Network Working Group
Internet Draft
Intended status: Standard
Expires: May 2, 2021

L. Dunbar
H. Song
J. Kaippallimalil
Futurewei

November 2, 2020

IP Layer Metrics for 5G Edge Computing Service
draft-dunbar-ippm-5g-edge-compute-ip-layer-metrics-01

Abstract

This draft describes the IP Layer metrics and methods to measure the Edge Computing Servers running status and environment for IP networks to select the optimal Edge Computing server location in 5G Edge Computing (EC) environment. Those measurements are for IP network to dynamically optimize the forwarding of 5G edge computing service without any knowledge above IP layer.

This Internet-Draft will expire on April 7, 2021.

Copyright Notice

Table of Contents

1. Introduction

1.1. 5G Edge Computing Background

In 5G Edge Computing environment [3GPP-EdgeComputing], one Application can have multiple Application Servers hosted in different Edge Computing data centers that are close in proximity. Those Edge Computing (mini) data centers are usually very close to, or co-located with, 5G base stations, with the goal to minimize latency and optimize the user experience.

When a UE (User Equipment) initiates application packets using the destination address from a DNS reply or from its own cache, the packets from the UE are carried in a PDU session through 5G Core [5GC] to the 5G UPF-PSA (User Plan Function - PDU Session Anchor). The UPF-PSA decapsulate the 5G GTP outer header and forwards the packets from the UEs to the Ingress router of the Edge Computing (EC) Local Data Network (LDN). The LDN for 5G EC, which is the IP Networks from 5GC perspective, is responsible for forwarding the packets to the intended destinations.

Routers in the local IP network should be able to select the "best" or "closest" application server location out of many. However, simply using distance alone as a metric may not be sufficient as there may be many locations in close proximity. Moreover, one of the main aims of locating application servers so close to the user is to provide lower latency. When a user moves and attaches from another access router (UPF), the local IP network should be able to continue routing to the established application server. As a user keeps moving further away, a closer application server maybe able to serve the user better. Network measurements, including latency of various paths are provided to the application domain to assist in reselection. These problems are outlined in sections 1.2, and 1.3.

1.2. Problem 1: Selecting 5G Edge Application Location

Having multiple locations closer to UEs to host one Application server can greatly improve the user experience. But selecting an optimal location for the application traffic from a UE may not be that simple.

Using DNS to reply with the address of the Application Server location closest to the requesting UE can encounter issues like:

- UE can cache results indefinitely, when the UE moves to a 5G cell site very far away, the cached address may still be used, which can incur large network delay.
- The Application Server at a specific location whose address replied by the DNS might be heavily loaded causing slow or no response, when there are available low utilized Application Server, for the same application, at different locations very close in proximity.
- No inherent leverage of proximity information present in the network (routing) layer, resulting in loss of performance
- Local DNS resolver become the unit of traffic management

Increasingly, Anycast is used extensively by various application providers and CDNs because ANYCAST makes it possible to dynamically load balance across locations that host the Application server based on network conditions. Application server location selection using Anycast address leverages the proximity information present in the network (routing) layer and eliminates the single point of failure and bottleneck at the DNS resolvers and application layer load balancers. Another benefit of using ANYCAST address is removing the dependency on UEs that use their cached destination IP addresses for extended period.

But selection of an ANYCAST location purely based on the network condition can encounter issue of the location selected by network routing information being overutilized while there are available underutilized locations close by.

1.3. Problem 2: UE mobility creates unbalanced anycast distribution

Another problem of using ANYCAST address for multiple locations of an Application Server in 5G environment is that UEs' frequent moving from one 5G site to another. The frequent move of UEs can make it difficult to plan where Application server should be hosted. When a large number of UEs using a particular Application congregate together unpredictably, the ANYCAST location selected based on routing distance can be heavily utilized, while the same Application Server at other locations close-by are underutilized.

```
   +--+
   |UE|---\+---------+                  +------------------+
   +--+    |  5G     |  +-----------+   | S1: aa08::4450   |
   +--+    | Site A +----+         +----+                  |
   |UE|----|        | Ra |         | R1 | S2: aa08::4460   |
   +--+    |        +----+         +----+                  |
  +---+    |         |  |           |   | S3: aa08::4470   |
  |UE1|---/+---------+  |           |   +------------------+
  +---+                 |IP Network |       L-DN1
                        |(3GPP N6)  |
     |                  |           |   +------------------+
     |                  |           |   | S1: aa08::4450   |
     |                  |          +----+                  |
     |                  |          | R3 | S2: aa08::4460   |
     v                  |          +----+                  |
                        |           |   | S3: aa08::4470   |
                        |           |   +------------------+
                        |           |      L-DN3
   +--+                 |           |
   |UE|---\+---------+  |           |   +------------------+
   +--+    |  5G     |  |           |   | S1: aa08::4450   |
   +--+    | Site B +----+         +----+                  |
   |UE|----|        | Rb |         | R2 | S2: aa08::4460   |
   +--+    |        +----+         +----+                  |
   +--+    |         |  +-----------+   | S3: aa08::4470   |
   |UE|---/+---------+                  +------------------+
   +--+                                     L-DN2
```

Figure 1: multiple ANYCAST instances in different edge DCs

This document describes the measurements at IP Layer that can reflect the application server running status and environment at the specific locations. This document also describes the method of incorporating those measurements with IP routing cost to come up with a more optimal criteria in selecting ANYCAST locations.

Note: for the ease of description, the Edge Application Server and Application Server are used interchangeably throughout this document.

