic network are denoted as T_1, T_2, \dots, T_n , respectively. The timestamps when the centralized control node receives the flow information of each network element are t_1', t_2', \dots, t_n' .

Taking the calculation of the one-way transmission delay of traffic from network element 1 to network element 2 as an example, the one-way transmission delay can be calculated in the following way. Firstly, because the clocks of network element 1 and network element 2 are not synchronized, suppose the time deviation between the two is Δt . Then the one-way transmission delay of traffic from network element 1 to network element 2 satisfies the following formula (1). Among them, Delay represents the one-way transmission delay of traffic from network element 1 to network element 2.

Formula (1): $\text{Delay} = t_2 - t_1 - \Delta t$

Secondly, because the clocks between network element 1 and the centralized control node are not synchronized, assuming that the time deviation between the two is $\Delta t'$, the time for the traffic information collected from the network element 1 to reach the centralized control node through the delay deterministic network satisfies the following formula (2).

Formula (2): $t_1' = t_1 + T_1 + \Delta t'$

Thirdly, the clocks between network element 2 and the centralized control node are not synchronized, and the time deviation between network element 2 and the centralized control node is $\text{delta_t}' - \text{delta_t}$. The time $t2'$ for the collected traffic to reach the centralized control node satisfies the following formula (3).

Formula (3): $t2' = t2 + T2 + \text{delta_t}' - \text{delta_t}$

Forthly, subtracting the formula (2) from the above formula (3), we can obtain the following formula (4).

Formula (4): $t2 - t1 - \text{delta_t} = t2' - t1' + T1 - T2$

Fifthly, substituting the above formula (4) into the above formula (1), the following formula (5) can be obtained.

Formula (5): $\text{Delay} = t2' - t1' + T1 - T2$

So far, the one-way transmission delay of traffic from network element 1 to network element 2 is obtained. Taking the calculation of one-way transmission delay of traffic from network element 1 to network element 3 as an example, the one-way transmission delay can be calculated in the following way: I) Referring to the above formula (5), the one-way transmission delay of traffic from network element 1 to network element 2 is: $\text{Delay}_{12} = t2' - t1' + T1 - T2$. II) Referring to the above formula (5), the one-way transmission delay of traffic from network element 2 to network element 3 is: $\text{Delay}_{23} = t3' - t2' + T2 - T3$. III) The one-way transmission delay of traffic from network element 1 to network element 3 is: $\text{Delay}_{13} = \text{Delay}_{12} + \text{Delay}_{23} = t2' - t1' + T1 - T2 + t3' - t2' + T2 - T3 = t3' - t1' + T1 - T3$. It can be seen that the one-way transmission delay between any two network elements can be calculated similarly to the above formula (5). For example, taking network element m and network element n as an example, the transmission delay of traffic from network element m to network element n is: $\text{Delay} = tn' - tm' + Tm - Tn$, where tn' and tm' are the time when the traffic information of network element m and network n reaches the centralized control node, and Tm and Tn are transmission delay of the traffic information from network element m and network element n to the centralized control node respectively through delay deterministic network.

4. Procedures of the One-way Delay Measurement Method

In this section, the procedures of the proposed one-way delay measurement method will be elaborated. Assume there are two network element. It is determined that the time when the centralized control node receives the first flow information is the first time, and the time when the second flow information is received by the centralized control node is determined to be the second time. The first flow information is sent to the centralized control node via delay deterministic network, and the second flow information is also sent to the centralized control node via delay deterministic network. The procedures of the one-way delay measurement method is shown in Figure 2.

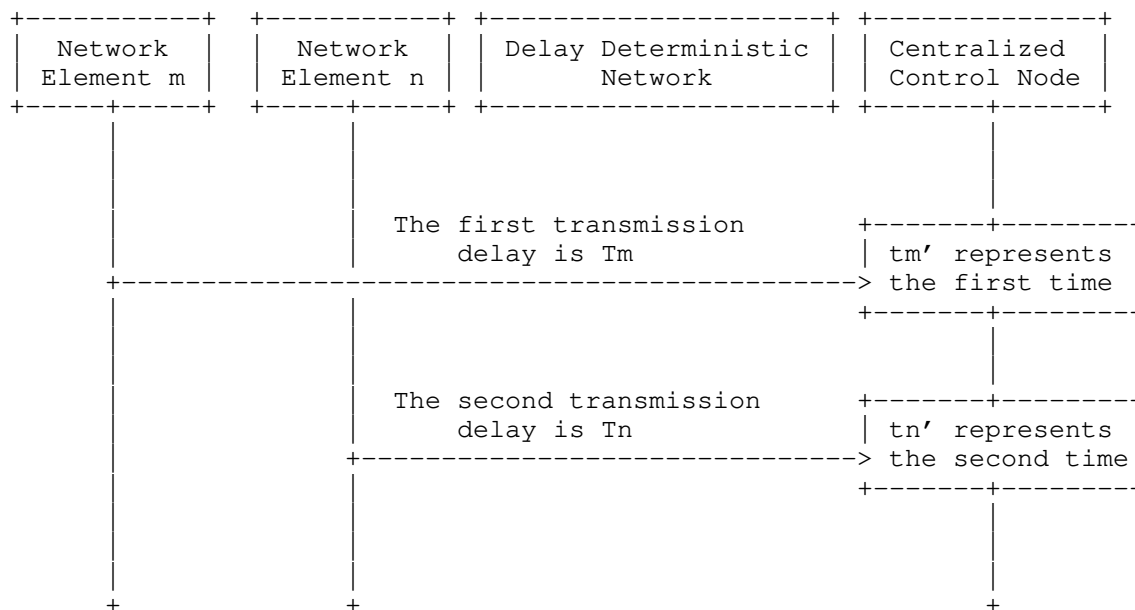


Figure 2: Figure 2: Procedures of the one-way delay measurement method

The transmission delay of traffic from the first network element to the second network element can be determined based on the first time, the second time, the first transmission delay, and the second transmission delay.

The first traffic information is sent by the first network element to the centralized control node via a delay deterministic network at the moment when the traffic passes through the first network element. And the time when the traffic passes through the first network

element refers to the moment when traffic enters the first network element or the time when traffic flows out of the first network element.

The second traffic information is sent by the second network element to the centralized control node via a delay deterministic network at the moment when the traffic passes through the second network element. And the time when the traffic passes through the second network element refers to the moment when traffic enters the second network element or the time when traffic flows out of the second network element.

It is determined that the transmission delay of the first traffic information from the first network element to the centralized control node is the first transmission delay, and it is determined that the transmission delay of the second traffic information from the second network element to the centralized control node is the second transmission delay. The transmission delay of traffic from the first network element to the second network element can be determined based on the following formula: $\text{Delay} = t_n' - t_m' + T_m - T_n$. Wherein, t_n' represents the second time, t_m' represents the first time, T_m represents the first transmission delay, T_n represents the second transmission delay, and Delay represents transmission delay of the traffic from the first network element to the second network element. In the above method, the delay deterministic network is used to ensure that the first transmission delay and the second transmission delay are fixed delays.

5. Security Considerations

TBD.

6. IANA Considerations

TBD.

7. Normative References

[IEEE.1588.2008]

IEEE, "IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems", July 2008.

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

- [RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <<https://www.rfc-editor.org/info/rfc5905>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8186] Mirsky, G. and I. Meilik, "Support of the IEEE 1588 Timestamp Format in a Two-Way Active Measurement Protocol (TWAMP)", RFC 8186, DOI 10.17487/RFC8186, June 2017, <<https://www.rfc-editor.org/info/rfc8186>>.
- [RFC8655] Finn, N., Thubert, P., Varga, B., and J. Farkas, "Deterministic Networking Architecture", RFC 8655, DOI 10.17487/RFC8655, October 2019, <<https://www.rfc-editor.org/info/rfc8655>>.

Authors' Addresses

Yang Li
China Mobile
Beijing
100053
China

Email: liyangzn@chinamobile.com

Hongwei Yang
China Mobile
Beijing
100053
China

Email: yanghongwei@chinamobile.com

Tao Sun
China Mobile
Beijing
100053
China

Email: suntao@chinamobile.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: 14 August 2022

Y. Li
T. Sun
H. Yang
D. Chen
China Mobile
Y. Wang
Huawei
10 February 2022

One-way Delay Measurement Based on Reference Delay
draft-li-ippm-ref-delay-measurement-02

Abstract

The end-to-end network one-way delay is an important performance metric in the 5G network. For realizing the accurate one-way delay measurement, existing methods requires the end-to-end deployment of accurate clock synchronization mechanism, such as PTP or GPS, which results in relatively high deployment cost. Another method can derive the one-way delay from the round-trip delay. In this case, since the delay of the downlink and uplink may be asymmetric, the measurement accuracy is relatively low. Hence, this document introduces a method to measure the end-to-end network one-way delay based on a reference delay guaranteed by deterministic networking without clock synchronization. The advantage of this solution is that it has high measurement accuracy and can test any flow type.

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 14 August 2022.

Copyright Notice

Copyright (c) 2022 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 Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction	2
2. Conventions Used in This Document	4
2.1. Terminology	4
2.2. Requirements Language	4
3. The method of One-way Delay Measurement Based on Reference Delay	4
3.1. One-way Delay Measurement Method	5
3.2. Packet and Measurement Header Format	7
4. Acquisition of Reference Delay	8
5. Security Considerations	8
6. IANA Considerations	8
7. Normative References	8
Authors' Addresses	9

1. Introduction

With the gradual promotion of new-generation network technologies (such as 5G networks) and their application in various industries, SLA guarantees for network quality become more and more important. For example, different 5G services have different requirements for network performance indicators such as delay, jitter, packet loss, and bandwidth. Among them, the 5G network delay is defined as end-to-end one-way delay of the network. Real-time and accurate measurement of the end-to-end one-way delay is very important for the SLA guarantee of network services, and has become an urgent and important requirement.

As shown in figure 1, 5G network HD video surveillance service is a common scenario having requirement of end-to-end one-way delay measurement. In this case, one end of the network is a high-definition surveillance camera in the wireless access side, and the other end of the network is a video server. The end-to-end one-way

delay from the surveillance camera to the video server is the sum of T1, T2, T3 and T4, which is composed of delay in wireless access network, optical transmission network, 5G core network, and IP data network.

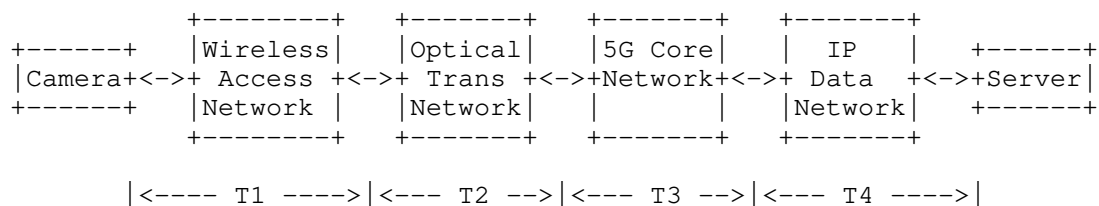


Figure 1: Figure 1:A Scenario for End-to-end One-way Delay

The existing one-way delay measurement solutions are divided into two types. One type of mechanism to calculate one-way delay is based on the measurement of round-trip delay. However, for example, because upstream traffic and downstream traffic do not share the same path in 5G network, the accuracy of the end-to-end one-way delay calculated from the round-trip delay is low. Another type of mechanism is in-band OAM with accurate network time synchronization mechanism, such as NTP[RFC5905] or PTP[IEEE.1588.2008].

The one-way delay measurement solution based on precise network time synchronization requires the deployment of an end-to-end time synchronization mechanism. The current time synchronization accuracy based on the NTP protocol can only reach millisecond level, which cannot fully meet the measurement accuracy requirements. The time synchronization accuracy based on the GPS module or the PTP protocol can meet the requirements. However, because many data centers are actually located underground or in rooms without GPS signals, so GPS clock information cannot be continuously obtained for time synchronization. For time synchronization solutions based on the PTP protocol, each device in the wireless access network, 5G transport network, and 5G core network must support the PTP protocol, which is unrealistic at the moment. So the one-way delay measurement solution based on precise end-to-end time synchronization is expensive and difficult to be deployed.

This document introduces a one-way delay measurement mechanism for Deterministic Networking (DetNet) [RFC8655]. The one-way delay measurement is based on a stable one-way delay of a reference DetNet packet, named as reference delay, which is known in advance and has extremely low jitter. We can use the reference delay provided by the reference DetNet packet to derive the one-way delay of other common service packets.

2. Conventions Used in This Document

2.1. Terminology

- NTP Network Time Protocol
- PTP Precision Time Protocol
- SLA Service Level Agreement

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. The method of One-way Delay Measurement Based on Reference Delay

The end-to-end one-way delay of a reference packet with a stable delay in the network can be used as a reference delay, denoted as Dref, which is known in advance and has extremely low jitter. This section will describe in detail the end-to-end one-way delay measurement method based on reference delay of the reference packet. Assume that the end-to-end one-way delay from the sender to the receiver is measured, as shown in figure 2. The intermediate network devices other than the sender and receiver are hidden in the figure.

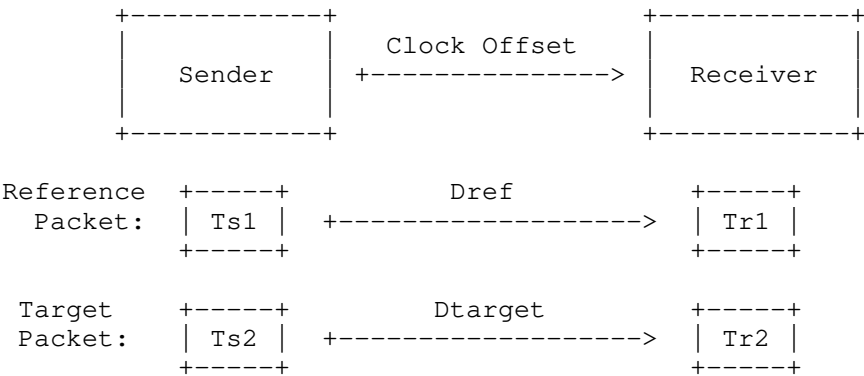


Figure 2: Figure 2:Topology of One-way Delay Measurement

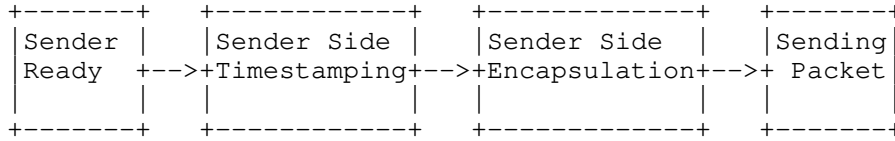
3.1. One-way Delay Measurement Method

The measurement steps are shown in figure 3, which describe the measurement steps at the sender side and receiver side respectively. For the sender side, a reference packet is sent. In the first step, the sender gets ready to send a reference packet; in the second step, the sender marks an egress timestamp $Ts1$ for the reference packet; in the third step, the sender encapsulates the egress timestamp of the reference packet in the measurement header of the reference packet; in the fourth step, the sender sends the reference packet. For the target packet, the sender side procedures are the same, we omit it for simplicity. The sending time of the target packet is according to the traffic model of real applications. On the other hand, the sender can send the reference packet according to a fixed frequency or adjust the sending frequency according to the link usage rate, so that the target packet can always find a nearby reference packet to make sure that the sending time interval between the reference packet and the target packet is small.

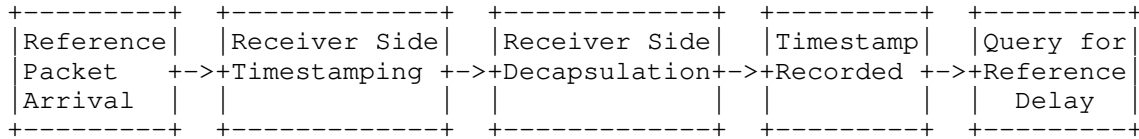
For the reference packet, the processing steps at the receiver are shown in figure 3. In the first step, the reference packet arrives at the receiver, and the receiver receives the reference packet; in the second step, the receiver timestamps the reference packet at the entrance, which is denoted as $Tr1$; in the third step, the receiver decapsulates the measurement header of the reference packet to obtain the sender side timestamp $Ts1$; in the fourth step, the receiver records the timestamp information of $Ts1$ and $Tr1$; in the fifth step, the receiver uses the source/destination pair obtained by decapsulation in the third step as the search key, queries the reference delay table and records the reference delay search result, denoted as $Dref$.

For the target packet, the processing steps at the receiver are also shown in figure 3. In the first step, the target packet arrives at the receiver, and the receiver receives the target packet; in the second step, the receiver timestamps the target packet at the entrance, which is denoted as $Tr2$; in the third step, the receiver decapsulates the measurement header of the target packet to obtain the sender side timestamp $Ts2$; in the fourth step, the receiver records the timestamp information of $Ts2$ and $Tr2$; in the fifth step, the receiver calculates the target one-way delay, which we want to measure, according to the recorded timestamp information $Ts1$, $Ts2$, $Tr1$, $Tr2$ and reference delay information $Dref$. The target one-way delay of the target packet is recorded as $Dtarget$.

Sender Side Procedures for both Reference and Target Packet:



Receiver Side Procedures for Reference Packet:



Receiver Side Procedures for Target Packet:

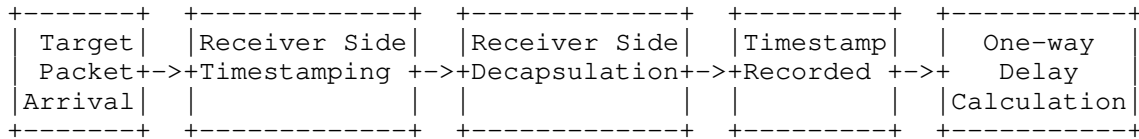


Figure 3: Figure 3: Measurement steps for Sender and Receiver
Respectively

Now we describe the fifth step of the receiver procedures for the target packet in figure 3, that is, calculating the one-way delay D_{target} of the target packet based on the recorded timestamp information $Ts1$, $Ts2$, $Tr1$, $Tr2$ and the reference delay information D_{ref} . The calculation method is the core of this solution. For the reference packet, leveraging the receiver timestamp minus the sender timestamp, we can get Equation 1.

$$\text{Equation 1: } Tr1 - Ts1 = D_{ref} + \text{Offset1}$$

where Offset1 is the time offset between the sender and the receiver when the reference packet transmission occurs. Similarly, for the target packet, we can get Equation 2 using the same method.

$$\text{Equation 2: } Tr2 - Ts2 = D_{target} + \text{Offset2}$$

where Offset2 is the time offset between the sender and the receiver when the target packet transmission occurs. Assuming that the sending time interval between the reference packet and the target packet is very small, we can get that $\text{Offset1} = \text{Offset2}$. By (Equation 2 - Equation 1), we can get Equation 3.

Equation 3: $D_{\text{target}} = (Tr_2 + Ts_1) - (Tr_1 + Ts_2) + D_{\text{ref}}$

So the one-way delay of the target packet can be calculated by Equation 3.

3.2. Packet and Measurement Header Format

The sender encapsulates the timestamp information and sender-receiver pair information in the measurement header of the sent packet, as shown in figure 4. The position of measurement header is in the option field of the TCP protocol header. The delay measurement option format is defined in figure 5. The Length value is 8 octets, which is in accordance with TCP option. The sender ID is one octet, and the receiver ID is also one octet. The sender side timestamp is 4 octets, which can store accurate timestamp information.

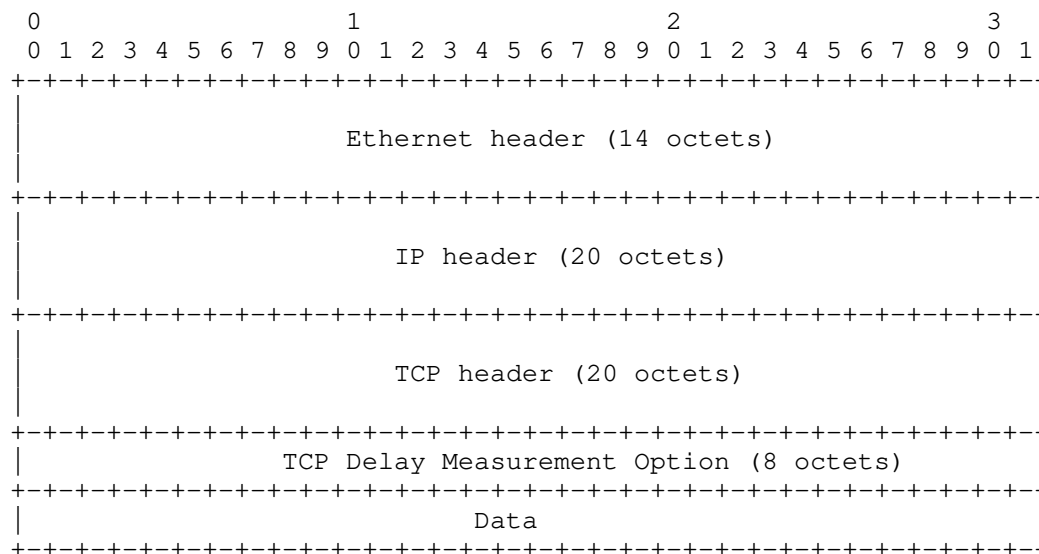


Figure 4: Figure 4: Format of Reference or Target Packet

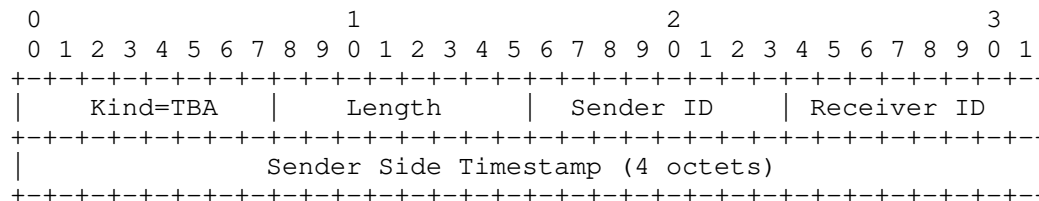


Figure 5: Figure 5: TCP Delay Measurement Option Format

4. Acquisition of Reference Delay

The end-to-end one-way delay includes three parts, namely the transmission delay, the internal processing delay of the network devices, and the internal queueing delay of the network devices. Among them, fixed parts of the delay include transmission delay and internal processing delay. The transmission delay is related to transmission distance and transmission media. For example, in optical fiber, it is about 5ns per meter. With transmission path and media determined, it is basically a fixed value. The internal processing delay of a network device includes processing delay of the device's internal pipeline or processor and serial-to-parallel conversion delay of the interface, which is related to in/out port rate of the device, message length and forwarding behavior. The magnitude of the internal processing delay is at microsecond level, and it is basically a fixed value related to the chip design specifications of a particular network device. Variable part of the delay is the internal queueing delay. The queueing delay of the device internal buffer is related to the queue depth, queue scheduling algorithm, message priority and message length. For each device along the end-to-end path, the queueing delay can reach microsecond or even millisecond level, depending on values of the above parameters and network congestion state.

With the continuous development of networking technologies and application requirements, a series of new network technologies have emerged which can guarantee bounded end-to-end delay and ultra small jitter. For example, deterministic network[RFC8655], by leveraging novel scheduling algorithms and packet priority settings, can stabilize queuing delay of network device on the end-to-end path. As a result, the end-to-end one-way delay is extremely low and bounded. So packets transmitted by a deterministic network with delay guarantee can be used as reference packets, and their end-to-end one-way delay can be used as reference delays. The acquisition method of reference delay is not limited to the above method based on deterministic network technology.

5. Security Considerations

TBD.

6. IANA Considerations

This document requests IANA to assign a Kind Number in TCP Option to indicate TCP Delay Measurement option.

7. Normative References

- [IEEE.1588.2008]
IEEE, "IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems", July 2008.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <<https://www.rfc-editor.org/info/rfc5905>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8655] Finn, N., Thubert, P., Varga, B., and J. Farkas, "Deterministic Networking Architecture", RFC 8655, DOI 10.17487/RFC8655, October 2019, <<https://www.rfc-editor.org/info/rfc8655>>.

Authors' Addresses

Yang Li
China Mobile
Beijing
100053
China

Email: liyangzn@chinamobile.com

Tao Sun
China Mobile
Beijing
100053
China

Email: suntao@chinamobile.com

Hongwei Yang
China Mobile
Beijing
100053
China

Email: yanghongwei@chinamobile.com

Danyang Chen
China Mobile
Beijing
100053
China

Email: chendanyang@chinamobile.com

Yali Wang
Huawei
156 Beiqing Rd., Haidian District
Beijing
China

Email: wangyalil1@huawei.com

IPPM
Internet-Draft
Intended status: Informational
Expires: January 13, 2022

M. Cociglio
Telecom Italia - TIM
A. Ferrieux
Orange Labs
G. Fioccola
Huawei Technologies
I. Lubashev
Akamai Technologies
F. Bulgarella
Telecom Italia - TIM
I. Hamchaoui
Orange Labs
M. Nilo
Telecom Italia - TIM
R. Sisto
Politecnico di Torino
D. Tikhonov
LiteSpeed Technologies
July 12, 2021

Explicit Flow Measurements Techniques
draft-mdt-ippm-explicit-flow-measurements-02

Abstract

This document describes protocol independent methods called Explicit Flow Measurement Techniques that employ few marking bits, inside the header of each packet, for loss and delay measurement. The endpoints, marking the traffic, signal these metrics to intermediate observers allowing them to measure connection performance, and to locate the network segment where impairments happen. Different alternatives are considered within this document. These signaling methods apply to all protocols but they are especially valuable when applied to protocols that encrypt transport header and do not allow traditional methods for delay and loss detection.

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 13, 2022.

Copyright Notice

Copyright (c) 2021 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
2. Notational Conventions	4
3. Latency Bits	5
3.1. Spin Bit	5
3.2. Delay Bit	6
3.2.1. Generation Phase	8
3.2.2. Reflection Phase	8
3.2.3. T_Max Selection	9
3.2.4. Delay Measurement using Delay Bit	10
3.2.5. Observer's Algorithm	12
3.2.6. Two Bits Delay Measurement: Spin Bit + Delay Bit . .	13
3.2.7. Hidden Delay Bit - Delay Bit with Privacy Protection	13
4. Loss Bits	13
4.1. T Bit - Round Trip Loss Bit	14
4.1.1. Round Trip Packet Loss Measurement	15
4.1.2. Setting the Round Trip Loss Bit on Outgoing Packets .	17
4.1.3. Observer's Logic for Round Trip Loss Signal	18
4.1.4. Loss Coverage and Signal Timing	19
4.2. Q Bit - Square Bit	19
4.2.1. Q Block Length Selection	19
4.2.2. Upstream Loss	20
4.2.3. Identifying Q Block Boundaries	21
4.3. L Bit - Loss Event Bit	21
4.3.1. End-To-End Loss	22

4.3.2. Loss Profile Characterization	22
4.4. L+Q Bits - Upstream, Downstream, and End-to-End Loss Measurements	22
4.4.1. Correlating End-to-End and Upstream Loss	23
4.5. R Bit - Reflection Square Bit	24
4.5.1. R+Q Bits - Using R and Q Bits for Passive Loss Measurement	25
4.5.2. Enhancement of R Block Length Computation	29
4.5.3. Improved Resilience to Packet Reordering	29
4.6. Improved Q and R Bits Resilience to Burst Losses	29
5. Summary of Delay and Loss Marking Methods	30
6. ECN-Echo Event Bit	32
6.1. Setting the ECN-Echo Event Bit on Outgoing Packets	32
6.2. Using E Bit for Passive ECN-Reported Congestion Measurement	32
7. Protocol Ossification Considerations	33
8. Examples of Application	33
8.1. QUIC	33
8.2. TCP	34
9. Security Considerations	34
9.1. Optimistic ACK Attack	35
10. Privacy Considerations	35
11. IANA Considerations	36
12. Change Log	36
13. Contributors	36
14. Acknowledgements	36
15. References	36
15.1. Normative References	36
15.2. Informative References	37
Authors' Addresses	39

1. Introduction

Packet loss and delay are hard and pervasive problems of day-to-day network operation. Proactively detecting, measuring, and locating them is crucial to maintaining high QoS and timely resolution of crippling end-to-end throughput issues. To this effect, in a TCP-dominated world, network operators have been heavily relying on information present in the clear in TCP headers: sequence and acknowledgment numbers and SACKs when enabled (see [RFC8517]). These allow for quantitative estimation of packet loss and delay by passive on-path observation. Additionally, the problem can be quickly identified in the network path by moving the passive observer around.

With encrypted protocols, the equivalent transport headers are encrypted and passive packet loss and delay observations are not possible, as described in [TRANSPORT-ENCRYPT].

Measuring TCP loss and delay between similar endpoints cannot be relied upon to evaluate encrypted protocol loss and delay. Different protocols could be routed by the network differently, and the fraction of Internet traffic delivered using protocols other than TCP is increasing every year. It is imperative to measure packet loss and delay experienced by encrypted protocol users directly.

This document defines Explicit Flow Measurement Techniques. These hybrid measurement path signals (see [IPM-Methods]) are to be embedded into a transport layer protocol and are explicitly intended for exposing RTT and loss rate information to on-path measurement devices. These measurement mechanisms are applicable to any transport-layer protocol, and, as an example, the document describes QUIC and TCP bindings.

The Explicit Flow Measurement Techniques described in this document can be used alone or in combination with other Explicit Flow Measurement Techniques. Each technique uses a small number of bits and exposes a specific measurement.

Following the recommendation in [RFC8558] of making path signals explicit, this document proposes adding a small number of dedicated measurement bits to the clear portion of the protocol headers. These bits can be added to an encrypted portion of a header belonging to any protocol layer, e.g. IP (see [IP]) and IPv6 (see [IPv6]) headers or extensions, such as [IPv6AltMark], UDP surplus space (see [UDP-OPTIONS] and [UDP-SURPLUS]), reserved bits in a QUIC v1 header (see [QUIC-TRANSPORT]).

The measurements are not designed for use in automated control of the network in environments where signal bits are set by untrusted hosts. Instead, the signal is to be used for troubleshooting individual flows as well as for monitoring the network by aggregating information from multiple flows and raising operator alarms if aggregate statistics indicate a potential problem.

The spin bit, delay bit and loss bits explained in this document are inspired by [AltMark], [SPIN-BIT], [I-D.trammell-tsvwg-spin] and [I-D.trammell-ippm-spin].

Additional details about the Performance Measurements for QUIC are described in the paper [ANRW19-PM-QUIC].

2. Notational 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].

3. Latency Bits

This section introduces bits that can be used for round trip latency measurements. Whenever this section of the specification refers to packets, it is referring only to packets with protocol headers that include the latency bits.

[QUIC-TRANSPORT] introduces an explicit per-flow transport-layer signal for hybrid measurement of RTT. This signal consists of a spin bit that toggles once per RTT. [SPIN-BIT] discusses an additional two-bit Valid Edge Counter (VEC) to compensate for loss and reordering of the spin bit and increase fidelity of the signal in less than ideal network conditions.

This document introduces a stand-alone single-bit delay signal that can be used by passive observers to measure the RTT of a network flow, avoiding the spin bit ambiguities that arise as soon as network conditions deteriorate.

3.1. Spin Bit

This section is a small recap of the spin bit working mechanism. For a comprehensive explanation of the algorithm, please see [SPIN-BIT].

The spin bit is an alternate marking [AltMark] generated signal, where the size of the alternation changes with the flight size each RTT.

The latency spin bit is a single bit signal that toggles once per RTT, enabling latency monitoring of a connection-oriented communication from intermediate observation points.

A "spin period" is a set of packets with the same spin bit value sent during one RTT time interval. A "spin period value" is the value of the spin bit shared by all packets in a spin period.

The client and server maintain an internal per-connection spin value (i.e. 0 or 1) used to set the spin bit on outgoing packets. Both endpoints initialize the spin value to 0 when a new connection starts. Then:

- when the client receives a packet with the packet number larger than any number seen so far, it sets the connection spin value to the opposite value contained in the received packet;
- when the server receives a packet with the packet number larger than any number seen so far, it sets the connection spin value to the same value contained in the received packet.

The computed spin value is used by the endpoints for setting the spin bit on outgoing packets. This mechanism allows the endpoints to generate a square wave such that, by measuring the distance in time between pairs of consecutive edges observed in the same direction, a passive on-path observer can compute the round trip delay of that network flow.

Spin bit enables round trip latency measurement by observing a single direction of the traffic flow.

Note that packet reordering can cause spurious edges that require heuristics to correct. The spin bit performance deteriorates as soon as network impairments arise as explained in Section 3.2.

3.2. Delay Bit

The delay bit has been designed to overcome accuracy limitations experienced by the spin bit under difficult network conditions:

- packet reordering leads to generation of spurious edges and errors in delay estimation;
- loss of edges causes wrong estimation of spin periods and therefore wrong RTT measurements;
- application-limited senders cause the spin bit to measure the application delays instead of network delays.

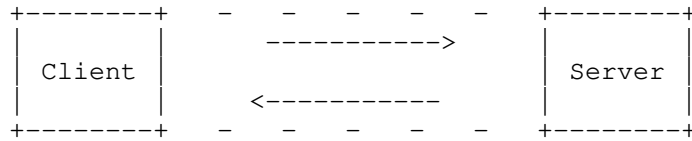
Unlike the spin bit, which is set in every packet transmitted on the network, the delay bit is set only once per round trip.

When the delay bit is used, a single packet with a marked bit (the delay bit) bounces between a client and a server during the entire connection lifetime. This single packet is called "delay sample".

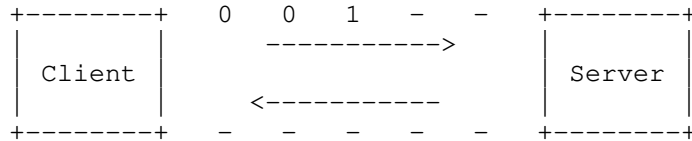
An observer placed at an intermediate point, observing a single direction of traffic, tracking the delay sample and the relative timestamp, can measure the round trip delay of the connection.

The delay sample lifetime is comprised of two phases: initialization and reflection. The initialization is the generation of the delay sample, while the reflection realizes the bounce behavior of this single packet between the two endpoints.

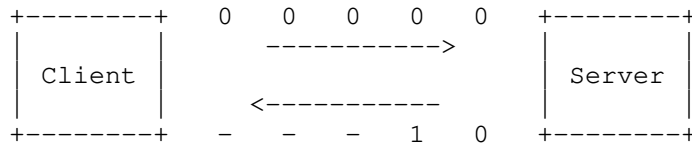
The next figure describes the elementary Delay bit mechanism.



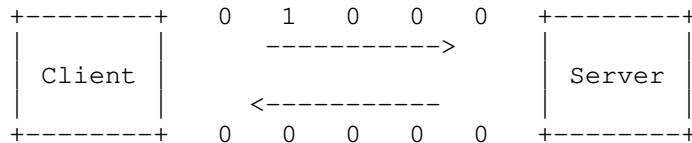
(a) No traffic at beginning.



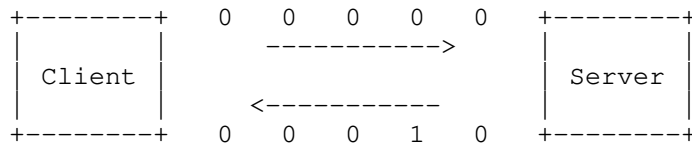
(b) The Client starts sending data and sets the first packet as Delay Sample.



(c) The Server starts sending data and reflects the Delay Sample.



(d) The Client reflects the Delay Sample.



(e) The Server reflects the Delay Sample and so on.

Delay bit mechanism

3.2.1. Generation Phase

Only client is actively involved in the generation phase. It maintains an internal per-flow timestamp variable ("ds_time") updated every time a delay sample is transmitted.

When connection starts, the client generates a new delay sample initializing the delay bit of the first outgoing packet to 1. Then it updates the "ds_time" variable with the timestamp of its transmission.

The server initializes the delay bit to 0 at the beginning of the connection, and its only task during the connection is described in Section 3.2.2.

In absence of network impairments, the delay sample should bounce between client and server continuously, for the entire duration of the connection. That is highly unlikely for two reasons:

1. the packet carrying the delay bit might be lost;
2. an endpoint could stop or delay sending packets because the application is limiting the amount of traffic transmitted;

To deal with these problems, the client generates a new delay sample if more than a predetermined time ("T_Max") has elapsed since the last delay sample transmission (including reflections). Note that "T_Max" should be greater than the max measurable RTT on the network. See Section 3.2.3 for details.

3.2.2. Reflection Phase

Reflection is the process that enables the bouncing of the delay sample between a client and a server. The behavior of the two endpoints is almost the same.

- Server side reflection: when a delay sample arrives, the server marks the first packet in the opposite direction as the delay sample.
- Client side reflection: when a delay sample arrives, the client marks the first packet in the opposite direction as the delay sample. It also updates the "ds_time" variable when the outgoing delay sample is actually forwarded.

In both cases, if the outgoing delay sample is being transmitted with a delay greater than a predetermined threshold after the reception of

the incoming delay sample (1ms by default), the delay sample is not reflected, and the outgoing delay bit is kept at 0.

By doing so, the algorithm can reject measurements that would overestimate the delay due to lack of traffic on the endpoints. Hence, the maximum estimation error would amount to twice the threshold (e.g. 2ms) per measurement.

3.2.3. T_Max Selection

The internal "ds_time" variable allows a client to identify delay sample losses. Considering that a lost delay sample is regenerated at the end of an explicit time ("T_Max") since the last generation, this same value can be used by an observer to reject a measure and start a new one.

In other words, if the difference in time between two delay samples is greater or equal than "T_Max", then these cannot be used to produce a delay measure. Therefore the value of "T_Max" must also be known to the on-path network probes.

There are two alternatives to select the "T_Max" value so that both client and observers know it. The first one requires that "T_Max" is known a priori ("T_Max_p") and therefore set within the protocol specifications that implements the marking mechanism (e.g. 1 second which usually is greater than the max expectable RTT). The second alternative requires a dynamic mechanism able to adapt the duration of the "T_Max" to the delay of the connection ("T_Max_c").

For instance, client and observers could use the connection RTT as a basis for calculating an effective "T_Max". They should use a predetermined initial value so that "T_Max = T_Max_p" (e.g. 1 second) and then, when a valid RTT is measured, change "T_Max" accordingly so that "T_Max = T_Max_c". In any case, the selected "T_Max" should be large enough to absorb any possible variations in the connection delay.

"T_Max_c" could be computed as two times the measured "RTT" plus a fixed amount of time ("100ms") to prevent low "T_Max" values in case of very small RTTs. The resulting formula is: "T_Max_c = 2RTT + 100ms". If "T_Max_c" is greater than "T_Max_p" then "T_Max_c" is forced to "T_Max_p" value.

Note that the observer's "T_Max" should always be less than or equal to the client's "T_Max" to avoid considering as a valid measurement what is actually the client's "T_Max". To obtain this result, the client waits for two consecutive incoming samples and computes the two related RTTs. Then it takes the largest of them as the basis of

the "T_Max_c" formula. At this point, observers have already measured a valid RTT and then computed their "T_Max_c".

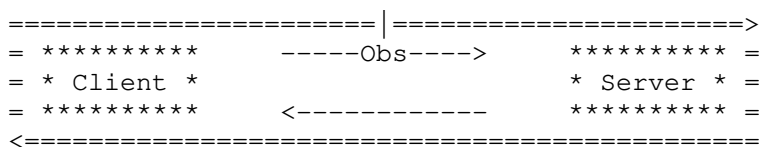
3.2.4. Delay Measurement using Delay Bit

When the Delay Bit is used, a passive observer can use delay samples directly and avoid inherent ambiguities in the calculation of the RTT as can be seen in spin bit analysis.

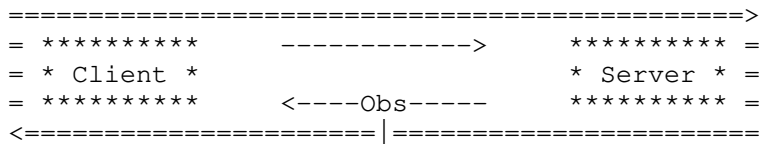
3.2.4.1. RTT Measurement

The delay sample generation process ensures that only one packet marked with the delay bit set to 1 runs back and forth between two endpoints per round trip time. To determine the RTT measurement of a flow, an on-path passive observer computes the time difference between two delay samples observed in a single direction.

To ensure a valid measurement, the observer must verify that the distance in time between the two samples taken into account is less than "T_Max".



(a) client-server RTT



(b) server-client RTT

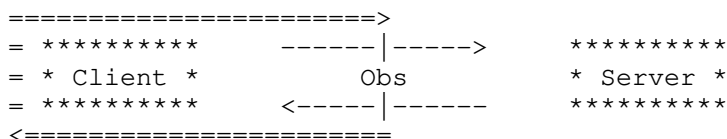
Round-trip time (both direction)

3.2.4.2. Half-RTT Measurement

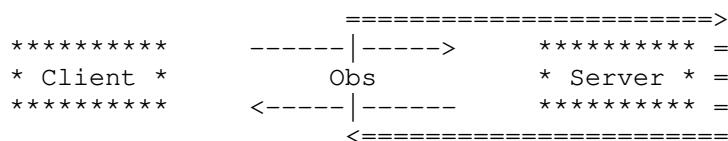
An observer that is able to observe both forward and return traffic directions can use the delay samples to measure "upstream" and "downstream" RTT components, also known as the half-RTT measurements. It does this by measuring the time between a delay sample observed in one direction and the delay sample previously observed in the opposite direction.

As with RTT measurement, the observer must verify that the distance in time between the two samples taken into account is less than "T_Max".

Note that upstream and downstream sections of paths between the endpoints and the observer, i.e. observer-to-client vs client-to-observer and observer-to-server vs server-to-observer, may have different delay characteristics due to the difference in network congestion and other factors.



(a) client-observer half-RTT

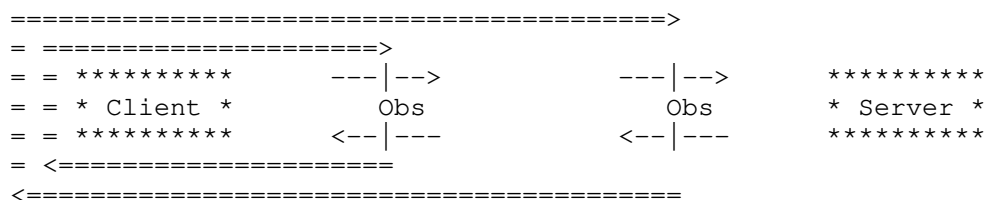


(b) observer-server half-RTT

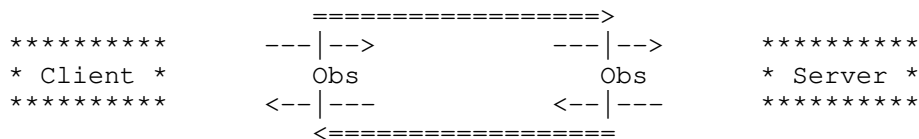
Half Round-trip time (both direction)

3.2.4.3. Intra-Domain RTT Measurement

Intra-domain RTT is the portion of the entire RTT used by a flow to traverse the network of a provider. To measure intra-domain RTT, two observers capable of observing traffic in both directions must be employed simultaneously at ingress and egress of the network to be measured. Intra-domain RTT is difference between the two computed upstream (or downstream) RTT components.



(a) client-observer RTT components (half-RTTs)



(b) the intra-domain RTT resulting from the subtraction of the above RTT components

Intra-domain Round-trip time (client-observer: upstream)

3.2.5. Observer's Algorithm

An on-path observer maintains an internal per-flow variable to keep track of time at which the last delay sample has been observed.

A unidirectional observer, upon detecting a delay sample:

- if a delay sample was also detected previously in the same direction and the distance in time between them is less than " $T_{Max} - K$ ", then the two delay samples can be used to calculate RTT measurement. " K " is a protection threshold to absorb differences in " T_{Max} " computation and delay variations between two consecutive delay samples (e.g. " $K = 10\% T_{Max}$ ").

If the observer can observe both forward and return traffic flows, and it is able to determine which direction contains the client and the server (e.g. by observing the connection handshake), upon detecting a delay sample:

- if a delay sample was also detected in the opposite direction and the distance in time between them is less than " $T_{Max} - K$ ", then the two delay samples can be used to measure the observer-client half-RTT or the observer-server half-RTT, according to the direction of the last delay sample observed.

3.2.6. Two Bits Delay Measurement: Spin Bit + Delay Bit

Spin and Delay bit algorithms work independently. If both marking methods are used in the same connection, observers can choose the best measurement between the two available:

- when a precise measurement can be produced using the delay bit, observers choose it;
- when a delay bit measurement is not available, observers choose the approximate spin bit one.

3.2.7. Hidden Delay Bit - Delay Bit with Privacy Protection

Theoretically, delay measurements can be used to roughly evaluate the distance of the client from the server (using the RTT) or from any intermediate observer (using the client-observer half-RTT). To protect users privacy, the algorithm of the delay bit can be slightly modified to mask the RTT of the connection to an intermediate observer. This result can be achieved using a simple expedient which consists in delaying the client-side reflection of the delay sample by a predetermined time value. This would lead an intermediate observer to inevitably measure a delay greater than the real one.

The Additional Delay should be randomly selected by the client and kept constant for a certain amount of time across multiple connections. This ensures that the client-server jitter remains the same as if no Additional Delay had been inserted. For instance, a new Additional Delay value could be generated whenever the client's IP address changes.

Using this technique, despite the Additional Delay introduced, it is still possible to correctly measure the right component of RTT (observer-server) and all the intra-domain measurements used to distribute the delay in the network. Furthermore, differently from the Delay Bit, the hidden Delay Bit makes the use of the client reflection threshold (1ms) redundant. Removing this threshold leads to the further advantage of increasing the number of valid measurements produced by the algorithm.

4. Loss Bits

This section introduces bits that can be used for loss measurements. Whenever this section of the specification refers to packets, it is referring only to packets with protocol headers that include the loss bits - the only packets whose loss can be measured.

- T: the "round Trip loss" bit is used in combination with the Spin bit to measure round-trip loss. See Section 4.1.
- Q: the "Square signal" bit is used to measure upstream loss. See Section 4.2.
- L: the "Loss event" bit is used to measure end-to-end loss. See Section 4.3.
- R: the "Reflection square signal" bit is used in combination with Q bit to measure end-to-end loss. See Section 4.1.

Loss measurements enabled by T, Q, and L bits can be implemented by those loss bits alone (T bit requires a working Spin Bit). Two-bit combinations Q+L and Q+R enable additional measurement opportunities discussed below.

Each endpoint maintains appropriate counters independently and separately for each separately identifiable flow (each sub-flow for multipath connections).

Since loss is reported independently for each flow, all bits (except for L bit) require a certain minimum number of packets to be exchanged per flow before any signal can be measured. Therefore, loss measurements work best for flows that transfer more than a minimal amount of data.

4.1. T Bit - Round Trip Loss Bit

The round Trip loss bit is used to mark a variable number of packets exchanged twice between the endpoints realizing a two round-trip reflection. A passive on-path observer, observing either direction, can count and compare the number of marked packets seen during the two reflections, estimating the loss rate experienced by the connection. The overall exchange comprises:

- The client selects, generates and consequently transmits a first train of packets, by setting the T bit to 1;
- The server, upon receiving each packet included in the first train, reflects to the client a respective second train of packets of the same size as the first train received, by setting the T bit to 1;
- The client, upon receiving each packet included in the second train, reflects to the server a respective third train of packets of the same size as the second train received, by setting the T bit to 1;

- The server, upon receiving each packet included in the third train, finally reflects to the client a respective fourth train of packets of the same size as the third train received, by setting the T bit to 1.