2. Conventions used in this document

A-ER: Egress Router to an Application Server Instance, [A-ER] is used to describe the last router that the application instance is attached. For 5G EC environment, the A-ER can be the gateway router to the Edge Computing Data Center.

ANYCAST Instance: refer to the Application Server Gateway at a specific location which is reachable by the ANYCAST address.

Application Server: An application server is a physical or virtual server that host the software system for the application.

Application Server Location: Represent a cluster of servers at one location serving the same Application. One application may have a Layer 7 Load balancer, whose address(es) are reachable from external IP network, in front of a set of application servers. From IP network perspective, this whole group of servers are considered as the Application server at the location.

EC: Edge Computing

Edge Application Server: used interchangeably with Application Server throughout this document.

Edge Computing Hosting Environment: An environment, such as psychical or virtual machines, providing support required for Edge Application Server's execution.

NOTE: The above terminologies are the same as those used in 3GPP TR 23.758

Edge DC: Edge Data Center, which provides the Edge Hosting Environment. It might be co-located with 5G Base Station and not only host 5G core functions, but also host frequently used Edge server instances.

gNB next generation Node B

Instance: the term "Instance" if used in the context of Application Server, is referring to one location of an application server; if used in the ANYCAST context, is referring to one location of the Application server with the same ANYCAST address.

L-DN: Local Data Network

PSA: PDU Session Anchor (UPF)

RTT: Round Trip Time

RTT-ANYCAST: A list of Round trip times to a group of routers that have the ANYCAST instances directly attached.

SSC: Session and Service Continuity

UE: User Equipment

UPF: User Plane Function

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. IP-Layer Metrics Definitions for 5G EC services

3.1. IP-Layer Application ID

From network perspective, an application server has an Identifier, or IP Layer Application Server ID, which is usually an ANYCAST address that can represent multiple locations that host the application server.

3.2. IP-Layer metric for App Server Load Measurement

There are many network techniques and protocols to optimize forwarding and ensuring QoS for applications, such as DSCP/DiffServ, Traffic Engineered (TE) solutions, Segment Routing, etc. But the reality is that most application servers don't expose their internal logics to network operators. Their communications are generally encrypted. Most of them do not even respond to PING or ICMP messages initiated by routers or network gears.

The proposed IP Layer Metrics and algorithms enable the IP networks to dynamically optimize the forwarding of 5G edge computing service without any knowledge above IP layer.

In a way, the proposed IP Layer Metrics and algorithm enable the IP networks to be more aware of Application behavior without dependency on getting information from Applications themselves. Without knowledge of application internal logics, network layer or IP Layer can monitor the traffic patterns to/from the Application Server at each location to gauge the running status of the application server at the location.

First, the network needs to discover which router(s) has the application server attached. Those routers are called Application Server Egress Router, or A-ER for short. A-ER is usually the Gateway Router to an Edge Computing Data Center. To discover if a (Gateway) router is the A-ER for a specific Edge Application Server, the (Gateway) router can periodically send reverse ARP (IPv4) or Neighbor Discovery scan with the address of the Application Server to discover if the Application Servers are hosted in its edge computing data center. If yes, the router or routers are identified as the A-ER for the Application Server. For one Application Server, there can be many A-ERs at different EC Data Centers.

For an Application Server at a specific location, which is identified by the address of the application server at the IP layer, the A-ER can measure the amount of traffic destined towards the address & the amount of the traffic from the specific address, such as:

- Total number of packets to the attached App Server (ToPackets);
- Total number of packets from the attached App Server (FromPackets);
- Total number of Bytes to the attached App Server (ToBytes);
- Total number of bytes from the attached App Server (FromBytes);

The actual load measurement to the App Server attached to an A-ER can be based on one of the metrics above or including all four metrics with different weights applied to each, such as:

LoadIndex =
w1*ToPackets+w2*FromPackes+w3*ToBytes+w4*FromBytes

Where 0<= wi <=1 and w1+ w2+ w3+ w4 = 1.

The weights of each metric contributing to the load index of the App Server attached to an A-ER can be configured or learned by self-adjusting based on user feedbacks.

The raw measurement is useful when the A-ER routers cannot be configured with a consistent algorithm to compute the aggregated load index and the raw measurements are needed by a central analytic system.

The A-ER can advertise either the aggregated Load Index or the raw measurements periodically, by BGP UPDATE messages or OSPF/ISIS Link statement Advertisements, to a group of routers that have traffic destined towards the ANYCAST addresses of those application servers.

Even though it would be better to have applications or their controllers directly reporting their own workload running status to network, it is not easy to have third party applications to provide information to network operators in addition to that applications servers can be running anywhere and.

The IP-Layer Load Measurements provides an intelligent estimate of the application server running status at a specific location without requiring cooperation from third party Applications or Application controllers.

3.3. Capacity Index in the overall cost

Given that different Edge Computing DCs may have different capacity, the following metrics need to be included to gauge an application Server's running status at a specific location:

- Capacity Index:
  Capacity Index is used to differentiate the running environment of the application server. Some data centers can have hundreds, or thousands, of servers behind an Application Server's App Layer Load Balancer that is reachable from external world. Other data centers can have very small number of servers for the application server. "Capacity Index", which is a numeric number, is used to represent the capacity of the application server in a specific location.

  "Capacity Index" can be a configured value indicating the capacity of a specific Application Server at a specific location, e.g. an Edge Computing DC. The Capacity Index is Application Server specific, meaning at one location, one Application Server can have the Capacity Index to be 10 and another server can be 2.