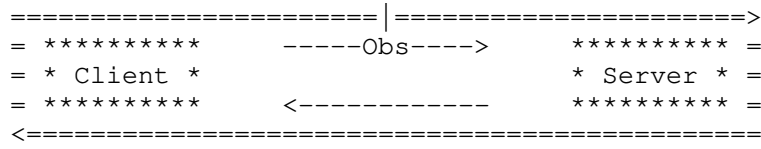
Packets belonging to the first round trip (first and second train) represent the Generation Phase, while those belonging to the second round trip (third and fourth train) represent the Reflection Phase.

A passive on-path observer can count and compare the number of marked packets seen during the two round trips (i.e. the first and third or the second and the fourth trains of packets, depending on which direction is observed) and estimate the loss rate experienced by the connection. This process is repeated continuously to obtain more measurements as long as the endpoints exchange traffic. These measurements can be called Round Trip losses.

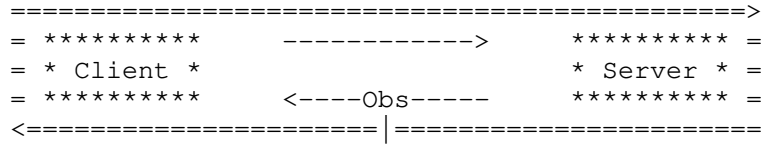
Since packet rates in two directions may be different, the number of marked packets in the train is determined by the direction with the lowest packet rate. See Section 4.1.2 for details on packet generation and for a mechanism to allow an observer to distinguish between trains belonging to different phases (Generation and Reflection).

4.1.1. Round Trip Packet Loss Measurement

Since the measurements are performed on a portion of the traffic exchanged between the client and the server, the observer calculates the end-to-end Round Trip Packet Loss (RTPL) that, statistically, will correspond to the loss rate experienced by the connection along the entire network path.



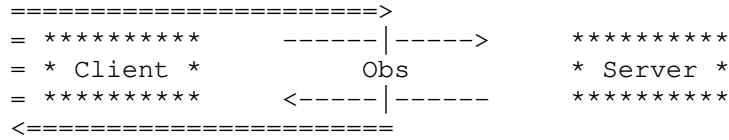
(a) client-server RTPL



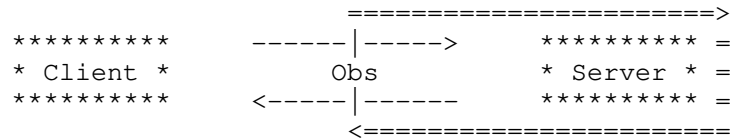
(b) server-client RTPL

Round-trip packet loss (both direction)

This methodology also allows the Half-RTPL measurement and the Intra-domain RTPL measurement in a way similar to RTT measurement.

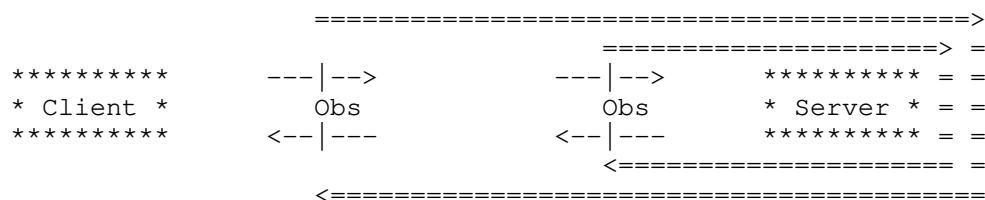


(a) client-observer half-RTPL

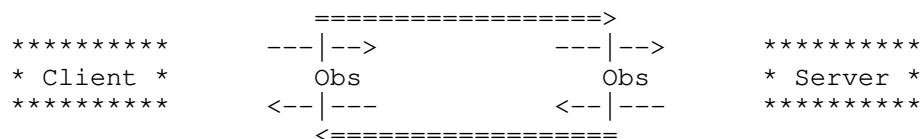


(b) observer-server half-RTPL

Half Round-trip packet loss (both direction)



(a) observer-server RTPL components (half-RTPLs)



(b) the intra-domain RTPL resulting from the subtraction of the above RTPL components

Intra-domain Round-trip packet loss (observer-server)

4.1.2. Setting the Round Trip Loss Bit on Outgoing Packets

The round Trip loss signal requires a working Spin-bit signal to separate trains of marked packets (packets with T bit set to 1). A "pause" of at least one empty spin-bit period between each phase of the algorithm serves as such separator for the on-path observer.

The client is in charge of launching trains of marked packets and does so according to the algorithm:

1. Generation Phase. The client starts generating marked packets for two consecutive spin-bit periods; it maintains a "generation token" count that is reset to zero at the beginning of the algorithm phase and is incremented every time a packet arrives. When the client transmits a packet and a "generation token" is available, the client marks the packet and retires a "generation token". If no token is available, the outgoing packet is transmitted unmarked. At the end of the first spin-bit period spent in generation, the reflection counter is unlocked to start counting incoming marked packets that will be reflected later;
2. Pause Phase. When the generation is completed, the client pauses till it has observed one entire spin bit period with no marked packets. That spin bit period is used by the observer as a separator between generated and reflected packets. During this marking pause, all the outgoing packets are transmitted with T

bit set to 0. The reflection counter is still incremented every time a marked packet arrives;

3. Reflection Phase. The client starts transmitting marked packets, decrementing the reflection counter for each transmitted marked packet until the reflection counter reached zero. The "generation token" method from the generation phase is used during this phase as well. At the end of the first spin-period spent in reflection, the reflection counter is locked to avoid incoming reflected packets incrementing it;
4. Pause Phase 2. The pause phase is repeated after the reflection phase and serves as a separator between the reflected packet train and a new packet train.

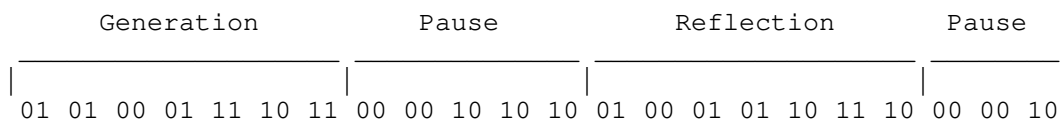
The generation token counter should be capped to limit the effects of a subsequent sudden reduction in the other endpoint's packet rate that could prevent that endpoint from reflecting collected packets. The most conservative cap value is "1".

A server maintains a "marking counter" that starts at zero and is incremented every time a marked packet arrives. When the server transmits a packet and the "marking counter" is positive, the server marks the packet and decrements the "marking counter". If the "marking counter" is zero, the outgoing packet is transmitted unmarked.

4.1.3. Observer's Logic for Round Trip Loss Signal

The on-path observer counts marked packets and separates different trains by detecting spin-bit periods (at least one) with no marked packets. The Round Trip Packet Loss (RTPL) is the difference between the size of the Generation train and the Reflection train.

In the following example, packets are represented by two bits (first one is the spin bit, second one is the loss bit):



Round Trip Loss signal example

Note that 5 marked packets have been generated of which 4 have been reflected.

4.1.4. Loss Coverage and Signal Timing

A cycle of the round Trip loss signaling algorithm contains 2 RTTs of Generation phase, 2 RTTs of Reflection phase, and two Pause phases at least 1 RTT in duration each. Hence, the loss signal is delayed by about 6 RTTs since the loss events.

The observer can only detect loss of marked packets that occurs after its initial observation of the Generation phase and before its subsequent observation of the Reflection phase. Hence, if the loss occurs on the path that sends packets at a lower rate (typically ACKs in such asymmetric scenarios), "2/6" ("1/3") of the packets will be sampled for loss detection.

If the loss occurs on the path that sends packets at a higher rate, " $\text{lowPacketRate}/(3*\text{highPacketRate})$ " of the packets will be sampled for loss detection. For protocols that use ACKs, the portion of packets sampled for loss in the higher rate direction during unidirectional data transfer is " $1/(3*\text{packetsPerAck})$ ", where the value of "packetsPerAck" can vary by protocol, by implementation, and by network conditions.

4.2. Q Bit - Square Bit

The sSquare bit (Q bit) takes its name from the square wave generated by its signal. Every outgoing packet contains the Q bit value, which is initialized to the 0 and inverted after sending N packets (a sSquare Block or simply Q Block). Hence, Q Period is $2*N$. The Q bit represents "packet color" as defined by [AltMark].

Observation points can estimate upstream losses by watching a single direction of the traffic flow and counting the number of packets in each observed Q Block, as described in Section 4.2.2.

4.2.1. Q Block Length Selection

The length of the block must be known to the on-path network probes. There are two alternatives to selecting the Q Block length. The first one requires that the length is known a priori and therefore set within the protocol specifications that implements the marking mechanism. The second requires the sender to select it.

In this latter scenario, the sender is expected to choose N (Q Block length) based on the expected amount of loss and reordering on the path. The choice of N strikes a compromise - the observation could become too unreliable in case of packet reordering and/or severe loss if N is too small, while short flows may not yield a useful upstream loss measurement if N is too large (see Section 4.2.2).

The value of N should be at least 64 and be a power of 2. This requirement allows an Observer to infer the Q Block length by observing one period of the square signal. It also allows the Observer to identify flows that set the loss bits to arbitrary values (see Section 7).

If the sender does not have sufficient information to make an informed decision about Q Block length, the sender should use $N=64$, since this value has been extensively tried in large-scale field tests and yielded good results. Alternatively, the sender may also choose a random power-of-2 N for each flow, increasing the chances of using a Q Block length that gives the best signal for some flows.

The sender must keep the value of N constant for a given flow.

4.2.2. Upstream Loss

Blocks of N (Q Block length) consecutive packets are sent with the same value of the Q bit, followed by another block of N packets with an inverted value of the Q bit. Hence, knowing the value of N , an on-path observer can estimate the amount of upstream loss after observing at least N packets. The upstream loss rate ("uloss") is one minus the average number of packets in a block of packets with the same Q value ("p") divided by N ("uloss= $1-\text{avg}(p)/N$ ").

The observer needs to be able to tolerate packet reordering that can blur the edges of the square signal, as explained in Section 4.2.3.

```

=====>
*****      -----Obs----->      *****
* Client *                               * Server *
*****      <-----              *****

```

(a) in client-server channel (uloss_up)

```

*****      ----->      *****
* Client *                               * Server *
*****      <----Obs----- *****
<=====

```

(b) in server-client channel (uloss_down)

Upstream loss

4.2.3. Identifying Q Block Boundaries

Packet reordering can produce spurious edges in the square signal. To address this, the observer should look for packets with the current Q bit value up to X packets past the first packet with a reverse Q bit value. The value of X, a "Marking Block Threshold", should be less than "N/2".

The choice of X represents a trade-off between resiliency to reordering and resiliency to loss. A very large Marking Block Threshold will be able to reconstruct Q Blocks despite a significant amount of reordering, but it may erroneously coalesce packets from multiple Q Blocks into fewer Q Blocks, if loss exceeds 50% for some Q Blocks.

4.3. L Bit - Loss Event Bit

The Loss Event bit uses an Unreported Loss counter maintained by the protocol that implements the marking mechanism. To use the Loss Event bit, the protocol must allow the sender to identify lost packets. This is true of protocols such as QUIC, partially true for TCP and SCTP (losses of pure ACKs are not detected) and is not true of protocols such as UDP and IP/IPv6.

The Unreported Loss counter is initialized to 0, and L bit of every outgoing packet indicates whether the Unreported Loss counter is positive (L=1 if the counter is positive, and L=0 otherwise).

The value of the Unreported Loss counter is decremented every time a packet with L=1 is sent.

The value of the Unreported Loss counter is incremented for every packet that the protocol declares lost, using whatever loss detection machinery the protocol employs. If the protocol is able to rescind the loss determination later, a positive Unreported Loss counter may be decremented due to the rescission, but it should NOT become negative due to the rescission.

This loss signaling is similar to loss signaling in [ConEx], except the Loss Event bit is reporting the exact number of lost packets, whereas Echo Loss bit in [ConEx] is reporting an approximate number of lost bytes.

For protocols, such as TCP ([TCP]), that allow network devices to change data segmentation, it is possible that only a part of the packet is lost. In these cases, the sender must increment Unreported Loss counter by the fraction of the packet data lost (so Unreported

Loss counter may become negative when a packet with L=1 is sent after a partial packet has been lost).

Observation points can estimate the end-to-end loss, as determined by the upstream endpoint, by counting packets in this direction with the L bit equal to 1, as described in Section 4.3.1.

4.3.1. End-To-End Loss

The Loss Event bit allows an observer to estimate the end-to-end loss rate by counting packets with L bit value of 0 and 1 for a given flow. The end-to-end loss rate is the fraction of packets with L=1.

The assumption here is that upstream loss affects packets with L=0 and L=1 equally. If some loss is caused by tail-drop in a network device, this may be a simplification. If the sender's congestion controller reduces the packet send rate after loss, there may be a sufficient delay before sending packets with L=1 that they have a greater chance of arriving at the observer.

4.3.2. Loss Profile Characterization

In addition to measuring the end-to-end loss rate, the Loss Event bit allows an observer to characterize loss profile, since the distribution of observed packets with L bit set to 1 roughly corresponds to the distribution of packets lost between 1 RTT and 1 RTO before (see Section 4.4.1). Hence, observing random single instances of L bit set to 1 indicates random single packet loss, while observing blocks of packets with L bit set to 1 indicates loss affecting entire blocks of packets.

4.4. L+Q Bits - Upstream, Downstream, and End-to-End Loss Measurements

Combining L and Q bits allows a passive observer watching a single direction of traffic to accurately measure:

- upstream loss: sender-to-observer loss (see Section 4.2.2)
- downstream loss: observer-to-receiver loss (see Section 4.4.1.1)
- end-to-end loss: sender-to-receiver loss on the observed path (see Section 4.3.1) with loss profile characterization (see Section 4.3.2)

4.4.1. Correlating End-to-End and Upstream Loss

Upstream loss is calculated by observing packets that did not suffer the upstream loss (Section 4.2.2). End-to-end loss, however, is calculated by observing subsequent packets after the sender's protocol detected the loss. Hence, end-to-end loss is generally observed with a delay of between 1 RTT (loss declared due to multiple duplicate acknowledgments) and 1 RTO (loss declared due to a timeout) relative to the upstream loss.

The flow RTT can sometimes be estimated by timing protocol handshake messages. This RTT estimate can be greatly improved by observing a dedicated protocol mechanism for conveying RTT information, such as the Spin bit (see Section 3.1) or Delay bit (see Section 3.2).

Whenever the observer needs to perform a computation that uses both upstream and end-to-end loss rate measurements, it should use upstream loss rate leading the end-to-end loss rate by approximately 1 RTT. If the observer is unable to estimate RTT of the flow, it should accumulate loss measurements over time periods of at least 4 times the typical RTT for the observed flows.

If the calculated upstream loss rate exceeds the end-to-end loss rate calculated in Section 4.3.1, then either the Q Period is too short for the amount of packet reordering or there is observer loss, described in Section 4.4.1.2. If this happens, the observer should adjust the calculated upstream loss rate to match end-to-end loss rate, unless the following applies.

In case of a protocol like TCP and SCTP that does not track losses of pure ACK packets, observing a direction of traffic dominated by pure ACK packets could result in measured upstream loss that is higher than measured end-to-end loss, if said pure ACK packets are lost upstream. Hence, if the measurement is applied to such protocols, and the observer can confirm that pure ACK packets dominate the observed traffic direction, the observer should adjust the calculated end-to-end loss rate to match upstream loss rate.

4.4.1.1. Downstream Loss

Because downstream loss affects only those packets that did not suffer upstream loss, the end-to-end loss rate ("eloss") relates to the upstream loss rate ("uloss") and downstream loss rate ("dloss") as $(1-uloss)(1-dloss)=1-eloss$. Hence, $dloss=(eloss-uloss)/(1-uloss)$.

4.4.1.2. Observer Loss

A typical deployment of a passive observation system includes a network tap device that mirrors network packets of interest to a device that performs analysis and measurement on the mirrored packets. The observer loss is the loss that occurs on the mirror path.

Observer loss affects upstream loss rate measurement, since it causes the observer to account for fewer packets in a block of identical Q bit values (see Section 4.2.2). The end-to-end loss rate measurement, however, is unaffected by the observer loss, since it is a measurement of the fraction of packets with the L bit value of 1, and the observer loss would affect all packets equally (see Section 4.3.1).

The need to adjust the upstream loss rate down to match end-to-end loss rate as described in Section 4.4.1 is an indication of the observer loss, whose magnitude is between the amount of such adjustment and the entirety of the upstream loss measured in Section 4.2.2. Alternatively, a high apparent upstream loss rate could be an indication of significant packet reordering, possibly due to packets belonging to a single flow being multiplexed over several upstream paths with different latency characteristics.

4.5. R Bit - Reflection Square Bit

R bit requires a deployment alongside Q bit. Unlike the square signal for which packets are transmitted into blocks of fixed size, the Reflection square signal (being an alternate marking signal too) produces blocks of packets whose size varies according to these rules:

- when the transmission of a new block starts, its size is set equal to the size of the last Q Block whose reception has been completed;
- if, before transmission of the block is terminated, the reception of at least one further Q Block is completed, the size of the block is updated to the average size of the further received Q Blocks. Implementation details follow.

The Reflection square value is initialized to 0 and is applied to the R-bit of every outgoing packet. The Reflection square value is toggled for the first time when the completion of a Q Block is detected in the incoming square signal (produced by the opposite node using the Q-bit). When this happens, the number of packets ("p"), detected within this first Q Block, is used to generate a reflection

square signal which toggles every "M=p" packets (at first). This new signal produces blocks of M packets (marked using the R-bit) and each of them is called "Reflection Block" (R Block).

The M value is then updated every time a completed Q Block in the incoming square signal is received, following this formula:
"M=round(avg(p))".

The parameter "avg(p)" is the average number of packets in a marking period computed considering all the Q Blocks received since the beginning of the current R Block.

To ensure a proper computation of the M value, endpoints implementing the R bit must identify the boundaries of incoming Q Blocks. The same approach described in {#endmarkingblock} should be used.

Looking at the R-bit, unidirectional observation points have an indication of losses experienced by the entire unobserved channel plus those occurred in the path from the sender up to them.

Since the Q Block is sent in one direction, and the corresponding reflected R Block is sent in the opposite direction, the reflected R signal is transmitted with the packet rate of the slowest direction. Namely, if the observed direction is the slowest, there can be multiple Q Blocks transmitted in the unobserved direction before a complete R Block is transmitted in the observed direction. If the unobserved direction is the slowest, the observed direction can be sending R Blocks of the same size repeatedly before it can update the signal to account for a newly-completed Q Block.

4.5.1. R+Q Bits - Using R and Q Bits for Passive Loss Measurement

Since both sSquare and Reflection square bits are toggled at most every N packets (except for the first transition of the R-bit as explained before), an on-path observer can count the number of packets of each marking block and, knowing the value of N, can estimate the amount of loss experienced by the connection. An observer can calculate different measurements depending on whether it is able to observe a single direction of the traffic or both directions.

Single directional observer:

- upstream loss in the observed direction: the loss between the sender and the observation point (see Section 4.2.2)

- "three-quarters" connection loss: the loss between the receiver and the sender in the unobserved direction plus the loss between the sender and the observation point in the observed direction
- end-to-end loss in the unobserved direction: the loss between the receiver and the sender in the opposite direction

Two directions observer (same metrics seen previously applied to both direction, plus):

- client-observer half round-trip loss: the loss between the client and the observation point in both directions
- observer-server half round-trip loss: the loss between the observation point and the server in both directions
- downstream loss: the loss between the observation point and the receiver (applicable to both directions)

4.5.1.1. Three-Quarters Connection Loss

Except for the very first block in which there is nothing to reflect (a complete Q Block has not been yet received), packets are continuously R-bit marked into alternate blocks of size lower or equal than N. Knowing the value of N, an on-path observer can estimate the amount of loss occurred in the whole opposite channel plus the loss from the sender up to it in the observation channel. As for the previous metric, the "three-quarters" connection loss rate ("tqloss") is one minus the average number of packets in a block of packets with the same R value ("t") divided by "N" ("tqloss=1-avg(t)/N").

```

=====>
= *****      -----Obs----->      *****
= * Client *                               * Server *
= *****      <-----              *****
<=====

```

(a) in client-server channel (tqloss_up)

```

=====>
*****      ----->      ***** =
* Client *                               * Server * =
*****      <-----Obs-----      ***** =
<=====

```

(b) in server-client channel (tqloss_down)

Three-quarters connection loss

The following metrics derive from this last metric and the upstream loss produced by the Q Bit.

4.5.1.2. End-To-End Loss in the Opposite Direction

End-to-end loss in the unobserved direction ("eloss_unobserved") relates to the "three-quarters" connection loss ("tqloss") and upstream loss in the observed direction ("uloss") as $(1 - \text{eloss_unobserved})(1 - \text{uloss}) = 1 - \text{tqloss}$. Hence, $\text{eloss_unobserved} = (\text{tqloss} - \text{uloss}) / (1 - \text{uloss})$.

```

*****      -----Obs----->      *****
* Client *                               * Server *
*****      <-----              *****
<=====

```

(a) in client-server channel (eloss_down)

```

=====>
*****      ----->      *****
* Client *                               * Server *
*****      <-----Obs-----      *****

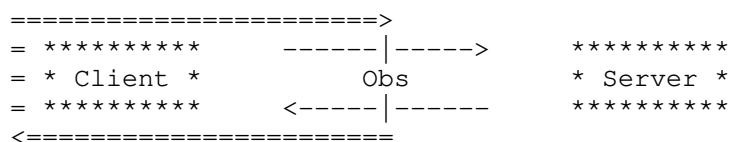
```

(b) in server-client channel (eloss_up)

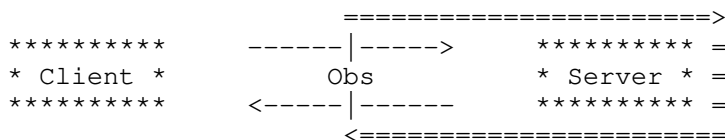
End-To-End loss in the opposite direction

4.5.1.3. Half Round-Trip Loss

If the observer is able to observe both directions of traffic, it is able to calculate two "half round-trip" loss measurements - loss from the observer to the receiver (in a given direction) and then back to the observer in the opposite direction. For both directions, "half round-trip" loss ("hrtloss") relates to "three-quarters" connection loss ("tqloss_opposite") measured in the opposite direction and the upstream loss ("uloss") measured in the given direction as $(1-uloss)(1-hrtloss)=1-tqloss_opposite$. Hence, $hrtloss=(tqloss_opposite-uloss)/(1-uloss)$.



(a) client-observer half round-trip loss (hrtloss_co)



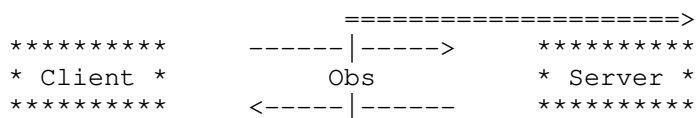
(b) observer-server half round-trip loss (hrtloss_os)

Half Round-trip loss (both direction)

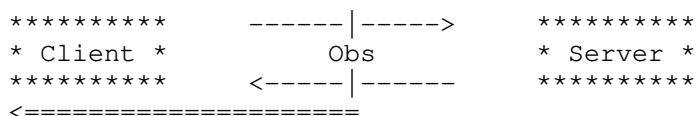
4.5.1.4. Downstream Loss

If the observer is able to observe both directions of traffic, it is able to calculate two downstream loss measurements using either end-to-end loss and upstream loss, similar to the calculation in Section 4.4.1.1 or using "half round-trip" loss and upstream loss in the opposite direction.

For the latter, $dloss=(hrtloss-uloss_opposite)/(1-uloss_opposite)$.



(a) in client-server channel (dloss_up)



(b) in server-client channel (dloss_down)

Downstream loss

4.5.2. Enhancement of R Block Length Computation

The use of the rounding function used in the M computation introduces errors that can be minimized by storing the rounding applied each time M is computed, and using it during the computation of the M value in the following R Block.

This can be achieved introducing the new "r_avg" parameter in the computation of M. The new formula is "Mr=avg(p)+r_avg; M=round(Mr); r_avg=Mr-M" where the initial value of "r_avg" is equal to 0.

4.5.3. Improved Resilience to Packet Reordering

When a protocol implementing the marking mechanism is able to detect when packets are received out of order, it can improve resilience to packet reordering beyond what is possible using methods described in Section 4.2.3.

This can be achieved by updating the size of the current R Block while this is being transmitted. The reflection block size is then updated every time an incoming reordered packet of the previous Q Block is detected. This can be done if and only if the transmission of the current reflection block is in progress and no packets of the following Q Block have been received.

4.6. Improved Q and R Bits Resilience to Burst Losses

Burst losses can affect Q and R measurements accuracy. Generally, burst losses can be absorbed and correctly measured if smaller than the established Q Block length. On the other hand, entire periods might be wiped out if the burst sizes become too large thus making the observer completely unaware of their loss.

To improve burst loss resilience, an observer might consider a received Q or R Block larger than the selected Q Block length as a burst loss event. Then compute the loss as three times Q Block length minus the measured block length. By doing so, an observer can detect burst losses of less than two blocks (e.g., less than 128 packets for Q Block length of 64 packets). A burst loss equal or greater than two consecutive periods would still remain unnoticed by the observer (or underestimated if a period longer than Q Block length were formed).

5. Summary of Delay and Loss Marking Methods

This section summarizes the marking methods described in this draft.

For the Delay measurement, it is possible to use the spin bit and/or the delay bit. A unidirectional or bidirectional observer can be used.

Method	# of bits	Available Delay Metrics		Impairments Resiliency	# of meas.
		UNIDIR Observer	BIDIR Observer		
S: Spin Bit	1	RTT	x2 Half RTT	low	very high
D: Delay Bit	1	RTT	x2 Half RTT	high	medium
D [^] : Hidden Delay Bit	1	RTT [^]	x2 Left Half [^] Right Half	high	high
SD: Spin Bit & Delay Bit *	2	RTT	x2 Half RTT	high	very high

x2 Same metric for both directions

* Both algorithms work independtly; an observer could use approximate spin bit measures when delay bit ones aren't available

[^] Masked metric (real value can be calculated only by those who know the Additional Delay)

Figure 1: Delay Comparison

For the Loss measurement, each row in the table of Figure 2 represents a loss marking method. For each method the table specifies the number of bits required in the header, the available metrics using an unidirectional or bidirectional observer, applicable protocols, measurement fidelity and delay.

Method	Bits	Available Loss Metrics		Protocols	Measurement Aspects	
		UNIDIR Observer	BIDIR Observer		Fidelity	Delay
T: Round Trip Loss Bit	\$ 1	RT	x2 Half RT	*	Rate by sampling 1/3 to 1/(3*ppa) of pkts over 2 RTT	~6 RTT
Q: Square Bit	1	Upstream	x2	*	Rate over N pkts (e.g. 64)	N pkts (e.g. 64)
L: Loss Event Bit	1	E2E	x2	#	Loss shape (and rate)	Min: RTT Max: RTO
QL: Square + Loss Ev. Bits	2	Upstream Downstream E2E	x2 x2 x2	#	-> see Q -> see Q L -> see L	Up: see Q Others: see L
QR: Square + Ref. Sq. Bits	2	Upstream 3/4 RT !E2E	x2 x2 E2E Downstream Half RT	*	Rate over N*ppa pkts (see Q bit for N)	Up: see Q Others: N*ppa pk (see Q for N)

* All protocols

Protocols employing loss detection (w/ or w/o pure ACK loss detection)

\$ Require a working spin bit

! Metric relative to the opposite channel

x2 Same metric for both directions

ppa Packets-Per-Ack

Q|L See Q if Upstream loss is significant; L otherwise

Figure 2: Loss Comparison

6. ECN-Echo Event Bit

While the primary focus of the draft is on exposing packet loss and delay, modern networks can report congestion before they are forced to drop packets, as described in [ECN]. When transport protocols keep ECN-Echo feedback under encryption, this signal cannot be observed by the network operators. When tasked with diagnosing network performance problems, knowledge of a congestion downstream of an observation point can be instrumental.

If downstream congestion information is desired, this information can be signaled with an additional bit.

- E: The "ECN-Echo Event" bit is set to 0 or 1 according to the Unreported ECN Echo counter, as explained below in Section 6.1.

6.1. Setting the ECN-Echo Event Bit on Outgoing Packets

The Unreported ECN-Echo counter operates identically to Unreported Loss counter (Section 4.3), except it counts packets delivered by the network with CE markings, according to the ECN-Echo feedback from the receiver.

This ECN-Echo signaling is similar to ECN signaling in [ConEx]. ECN-Echo mechanism in QUIC provides the number of packets received with CE marks. For protocols like TCP, the method described in [ConEx-TCP] can be employed. As stated in [ConEx-TCP], such feedback can be further improved using a method described in [ACCURATE].

6.2. Using E Bit for Passive ECN-Reported Congestion Measurement

A network observer can count packets with CE codepoint and determine the upstream CE-marking rate directly.

Observation points can also estimate ECN-reported end-to-end congestion by counting packets in this direction with a E bit equal to 1.

The upstream CE-marking rate and end-to-end ECN-reported congestion can provide information about downstream CE-marking rate. Presence of E bits along with L bits, however, can somewhat confound precise estimates of upstream and downstream CE-markings in case the flow contains packets that are not ECN-capable.

7. Protocol Ossification Considerations

Accurate loss and delay information is not critical to the operation of any protocol, though its presence for a sufficient number of flows is important for the operation of networks.

The delay and loss bits are amenable to "greasing" described in [RFC8701], if the protocol designers are not ready to dedicate (and ossify) bits used for loss reporting to this function. The greasing could be accomplished similarly to the Latency Spin bit greasing in [QUIC-TRANSPORT]. Namely, implementations could decide that a fraction of flows should not encode loss and delay information and, instead, the bits would be set to arbitrary values. The observers would need to be ready to ignore flows with delay and loss information more resembling noise than the expected signal.

8. Examples of Application

8.1. QUIC

The binding of a delay signal to QUIC is partially described in [QUIC-TRANSPORT], which adds the spin bit to the first byte of the short packet header, leaving two reserved bits for future experiments.

To implement the additional signals discussed in this document, the first byte of the short packet header can be modified as follows:

- the delay bit (D) can be placed in the first reserved bit (i.e. the fourth most significant bit `_0x10_`) while the round trip loss bit (T) in the second reserved bit (i.e. the fifth most significant bit `_0x08_`); the proposed scheme is:

```

  0 1 2 3 4 5 6 7
+---+---+---+---+
|0|1|S|D|T|K|P|P|
+---+---+---+---+
```

Scheme 1

- alternatively, a two bits loss signal (QL or QR) can be placed in both reserved bits; the proposed schemes, in this case, are:

```

  0 1 2 3 4 5 6 7
+---+---+---+---+---+---+
|0|1|S|Q|L|K|P|P|
+---+---+---+---+---+---+

```

Scheme 2A

```

  0 1 2 3 4 5 6 7
+---+---+---+---+---+---+
|0|1|S|Q|R|K|P|P|
+---+---+---+---+---+---+

```

Scheme 2B

A further option would be to substitute the spin bit with the delay bit (or hidden delay bit) leaving the two reserved bits for loss detection. The proposed schemes are:

```

  0 1 2 3 4 5 6 7          0 1 2 3 4 5 6 7
+---+---+---+---+---+---+  +---+---+---+---+---+---+
|0|1|D|Q|L|K|P|P|  OR  |0|1|D^|Q|L|K|P|P|
+---+---+---+---+---+---+  +---+---+---+---+---+---+

```

Scheme 3A

```

  0 1 2 3 4 5 6 7          0 1 2 3 4 5 6 7
+---+---+---+---+---+---+  +---+---+---+---+---+---+
|0|1|D|Q|R|K|P|P|  OR  |0|1|D^|Q|R|K|P|P|
+---+---+---+---+---+---+  +---+---+---+---+---+---+

```

Scheme 3B

8.2. TCP

The signals can be added to TCP by defining bit 4 of byte 13 of the TCP header to carry the spin bit or the delay bit, and possibly bits 5 and 6 to carry additional information, like the delay bit and the round-trip loss bit (DT), or a two bits loss signal (QL or QR).

9. Security Considerations

Passive loss and delay observations have been a part of the network operations for a long time, so exposing loss and delay information to the network does not add new security concerns for protocols that are currently observable.

In the absence of packet loss, Q and R bits signals do not provide any information that cannot be observed by simply counting packets

transiting a network path. In the presence of packet loss, Q and R bits will disclose the loss, but this is information about the environment and not the endpoint state. The L bit signal discloses internal state of the protocol's loss detection machinery, but this state can often be gleamed by timing packets and observing congestion controller response.

Hence, loss bits do not provide a viable new mechanism to attack data integrity and secrecy.

9.1. Optimistic ACK Attack

A defense against an Optimistic ACK Attack, described in [QUIC-TRANSPORT], involves a sender randomly skipping packet numbers to detect a receiver acknowledging packet numbers that have never been received. The Q bit signal may inform the attacker which packet numbers were skipped on purpose and which had been actually lost (and are, therefore, safe for the attacker to acknowledge). To use the Q bit for this purpose, the attacker must first receive at least an entire Q Block of packets, which renders the attack ineffective against a delay-sensitive congestion controller.

A protocol that is more susceptible to an Optimistic ACK Attack with the loss signal provided by Q bit and uses a loss-based congestion controller, should shorten the current Q Block by the number of skipped packets numbers. For example, skipping a single packet number will invert the square signal one outgoing packet sooner.

Similar considerations apply to the R Bit, although a shortened R Block along with a matching skip in packet numbers does not necessarily imply a lost packet, since it could be due to a lost packet on the reverse path along with a deliberately skipped packet by the sender.

10. Privacy Considerations

To minimize unintentional exposure of information, loss bits provide an explicit loss signal - a preferred way to share information per [RFC8558].

New protocols commonly have specific privacy goals, and loss reporting must ensure that loss information does not compromise those privacy goals. For example, [QUIC-TRANSPORT] allows changing Connection IDs in the middle of a connection to reduce the likelihood of a passive observer linking old and new sub-flows to the same device. A QUIC implementation would need to reset all counters when it changes the destination (IP address or UDP port) or the Connection ID used for outgoing packets. It would also need to avoid

incrementing Unreported Loss counter for loss of packets sent to a different destination or with a different Connection ID.

11. IANA Considerations

This document makes no request of IANA.

12. Change Log

TBD

13. Contributors

The following people provided valuable contributions to this document:

- Marcus Ihlar, Ericsson, marcus.ihlar@ericsson.com
- Jari Arkko, Ericsson, jari.arkko@ericsson.com
- Emile Stephan, Orange, emile.stephan@orange.com

14. Acknowledgements

TBD

15. References

15.1. Normative References

- [ConEx] Mathis, M. and B. Briscoe, "Congestion Exposure (ConEx) Concepts, Abstract Mechanism, and Requirements", RFC 7713, DOI 10.17487/RFC7713, December 2015, <<https://www.rfc-editor.org/info/rfc7713>>.
- [ConEx-TCP] Kuehlewind, M., Ed. and R. Scheffenegger, "TCP Modifications for Congestion Exposure (ConEx)", RFC 7786, DOI 10.17487/RFC7786, May 2016, <<https://www.rfc-editor.org/info/rfc7786>>.
- [ECN] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, DOI 10.17487/RFC3168, September 2001, <<https://www.rfc-editor.org/info/rfc3168>>.

- [IP] Postel, J., "Internet Protocol", STD 5, RFC 791, DOI 10.17487/RFC0791, September 1981, <<https://www.rfc-editor.org/info/rfc791>>.
- [IPM-Methods] Morton, A., "Active and Passive Metrics and Methods (with Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799, May 2016, <<https://www.rfc-editor.org/info/rfc7799>>.
- [IPv6] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <<https://www.rfc-editor.org/info/rfc8200>>.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC8558] Hardie, T., Ed., "Transport Protocol Path Signals", RFC 8558, DOI 10.17487/RFC8558, April 2019, <<https://www.rfc-editor.org/info/rfc8558>>.
- [TCP] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <<https://www.rfc-editor.org/info/rfc793>>.

15.2. Informative References

- [ACCURATE] Briscoe, B., Kuehlewind, M., and R. Scheffenegger, "More Accurate ECN Feedback in TCP", draft-ietf-tcpm-accurate-ecn-14 (work in progress), February 2021.
- [AltMark] Fioccola, G., Ed., Capello, A., Cociglio, M., Castaldelli, L., Chen, M., Zheng, L., Mirsky, G., and T. Mizrahi, "Alternate-Marking Method for Passive and Hybrid Performance Monitoring", RFC 8321, DOI 10.17487/RFC8321, January 2018, <<https://www.rfc-editor.org/info/rfc8321>>.
- [ANRW19-PM-QUIC] Bulgarella, F., Cociglio, M., Fioccola, G., Marchetto, G., and R. Sisto, "Performance measurements of QUIC communications", Proceedings of the Applied Networking Research Workshop, DOI 10.1145/3340301.3341127, July 2019.

- [I-D.trammell-ippm-spin]
Trammell, B., "An Explicit Transport-Layer Signal for Hybrid RTT Measurement", draft-trammell-ippm-spin-00 (work in progress), January 2019.
- [I-D.trammell-tsvwg-spin]
Trammell, B., "A Transport-Independent Explicit Signal for Hybrid RTT Measurement", draft-trammell-tsvwg-spin-00 (work in progress), July 2018.
- [IPv6AltMark]
Fioccola, G., Zhou, T., Cociglio, M., Qin, F., and R. Pang, "IPv6 Application of the Alternate Marking Method", draft-ietf-6man-ipv6-alt-mark-04 (work in progress), March 2021.
- [QUIC-TRANSPORT]
Iyengar, J. and M. Thomson, "QUIC: A UDP-Based Multiplexed and Secure Transport", draft-ietf-quic-transport-34 (work in progress), January 2021.
- [RFC8517] Dolson, D., Ed., Snellman, J., Boucadair, M., Ed., and C. Jacquenet, "An Inventory of Transport-Centric Functions Provided by Middleboxes: An Operator Perspective", RFC 8517, DOI 10.17487/RFC8517, February 2019, <<https://www.rfc-editor.org/info/rfc8517>>.
- [RFC8701] Benjamin, D., "Applying Generate Random Extensions And Sustain Extensibility (GREASE) to TLS Extensibility", RFC 8701, DOI 10.17487/RFC8701, January 2020, <<https://www.rfc-editor.org/info/rfc8701>>.
- [SPIN-BIT]
Trammell, B., Vaere, P. D., Even, R., Fioccola, G., Fossati, T., Ihlar, M., Morton, A., and E. Stephan, "Adding Explicit Passive Measurability of Two-Way Latency to the QUIC Transport Protocol", draft-trammell-quic-spin-03 (work in progress), May 2018.
- [TRANSPORT-ENCRYPT]
Fairhurst, G. and C. Perkins, "Considerations around Transport Header Confidentiality, Network Operations, and the Evolution of Internet Transport Protocols", draft-ietf-tsvwg-transport-encrypt-21 (work in progress), April 2021.

[UDP-OPTIONS]

Touch, J., "Transport Options for UDP", draft-ietf-tsvwg-udp-options-12 (work in progress), May 2021.

[UDP-SURPLUS]

Herbert, T., "UDP Surplus Header", draft-herbert-udp-space-hdr-01 (work in progress), July 2019.

Authors' Addresses

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

Email: mauro.cociglio@telecomitalia.it

Alexandre Ferrieux
Orange Labs

Email: alexandre.ferrieux@orange.com

Giuseppe Fioccola
Huawei Technologies
Riesstrasse, 25
Munich 80992
Germany

Email: giuseppe.fioccola@huawei.com

Igor Lubashev
Akamai Technologies

Email: ilubashe@akamai.com

Fabio Bulgarella
Telecom Italia - TIM
Via Reiss Romoli, 274
Torino 10148
Italy

Email: fabio.bulgarella@guest.telecomitalia.it

Isabelle Hamchaoui
Orange Labs

EMail: isabelle.hamchaoui@orange.com

Massimo Nilo
Telecom Italia - TIM
Via Reiss Romoli, 274
Torino 10148
Italy

EMail: massimo.nilo@telecomitalia.it

Riccardo Sisto
Politecnico di Torino

EMail: riccardo.sisto@polito.it

Dmitri Tikhonov
LiteSpeed Technologies

EMail: dtikhonov@litespeedtech.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: 27 April 2022

G. Mirsky
J. Halpern
Ericsson
X. Min
ZTE Corp.
L. Han
China Mobile
24 October 2021

Error Performance Measurement in Packet-switched Networks
draft-mirsky-ippm-epm-04

Abstract

This document describes the use of the error performance metric to characterize a packet-switched network's conformance to the pre-defined set of performance objectives. In this document, metrics that characterize error performance in a packet-switched network (PSN) are defined, as well as methods to measure and calculate them. Also, the requirements for an active Operation, Administration, and Maintenance protocol to support the error performance measurement in PSN are discussed, and potential candidate protocols are analyzed. All metrics and measurement methods are equally applicable to underlay and overlay networks.

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 27 April 2022.

Copyright Notice

Copyright (c) 2021 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 and Acronyms	3
2.2. Requirements Language	4
3. Error Performance Metrics	4
3.1. Measure Error Performance Metrics	4
3.2. Calculate Error Performance Metrics	5
4. Requirements to EPM	5
5. Active OAM Protocol for EPM	6
6. Availability of Anything-as-a-Service	6
7. IANA Considerations	7
8. Security Considerations	8
9. Acknowledgments	8
10. References	8
10.1. Normative References	8
10.2. Informative References	8
Authors' Addresses	9

1. Introduction

Operations, Administration, and Maintenance (OAM) is a collection of methods to detect, characterize, localize failures in a network, and monitor the network's performance using various measurement methods. Traditionally, the former set of OAM tools identified as Fault Management (FM) OAM. The latter - Performance Monitoring (PM) OAM. Some OAM protocols can be used for both groups of tasks, while some serve one particular group. But regardless of how many OAM protocols are in use, network operators and network users are faced with multiple metrics that characterize the network conditions. This document describes a new component of packet-switched network (PSN) OAM.

Error performance measurement (EPM) is a part of an OAM toolset that provides an operator with information related to network measurements for a uni-directional or a bidirectional connection between two systems. In current technology, EPM has been defined only for data communication methods that have a constant bit-rate transmission

[ITU.G.826] and not for PSN, where transmissions are statistically random. As a statistically multiplexed network in a PSN, a receiver node does not expect a packet to arrive from a sender node at a specific moment, less from a particular sender. That is what differentiates PSN from networks built on a constant bit-rate transmission, where a stream of bits between two nodes is always present, whether it represents data or not. That provides the receiver with a predictable number of measurements in a series of measurement intervals. In PSN, on-path OAM methods, i.e., measurement methods that use data flow, cannot provide such predictability and thus be used for EPM. In PSN, EPM needs to use active OAM methods, per definition in [RFC7799]. This document identifies metrics that characterize PSN error performance and methods to measure and calculate them. Also, the requirements for an active OAM protocol to support EPM in PSN are discussed, and potential candidate protocols are analyzed.

2. Conventions used in this document

2.1. Terminology and Acronyms

OAM Operations, Administration, and Maintenance

EP Error Performance

EPM Error Performance Measurement

ES Errored Second

ESR Errored Second Ratio

SES Severely Errored Second

SESR Severely Errored Second Ratio

EFS Error-Free Second

PSN Packet-switched Network

FM Fault Management

PM Performance Monitoring

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. Error Performance Metrics

When analyzing the error performance of a path between two nodes, we need to select a time interval as the unit of EPM. In [ITU.G.826], a time interval of one second is used. It is reasonable to use the same time interval for EPM for PSNs. Further, for the purpose of EPM, each time interval, i.e., second, is classified either as Errored Second (ES), Severely Errored Second (SES), or Error-Free Second (EFS). These are defined as follows:

- * An ES is a time interval during which at least one of the performance parameters degraded below its optimal level threshold or a defect was detected.
- * An SES is a time interval during which at least one the performance parameters degraded below its critical threshold or a defect was detected.
- * Consequently, an EFS is a time interval during which all performance objectives are at or above their respective optimal levels, and no defect has been detected.

The definition of a state of a defect in the network is also necessary for understanding the EPM. In this document, the defect is interpreted as the state of inability to communicate between a particular set of nodes. It is important to note that it is being defined as a state, and thus, it has conditions that define entry into it and exit out of it. Also, the state of defect exists only in connection to the particular group of nodes in the network, not the network as a domain.

3.1. Measure Error Performance Metrics

The definitions of ES, SES, and EFS allow for characterization of the communication between two nodes relative to the level of required and acceptable performance and when performance degrades below the acceptable level. The former condition in this document referred to as network availability. The latter - network unavailability. Based on the definitions, SES is the one-second of network unavailability while ES and EFS present an interval of network availability. But

since the conditions of network are everchanging periods of network availability and unavailability need to be defined with duration larger than one-second interval to reduce the number of state changes while correctly reflecting the network condition. The method to determine the state of the network in terms of EPM OAM is described below:

- * If ten consecutive SES intervals been detected, then the EPM state of the network determined as unavailability and the beginning of that period of unavailability state is at the start of the first SES in the sequence of the consecutive SES intervals.
- * Similarly, ten consecutive non-SES intervals, i.e., either ES or EFS, indicate that the network is in the availability period, i.e., available. The start of that period is at the beginning of the first non-SES interval.
- * Resulting from these two definitions, a sequence of less than ten consecutive SES or non-SES intervals does not change the EPM state of the network. For example, if the EPM state is determined as unavailability, a sequence of seven EFS intervals is not viewed as an availability period.

3.2. Calculate Error Performance Metrics

Determining the period in which the path is currently EP-wise is helpful. But because switching between periods requires ten consecutive one-second intervals, conditions that last shorter intervals may not be adequately reflected. Two additional EP OAM metrics can be used, and they are defined as follows:

- * errored second ratio (ESR) is the ratio of ES to the total number of seconds in a time of the availability periods during a fixed measurement interval.
- * severely errored second ratio (SESR) - is the ratio of SES to the total number of seconds in a time of the availability periods during a fixed measurement interval.

4. Requirements to EPM

TBA

5. Active OAM Protocol for EPM

Digital communication methods characterized as the constant-bit rate digital paths and connections allow measurement of the error performance without using an active OAM. That is possible because a predictable flow of digital signals is expected at an egress system. That is not the case for packet-switched networks that are based on the principle of statistical multiplexing flows. The latter usually improves the utilization of the communication network's resources, but it also makes the flow unpredictable for the egress system. For that reason, an active OAM has to be used in measuring the error performance in a network. A combination of OAM protocols can provide the necessary for EPM functionality. For example, Bidirectional Forwarding Detection (BFD) [RFC5880] can be used to monitor the continuity of a path between the ingress and egress systems. And STAMP [RFC8762] can be used to measure and calculate performance metrics that are used as Service Level Objectives. But using two protocols and correlating the state of the network from them adds to the complexity in network operation.

6. Availability of Anything-as-a-Service

Anything as a service (XaaS) describes a general category of services related to cloud computing and remote access. These services include the vast number of products, tools, and technologies that are delivered to users as a service over the Internet. In this document, the availability of XaaS is viewed as the ability to access the service over a period of time with pre-defined performance objectives. Among the advantages of the XaaS model are:

- * Improving the expense model by purchasing services from providers on a subscription basis rather than buying individual products, e.g., software, hardware, servers, security, infrastructure, and install them on-site, and then link everything together to create networks.
- * Speeding new apps and business processes by quickly adapting to changing market conditions with new applications or solutions.
- * Shifting IT resources to specialized higher-value projects that use the core expertise of the company.

But XaaS model also has potential challenges:

- * Possible downtime resulting from issues of internet reliability, resilience, provisioning, and managing the infrastructure resources.

- * Performance issues caused by depleted resources like bandwidth, computing power, inefficiencies of virtualized environments, ongoing management and security of multi-cloud services.
- * Complexity impacts enterprise IT team that must remain in the process of the continued learning of the provided services.

The framework and metrics of the EPM defined in Section 3 allow a provider of XaaS and their customers to quantify, measure, monitor for conformance what is often referred to as an ephemeral - availability of the service to be delivered. There are other definitions and methods of expressing availability. For example, [HighAvailability-WP] uses the following equation:

Availability Average = $MTBF / (MTBF + MTRR)$,

where:

MTBF (Mean Time Between Failures) - mean time between individual component failures. For example, a hard drive malfunction or hypervisor reboot.