  If the Application Server capacity configuration is not available, a network analytics tool can use the historic measurements as the basis to estimate the site capacity. If an Application Server at a specific site has high volume for extended period historically, the site capacity can be considered as higher than the other site with historic low volume. This is under the assumption that application controllers monitor utilization of the application servers at different locations. If an application server has prolonged over-utilized servers at some locations, the application controller will trigger manual intervention to increase the computing powers at those locations. However, the manual intervention cycles can be in weeks/months. That is why the IP layer metrics and algorithms that can change flows direction in minutes become very essential.

3.4. Site Preference Index in the overall cost

As described in [IPv6-StickyService] and [ISPF-EXT-EC], an EC sticky service needs to connect a UE to the application server that has been serving the UE before the UE moves to a new 5G Site, unless there is failure to that location.

To achieve the goal of sticking a flow from one specific UE to a specific site, a "site Preference Index" is created. The value of the Site Preference Index can be manipulated for packets of some flows to be steered towards an application server location farther away in routing distance. The "Site Preference Index" enables some sites to be more preferred for handling the UE traffic to a application server than others.

3.5. RTT to an ANYCAST Address in 5G EC

ANYCAST used in 5G Edge computing environment is slightly different from the typical ANYCAST address being deployed. Typical ANYCAST address is used to represent instances in vast different geographical locations, such as different continents. ANCAST address for "app.net" for Asia lead packets to a server instance of "app.net" hosted in Asia. Therefore, the RTT for "app.net" in Asia, is a single value that represent the round time trip to the server in Asia that host the "app.net".

5G Edge Computing environment can have one Application server hosted in multiple Edge Computing DCs close in proximity. Routers, i.e. the ingress router to 5G LDN (Local Data Network), can forward packets for the ANYCAST address of "app.net" to different egress routers that have "app.net" servers attached.

If "app.net" is hosted in four different 5G Edge Computing Data Centers. All those DCs have the same ANYCAST address for the "app.net". The RTT to "app.net" ANYCAST address need to be a group of values (instead of one RTT value to a unicast address). The RTT group value should include the A-ER router’s specific unicast address (e.g. the loopback address) to which the Application Server is attached.

RTT to "app.net" ANYCAST Address is represented as:

    List of {Egress Router address, RTT value}

This list is called "RTT-ANYCAST".

In order to better optimize the ANYCAST traffic, each router adjacent to 5G PSA needs to periodically measure RTT to a list of A-ER routers that advertise the ANYCAST address. The RTT to egress router at Site-i is considered as the RTT to the ANYCAST instance at the Site-i.

4. Algorithm in Selecting the optimal Target Location

The goal of the algorithm is to equalize the traffic among multiple locations of the same ANYCAST address.

The main benefit of using ANYCAST is to leverage the IP-layer information to equalize the traffic among multiple locations of the same Application Server, usually identified by one or a group of ANYCAST addresses.

For 5G Edge Computing environment, the ingress router to each LDN needs to be notified of the Load Index and Capacity Index of the Application Servers at different EC site to make the intelligent decision on where to forward the traffic from UEs for an application.

The Algorithm needs to take the following attributes into consideration:

- Load Measurement Index [Section 3.2],
- capacity index [Section 3.3],
- Preference Index [Section 3.4], and
- network delay [Section 3.5].

Here is an algorithm for a router, e.g. the router directly attached to the 5G PSA, to compare the cost to reach the Application Server between the Site-i or Site-j:

$$\text{Cost-i}=\min\left(w *\left(\frac{\text{Load-i} * \text{CP-j}}{\text{Load-j} * \text{CP-i}}\right) + (1-w) *\left(\frac{\text{Pref-j} * \text{Delay-i}}{\text{Pref-i} * \text{Delay-j}}\right)\right)$$

Load-i: Load Index at Site-i, it is the weighted combination of the total packets and bytes sent to and received from the Application Server at Site-i during a fixed time period.

CP-i (Capacity-i) (lower value means higher capacity): capacity index at the site i.

Delay-i: Network latency measurement (RTT) to the A-ER that has the Application Server attached at the site-i.

Pref-i (Preference Index: lower value means higher preference): Network Preference index for the site-I.

w: Weight for load and site information, which is a value between 0 and 1. If smaller than 0.5, Network latency and the site Preference have more influence; otherwise, Server load and its capacity have more influence.

5. Scope of IP Layer Metrics Advertisement

Each Application Server might be used by a small group of UEs. Therefore, it is not necessary for A-ER router to advertise the IP layer metrics to all other routers in the 5G LDN. Likewise, each EC Data Center may only host a small number of application servers.

"Application Bound Group Routers" is used to refer a group of routers that are interested in a group of specific ANYCAST addresses. The IP Layer Metrics for an Application Server should be advertised among the routers in the "Application bound Group Routers".

BGP RT Constrained Distribution [RFC4684] can be used to form the "Application Bound Group Routers".

Since there are much more Application Servers than the number of routers in 5G LDN, a more practical way to form the "Application Bound Group of Routers" is for each ingress router to query a network controller upon receiving the first packet to a specific ANYCAST address to be included in the "Application Bound Group Routers". There should be a timer associated with Ingress router, as the UE that uses the Application Server might move away. Upon timer expires, the Ingress Router is removed from the "Application Bound Group of Routers".

6. Manageability Considerations

To be added.

7. Security Considerations

To be added.

8. IANA Considerations

To be added.

9. References

9.1. Normative References

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

[RFC4364] E. rosen, Y. Rekhter, "BGP/MPLS IP Virtual Private networks (VPNs)", Feb 2006.