MTRR (Mean Time To Repair) - refers to how long it takes to fix the broken component or the application to come back online,

While this approach estimates the expected availability of a XaaS, the EPM reflects near-real-time availability of a service as experienced by a user. It also provides valuable data for more accurate and realistic MTBF and MTRR in the particular environment, and simplifies comparison of different solutions that may use redundant servers (web and database), load balancers.

In another field of communication, mobile voice and data services, the definition of service availability is understood as "the probability of successful service reception: a given area is declared "in-coverage" if the service in that area is available with a pre-specified minimum rate of success. Service availability has the advantage of being more easily understandable for consumers and is expressed as a percentage of the number of attempts to access a given service." [BEREC-CP]. The definition of the availability used in the EPM throughout this document is close to the quoted above. It might be considered as the extension that allows regulators, operators, and consumers to compare not only the rate of successfully establishing a connection but the quality of the connection during its lifetime.

7. IANA Considerations

TBA

8. Security Considerations

TBA

9. Acknowledgments

TBA

10. References

10.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

10.2. Informative References

- [BEREC-CP] Body of European Regulators for Electronic Communications, "BEREC Common Position on information to consumers on mobile coverage", Common Approaches/Positions BoR (18) 237, June 2018, <https://berec.europa.eu/eng/document_register/subject_matter/berec/regulatory_best_practices/common_approaches_positions/8315-berec-common-position-on-information-to-consumers-on-mobile-coverage>.
- [HighAvailability-WP] Avi Freedman, Server Central, "High Availability in Cloud and Dedicated Infrastructure", <<https://www.deft.com/wp-content/uploads/pdf/SCTG-High-Availability-White-Paper-Part-2.pdf>>.
- [ITU.G.826] ITU-T, "End-to-end error performance parameters and objectives for international, constant bit-rate digital paths and connections", ITU-T G.826, December 2002.
- [RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <<https://www.rfc-editor.org/info/rfc5880>>.

- [RFC7799] Morton, A., "Active and Passive Metrics and Methods (with Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799, May 2016, <<https://www.rfc-editor.org/info/rfc7799>>.
- [RFC8762] Mirsky, G., Jun, G., Nydell, H., and R. Foote, "Simple Two-Way Active Measurement Protocol", RFC 8762, DOI 10.17487/RFC8762, March 2020, <<https://www.rfc-editor.org/info/rfc8762>>.

Authors' Addresses

Greg Mirsky
Ericsson

Email: gregimirsky@gmail.com

Joel Halpern
Ericsson

Email: joel.halpern@ericsson.com

Xiao Min
ZTE Corp.

Email: xiao.min2@zte.com.cn

Liuyan Han
China Mobile
32 XuanWuMenXi Street
Beijing
100053
China

Email: hanliuyan@chinamobile.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: July 17, 2022

T. Mizrahi
Huawei
J. Iurman
ULiege
F. Brockners
Cisco
January 13, 2022

In Situ OAM Profile for the Linux Kernel Implementation
draft-mizrahi-ippm-ioam-linux-profile-02

Abstract

In Situ Operations, Administration and Maintenance (IOAM) is used for monitoring network performance and for detecting traffic bottlenecks and anomalies. This document defines an IOAM profile that is used in the Linux kernel implementation, starting from the Linux 5.15 kernel.

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 July 17, 2022.

Copyright Notice

Copyright (c) 2022 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. The Linux IOAM Profile	2
2.1. Use Cases	2
2.2. IOAM Version	3
2.3. IOAM Options	3
2.4. Encapsulation	3
2.5. IOAM Supported Data Fields	4
2.6. Trace Option-Type Flags	5
2.7. Timestamp Format	5
2.8. Profile Coexistence	5
2.9. Validity	5
3. Notes about the IOAM Support in Linux	5
4. IANA Considerations	6
5. Security Considerations	6
6. Normative References	6
Authors' Addresses	7

1. Introduction

IOAM [I-D.ietf-ippm-ioam-data] is used for monitoring traffic in the network by incorporating IOAM data fields into in-flight data packets.

An IOAM profile [I-D.mizrahi-ippm-ioam-profile] defines a use case or a set of use cases for IOAM, and an associated set of rules that restrict the scope and features of the IOAM specification, thereby limiting it to a subset of the full functionality.

This document introduces a profile of IOAM that is used in the Linux kernel implementation. The profile is intended to formally specify the subset of features that are in scope, and to enable other implementations to interoperate with the Linux implementation.

2. The Linux IOAM Profile

2.1. Use Cases

The Linux kernel implementation enables the functionality of any of the following nodes:

- o IOAM encapsulating node
- o IOAM transit node

- o IOAM decapsulating node

One possible use case is a set of Linux-based hosts that function as IOAM encapsulating and decapsulating nodes, interconnected by IOAM transit nodes that are not necessarily Linux-based. Thus, Linux-based implementations are expected to interoperate with other implementations that comply to this profile.

Another possible use case is a homogenous setting in which all IOAM nodes are Linux-based.

2.2. IOAM Version

The current profile is based on [I-D.ietf-ippm-ioam-data-15], which is a work-in-progress version of IOAM.

2.3. IOAM Options

The current profile uses the Pre-allocated Trace Option-Type. It is assumed that one IOAM option is used in an IOAM encapsulated packet.

2.4. Encapsulation

This profile uses an IPv6 encapsulation for the IOAM option, i.e., the option is encapsulated in an IPv6 Extension Header. Generally speaking, this extension header may be an extension of an IPv6 tunnel header, or it may be an extension of the end-to-end IPv6 header. Both cases are discussed below. The extension header is used for the IOAM Pre-allocated Trace Option-Type, as defined in [I-D.ietf-ippm-ioam-ipv6-options-06], which is a work-in-progress version of the IPv6 IOAM option.

The IPv6 Extension Header is a Hop-by-Hop Options header, that contains the IOAM Trace Option-Type. The Hop-by-Hop Options header can include one or more options, such that one of these options is the IOAM Pre-allocated Trace Option-Type. Figure 1 illustrates the format of this Hop-by-Hop Options header when the IOAM Pre-allocated Trace Option-Type is the only Hop-by-Hop option. If more options are present the format will change accordingly.

As illustrated in Figure 1, the first 2 octets are the Hop-by-Hop Options header [RFC8200], followed by a 2 octet Padding field. The following 4 octets are the IOAM IPv6 option header [I-D.ietf-ippm-ioam-ipv6-options-06]. The IOAM Option includes the 8 octet Pre-allocated Trace Option-Type header [I-D.ietf-ippm-ioam-data-15], followed by the Option Data.

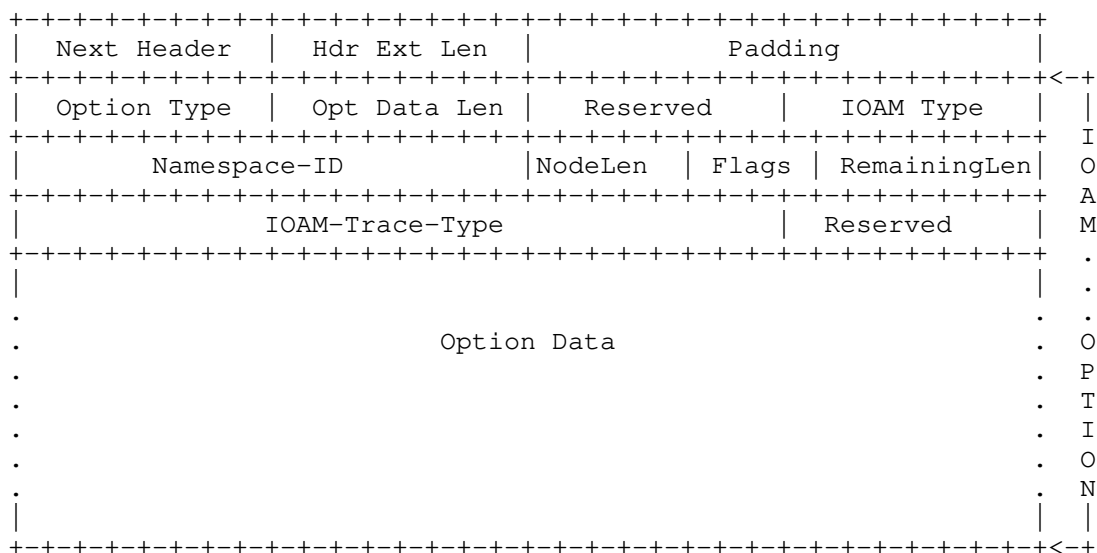


Figure 1: IPv6 IOAM Extension Header Format

Starting from the Linux 5.16 kernel, the IOAM encapsulation might also use an IPv6 tunnel for in-transit packets, as defined in [RFC2473], and as illustrated in Figure 2. The kernel supports both alternatives: either encapsulation using an IPv6 tunnel, or pushing the IOAM option as an IPv6 extension header of the existing IPv6 header.

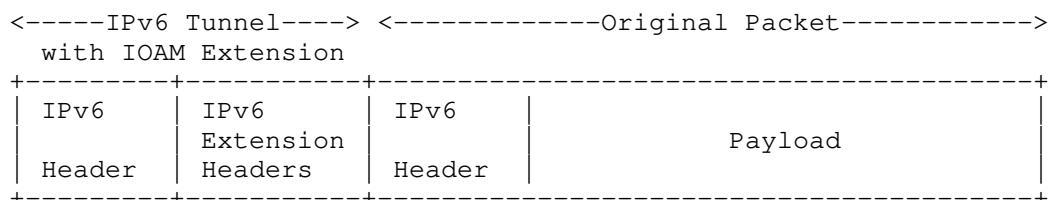


Figure 2: IOAM in IPv6 Tunnel Encapsulation

2.5. IOAM Supported Data Fields

The current profile supports all the data field types that are defined in [I-D.ietf-ippm-ioam-data-15] for the Pre-allocated Trace Option-Type, except for the Checksum Complement field, which is not required in this profile, since the IOAM Trace Option is encapsulated directly in an IPv6 Extension Header, without any additional layers that use a checksum.

2.6. Trace Option-Type Flags

This profile only uses the Overflow flag.

2.7. Timestamp Format

This profile uses the POSIX timestamp format.

2.8. Profile Coexistence

It is assumed that the current profile is used in a confined administrative domain in which no other IOAM profiles are used. Therefore, it is assumed that the current profile does not coexist with other profiles.

2.9. Validity

An IOAM transit/decapsulating node that receives a packet with IOAM options that do not comply to the current profile is expected to forward/decapsulate the packet without IOAM processing, if it is able to do so. If a decapsulating node is not able to decapsulate an IOAM option that is not compliant to the current profile, the packet is discarded.

3. Notes about the IOAM Support in Linux

The current Linux implementation supports all the data field types defined in [I-D.ietf-ippm-ioam-data-15] for the Pre-allocated Trace Option-Type. Specifically, the Linux implementation does not update the transit delay, the queue depth, the checksum complement and the buffer occupancy. These four data field types are passively supported, meaning the Linux implementation can add the Pre-allocated Trace Option-Type including these fields, but cannot populate them with system information. They are populated with empty values and, therefore, interoperability is possible with other IOAM nodes that support these fields.

The following table summarizes the data field type support in the Linux implementation.

Data field type	Status
Hop_Lim and node_id (short format)	Supported
Ingress_if_id and egress_if_id (short format)	Supported
Timestamp seconds	Supported
Timestamp fraction	Supported
Transit delay	Passive support
Namespace specific data (short format)	Supported
Queue depth	Supported*
Checksum complement	Passive support
Hop_Lim and node_id (wide format)	Supported
Ingress_if_id and egress_if_id (wide format)	Supported
Namespace specific data (wide format)	Supported
Buffer occupancy	Passive support
Opaque State Snapshot	Supported

*: Queue depth is supported starting from the Linux 5.17 kernel, and has passive support for older kernel versions.

Both the Opaque State Snapshot and the Namespace specific data are supported in the Linux implementation by incorporating configurable values into these fields. Notably, Linux-based IOAM nodes can interoperate with other nodes that use the Opaque State Snapshot and/or the Namespace specific data in a more flexible way.

If an IOAM transit node receives a packet with one or more undefined bits of the trace type set to 1, it will add corresponding node data filled with the reserved value 0xFFFFFFFF, as defined in [I-D.ietf-ippm-ioam-data-15]. This allows for some flexibility from an interoperability point of view.

4. IANA Considerations

This document does not include any requests from IANA.

5. Security Considerations

The security considerations of IOAM profiles are discussed in [I-D.mizrahi-ippm-ioam-profile]. The current document does not present any new security considerations.

6. Normative References

- [I-D.ietf-ippm-ioam-data]
 Brockners, F., Bhandari, S., and T. Mizrahi, "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data-17 (work in progress), December 2021.

- [I-D.ietf-ippm-ioam-data-15]
Brockners, F., Bhandari, S., and T. Mizrahi, "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data-15 (work in progress), October 2021.
- [I-D.ietf-ippm-ioam-ipv6-options-06]
Bhandari, S. and F. Brockners, "In-situ OAM IPv6 Options", draft-ietf-ippm-ioam-ipv6-options-06 (work in progress), July 2021.
- [I-D.mizrahi-ippm-ioam-profile]
Mizrahi, T., Brockners, F., Bhandari, S., Sivakolundu, R., Pignataro, C., Kfir, A., Gafni, B., Spiegel, M., Zhou, T., and J. Lemon, "In Situ OAM Profiles", draft-mizrahi-ippm-ioam-profile-05 (work in progress), August 2021.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC2473] Conta, A. and S. Deering, "Generic Packet Tunneling in IPv6 Specification", RFC 2473, DOI 10.17487/RFC2473, December 1998, <<https://www.rfc-editor.org/info/rfc2473>>.
- [RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <<https://www.rfc-editor.org/info/rfc8200>>.

Authors' Addresses

Tal Mizrahi
Huawei
8-2 Matam
Haifa
Israel

Email: tal.mizrahi.phd@gmail.com

Justin Iurman
Universite de Liege
10, Allee de la decouverte (B28)
Sart-Tilman, LIEGE 4000
Belgium

Email: justin.iurman@uliege.be

Frank Brockners
Cisco Systems, Inc.
Hansaallee 249, 3rd Floor
DUESSELDORF, NORDRHEIN-WESTFALEN 40549
Germany

Email: fbrockne@cisco.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 28, 2022

L. Ciavattone
A. Morton
AT&T Labs
October 25, 2021

Test Protocol for One-way IP Capacity Measurement
draft-morton-ippm-capacity-metric-protocol-02

Abstract

This memo addresses the problem of protocol support for measuring Network Capacity metrics in RFC xxxx: draft-ietf-ippm-capacity-metric-method, where the method deploys a feedback channel from the receiver to control the sender's transmission rate in near-real-time. It supplies a simple protocol to perform the measurements.

See Section 10: The authors seek feedback to determine what additional features will be necessary for an IETF Standards Track Protocol, beyond what is present in the running code available now.

Status of This Memo

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

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

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

This Internet-Draft will expire on April 28, 2022.

Copyright Notice

Copyright (c) 2021 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
1.1. Requirements Language	3
2. Scope, Goals, and Applicability	3
3. Protocol Overview	4
4. General Parameters and Definitions	5
5. Setup Request and Response Exchange	7
5.1. Setup Response Processing at the Client	11
6. Test Activation Request and Response	11
6.1. Test Activation Request at the client	11
6.2. Test Activation Request - server response	13
6.3. Test Activation Response action at the client	15
7. Test Stream Transmission and Measurement Feedback Messages .	15
7.1. Test Packet PDU and Roles	15
7.2. Status PDU	18
8. Stopping the Test	23
9. Method of Measurement	24
9.1. Running Code	24
10. Security Considerations	24
11. IANA Considerations	26
12. Acknowledgments	27
13. References	27
13.1. Normative References	27
13.2. Informative References	28
Authors' Addresses	30

1. Introduction

The IETF's efforts to define Network and Bulk Transport Capacity have been chartered and finally progressed after over twenty years.

Over that time, the performance community has seen development of Informative definitions in [RFC3148] for Framework for Bulk Transport Capacity (BTC), RFC 5136 for Network Capacity and Maximum IP-layer Capacity, and the Experimental metric definitions and methods in [RFC8337], Model-Based Metrics for BTC.

This memo looks at the problem of measuring Network Capacity metrics defined in [I-D.ietf-ippm-capacity-metric-method] where the method deploys a feedback channel from the receiver to control the sender's transmission rate in near-real-time.

Although there are several test protocol already available for support and manage active measurements, this protocol is a major departure from their operation:

1. UDP transport is used for all setup, test activation, and control messages, and for results feedback (not TCP), simplifying operations.
2. TWAMP [RFC5357] and STAMP [RFC8762] use the philosophy that one host is a Session-Reflector, sending test packets every time they receive a test packet. This protocol supports a one-way test with periodic status messages returned to the sender. These messages are also a basis for on-path Round-trip delay measurements, which are a key input to the load adjustment search algorithm.
3. OWAMP [RFC4656] supports one-way testing with results Fetch at the end of the test session. This protocol supports a one-way test and requires periodic status messages returned to the sender to support the load adjustment search algorithm.
4. The security features of OWAMP [RFC4656] and TWAMP [RFC5357] have been described as "unusual", to the point that IESG approved their use while also asking that these methods not be used again. Further, the common OWAMP [RFC4656] and TWAMP [RFC5357] approach to security is over 15 years old at this time.

Note: the -02 update of this draft will be the last that describes version 8 of the protocol in the running code. Future updates of the draft will correspond to protocol version 9 and higher versions.

1.1. Requirements Language

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

2. Scope, Goals, and Applicability

The scope of this memo is to define a protocol to measure the Maximum IP-Layer Capacity metric and according to the standardized method.

The continued goal is to harmonize the specified metric and method across the industry, and this protocol supports the specifications of IETF and other Standards Development Organizations.

All active testing protocols currently defined by the IPPM WG are UDP-based, but this protocol specifies both control and test protocols using UDP transport. Also, the control protocol continues operating during testing to convey results and dynamic configurations.

The primary application of the protocol described here is the same as in Section 2 of [RFC7497] where:

- o The access portion of the network is the focus of this problem statement. The user typically subscribes to a service with bidirectional access partly described by rates in bits per second.

3. Protocol Overview

This section gives an informative overview of the communication protocol between two test end-points (without expressing requirements: later sections provide details and requirements).

One end-point takes the role of server, awaiting connection requests on a well-known port from the other end-point, the client.

The client requires configuration of a test direction parameter (upstream or downstream test) as well as the hostname or IP address of the server in order to begin the setup and configuration exchanges with the server.

The protocol uses UDP transport and has four phases:

1. Setup Request and Response Exchange: The client requests to begin a test by communicating its protocol version, intended security mode, and jumbo datagram support. The server either confirms matching configuration or rejects the connection. The server also communicates the ephemeral port for further communication when accepting the client's request.
2. Test Activation Request and Response: the client composes a request conveying parameters such as the testing direction, the duration of the test interval and test sub-intervals, and various thresholds. The server then chooses to accept, ignore or modify any of the test parameters, and communicates the set that will be used unless the client rejects the modifications. Note that the client assumes that the Test Activation exchange has opened any co-located firewalls and network address/port translators for the test connection (in response to the Request packet on the ephemeral port) and the traffic that follows. If the Test Activation is rejected or fails, the client assumes that the

firewall will close the address/port combination after the firewall's configured idle traffic time-out.

3. Test Stream Transmission and Measurement Feedback Messages:
Testing proceeds with one end-point sending load PDUs and the other end-point receiving the load PDUs and sending frequent status messages to communicate status and transmission conditions there. The feedback messages are input to a load-control algorithm at the server, which controls future sending rates at either end-point as needed. The choice to locate the load-control algorithm at the server, regardless of transmission direction, means that the algorithm can be updated more easily at a host within the network, and at a fewer number of hosts than the number of clients.
4. Stopping the Test: The server initiates the phase to stop the test by setting the STOP1 indication in load PDUs or sstatus feedback messages. The client acknowledges by setting the STOP2 in further load PDUs or messages, and a graceful connection termination at each end-point follows. (Since the load PDUs and feedback messages are used, this phase is kind of a sub-phase of 3.) If the Test traffic stops or the communication path fails, the client assumes that the firewall will close the address/port combination after the firewall's configured idle traffic time-out.

4. General Parameters and Definitions

This section lists the REQUIRED input factors to specify a Sender or Receiver metric.

- o Src, the address of a host (such as the globally routable IP address).
- o Dst, the address of a host (such as the globally routable IP address).
- o MaxHops, 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 (implemented in the TTL or Hop Limit).
- o T0, the time at the start of measurement interval, when packets are first transmitted from the Source.
- o I, the nominal duration of a measurement interval at the destination (default 10 sec)

- o dt, the nominal duration of m equal sub-intervals in I at the destination (default 1 sec)
- o dtn, the beginning boundary of a specific sub-interval, n, one of m sub-intervals in I
- o FT, the feedback time interval between status feedback messages communicating measurement results, sent from the receiver to control the sender. The results are evaluated to determine how to adjust the current offered load rate at the sender (default 50ms)
- o Tmax, a maximum waiting time for test packets to arrive at the destination, set sufficiently long to disambiguate packets with long delays from packets that are discarded (lost), such that the distribution of one-way delay is not truncated.
- o F, the number of different flows synthesized by the method (default 1 flow)
- o 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). Note: The IPv6 flow label MAY be included in the flow definition when routers have complied with [RFC6438] guidelines.
- o Type-P, the complete description of the test packets for which this assessment applies (including the flow-defining fields). Note that the UDP transport layer is one requirement for test packets specified below. Type-P is a parallel concept to "population of interest" defined in clause 6.1.1 of[Y.1540].
- o PM, a list of fundamental metrics, such as loss, delay, and reordering, and corresponding target performance threshold. At least one fundamental metric and target performance threshold MUST be supplied (such as One-way IP Packet Loss [RFC7680] equal to zero).

A non-Parameter which is required for several metrics is defined below:

- o T, the host time of the *first* test packet's *arrival* as measured at the destination Measurement Point, or MP(Dst). There may be other packets sent between source and destination hosts that are excluded, so this is the time of arrival of the first packet used for measurement of the metric.

Note that time stamp format and resolution, sequence numbers, etc. will be established by this memo.

5. Setup Request and Response Exchange

All messages defined in this section SHALL use UDP transport. The hosts SHALL calculate and include the UDP checksum, or check the UDP checksum as necessary.

The client SHALL begin the Control protocol connection by sending a Setup Request message to the server's control port.

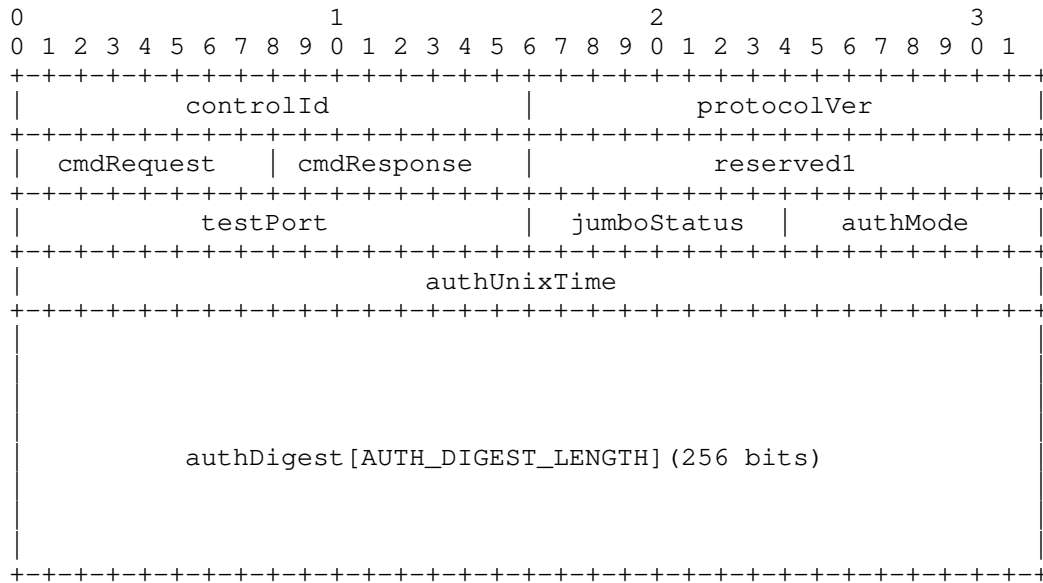
The client SHALL simultaneously start a test initiation timer so that if the control protocol fails to complete all exchanges in the allocated time, the client software SHALL exit (close the UDP socket and indicate an error message to the user).

(Note: in version 8, the watchdog time-out is configured, in udpst.h, as `#define WARNING_NOTRAFFIC 1` // Receive traffic stopped warning threshold (sec) `#define TIMEOUT_NOTRAFFIC (WARNING_NOTRAFFIC + 4)` or 5 seconds)

The Setup Request message PDU SHALL be organized as follows:

```
uint16_t controlId;    // Control ID = 0xACE1
uint16_t protocolVer;  // Protocol version = 0x08
uint8_t cmdRequest;    // Command request = 1 (request)
uint8_t cmdResponse;   // Command response = 0
uint16_t reserved1;    // Reserved (alignment)
uint16_t testPort;     // Test port on server (=0 for Request)
uint8_t jumboStatus;   // Jumbo datagram support status (BOOL)
uint8_t authMode;      // Authentication mode
uint32_t authUnixTime; // Authentication time stamp
unsigned char authDigest[AUTH_DIGEST_LENGTH] // SHA256_DIGEST_LENGTH = 32
oct
```

The UDP PDU format layout SHALL be as follows (big-endian AB):



When the server receives the Setup Request it SHALL validate the request by checking the protocol version, the jumbo datagram support indicator, and the authentication data if utilized. If the client has selected options for:

- o Jumbo datagram support status (BOOL),
- o Authentication mode, and
- o Authentication time stamp

that do not match the server configuration, the server MUST reject the Setup Request.

(Note: in version 8, the watchdog time is configured, in udpst.h, as `#define WARNING_NOTRAFFIC 1 // Receive traffic stopped warning threshold (sec) #define TIMEOUT_NOTRAFFIC (WARNING_NOTRAFFIC + 4) or 5 seconds`)

If the Setup Request must be rejected (due to any of the reasons in the Command response codes listed below), a Setup Response SHALL be sent back to the client with a corresponding command response value indicating the reason for the rejection.

```

uint16_t controlId;    // Control ID = 0xACE1
uint16_t protocolVer;  // Protocol version = 0x08
uint8_t cmdRequest;    // Command request = 2 (reply)
uint8_t cmdResponse;   // Command response = <see table below>
uint16_t reserved1;    // Reserved (alignment)
uint16_t testPort;     // Test port on server (available port in Response
)

uint8_t jumboStatus;   // Jumbo datagram support status (BOOL)
uint8_t authMode;      // Authentication mode
uint32_t authUnixTime; // Authentication time stamp
unsigned char authDigest[AUTH_DIGEST_LENGTH] // 32 octets, MBZ

```

Command Response Codes

Control Header Setup Request Code CHSR_CRSP_NONE	0 = None
Control Header Setup Request Code CHSR_CRSP_ACKOK	1 = Acknowledgement
Control Header Setup Request Code CHSR_CRSP_BADVER	2 = Bad Protocol Version
Control Header Setup Request Code CHSR_CRSP_BADJS	3 = Invalid Jumbo datagram option
Control Header Setup Request Code CHSR_CRSP_AUTHNC	4 = Unexpected Authentication in Setup Request
Control Header Setup Request Code CHSR_CRSP_AUTHREQ	5 = Authentication missing in Setup Request
Control Header Setup Request Code CHSR_CRSP_AUTHINV	6 = Invalid authentication method
Control Header Setup Request Code CHSR_CRSP_AUTHFAIL	7 = Authentication failure
Control Header Setup Request Code CHSR_CRSP_AUTHTIME	8 = Authentication time is invalid in Setup Request

@@@ To Do: How do we communicate multiple errors when the server sends the Setup Response? Is an error hierarchy possible, where Bad Protocol Version means that none of the other aspects (higher error numbers) were checked? New text to address this issue appears below:

There is a hierarchy of Command Response codes, beginning with: "2 = Bad Protocol Version", which SHALL be checked first because it is a fixed field length and the most reliable check. The server SHOULD communicate the first error condition detected in the order listed below:

```

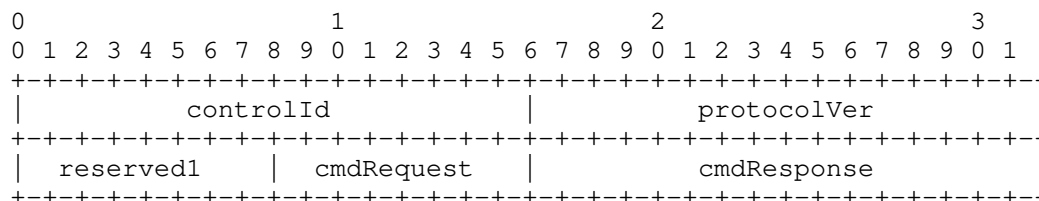
3 = Invalid Jumbo datagram option
5 = Authentication missing in Setup Request
4 = Unexpected Authentication in Setup Request
6 = Invalid authentication method (SHA-256 not used)
7 = Authentication failure (both shared secret and time)
8 = Authentication time is invalid in Setup Request (replay attack)

```

The only circumstance when a server would not communicate the appropriate Command Response Code for an error condition above is when an attack has been detected, in which case the server will allow setup attempts with errors to terminate silently.

@@@ Or, if multiple error codes are wanted, would a flag system work better? For an expanded field multiple error codes, we could use decimal values that set 1 bit in the cmdResponse (0,1,2,4,8,...) for

each code, and expand the cmdResponse field to 16 bits by using the nearby reserved field as shown below (and moving the fields within the 32-bit word):



@@@ - end text for discussion -

If the server finds that the Setup Request matches its configuration and is otherwise acceptable, the server SHALL initiate a new connection for the client, using a new UDP socket allocated from the UDP ephemeral port range. Then, the server SHALL start a watchdog timer (to terminate the connection in case the client goes silent), and sends the Setup Response back to the client (see below for composition).

If the Setup Request is accepted by the server, a Setup Response SHALL be sent back to the client with a corresponding command response value indicating 1 = Acknowledgement.

```

uint16_t controlId;    // Control ID = 0xACE1
uint16_t protocolVer;  // Protocol version = 0x08
uint8_t cmdRequest;    // Command request = 2 (reply)
uint8_t cmdResponse;   // Command response = 1 (Acknowledgement)
uint16_t reserved1;    // Reserved (alignment)
uint16_t testPort;     // Test port on server (available port in Response
)
uint8_t jumboStatus;   // Jumbo datagram support status (BOOL)
uint8_t authMode;      // Authentication mode
uint32_t authUnixTime; // Authentication time stamp
unsigned char authDigest[AUTH_DIGEST_LENGTH] // 32 octets, MBZ
...

```

The new connection is associated with a new UDP socket allocated from the UDP ephemeral port range at the server. The server SHALL set a timer for the new connection as a watchdog (in case the client goes quiet) and send the Setup response back to the client.

(Note: in version 8, the watchdog time-out is configured at 5 seconds)

The Setup Response SHALL include the port number at the server for the new socket, and this UDP port-pair SHALL be used for all

subsequent communication. The server SHALL also include the values of:

- o Jumbo datagram support status (BOOL),
- o Authentication mode, and
- o Authentication time stamp

for the client's use on the new connection in its Setup Response, and the remaining 32 octets MUST Be Zero (MBZ).

Finally, the new UDP connection associated with the new socket and port number is opened, and the server awaits communication there.

If a Test Activation request is not subsequently received from the client on this new port number before the watchdog timer expires, the server SHALL close the socket and deallocate the port.

5.1. Setup Response Processing at the Client

When the client receives the Setup response from the server it first checks the cmdResponse value. If this value indicates an error the client SHALL display/report a relevant message to the user or management process and exit. If the client receives a Command Response code (CRSP) that is not equal to one of the codes defined above, then the client MUST terminate the connection and terminate operation of the current Setup Request. If the Command Response code (CRSP) value indicates success the client SHALL compose a Test Activation Request with all the test parameters it desires such as the test direction, the test duration, etc.

6. Test Activation Request and Response

This section is divided according to the sending and processing of the client, server, and again at the client.

All messages defined in this section SHALL use UDP transport. The hosts SHALL calculate and include the UDP checksum, or check the UDP checksum as necessary.

6.1. Test Activation Request at the client

Upon a successful setup, the client SHALL then send the Test Activation Request to the UDP port number the server communicated in the Setup Response.

@@@ To Do: Add Options for UDP payload content (beyond the Test PDU), such as all zeroes, all ones, alternating one and zero, and pseudo-random.

The client SHALL compose Test Activation Request as follows:

```

uint16_t controlId;    // Control ID
uint16_t protocolVer;  // Protocol version
uint8_t cmdRequest;    // Command request, 1 = upstream, 2 = downstream
uint8_t cmdResponse;   // Command response (set to 0)
uint16_t lowThresh;    // Low delay variation threshold
uint16_t upperThresh;  // Upper delay variation threshold
uint16_t trialInt;     // Status feedback/trial interval (ms)
uint16_t testIntTime;  // Test interval time (sec)
uint8_t subIntPeriod;  // Sub-interval period (sec)
uint8_t ipTosByte;     // IP ToS byte for testing
uint16_t srIndexConf;  // Configured sending rate index (see Note below)
uint8_t useOwDelVar;   // Use one-way delay instead of RTT
uint8_t highSpeedDelta; // High-speed row adjustment delta
uint16_t slowAdjThresh; // Slow rate adjustment threshold
uint16_t seqErrThresh; // Sequence error threshold
uint8_t ignoreOooDup;  // Ignore Out-of-Order/Duplicate datagrams
uint8_t reserved1;     // (Alignment)
uint16_t reserved2;    // (Alignment)

```

Control Header Test Activation Command Request Values:

```

CHTA_CREQ_NONE      0 = No Request
CHTA_CREQ_TESTACTUS 1 = Request test in Upstream direction (client to server, client takes the role of sending test packets)
CHTA_CREQ_TESTACTDS 2 = Request test in Downstream direction (server to client, client takes the role of receiving test packets)

```

Control Header Test Activation Command Response Values:

```

CHTA_CRSP_NONE      0 = Used by client when making a Request
CHTA_CRSP_ACKOK     1 = Used by Server in affirmative Response
CHTA_CRSP_BADPARAM  2 = Used by Server to indicate an error; bad parameter; reject;

```

Note: uint16_t srIndexConf is the table index of the configured *fixed* sending rate index to use. The client can request the specified rate, or the server can use this field to coerce a maximum rate in its response. If the server sets to 0 in its response, client SHALL not use fixed rate.

The UDP PDU format of the Test Activation Request is as follows (big-endian AB):

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								
controlId										protocolVer																													
cmdRequest					cmdResponse					lowThresh																													
upperThresh										trialInt																													
testIntTime										subIntPeriod					ipTosByte																								
srIndexConf										useOwDelVar					highSpeedDelta																								
slowAdjThresh										seqErrThresh																													
ignoreOooDup					reserved1					reserved2																													

Note: This is only 28 octets of the 56 octet PDU sent, the rest are MBZ for a Test Activation Request.

The client SHALL use the configuration for

- o Jumbo datagram support status (BOOL),
- o Authentication mode, and
- o Authentication time stamp

requested and confirmed by the server.

6.2. Test Activation Request - server response

After the server receives the Test Activation request on the new connection, it MUST choose to accept, ignore or modify any of the test parameters.

When the server sends the Test Activation response back, it SHALL set the cmd Response field to:

```
uint8_t cmdResponse; // Command response (set to 1, ACK, or 2 error)
```

The server SHALL include all the test parameters again to make the client aware of any changes.

If the client has requested an upstream test, the server SHALL include the transmission parameters from the first row of the sending rate table.

The remaining 28 octets of the Test Activation Request (normally read from the first row of the sending rate table) are called the Sending Rate Structure, and SHALL be organized as follows:

```
uint32_t txIntervall; // Transmit interval (us)
uint32_t udpPayload1; // UDP payload (bytes)
uint32_t burstSize1; // UDP burst size per interval
uint32_t txInterval2; // Transmit interval (us)
uint32_t udpPayload2; // UDP payload (bytes)
uint32_t burstSize2; // UDP burst size per interval
uint32_t udpAddon2; // UDP add-on (bytes)
```

with

```
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     txIntervall1                             |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     udpPayload1                             |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     burstSize1                             |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     txInterval2                             |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     udpPayload2                             |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     burstSize2                             |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     udpAddon2                             |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
```

Note that the server additionally has the option of completely rejecting the request and sending back an appropriate command response value:

```
uint8_t cmdResponse; // Command response (set to 2, error)
```

If activation continues, the new connection is prepared for an upstream OR downstream test.

In the case of a downstream test, the server prepares to send with either a single timer to send status PDUs at the specified interval OR dual timers to send load PDUs based on the first row of the sending rate table.

The server SHALL then send a Test Activation response back to the client, update the watchdog timer with a new time-out value, and set a test duration timer to eventually stop the test.

The new connection is now ready for testing.

6.3. Test Activation Response action at the client

When the client receives the Test Activation response, it first checks the command response value.

If the client receives a Test Activation command response value that indicates an error, the client SHALL display/report a relevant message to the user or management process and exit.

If the client receives a Test Activation command response value that is not equal to one of the codes defined above, then the client MUST terminate the connection and terminate operation of the current Setup Request.

If the client receives a Test Activation command response value that indicates success (CHTA_CRSP_ACKOK) the client SHALL update its configuration to use any test parameters modified by the server.

Next, the client SHALL prepare its connection for either an upstream test with dual timers set to send load PDUs (based on the starting transmission parameters sent by the server), OR a downstream test with a single timer to send status PDUs at the specified interval.

Then, the client SHALL stop the test initiation timer, set a new time-out value for the watchdog timer, and start the timer (in case the server goes quiet).

The connection is now ready for testing.

7. Test Stream Transmission and Measurement Feedback Messages

This section describes the testing phase of the protocol. The roles of sender and receiver vary depending whether the direction of testing is from server to client, or the reverse.

All messages defined in this section SHALL use UDP transport. The hosts SHALL calculate and include the UDP checksum, or check the received UDP checksum before further processing, as necessary.

7.1. Test Packet PDU and Roles

Testing proceeds with one end point sending load PDUs, based on transmission parameters from the sending rate table, and the other end point receiving the load PDUs and sending status messages to communicate the traffic conditions at the receiver.

The watchdog timer at the receiver SHALL be reset each time a test PDU is received. See non-graceful test stop in Section 8 for handling the watchdog/NOTRAFFIC time-out expiration at each end-point.

When the server is sending Load PDUs in the role of sender, it SHALL use the transmission parameters directly from the sending rate table via the index that is currently selected (which was based on the feedback in its received status messages).

However, when the client is sending load PDUs in the role of sender, it SHALL use the discreet transmission parameters that were communicated by the server in its periodic status messages (and not referencing a sending rate table). This approach allows the server to control the individual sending rates as well as the algorithm used to decide when and how to adjust the rate.

The server uses a load adjustment algorithm which evaluates measurements, either it's own or the contents of received feedback messages. This algorithm is unique to udpst; it provides the ability to search for the Maximum IP Capacity that is absent from other testing tools. Although the algorithm depends on the protocol, it is not part of the protocol per se.

The current algorithm has three paths to its decision on the next sending rate:

1. When there are no impairments present (no sequence errors, low delay variation), resulting in sending rate increase.
2. When there are low impairments present (no sequence errors but higher levels of delay variation), so the same sending rate is retained.
3. When the impairment levels are above the thresholds set for this purpose and "congestion" is inferred, resulting in sending rate decrease.

The algorithm also has two modes for increasing/decreasing the sending rate:

- o A high-speed mode to achieve high sending rates quickly, but also back-off quickly when "congestion" is inferred from the measurements. Any two consecutive feedback intervals that have a sequence number anomaly and/or contain an upper delay variation threshold exception in both of the two consecutive intervals, count as the two consecutive feedback measurements required to declare "congestion" within a test.

- o A single-step mode where all rate adjustments use the minimum increase or decrease of one step in the sending rate table. The single step mode continues after the first inference of "congestion" from measured impairments.

On the other hand, the test configuration MAY use a fixed sending rate requested by the client, using the field below:

```
uint16_t srIndexConf; // Configured sending rate index
```

The client MAY communicate the desired fixed rate in it's activation request.

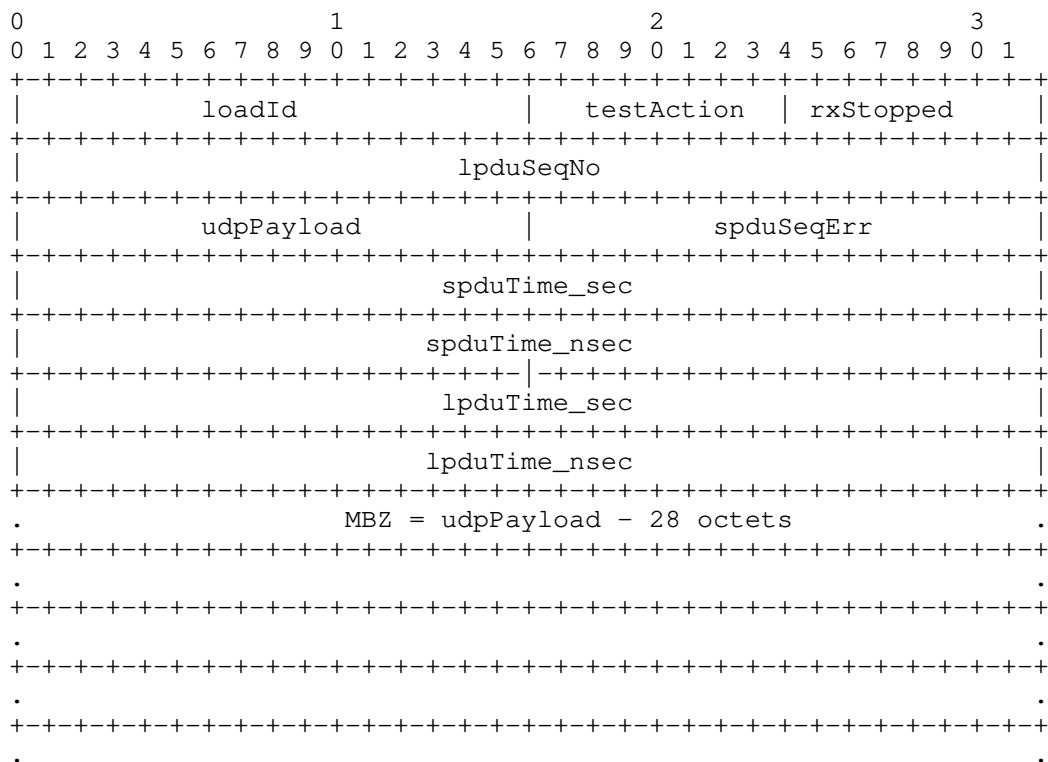
The Load PDU SHALL have the following format and field definitions:

```
uint16_t loadId; // Load ID (=0xBEEF for the LLoad PDU)
uint8_t testAction; // Test action (= 0x00 normally, until test stop)
uint8_t rxStopped; // Receive traffic stopped indicator (BOOL)
uint32_t lpduSeqNo; // Load PDU sequence number (starts at 1)
uint16_t udpPayload; // UDP payload LENGTH(bytes)
uint16_t spduSeqErr; // Status PDU sequence error count
//
uint32_t spduTime_sec; // Send time in last received status PDU
uint32_t spduTime_nsec; // Send time in last received status PDU
uint32_t lpduTime_sec; // Send time of this load PDU
uint32_t lpduTime_nsec; // Send time of this load PDU
```

Test Action Codes

```
TEST_ACT_TEST 0 // normal
TEST_ACT_STOP1 1 // normal stop at end of test: server sends in STATUS or Test P
DU
TEST_ACT_STOP2 2 // ACK of STOP1: sent by client in STATUS or Test PDU
```

The Test Load UDP PDU format is as follows (big-endian AB):



7.2. Status PDU

The receiver SHALL send a Status PDU to the sender during a test at the configured feedback interval.

The watchdog timer at the test PDU sender SHALL be reset each time a Status PDU is received. See non-graceful test stop in Section 8 for handling the watchdog/NOTRAFFIC time-out expiration at each end-point.

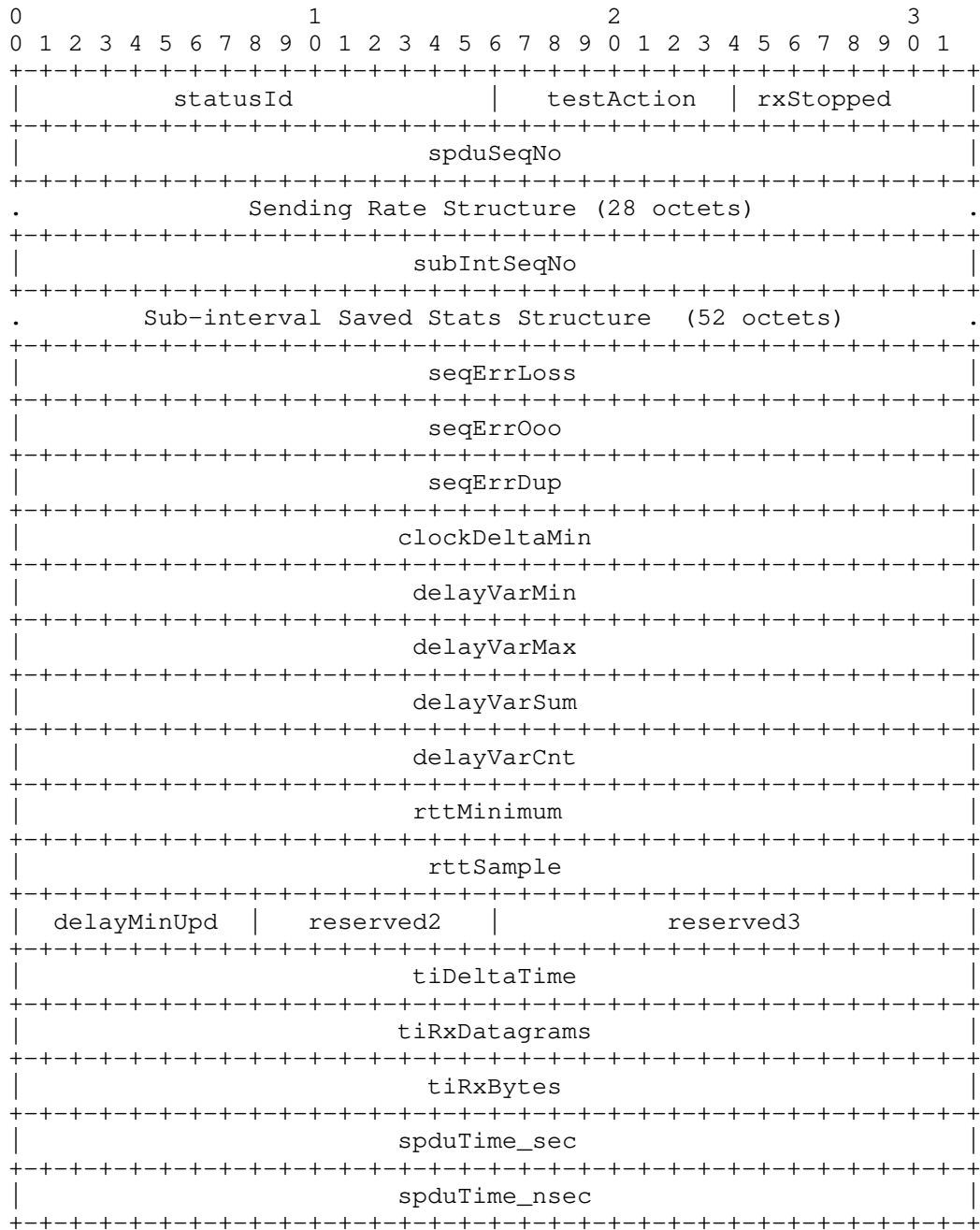
@@@ To Do: What protections from bit errors (checksum) or on-path attacks (something stronger) are warranted for teh Status PDUs? These PDUs are a key part of the server-client control loop. Added a requirement to calculate and include/check the UDP checksum.