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

[RFC8200] s. Deering R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", July 2017

9.2. Informative References

[3GPP-EdgeComputing] 3GPP TR 23.748, "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Study on enhancement of support for Edge Computing in 5G Core network (5GC)", Release 17 work in progress, Aug 2020.

[RFC5521] P. Mohapatra, E. Rosen, "The BGP Encapsulation Subsequent Address Family Identifier (SAFI) and the BGP Tunnel Encapsulation Attribute", April 2009.

[BGP-SDWAN-Port] L. Dunbar, H. Wang, W. Hao, "BGP Extension for SDWAN Overlay Networks", draft-dunbar-idr-bgp-sdwan-overlay-ext-03, work-in-progress, Nov 2018.

[SDWAN-EDGE-Discovery] L. Dunbar, S. Hares, R. Raszuk, K. Majumdar, "BGP UPDATE for SDWAN Edge Discovery", draft-dunbar-idr-sdwan-edge-discovery-00, work-in-progress, July 2020.

[Tunnel-Encap] E. Rosen, et al "The BGP Tunnel Encapsulation Attribute", draft-ietf-idr-tunnel-encaps-10, Aug 2018.

10. Acknowledgments

Acknowledgements to Donald Eastlake for their review and contributions.

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

Authors' Addresses

Linda Dunbar
Futurewei
Email: ldunbar@futurewei.com

HaoYu Song
Futurewei
Email: haoyu.song@futurewei.com

John Kaippallimalil
Futurewei
Email: john.kaippallimalil@futurewei.com

ippm B. Gafni
Internet-Draft Nvidia
Intended status: Standards Track H. Liu
Expires: May 6, 2021 R. Miao
Alibaba Group
M. Spiegel
Barefoot Networks, an Intel company
November 02, 2020

Additional data fields for IOAM Trace Option Types
draft-gafni-ippm-ioam-additional-data-fields-00

Abstract

In-situ Operations, Administration, and Maintenance (IOAM) records operational and telemetry information in the packet while the packet traverses a path between two points in the network. This document discusses additional data fields and associated data types to be added to the IOAM data fileds described in [I-D.ietf-ippm-ioam-data]. In-situ OAM data fields can be encapsulated into a variety of protocols such as NSH, Segment Routing, Geneve, IPv6 (via extension header), or IPv4.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

In-situ Operations, Administration, and Maintenance (IOAM) records operational and telemetry information in the packet while the packet traverses a path between two points in the network. This document is adding additional data fields that can be reported by the network as part of IOAM.

2. Conventions

2.1. Requirements Language

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

2.2. Abbreviations

Abbreviations used in this document:

IOAM: In-situ Operations, Administration, and Maintenance

3. Additional Data Fields

This draft extends [I-D.ietf-ippm-ioam-data] with additional data fields. The additional suggested data fields are:

- o Transmitted Bytes from an interface
- o Speed of an interface
- o Interface errors

The addition of these new data fields is intended to help network operators to better manage their networks, where more data is requried with regards to the activity and quality of the network ports. For example, one framework that may take advantage of these new data fileds is HPCC, which is proposed at [I-D.pan-tsvwg-hpccplus]. This section discusses the needed ammendments to the IOAM Trace header and the format of the added data fields themselves.

3.1. IOAM Trace Option-Types Ammendments

IOAM Trace Option-Types and their headers are defined in section 4.4 of [I-D.ietf-ippm-ioam-data]. As shown in section 4.4.1, the trace option header includes an IOAM-Trace-Type which is a "A 24-bit identifier which specifies which data types are used in this node data list". In order to extend [I-D.ietf-ippm-ioam-data] it is required to allocate respective bits specifying the additional data fields to be added to the packet. This draft is asking for the allocation of additional 2 bits:

Bit 12 When set indicates presence of Transmitted Bytes from an interface.

Bit 13 When set indicates presence of Speed of an interface and Interface errors.

Section 3.2 describes the new suggested data types and their formats.

3.2. The Additional IOAM Node Data Fields and Associated Formats

The Data fields and associated data types for each of the additional IOAM Data Fields are shown below:

Transmitted Bytes from an interface: 4-octet field defined as follows:

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

tx_bytes: 4-octet unsigned integer field. This field indicates how many bytes have been transmitted from the egress interface the packet is going out from. Note that this field may wrap around. As an example, for a 100Gbps port this field may wrap around within less than 3 seconds. This field is usable to determine the amount of data going through the path a flow is going through. Following multiple packets traversing the same interface, together with a timestamp, allows a network operator to gauge the amount of traffic going through the interface in total and relative to the flow it tracks. This data in turn may help to better control the traffic and take decisions related to the performance of the flow and the network.

Speed of an interface and Total errors of an interface: 4-octet field defined as follows:

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| interface_|                  interface_errors                 |
|   speed   |                                                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

interface_speed: 6 bits unsigned integer field. This field indicates the current operational speed of the interface. The procedure to allocate, manage and map the interface_speed values into the actual speed is beyond the scope of this document. This field is usable to detect whether a packet or a flow is going through a path which has enough capacity compared to the expectation of the operator. Changes in the speed of the connectivty may require changing routing decisions or troubleshooting the links under consideration. When an operator intends to take a decision about the amount of data to transmit per flow, this data is helpful as well to track.

interface_errors: 26 bits unsigned integer field. This field inciates how many errors, such as packet drops due to CRC errors, have been detected on the interface used to deliver the packet. This data is helpful in order to understand the risk associated with the packet, or the flow it belongs to, as it shows the quality of the interfaces it uses as part of its path in the network. It can also point out potential issues that other packets from the same flow might have experienced.