The Status Header PDU SHALL have the following format and field definitions:

```
// Status feedback header for UDP payload of status PDUs
//
```

```
    uint16_t statusId; // Status ID = 0xFEED
    uint8_t testAction; // Test action
    uint8_t rxStopped; // Receive traffic stopped indicator (BOOL)
    uint32_t spduSeqNo; // Status PDU sequence number (starts at 1)
    //
    struct sendingRate srStruct; // Sending Rate Structure (28 octets)
    //
    uint32_t subIntSeqNo; // Sub-interval sequence number
    struct subIntStats sisSav; // Sub-interval Saved Stats Structure (52 oct
ets)
    //
    uint32_t seqErrLoss; // Loss sum
    uint32_t seqErrOoo; // Out-of-Order sum
    uint32_t seqErrDup; // Duplicate sum
    //
    uint32_t clockDeltaMin; // Clock delta minimum (either RTT or 1-way delay
)
    uint32_t delayVarMin; // Delay variation minimum
    uint32_t delayVarMax; // Delay variation maximum
    uint32_t delayVarSum; // Delay variation sum
    uint32_t delayVarCnt; // Delay variation count
    uint32_t rttMinimum; // Minimum round-trip time sampled
    uint32_t rttSample; // Last round-trip time sample
    uint8_t delayMinUpd; // Delay minimum(s) updated observed, communicate
d in both directions.
    uint8_t reserved2; // (alignment)
    uint16_t reserved3; // (alignment)
    //
    uint32_t tiDeltaTime; // Trial interval delta time
    uint32_t tiRxDatagrams; // Trial interval receive datagrams
    uint32_t tiRxBytes; // Trial interval receive bytes
    //
    uint32_t spduTime_sec; // Send time of this status PDU
    uint32_t spduTime_nsec; // Send time of this status PDU
```

The Status feedback UDP payload PDUs format is as follows (big-endian AB):



Note that the Sending Rate Structure (28 octets) is defined in the Test Activation section.

Also note that the Sub-interval Saved Stats Structure (52 octets) SHALL be included (and populated as required when the server is in the receiver role) as defined below.

The Sub-interval saved statistics structure for received traffic measurements SHALL be organized and formatted as follows:

```
uint32_t rxDatagrams; // Received datagrams
uint32_t rxBytes;     // Received bytes
uint32_t deltaTime;  // Time delta
uint32_t seqErrLoss;  // Loss sum
uint32_t seqErrOoo;   // Out-of-Order sum
uint32_t seqErrDup;   // Duplicate sum
uint32_t delayVarMin; // Delay variation minimum
uint32_t delayVarMax; // Delay variation maximum
uint32_t delayVarSum; // Delay variation sum
uint32_t delayVarCnt; // Delay variation count
uint32_t rttMinimum;  // Minimum round-trip time
uint32_t rttMaximum;  // Maximum round-trip time
uint32_t accumTime;   // Accumulated time
```

```
-----
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
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     rxDatagrams                               |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     rxBytes                                   |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     deltaTime                               |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     seqErrLoss                             |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     seqErrOoo                             |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     seqErrDup                             |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     delayVarMin                             |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     delayVarMax                             |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     delayVarSum                             |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     delayVarCnt                             |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     rttMinimum                             |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     rttMaximum                             |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     accumTime                               |
+-----+-----+-----+-----+-----+-----+-----+-----+
```

Note that the 52 octet saved statistics structure above has slight differences from the 40 octets that follow in the status feedback PDU, particularly the time-related fields.

Upon receiving the Status Feedback PDU or expiration of the feedback interval, the server SHALL perform calculations required by the Load adjustment algorithm and adjust its sending rate, or signal that the client do so in its role as as sender.

@@@ To Do: Additional measurements, like interface byte counters from a client at a residential gateway, would change the Status Feedback PDU (and the protocol version number as a result). Interface byte counters seem useful for specific circumstances, such as when the client application has acces to an interface that sees all traffic to/from a service subscriber's location.

8. Stopping the Test

When the test duration timer on the server expires, it SHALL set the connection test action to STOP and also starts marking all outgoing load or status PDUs with a test action of STOP1.

```
uint8_t testAction; // Test action (server sets STOP1)
```

This is simply a non-reversible state for all future messages sent from the server.

When the client receives a load or status PDU with the STOP1 indication, it SHALL finalize testing, display the test results, and also mark its connection with a test action of STOP (so that any PDUs received subsequent to the STOP1 are ignored).

With the test action of the client's connection set to STOP, the very next expiry of a send timer for either a load or status PDU SHALL cause the client to schedule an immediate end time to exit.

The client SHALL then send all subsequent load or status PDUs with a test action of STOP2

```
uint8_t testAction; // Test action (client sets STOP2)
```

as confirmation to the server, and a graceful termination of the test can begin.

When the server receives the STOP2 confirmation in the load or status PDU, the server SHALL schedule an immediate end time for the connection which closes the socket and deallocates it.

In a non-graceful test stop, the watchdog/NOTRAFFIC time-outs at each end-point will expire (sometimes at one end-point first), notifications in logs, STDOUT, and/or formateed output SHALL be made,

and the test action of each end-point's connection SHALL be set to STOP.

9. Method of Measurement

The architecture of the method REQUIRES two cooperating hosts operating in the roles of Src (test packet sender) and Dst (receiver), with a measured path and return path between them.

The duration of a test duration, parameter I, MUST be constrained in a production network, since this is an active test method and it will likely cause congestion on the Src to Dst host path during a test.

9.1. Running Code

This section is for the benefit of the Document Shepherd's form, and will be deleted prior to final review.

Much of the development of the method and comparisons with existing methods conducted at IETF Hackathons and elsewhere have been based on the example udpst Linux measurement tool (which is a working reference for further development) [udpst]. The current project:

- o is a utility that can function as a client or server daemon
- o requires a successful client-initiated setup handshake between cooperating hosts and allows firewalls to control inbound unsolicited UDP which either go to a control port [expected and w/ authentication] or to ephemeral ports that are only created as needed. Firewalls protecting each host can both continue to do their job normally. This aspect is similar to many other test utilities available.
- o is written in C, and built with gcc (release 9.3) and its standard run-time libraries
- o allows configuration of most of the parameters described in Sections 4 and 7.
- o supports IPv4 and IPv6 address families.
- o supports IP-layer packet marking.

10. Security Considerations

Active metrics and measurements have a long history of security considerations. The security considerations that apply to any active

measurement of live paths are relevant here. 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.

There are some new considerations for Capacity measurement as described in this memo.

1. Cooperating source and destination hosts and agreements to test the path between the hosts are REQUIRED. Hosts perform in either the Src or Dst roles.
2. It is REQUIRED to have a user client-initiated setup handshake between cooperating hosts that allows firewalls to control inbound unsolicited UDP traffic which either goes to a control port [expected and w/authentication] or to ephemeral ports that are only created as needed. Firewalls protecting each host can both continue to do their job normally.
3. Client-server authentication and integrity protection for feedback messages conveying measurements is RECOMMENDED. To accomodate different host limitations and testing circumstances, different modes of operation are recommended:

A. Unauthenticated mode (for all phases)

AND

B. OPTIONAL Authenticated set-up only
SHA-256 HMAC time-window verification (5 min time stamp verification)
(could add silent failure option)

C. Encrypted setup and test-activation
(currently using OpenSSL Library, so KISS, but may be too slow for
test packets)

----- Old/lowpower host performance impacts -----

D. Encrypted feedback messages (maybe split into Integrity and encrypt?)

E. Integrity protection for test packets SHA-256 HMAC

F. Encrypted test packets (maybe also valuable to defeat compression on links)

4. Hosts MUST limit the number of simultaneous tests to avoid resource exhaustion and inaccurate results.
5. Senders MUST be rate-limited. This can be accomplished using a pre-built table defining all the offered load rates that will be supported (Section 8.1). The recommended load-control search algorithm results in "ramp up" from the lowest rate in the table.
6. Service subscribers with limited data volumes who conduct extensive capacity testing might experience the effects of Service Provider controls on their service. Testing with the Service Provider's measurement hosts SHOULD be limited in frequency and/or overall volume of test traffic (for example, the range of I duration values SHOULD be limited).

The exact specification of these features was hopefully accomplished during this protocol development.

11. IANA Considerations

This memo requests IANA to assign a UDP port.

12. Acknowledgments

Thanks to Ruediger Geib, Lincoln Lavoie, Can Desem, and Greg Mirsky for reviewing this draft and providing helpful suggestions and areas for further development.

13. References

13.1. Normative References

- [I-D.ietf-ippm-capacity-metric-method]
Morton, A., Geib, R., and L. Ciavattone, "Metrics and Methods for One-way IP Capacity", draft-ietf-ippm-capacity-metric-method-12 (work in progress), June 2021.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC2330] Paxson, V., Almes, G., Mahdavi, J., and M. Mathis, "Framework for IP Performance Metrics", RFC 2330, DOI 10.17487/RFC2330, May 1998, <<https://www.rfc-editor.org/info/rfc2330>>.
- [RFC2681] Almes, G., Kalidindi, S., and M. Zekauskas, "A Round-trip Delay Metric for IPPM", RFC 2681, DOI 10.17487/RFC2681, September 1999, <<https://www.rfc-editor.org/info/rfc2681>>.
- [RFC6438] Carpenter, B. and S. Amante, "Using the IPv6 Flow Label for Equal Cost Multipath Routing and Link Aggregation in Tunnels", RFC 6438, DOI 10.17487/RFC6438, November 2011, <<https://www.rfc-editor.org/info/rfc6438>>.
- [RFC7497] Morton, A., "Rate Measurement Test Protocol Problem Statement and Requirements", RFC 7497, DOI 10.17487/RFC7497, April 2015, <<https://www.rfc-editor.org/info/rfc7497>>.
- [RFC7680] Almes, G., Kalidindi, S., Zekauskas, M., and A. Morton, Ed., "A One-Way Loss Metric for IP Performance Metrics (IPPM)", STD 82, RFC 7680, DOI 10.17487/RFC7680, January 2016, <<https://www.rfc-editor.org/info/rfc7680>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

- [RFC8468] Morton, A., Fabini, J., Elkins, N., Ackermann, M., and V. Hegde, "IPv4, IPv6, and IPv4-IPv6 Coexistence: Updates for the IP Performance Metrics (IPPM) Framework", RFC 8468, DOI 10.17487/RFC8468, November 2018, <<https://www.rfc-editor.org/info/rfc8468>>.

13.2. Informative References

- [copycat] Edleine, K., Kuhlewind, K., Trammell, B., and B. Donnet, "copycat: Testing Differential Treatment of New Transport Protocols in the Wild (ANRW '17)", July 2017, <<https://irtf.org/anrw/2017/anrw17-final5.pdf>>.
- [LS-SG12-A] 12, I. S., "LS - Harmonization of IP Capacity and Latency Parameters: Revision of Draft Rec. Y.1540 on IP packet transfer performance parameters and New Annex A with Lab Evaluation Plan", May 2019, <<https://datatracker.ietf.org/liaison/1632/>>.
- [LS-SG12-B] 12, I. S., "LS on harmonization of IP Capacity and Latency Parameters: Consent of Draft Rec. Y.1540 on IP packet transfer performance parameters and New Annex A with Lab & Field Evaluation Plans", March 2019, <<https://datatracker.ietf.org/liaison/1645/>>.
- [RFC2544] Bradner, S. and J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices", RFC 2544, DOI 10.17487/RFC2544, March 1999, <<https://www.rfc-editor.org/info/rfc2544>>.
- [RFC3148] Mathis, M. and M. Allman, "A Framework for Defining Empirical Bulk Transfer Capacity Metrics", RFC 3148, DOI 10.17487/RFC3148, July 2001, <<https://www.rfc-editor.org/info/rfc3148>>.
- [RFC4656] Shalunov, S., Teitelbaum, B., Karp, A., Boote, J., and M. Zekauskas, "A One-way Active Measurement Protocol (OWAMP)", RFC 4656, DOI 10.17487/RFC4656, September 2006, <<https://www.rfc-editor.org/info/rfc4656>>.
- [RFC5136] Chimento, P. and J. Ishac, "Defining Network Capacity", RFC 5136, DOI 10.17487/RFC5136, February 2008, <<https://www.rfc-editor.org/info/rfc5136>>.

- [RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, DOI 10.17487/RFC5357, October 2008, <<https://www.rfc-editor.org/info/rfc5357>>.
- [RFC6815] Bradner, S., Dubray, K., McQuaid, J., and A. Morton, "Applicability Statement for RFC 2544: Use on Production Networks Considered Harmful", RFC 6815, DOI 10.17487/RFC6815, November 2012, <<https://www.rfc-editor.org/info/rfc6815>>.
- [RFC7312] Fabini, J. and A. Morton, "Advanced Stream and Sampling Framework for IP Performance Metrics (IPPM)", RFC 7312, DOI 10.17487/RFC7312, August 2014, <<https://www.rfc-editor.org/info/rfc7312>>.
- [RFC7594] Eardley, P., Morton, A., Bagnulo, M., Burbridge, T., Aitken, P., and A. Akhter, "A Framework for Large-Scale Measurement of Broadband Performance (LMAP)", RFC 7594, DOI 10.17487/RFC7594, September 2015, <<https://www.rfc-editor.org/info/rfc7594>>.
- [RFC7799] Morton, A., "Active and Passive Metrics and Methods (with Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799, May 2016, <<https://www.rfc-editor.org/info/rfc7799>>.
- [RFC8337] Mathis, M. and A. Morton, "Model-Based Metrics for Bulk Transport Capacity", RFC 8337, DOI 10.17487/RFC8337, March 2018, <<https://www.rfc-editor.org/info/rfc8337>>.
- [RFC8762] Mirsky, G., Jun, G., Nydell, H., and R. Foote, "Simple Two-Way Active Measurement Protocol", RFC 8762, DOI 10.17487/RFC8762, March 2020, <<https://www.rfc-editor.org/info/rfc8762>>.
- [RFC8972] Mirsky, G., Min, X., Nydell, H., Foote, R., Masputra, A., and E. Ruffini, "Simple Two-Way Active Measurement Protocol Optional Extensions", RFC 8972, DOI 10.17487/RFC8972, January 2021, <<https://www.rfc-editor.org/info/rfc8972>>.
- [TR-471] Morton, A., "Broadband Forum TR-471: IP Layer Capacity Metrics and Measurement", July 2020, <<https://www.broadband-forum.org/technical/download/TR-471.pdf>>.

- [udpst] udpst Project Collaborators, "UDP Speed Test Open Broadband project", December 2020, <<https://github.com/BroadbandForum/obudpst>>.
- [Y.1540] Y.1540, I. R., "Internet protocol data communication service - IP packet transfer and availability performance parameters", December 2019, <<https://www.itu.int/rec/T-REC-Y.1540-201912-I/en>>.
- [Y.Sup60] Morton, A., "Recommendation Y.Sup60, (09/20) Interpreting ITU-T Y.1540 maximum IP-layer capacity measurements", September 2020, <<https://www.itu.int/rec/T-REC-Y.Sup60/en>>.

Authors' Addresses

Len Ciavattone
AT&T Labs
200 Laurel Avenue South
Middletown,, NJ 07748
USA

Email: lencia@att.com

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

Phone: +1 732 420 1571
Fax: +1 732 368 1192
Email: acm@research.att.com

IPPM
Internet-Draft
Intended status: Informational
Expires: 13 November 2022

H. Song
Futurewei Technologies
G. Mirsky
Ericsson
C. Filsfils
A. Abdelsalam
Cisco Systems, Inc.
T. Zhou
Z. Li
Huawei
G. Mishra
Verizon Inc.
J. Shin
SK Telecom
K. Lee
LG U+
12 May 2022

In-Situ OAM Marking-based Direct Export
draft-song-ippm-postcard-based-telemetry-12

Abstract

The document describes a packet-marking variation of the IOAM DEX option, referred to as IOAM Marking. Similar to IOAM DEX, IOAM Marking does not carry the telemetry data in user packets but send the telemetry data through a dedicated packet. Unlike IOAM DEX, IOAM Marking does not require an extra instruction header. IOAM Marking raises some unique issues that need to be considered. This document formally describes the high level scheme and cover the common requirements and issues when applying IOAM Marking in different networks. IOAM Marking is complementary to the other on-path telemetry schemes such as IOAM trace and E2E options.

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 13 November 2022.

Copyright Notice

Copyright (c) 2022 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 Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Motivation	3
2. IOAM Marking: Marking-based IOAM Direct Export	3
3. New Challenges	5
4. IOAM Marking Design Considerations	6
4.1. Packet Marking	6
4.2. Flow Path Discovery	7
4.3. Packet Identity for Export Data Correlation	7
4.4. Control the Load	8
5. Implementation Recommendation	8
5.1. Configuration	8
5.2. Postcard Format	8
5.3. Data Correlation	9
6. Use Cases	9
7. Security Considerations	10
8. IANA Considerations	10
9. Contributors	10
10. Acknowledgments	10
11. Informative References	10
Authors' Addresses	12

1. Motivation

To gain detailed data plane visibility to support effective network OAM, it is essential to be able to examine the trace of user packets along their forwarding paths. Such on-path flow data reflect the state and status of each user packet's real-time experience and provide valuable information for network monitoring, measurement, and diagnosis.

The telemetry data include but not limited to the detailed forwarding path, the timestamp/latency at each network node, and, in case of packet drop, the drop location, and the reason. The emerging programmable data plane devices allow user-defined data collection or conditional data collection based on trigger events. Such on-path flow data are from and about the live user traffic, which complements the data acquired through other passive and active OAM mechanisms such as IPFIX [RFC7011] and ICMP [RFC2925].

On-path telemetry was developed to cater to the need of collecting on-path flow data. There are two basic modes for on-path telemetry: the passport mode and the postcard mode. In the passport mode which is represented by IOAM trace option [I-D.ietf-ippm-ioam-data], each node on the path adds the telemetry data to the user packets (i.e., stamp the passport). The accumulated data-trace carried by user packets are exported at a configured end node. In the postcard mode which is represented by IOAM direct export option (DEX) [I-D.ietf-ippm-ioam-direct-export], each node directly exports the telemetry data using an independent packet (i.e., send a postcard) to avoid carrying the data with user packets. The postcard mode is complementary to the passport mode.

IOAM DEX uses an instruction header to explicitly instruct the telemetry data to be collected. This document describes another variation of the postcard mode on-path telemetry, IOAM Marking. Unlike IOAM DEX, IOAM Marking does not require a telemetry instruction header. However, IOAM Marking has unique issues that need to be considered. This document discusses the challenges and their solutions which are common to the high-level scheme of IOAM Marking.

2. IOAM Marking: Marking-based IOAM Direct Export

As the name suggests, IOAM Marking only needs a marking-bit in the existing headers of user packets to trigger the telemetry data collection and export. The sketch of IOAM Marking is as follows. If on-path data need to be collected, the user packet is marked at the path head node. At each IOAM Marking-aware node, if the mark is detected, a postcard (i.e., the dedicated OAM packet triggered by a

marked user packet) is generated and sent to a collector. The postcard contains the data requested by the management plane. The requested data are configured by the management plane. Once the collector receives all the postcards for a single user packet, it can infer the packet's forwarding path and analyze the data set. The path end node is configured to unmark the packets to its original format if necessary.

The overall architecture of IOAM Marking is depicted in Figure 1.

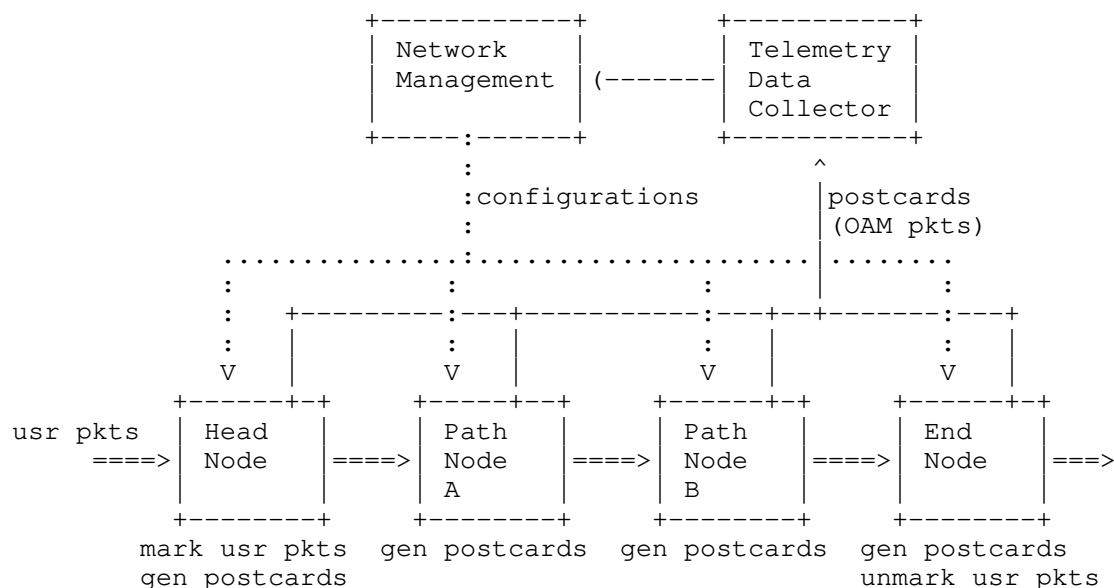


Figure 1: Architecture of IOAM Marking

The advantages of IOAM Marking are summarized as follows.

- * 1: IOAM Marking avoids augmenting user packets with new headers and the signaling for telemetry data collection remains in the data plane.
- * 2: IOAM Marking is extensible for collecting arbitrary new data to support possible future use cases. The data set to be collected can be configured through the management plane or control plane.

- * 3: IOAM Marking can avoid interfering with the normal forwarding. The collected data are free to be transported independently through in-band or out-of-band channels. The data collecting, processing, assembly, encapsulation, and transport are, therefore, decoupled from the forwarding of the corresponding user packets and can be performed in data-plane slow-path if necessary.
- * 4: For IOAM Marking, the types of data collected from each node can vary depending on application requirements and node capability.
- * 5: IOAM Marking makes it easy to secure the collected data without exposing it to unnecessary entities. For example, both the configuration and the telemetry data can be encrypted and/or authenticated before being transported, so passive eavesdropping and a man-in-the-middle attack can both be deterred.
- * 6: Even if a user packet under inspection is dropped at some node in the network, the postcards collected from the preceding nodes are still valid and can be used to diagnose the packet drop location and reason.

3. New Challenges

Although IOAM Marking has some unique features compared to the passport mode telemetry and the instruction-based IOAM DEX, it introduces a few new challenges.

- * Challenge 1 (Packet Marking): A user packet needs to be marked to trigger the path-associated data collection. Since IOAM Marking does not augment user packets with any new header fields, it needs to reserve or reuse bits from the existing header fields. This raises a similar issue as in the Alternate Marking Scheme [RFC8321]
- * Challenge 2 (Configuration): Since the packet header will not carry IOAM instructions anymore, the data plane devices need to be configured to know what data to collect. However, in general, the forwarding path of a flow packet (due to ECMP or dynamic routing) is unknown beforehand (note that there are some notable exceptions, such as segment routing). If the per-flow customized data collection is required, configuring the data set for each flow at all data plane devices might be expensive in terms of configuration load and data plane resources.
- * Challenge 3 (Data Correlation): Due to the variable transport latency, the dedicated postcard packets for a single packet may arrive at the collector out of order or be dropped in networks for

some reason. In order to infer the packet forwarding path, the collector needs some information from the postcard packets to identify the user packet affiliation and the order of path node traversal.

- * Challenge 4 (Load Overhead): Since each postcard packet has its header, the overall network bandwidth overhead of IOAM Marking can be high. A large number of postcards could add processing pressure on data collecting servers. That can be used as an attack vector for DoS.

4. IOAM Marking Design Considerations

To address the above challenges, we propose several design details of IOAM Marking.