4. Security Considerations

TBD

5. IANA Considerations

TBD

6. References

6.1. 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>.

6.2. Informative 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-10 (work in progress), July 2020.

[I-D.pan-tsvwg-hpccplus]
Miao, R., Liu, H., Pan, R., Lee, J., Kim, C., Gafni, B., and Y. Shpigelman, "HPCC++: Enhanced High Precision Congestion Control", draft-pan-tsvwg-hpccplus-02 (work in progress), September 2020.

Authors' Addresses

Barak Gafni
Nvidia
350 Oakmead Parkway, Suite 100
Sunnyvale, CA 94085
U.S.A.

Email: gbarak@nvidia.com

Hongqiang H. Liu
Alibaba Group
108th Ave NE, Suite 800
Bellevue, WA 98004
U.S.A.

Email: hongqiang.liu@alibaba-inc.com

Rui Miao
Alibaba Group
525 Almanor Ave, 4th Floor
Sunnyvale, CA 94085
USA

Email: miao.rui@alibaba-inc.com

Mickey Spiegel
Barefoot Networks, an Intel
company
4750 Patrick Henry Drive
Santa Clara, CA 95054
US

Email: mickey.spiegel@intel.com

IPPM Working Group R. Gandhi, Ed.
Internet-Draft C. Filsfils
Intended status: Standards Track Cisco Systems, Inc.
Expires: April 23, 2021 D. Voyer
Bell Canada
M. Chen
Huawei
B. Janssens
Colt
October 20, 2020

Simple TWAMP (STAMP) Extensions for Segment Routing Networks
draft-gandhi-ippm-stamp-srpm-00

Abstract

Segment Routing (SR) leverages the source routing paradigm. SR is applicable to both Multiprotocol Label Switching (SR-MPLS) and IPv6 (SRv6) data planes. This document specifies RFC 8762 (Simple Two-Way Active Measurement Protocol (STAMP)) extensions for Delay and Loss Measurement in Segment Routing networks, for both SR-MPLS and SRv6 data planes.

Table of Contents

1. Introduction

Segment Routing (SR) leverages the source routing paradigm and greatly simplifies network operations for Software Defined Networks (SDNs). SR is applicable to both Multiprotocol Label Switching (SR-MPLS) and IPv6 (SRv6) data planes. Built-in SR Performance Measurement (PM) is one of the essential requirements 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 using probe messages [RFC8762]. It eliminates the need for control-channel signaling by using configuration data model to provision a test-channel (e.g. UDP paths). [I-D.ietf-ippm-stamp-option-tlv] defines TLV extensions for STAMP messages.

The STAMP message with a TLV for "direct measurement" can be used for combined Delay + Loss measurement [I-D.ietf-ippm-stamp-option-tlv]. However, in order to use only for loss measurement purpose, it requires the node to support the delay measurement messages and support timestamp for these messages (which may also require clock synchronization). Furthermore, for hardware-based counter collection for direct-mode loss measurement, the optional TLV based processing adds unnecessary overhead (as counters are not at well-known locations).

This document specifies RFC 8762 (Simple Two-Way Active Measurement Protocol (STAMP)) extensions for Delay and Loss Measurement in Segment Routing networks, for both SR-MPLS and SRv6 data planes.

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

BSID: Binding Segment ID.

DM: Delay Measurement.

HMAC: Hashed Message Authentication Code.

LM: Loss Measurement.

MPLS: Multiprotocol Label Switching.

NTP: Network Time Protocol.

OWAMP: One-Way Active Measurement Protocol.

PM: Performance Measurement.

PTP: Precision Time Protocol.

SID: Segment ID.

SL: Segment List.

SR: Segment Routing.

SRH: Segment Routing Header.

SR-MPLS: Segment Routing with MPLS data plane.

SRv6: Segment Routing with IPv6 data plane.

SSID: STAMP Session Identifier.

STAMP: Simple Two-way Active Measurement Protocol.

2.3. Reference Topology

In the reference topology shown below, the sender node R1 initiates a performance measurement probe query message and the reflector node R5 sends a probe response message for the query message received. The probe response message is typically sent to the sender node R1.

```
                          t1                t2
                         /                   \
                +-------+       Query         +-------+
                |       | - - - - - - - - - ->|       |
                |   R1  |=====================|   R5  |
                |       |<- - - - - - - - - - |       |
                +-------+       Response      +-------+
                         \                   /
                          t4                t3
                 Sender                        Reflector

                         Reference Topology
```

3. Probe Query Message

3.1. Control Code Field Extension for STAMP Messages

In this document, the Control Code field is defined for delay and loss measurement probe query messages for STAMP protocol in unauthenticated and authenticated modes. The modified delay measurement probe query message format is shown in Figure 1. This message format is backwards compatible with the message format defined in STAMP [RFC8762] as its reflector MUST ignore the received field (previously identified as MBZ). With this field, the reflector node does not require any additional state for PM (recall that in SR networks, the state is in the probe packet and signaling of the parameters is undesired). The usage of the Control Code is not limited to the SR and can be used for non-SR network.

```
    .                                                               .
    .                                                               .
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                          Timestamp                            |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |         Error Estimate        | SSID                          |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |         MBZ                                   |Se Control Code|
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    .                                                               .
    .                                                               .
```

Figure 1: Sender Control Code in STAMP DM Message

Sender Control Code: Set as follows in STAMP probe query message.

In a Query:

0x0: Out-of-band Response Requested. Indicates that the probe response is not required over the same path in the reverse direction. This is also the default behavior.