4.1. Packet Marking

To trigger the path-associated data collection, usually, a single bit from some header field is sufficient. While no such bit is available, other packet-marking techniques are needed. We discuss several possible application scenarios.

- * IPv4. Alternate Marking (AM) [RFC8321] is an IP flow performance measurement framework that also requires a single bit for packet coloring. The difference is that AM does in-network measurement while IOAM Marking only collects and exports data at network nodes (i.e., the data analysis is done at the collector rather than in the network nodes). AM suggests to use some reserved bit of the Flag field or some unused bit of the TOS field. Actually, AM can be considered a sub-case of IOAM Marking, so that the same bit can be used for IOAM Marking. The management plane is responsible for configuring the actual operation mode.
- * SFC NSH. The OAM bit in the NSH header can be used to trigger the on-path data collection [RFC8300]. IOAM Marking does not add any other metadata to NSH.
- * MPLS. Instead of choosing a header bit, we take advantage of the synonymous flow label [I-D.bryant-mpls-synonymous-flow-labels] approach to mark the packets. A synonymous flow label indicates the on-path data should be collected and forwarded through a postcard.
- * SRv6: A flag bit in SRH can be reserved to trigger the on-path data collection [I-D.song-6man-srv6-pbt]. SRv6 OAM [I-D.ietf-6man-spring-srv6-oam] has adopted the O-bit in SRH flags as the marking bit to trigger the telemetry.

4.2. Flow Path Discovery

In case the path that a flow traverses is unknown in advance, all IOAM Marking-aware nodes should be configured to react to the marked packets by exporting some basic data, such as node ID and TTL before a data set template for that flow is configured. This way, the management plane can learn the flow path dynamically.

If the management plane wants to collect the on-path data for some flow, it configures the head node(s) with a probability or time interval for the flow packet marking. When the first marked packet is forwarded in the network, the IOAM Marking-aware nodes will export the basic data set to the collector. Hence, the flow path is identified. If other data types need to be collected, the management plane can further configure the data set's template to the target nodes on the flow's path. The IOAM Marking-aware nodes collect and export data accordingly if the packet is marked and a data set template is present.

If the flow path is changed for any reason, the new path can be quickly learned by the collector. Consequently, the management plane controller can be directed to configure the nodes on the new path. The outdated configuration can be automatically timed out or explicitly revoked by the management plane controller.

4.3. Packet Identity for Export Data Correlation

The collector needs to correlate all the postcard packets for a single user packet. Once this is done, the TTL (or the timestamp, if the network time is synchronized) can be used to infer the flow forwarding path. The key issue here is to correlate all the postcards for the same user packet.

The first possible approach includes the flow ID plus the user packet ID in the OAM packets. For example, the flow ID can be the 5-tuple IP header of the user traffic, and the user packet ID can be some unique information pertaining to a user packet (e.g., the sequence number of a TCP packet).

If the packet marking interval is large enough, the flow ID is enough to identify a user packet. As a result, it can be assumed that all the exported postcard packets for the same flow during a short time interval belong to the same user packet.

Alternatively, if the network is synchronized, then the flow ID plus the timestamp at each node can also infer the postcard affiliation. However, some errors may occur under some circumstances. For example, two consecutive user packets from the same flows are marked,

but one exported postcard from a node is lost. It is difficult for the collector to decide to which user packet the remaining postcard is related. In many cases, such a rare error has no catastrophic consequence. Therefore it is tolerable.

4.4. Control the Load

IOAM Marking should not be applied to all the packets all the time. It is better to be used in an interactive environment where the network telemetry applications dynamically decide which subset of traffic is under scrutiny. The network devices can limit the packet marking rate through sampling and metering. The postcard packets can be distributed to different servers to balance the processing load.

It is important to understand that the total amount of data exported by IOAM Marking is identical to that of IOAM trace option. The only extra overhead is the packet header of the postcards. In the case of IOAM trace option, it carries the data from each node throughout the path to the end node before exporting the aggregated data. On the other hand, IOAM Marking directly exports local data. The overall network bandwidth impact depends on the network topology and scale, and in some cases IOAM Marking could be more bandwidth efficient.

5. Implementation Recommendation

5.1. Configuration

The head node's ACL should be configured to filter out the target flows for telemetry data collection. Optionally, a flow packet sampling rate or probability could be configured to monitor a subset of the flow packets.

The telemetry data set that should be exported by postcards at each path node could be configured using the data set templates specified, for example, in IPFIX [RFC7011]. In future revisions, we will provide more details.

The IOAM Marking-aware path nodes could be configured to respond or ignore the marked packets.

5.2. Postcard Format

The postcard should use the same data export format as that used by IOAM. [I-D.spiegel-ippm-ioam-rawexport] proposes a raw format that can be interpreted by IPFIX. In future revisions, we will provide more details.

5.3. Data Correlation

Enough information should be included to help the collector to correlate and order the postcards for a single user packet. Section 4.3 provides several possible means. The application scenario and network protocol are important factors to determine the means to use. In future revisions, we will provide details for representative applications.

6. Use Cases

The MPLS Design Team has been investigating extensibility options for the MPLS data plane.

The challenge has been to continue to support existing MPLS architecture, backwards compatibility as well as not excessively increase the depth of the MPLS label stack with a variety of functional SPL labels and NAI indicators similar in concept to the MPLS Entropy label ELI, EL added to the label stack, as well as the MPLS extension headers being in Stack or post stack.

Reference Augmented Forwarding (RAF) [I-D.raszuk-mpls-raf-fwk] utilizes In Stack Data (ISD) with parity to Entropy Label stack {TL,RFI,RFV,AL} and control plane extension to distribute special network actions and forwarding behaviors.

Reference Augmented Forwarding (RAF) keeps the ISD and PSD stack depth in check by using an alternative means of carrying the IOAM data using IGP control plane extension TLV to carry the data to provide In-Situ IOAM on path telemetry using the postcard based telemetry.

The MPLS Design Team may come up with other alternatives to carry IOAM data such as the IGP extension mentioned and maybe other solutions, which will heavily rely on the the postcard based solution.

With Segment Routing SR-MPLS and SRv6 as Maximum SID Depth(MSD) as well as PMTU in SR Policy are critical issues for SR path instantiation by a controller, postcard based telemetry will become a critical solution to ensure that IOAM telemetry can be viable for operators by eliminating IOAM data from being carried in-situ in the SR-TE policy path.

This draft provides a critical optimization that fills the gaps with IOAM DEX related to packet marking triggers using existing mechanisms as well as flow path discovery mechanisms to avoid configuration of on path data plane node complexity and helps mitigate SR MSD and PMTU issues.

7. Security Considerations

Several security issues need to be considered.

- * Eavesdrop and tamper: the postcards can be encrypted and authenticated to avoid such security threats.
- * DoS attack: IOAM Marking can be limited to a single administrative domain. The mark must be removed at the egress domain edge. The node can rate-limit the extra traffic incurred by postcards.

8. IANA Considerations

No requirement for IANA is identified.

9. Contributors

We thank Alfred Morton who provided valuable suggestions and comments helping improve this draft.

10. Acknowledgments

TBD.

11. Informative References

[I-D.bryant-mpls-synonymous-flow-labels]

Bryant, S., Swallow, G., Sivabalan, S., Mirsky, G., Chen, M., and Z. Li, "RFC6374 Synonymous Flow Labels", Work in Progress, Internet-Draft, draft-bryant-mpls-synonymous-flow-labels-01, 4 July 2015, <<https://www.ietf.org/archive/id/draft-bryant-mpls-synonymous-flow-labels-01.txt>>.

[I-D.ietf-6man-spring-srv6-oam]

Ali, Z., Filsfils, C., Matsushima, S., Voyer, D., and M. Chen, "Operations, Administration, and Maintenance (OAM) in Segment Routing Networks with IPv6 Data plane (SRv6)", Work in Progress, Internet-Draft, draft-ietf-6man-spring-srv6-oam-13, 23 January 2022, <<https://www.ietf.org/archive/id/draft-ietf-6man-spring-srv6-oam-13.txt>>.

- [I-D.ietf-ippm-ioam-data]
Brockners, F., Bhandari, S., and T. Mizrahi, "Data Fields for In-situ OAM", Work in Progress, Internet-Draft, draft-ietf-ippm-ioam-data-17, 13 December 2021, <<https://www.ietf.org/archive/id/draft-ietf-ippm-ioam-data-17.txt>>.
- [I-D.ietf-ippm-ioam-direct-export]
Song, H., Gafni, B., Zhou, T., Li, Z., Brockners, F., Bhandari, S., Sivakolundu, R., and T. Mizrahi, "In-situ OAM Direct Exporting", Work in Progress, Internet-Draft, draft-ietf-ippm-ioam-direct-export-07, 13 October 2021, <<https://www.ietf.org/archive/id/draft-ietf-ippm-ioam-direct-export-07.txt>>.
- [I-D.raszuk-mpls-raf-fwk]
Raszuk, R., "Framework of MPLS Reference Augmented Forwarding", Work in Progress, Internet-Draft, draft-raszuk-mpls-raf-fwk-00, 25 April 2022, <<https://www.ietf.org/archive/id/draft-raszuk-mpls-raf-fwk-00.txt>>.
- [I-D.song-6man-srv6-pbt]
Song, H., "Support Postcard-Based Telemetry for SRv6 OAM", Work in Progress, Internet-Draft, draft-song-6man-srv6-pbt-01, 14 October 2019, <<https://www.ietf.org/archive/id/draft-song-6man-srv6-pbt-01.txt>>.
- [I-D.spiegel-ippm-ioam-rawexport]
Spiegel, M., Brockners, F., Bhandari, S., and R. Sivakolundu, "In-situ OAM raw data export with IPFIX", Work in Progress, Internet-Draft, draft-spiegel-ippm-ioam-rawexport-06, 21 February 2022, <<https://www.ietf.org/archive/id/draft-spiegel-ippm-ioam-rawexport-06.txt>>.
- [RFC2925] White, K., "Definitions of Managed Objects for Remote Ping, Traceroute, and Lookup Operations", RFC 2925, DOI 10.17487/RFC2925, September 2000, <<https://www.rfc-editor.org/info/rfc2925>>.
- [RFC7011] Claise, B., Ed., Trammell, B., Ed., and P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information", STD 77, RFC 7011, DOI 10.17487/RFC7011, September 2013, <<https://www.rfc-editor.org/info/rfc7011>>.

- [RFC8300] Quinn, P., Ed., Elzur, U., Ed., and C. Pignataro, Ed.,
"Network Service Header (NSH)", RFC 8300,
DOI 10.17487/RFC8300, January 2018,
<<https://www.rfc-editor.org/info/rfc8300>>.
- [RFC8321] Fioccola, G., Ed., Capello, A., Cociglio, M., Castaldelli,
L., Chen, M., Zheng, L., Mirsky, G., and T. Mizrahi,
"Alternate-Marking Method for Passive and Hybrid
Performance Monitoring", RFC 8321, DOI 10.17487/RFC8321,
January 2018, <<https://www.rfc-editor.org/info/rfc8321>>.

Authors' Addresses

Haoyu Song
Futurewei Technologies
2330 Central Expressway
Santa Clara, 95050,
United States of America
Email: hsong@futurewei.com

Greg Mirsky
Ericsson
Email: gregimirsky@gmail.com

Clarence Filsfils
Cisco Systems, Inc.
Belgium
Email: cfilsfil@cisco.com

Ahmed Abdelsalam
Cisco Systems, Inc.
Italy
Email: ahabdels@cisco.com

Tianran Zhou
Huawei
156 Beiqing Road
Beijing, 100095
P.R. China
Email: zhoutianran@huawei.com

Zhenbin Li
Huawei
156 Beiqing Road
Beijing, 100095
P.R. China
Email: lizhenbin@huawei.com

Gyan Mishra
Verizon Inc.
Email: hayabusagsm@gmail.com

Jongyoon Shin
SK Telecom
South Korea
Email: jongyoon.shin@sk.com

Kyungtae Lee
LG U+
South Korea
Email: coolee@lguplus.co.kr

IPPM
Internet-Draft
Intended status: Standards Track
Expires: September 1, 2022

T. Zhou, Ed.
G. Fioccola
Huawei
Y. Liu
China Mobile
M. Cociglio
Telecom Italia
S. Lee
LG U+
W. Li
Huawei
February 28, 2022

Enhanced Alternate Marking Method
draft-zhou-ippm-enhanced-alternate-marking-09

Abstract

This document extends the IPv6 Alternate Marking Option to provide enhanced capabilities and allow advanced functionalities. With this extension, it can be possible to perform thicker packet loss measurements and more dense delay measurements with no limitation for the number of concurrent flows under monitoring.

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 1, 2022.

Copyright Notice

Copyright (c) 2022 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
1.1. Requirements Language	3
2. Data Fields Format	3
3. Security Considerations	6
4. IANA Considerations	6
5. References	7
5.1. Normative References	7
5.2. Informative References	7
Authors' Addresses	8

1. Introduction

The Alternate Marking [RFC8321] and Multipoint Alternate Marking [RFC8889] define the Alternate Marking technique that is a hybrid performance measurement method, per [RFC7799] classification of measurement methods. This method is based on marking consecutive batches of packets and it can be used to measure packet loss, latency, and jitter on live traffic.

The IPv6 AltMark Option [I-D.ietf-6man-ipv6-alt-mark] applies the Alternate Marking Method to IPv6, and defines an Extension Header Option to encode the Alternate Marking Method for both the Hop-by-Hop Options Header and the Destination Options Header. Similarly, SRv6 AltMark [I-D.fz-spring-srv6-alt-mark] defines how Alternate Marking data is carried as a TLV in the Segment Routing Header.

While the IPv6 AltMark Option implements the basic alternate marking methodology, this document defines extended data fields for the AltMark Option and provides enhanced capabilities to overcome some challenges and enable future proof applications.

It is worth mentioning that the enhanced capabilities are intended for further use and are optional.

Some possible enhanced applications MAY be:

1. thicker packet loss measurements: the single marking method of the base AltMark Option can be extended with additional marking bits in order to get shortest marking periods under the same timing conditions.
2. more dense delay measurements: than double marking method of the base AltMark Option can be extended with additional marking bits in order to identify down to each packet as delay sample.
3. increase the number of concurrent flows under monitoring: if the 20-bit FlowMonID is set independently and pseudo randomly, there is a 50% chance of collision for 1206 flows. The size of FlowMonID can be extended to raise the entropy and therefore to increase the number of concurrent flows that can be monitored.

1.1. Requirements Language

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

2. Data Fields Format

The Data Fields format is represented in Figure 1. A 4-bit NH(NextHeader) field is allocated from the Reserved field of IPv6 AltMark Option [I-D.ietf-6man-ipv6-alt-mark]. It is worth highlighting that remaining bits of the former Reserved field continue to be reserved.

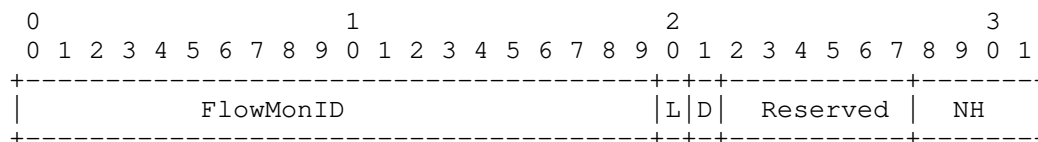


Figure 1: Data fields indicator for enhanced capabilities

The NH (NextHeader) field is used to indicate the extended data fields which are used for enhanced capabilities:

NextHeader value of 0x00 is reserved for backward compatibility. It means that there is no extended data field attached.

NextHeader values of 0x01-0x08 are reserved for private use or for experimentation.

NextHeader value of 0x09 indicates the extended data fields. The format is showed in Figure 2.

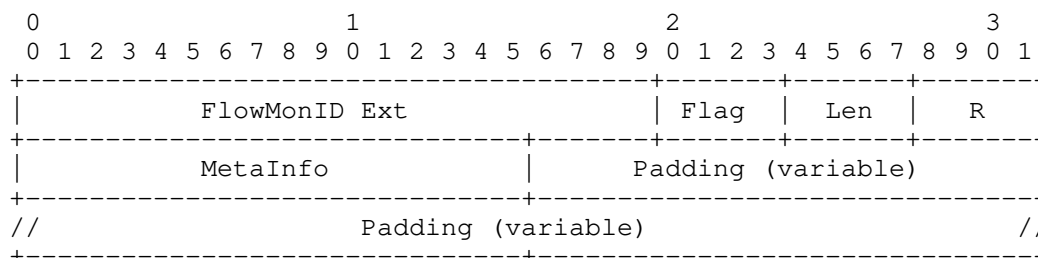


Figure 2: Data fields extension for enhanced alternate marking

where:

- o FlowMonID Ext - 20 bits unsigned integer. This is used to extend the FlowMonID in order to reduce the conflict when random allocation is applied. The disambiguation of the FlowMonID field is discussed in IPv6 AltMark Option [I-D.ietf-6man-ipv6-alt-mark].
- o Flag - A 4-bit flag to indicate the special purpose usage (see below).
- o Len - Length. It indicates the length of the enhanced alternate marking extension in bytes.
- o R - Reserved for further use. These bits MUST be set to zero on transmission and ignored on receipt.
- o MetaInfo - A 16-bit Bitmap to indicate more meta data attached for the enhanced function (see below).
- o Padding - These bits MUST be set to zero when not being used.

The Flag is defined in Figure 3 as:

- o bit 0 - Measurement mode, M bit. If M=0, it indicates that it is for hop-by-hop monitoring. If M=1, it indicates that it is for end-to-end monitoring.
- o bit 2 - Flow direction identification, F bit. This flag is used in the case backward direction flow monitoring is requested to be set up automatically. If F=1, it indicates that the flow direction is forward. If F=0, it indicates that the flow direction is backward.

- o others (shown as R) - Reserved. These bits MUST be set to zero and ignored on receipt.

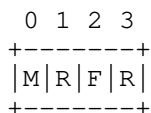


Figure 3: Flag data field

The MetaInfo is defined in the following Figure 4 as a bit map:

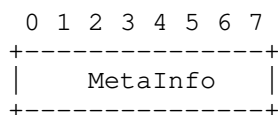


Figure 4: MetaInfo data field

- o bit 0: it indicates a 6 bytes Timestamp that is attached as Padding after the MetaInfo. Timestamp(s) stands for the number of seconds in the timestamp. It will overwrite the Padding after MetaInfo. Timestamp(ns) stands for the number of sub-seconds in the timestamp with the unit of nano second. This Timestamp is filled by the encapsulation node, and is taken all the way to the decapsulation node. So that all the intermediate nodes could compare it with its local time, and measure the one way delay.

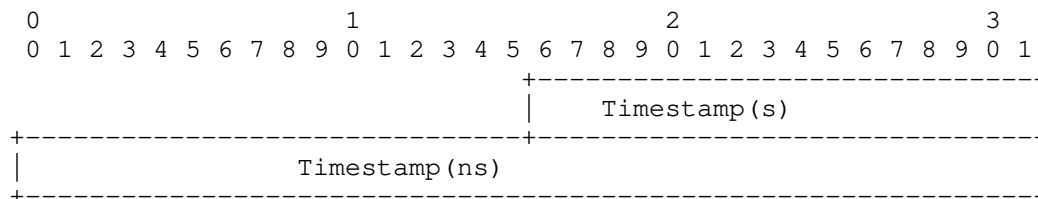


Figure 5: Timestamp data field

- o bit 1: it indicates the control information with the following data format that is attached as Padding after the MetaInfo:

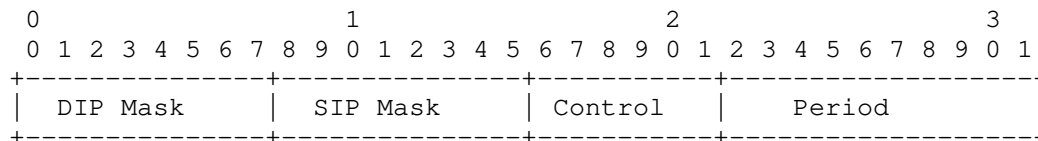


Figure 6: Control words for backward direction flow monitoring

This is used to set up the backward direction flow monitoring.
Where:

- * DIP Mask: it is the length of the destination IP prefix.
 - * SIP Mask: it is the length of the source IP prefix.
 - * Control: it indicates more match fields to set up the backward direction flow monitoring.
 - * Period: it indicates the alternate marking period with the unit of second.
- o bit 2: it indicates a 4 bytes Sequence number with the following data format that is attached as Padding after the MetaInfo. The unique Sequence could be used to detect the out-of-order packets, in addition to the normal loss measurement. More over, the Sequence can be used together with the latency measurement, so as to get the per packet timestamp.

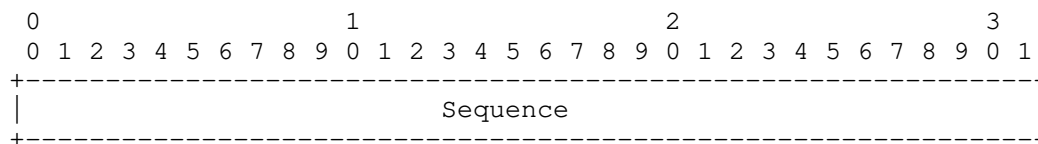


Figure 7: Sequence number data field

It is worth noting that the meta data information forming the Padding and specified above in Figure 5, Figure 6 and Figure 7 must be ordered according to the order of the MetaInfo bits.

3. Security Considerations

IPv6 AltMark Option [I-D.ietf-6man-ipv6-alt-mark] analyzes different security concerns and related solutions. These aspects are valid and applicable also to this document. In particular the fundamental security requirement is that Alternate Marking MUST only be applied in a specific limited domain, as also mentioned in [RFC8799].

4. IANA Considerations

This document has no request to IANA.

5. References

5.1. Normative References

- [I-D.fz-spring-srv6-alt-mark]
Fioccola, G., Zhou, T., and M. Cociglio, "Segment Routing Header encapsulation for Alternate Marking Method", draft-fz-spring-srv6-alt-mark-02 (work in progress), February 2022.
- [I-D.ietf-6man-ipv6-alt-mark]
Fioccola, G., Zhou, T., Cociglio, M., Qin, F., and R. Pang, "IPv6 Application of the Alternate Marking Method", draft-ietf-6man-ipv6-alt-mark-12 (work in progress), October 2021.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC7799] Morton, A., "Active and Passive Metrics and Methods (with Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799, May 2016, <<https://www.rfc-editor.org/info/rfc7799>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

5.2. Informative References

- [RFC8321] Fioccola, G., Ed., Capello, A., Cociglio, M., Castaldelli, L., Chen, M., Zheng, L., Mirsky, G., and T. Mizrahi, "Alternate-Marking Method for Passive and Hybrid Performance Monitoring", RFC 8321, DOI 10.17487/RFC8321, January 2018, <<https://www.rfc-editor.org/info/rfc8321>>.
- [RFC8799] Carpenter, B. and B. Liu, "Limited Domains and Internet Protocols", RFC 8799, DOI 10.17487/RFC8799, July 2020, <<https://www.rfc-editor.org/info/rfc8799>>.
- [RFC8889] Fioccola, G., Ed., Cociglio, M., Sapio, A., and R. Sisto, "Multipoint Alternate-Marking Method for Passive and Hybrid Performance Monitoring", RFC 8889, DOI 10.17487/RFC8889, August 2020, <<https://www.rfc-editor.org/info/rfc8889>>.

Authors' Addresses

Tianran Zhou
Huawei
156 Beiqing Rd.
Beijing 100095
China

Email: zhoutianran@huawei.com

Giuseppe Fioccola
Huawei
Riesstrasse, 25
Munich 80992
Germany

Email: giuseppe.fioccola@huawei.com

Yisong Liu
China Mobile
Beijing
China

Email: liuyisong@chinamobile.com

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

Email: mauro.cociglio@telecomitalia.it

Shinyoung Lee
LG U+
71, Magokjungang 8-ro, Gangseo-gu
Seoul
Republic of Korea

Email: leesy@lguplus.co.kr

Weidong Li
Huawei
156 Beiqing Rd.
Beijing 100095
China

Email: poly.li@huawei.com