0x1: In-band Response Requested. Indicates that this query has been sent over a bidirectional path and the probe response is required over the same path in the reverse direction.

0x2: No Response Requested.

3.2. Loss Measurement Query Message Extensions

In this document, STAMP probe query messages for loss measurement are defined as shown in Figure 2 and Figure 3. The message formats are hardware efficient due to well-known locations of the counters and payload small in size. They are stand-alone and similar to the delay measurement message formats (e.g. location of the Counter and Timestamp). They also do not require backwards compatibility and support for the existing DM message formats from [RFC8762] as different user-configured destination UDP port is used for loss measurement.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sequence Number                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Transmit Counter                       |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|X|B| Reserved  | Block Number  | SSID                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         MBZ                                   |Se Control Code|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                        MBZ (24 octets)                        |
|                                                               |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: STAMP LM Probe Query Message - Unauthenticated Mode

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sequence Number                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        MBZ (12 octets)                        |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Transmit Counter                       |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|X|B| Reserved  | Block Number  | SSID                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         MBZ                                   |Se Control Code|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        MBZ (64 octets)                        |
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                        HMAC (16 octets)                       |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: STAMP LM Probe Query Message - Authenticated Mode

Sequence Number (32-bit): As defined in [RFC8762].

Transmit Counter (64-bit): The number of packets or octets sent by the sender node in the query message and by the reflector node in the response message. The counter is always written at the well-known location in the probe query and response messages.

Receive Counter (64-bit): The number of packets or octets received at the reflector node. It is written by the reflector node in the probe response message.

Sender Counter (64-bit): This is the exact copy of the transmit counter from the received query message. It is written by the reflector node in the probe response message.

Sender Sequence Number (32-bit): As defined in [RFC8762].

Sender TTL: As defined in [RFC8762].

LM Flags: 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 an LM query and copied from an LM query to an LM response message. Set to 0 when the LM message 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 1-4 fields represent octet counts. The octet count applies to all packets within the LM scope, and the octet count of a packet sent 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.

Block Number (8-bit): The Loss Measurement using Alternate-Marking method defined in [RFC8321] requires to color the data traffic. To be able to correlate the transmit and receive traffic counters of the matching color, the Block Number (or color) of the traffic counters is carried by the probe query and response messages for loss measurement. The Block Number can also be used to aggregate performance metrics collected.

HMAC: The probe message in authenticated mode includes a key Hashed Message Authentication Code (HMAC) [RFC2104] hash. Each probe query and response messages are authenticated by adding Sequence Number with Hashed Message Authentication Code (HMAC) TLV. It can use HMAC-SHA-256 truncated to 128 bits (similarly to the use of it in IPSec defined in [RFC4868]); hence the length of the HMAC field is 16 octets.

HMAC uses its own key and the mechanism to distribute the HMAC key is outside the scope of this document.

In authenticated mode, only the sequence number is encrypted, and the other payload fields are sent in clear text. The probe message MAY include Comp.MBZ (Must Be Zero) variable length field to align the packet on 16 octets boundary.

4. Probe Response Message

4.1. Loss Measurement Response Message Extensions

In this document, STAMP probe response message formats are defined for loss measurement as shown in Figure 4 and Figure 5.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sequence Number                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Transmit Counter                       |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|X|B| Reserved  | Block Number  | SSID                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Receive Counter                        |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sender Sequence Number                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sender Counter                         |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|X|B| Reserved  |Sender Block Nu|   MBZ                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Sender TTL   |      MBZ                                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4: STAMP LM Probe Response Message - Unauthenticated Mode

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sequence Number                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        MBZ (12 octets)                        |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Transmit Counter                       |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|X|B| Reserved  | Block Number  | SSID                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        MBZ (4 octets)                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Receive Counter                        |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        MBZ (8 octets)                         |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sender Sequence Number                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        MBZ (12 octets)                        |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sender Counter                         |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|X|B| Reserved  |Sender Block Nu|   MBZ                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        MBZ (4 octets)                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Sender TTL   |                                               |
+-+-+-+-+-+-+-+-+                                               |
|                        MBZ (15 octets)                        |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                        HMAC (16 octets)                       |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 5: STAMP LM Probe Response Message - Authenticated Mode

5. Node Address TLV Extensions

In this document, Node Address TLV is defined for STAMP message [I-D.ietf-ippm-stamp-option-tlv] and has the following format shown in Figure 6:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|STAMP TLV Flags|     Type      |         Length                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Reserved                      | Address Family                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~                            Address                            ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 6: Node Address TLV Format

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 STAMP TLV Flags are set using the procedures described in [I-D.ietf-ippm-stamp-option-tlv].

The following Type is defined and it contains Node Address TLV:

Destination Node Address (value TBA1):

The Destination Node Address TLV is optional. The Destination Node Address TLV indicates the address of the intended recipient node of the probe message. The reflector node MUST NOT send response message if it is not the intended destination node of the probe query message.

6. Return Path TLV Extensions

For two-way performance measurement, the reflector node needs to send the probe response message on a specific reverse path. The sender node can request in the probe query message to the reflector node to send a response message back on a given reverse path (e.g. co-routed bidirectional path). This way the reflector node does not require any additional state for PM (recall that in SR networks, the state is in the probe packet and signaling of the parameters is undesired).

For one-way performance measurement, the sender node address may not be reachable via IP route from the reflector node. The sender node in this case needs to send its reachability path information to the reflector node.

[I-D.ietf-ippm-stamp-option-tlv] defines STAMP probe query messages that can include one or more optional TLVs. The TLV Type (value TBA2) is defined in this document for Return Path that carries reverse path for STAMP probe response messages (in the payload of the message). The format of the Return Path TLV is shown in Figure 7:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|STAMP TLV Flags|   Type=TBA2   |         Length                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Return Path Sub-TLVs                       |
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 7: Return Path TLV

The STAMP TLV Flags are set using the procedures described in [I-D.ietf-ippm-stamp-option-tlv].

The following Type defined for the Return Path TLV contains the Node Address sub-TLV using the format shown above in Figure 7:

o Type (value 0): Return Address. Target node address of the response message different than the Source Address in the query

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|STAMP TLV Flags|     Type      |         Length                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Segment(1)                                 |
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.                                                               .
.                                                               .

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Segment(n)                                 |
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 8: Segment List Sub-TLV in Return Path TLV

The Segment List Sub-TLV (shown above in Figure 8) in the Return Path TLV can be one of the following Types:

- o Type (value 1): SR-MPLS Label Stack of the Reverse Path

- o Type (value 2): SR-MPLS Binding SID [I-D.ietf-pce-binding-label-sid] of the Reverse SR Policy

- o Type (value 3): SRv6 Segment List of the Reverse Path

- o Type (value 4): SRv6 Binding SID [I-D.ietf-pce-binding-label-sid] of the Reverse SR Policy

The Return Path TLV is optional. The sender node MUST only insert one Return Path TLV in the probe query message and the reflector node MUST only process the first Return Path TLV in the probe query message and ignore other Return Path TLVs if present. The reflector node MUST send probe response message back on the reverse path specified in the Return Path TLV and MUST NOT add Return Path TLV in the probe response message.

7. Security Considerations

The performance measurement is intended for deployment in well-managed private and service provider networks. As such, it assumes that a node involved in a measurement operation has previously verified the integrity of the path and the identity of the far-end reflector node.

If desired, attacks can be mitigated by performing basic validation and sanity checks, at the sender, of the counter or timestamp fields in received measurement response messages. The minimal state associated with these protocols also limits the extent of measurement disruption that can be caused by a corrupt or invalid message to a single query/response cycle.

Use of HMAC-SHA-256 in the authenticated mode protects the data integrity of the probe messages. Cryptographic measures may be enhanced by the correct configuration of access-control lists and firewalls.

8. IANA Considerations

IANA will create a "STAMP TLV Type" registry for [I-D.ietf-ippm-stamp-option-tlv]. IANA is requested to allocate a value for the following Destination Address TLV Type from the IETF Review TLV range of this registry. This TLV is to be carried in the probe messages.

o Type TBA1: Destination Node Address TLV

IANA is also requested to allocate a value for the following Return Path TLV Type from the IETF Review TLV range of the same registry. This TLV is to be carried in the probe query messages.

o Type TBA2: Return Path TLV

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 1:

Value	Description	Reference
0	Reserved	This document
1 - 175	Unassigned	This document
176 - 239	Unassigned	This document
240 - 251	Experimental	This document
252 - 254	Private Use	This document
255	Reserved	This document

Table 1: Return Path Sub-TLV Type Registry

IANA is requested to allocate the values for the following Sub-TLV Types from this registry.

o Type (value 1): Return Address

o Type (value 2): SR-MPLS Label Stack of the Reverse Path

o Type (value 3): SR-MPLS Binding SID [I-D.ietf-pce-binding-label-sid] of the Reverse SR Policy

o Type (value 4): SRv6 Segment List of the Reverse Path

o Type (value 5): SRv6 Binding SID [I-D.ietf-pce-binding-label-sid] of the Reverse SR Policy

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>.

[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>.

[I-D.ietf-ippm-stamp-option-tlv] Mirsky, G., Min, X., Nydell, H., Foote, R., Masputra, A., and E. Ruffini, "Simple Two-way Active Measurement Protocol Optional Extensions", draft-ietf-ippm-stamp-option-tlv-09 (work in progress), August 2020.

9.2. Informative References

[RFC2104] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, DOI 10.17487/RFC2104, February 1997, <https://www.rfc-editor.org/info/rfc2104>.

[RFC4868] Kelly, S. and S. Frankel, "Using HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512 with IPsec", RFC 4868, DOI 10.17487/RFC4868, May 2007, <https://www.rfc-editor.org/info/rfc4868>.

[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>.

[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>.

[I-D.ietf-pce-binding-label-sid]
Filsfils, C., Sivabalan, S., Tantsura, J., Hardwick, J., Previdi, S., and C. Li, "Carrying Binding Label/Segment-ID in PCE-based Networks.", draft-ietf-pce-binding-label-sid-03 (work in progress), June 2020.

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 for reviewing this document and providing useful comments and suggestions. 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

IPPM Working Group
Internet-Draft
Intended status: Informational
Expires: April 23, 2021

R. Gandhi, Ed.
C. Filsfils
Cisco Systems, Inc.
D. Voyer
Bell Canada
M. Chen
Huawei
B. Janssens
Colt
October 20, 2020

TWAMP Light Extensions for Segment Routing Networks
draft-gandhi-ippm-twamp-srpm-00

Abstract

Segment Routing (SR) leverages the source routing paradigm. SR is applicable to both Multiprotocol Label Switching (SR-MPLS) and IPv6 (SRv6) data planes. This document describes RFC 5357 (Two-Way Active Measurement Protocol (TWAMP) Light) extensions for Delay and Loss Measurement in Segment Routing networks, for both SR-MPLS and SRv6 data planes.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Segment Routing (SR) leverages the source routing paradigm and greatly simplifies network operations for Software Defined Networks (SDNs). SR is applicable to both Multiprotocol Label Switching (SR-MPLS) and IPv6 (SRv6) data planes. Built-in SR Performance Measurement (PM) is one of the essential requirements to provide Service Level Agreements (SLAs).

The One-Way Active Measurement Protocol (OWAMP) defined in [RFC4656] and Two-Way Active Measurement Protocol (TWAMP) defined in [RFC5357] provide capabilities for the measurement of various performance metrics in IP networks using probe messages. These protocols rely on control-channel signaling to establish a test-channel over an UDP path. The TWAMP Light [Appendix I in RFC5357] [BBF.TR-390] provides simplified mechanisms for active performance measurement in Customer IP networks by provisioning UDP paths and eliminates the need for control-channel signaling. As described in Appendix A of [RFC8545], TWAMP Light mechanism is informative only. These protocols lack support for direct-mode Loss Measurement (LM) to detect actual Customer data traffic loss which is required in SR networks.

This document describes RFC 5357 (Two-Way Active Measurement Protocol (TWAMP) Light) extensions for Delay and Loss Measurement in Segment Routing networks, for both SR-MPLS and SRv6 data planes.

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

BSID: Binding Segment ID.

DM: Delay Measurement.

HMAC: Hashed Message Authentication Code.

LM: Loss Measurement.

MPLS: Multiprotocol Label Switching.

NTP: Network Time Protocol.

OWAMP: One-Way Active Measurement Protocol.

PM: Performance Measurement.

PTP: Precision Time Protocol.

SID: Segment ID.

SL: Segment List.

SR: Segment Routing.

SRH: Segment Routing Header.

SR-MPLS: Segment Routing with MPLS data plane.

SRv6: Segment Routing with IPv6 data plane.

TWAMP: Two-Way Active Measurement Protocol.

2.3. Reference Topology

In the reference topology shown below, the sender node R1 initiates a performance measurement probe query message and the reflector node R5 sends a probe response message for the query message received. The probe response message is typically sent to the sender node R1.

```
                   t1                  t2
                  /                      \
         +-------+       Query          +-------+
         |       | - - - - - - - - - ->|       |
         |   R1  |=====================|   R5  |
         |       |<- - - - - - - - - - |       |
         +-------+       Response       +-------+
                  \                      /
                   t4                  t3
          Sender                         Reflector
```

Reference Topology

3. Probe Query Message

3.1. Control Code Field Extension for TWAMP Light Messages

In this document, the Control Code field is defined for delay and loss measurement probe query messages for TWAMP Light in unauthenticated and authenticated modes. The modified delay measurement probe query message format is shown in Figure 1. This message format is backwards compatible with the message format defined in [RFC5357] as its reflector ignores the received field (previously identified as MBZ). With this field, the reflector node does not require any additional state for PM (recall that in SR networks, the state is in the probe packet and signaling of the parameters is undesired). The usage of the Control Code is not limited to the SR and can be used for non-SR network.

```
    .                                                               .
    .                                                               .
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                          Timestamp                            |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |         Error Estimate        |  MBZ                          |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |         MBZ                                    |Se Control Code|
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    .                                                               .
    .                                                               .
```

Figure 1: Sender Control Code in TWAMP Light DM Message

Sender Control Code: Set as follows in TWAMP Light probe query message.

In a Query:

0x0: Out-of-band Response Requested. Indicates that the probe response is not required over the same path in the reverse direction. This is also the default behavior.

0x1: In-band Response Requested. Indicates that this query has been sent over a bidirectional path and the probe response is required over the same path in the reverse direction.

0x2: No Response Requested.

3.2. Loss Measurement Query Message Extensions

In this document, TWAMP Light probe query messages for loss measurement are defined as shown in Figure 2 and Figure 3. The message formats are hardware efficient due to well-known locations of the counters and payload small in size. They are stand-alone and similar to the delay measurement message formats (e.g. location of the Counter and Timestamp). They also do not require backwards compatibility and support for the existing DM message formats from [RFC5357] as different user-configured destination UDP port is used for loss measurement.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sequence Number                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Transmit Counter                       |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|X|B| Reserved  | Block Number  | MBZ                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         MBZ                                   |Se Control Code|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
.                                                               .
.                        Packet Padding                         .
.                                                               .
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: TWAMP Light LM Probe Query Message - Unauthenticated Mode

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                        Sequence Number                        |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                        MBZ (12 octets)                        |
   |                                                               |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                        Transmit Counter                       |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |X|B| Reserved  | Block Number  | MBZ                           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |         MBZ                                   |Se Control Code|
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                        HMAC (16 octets)                       |
   |                                                               |
   |                                                               |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   .                                                               .
   .                        Packet Padding                         .
   .                                                               .
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: TWAMP Light LM Probe Query Message - Authenticated Mode

Sequence Number (32-bit): As defined in [RFC5357].

Transmit Counter (64-bit): The number of packets or octets sent by the sender node in the query message and by the reflector node in the response message. The counter is always written at the well-known location in the probe query and response messages.

Receive Counter (64-bit): The number of packets or octets received at the reflector node. It is written by the reflector node in the probe response message.

Sender Counter (64-bit): This is the exact copy of the transmit counter from the received query message. It is written by the reflector node in the probe response message.

Sender Sequence Number (32-bit): As defined in [RFC5357].

Sender TTL: As defined in [RFC5357].

LM Flags: 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 an LM query and copied from an LM query to an LM response message. Set to 0 when the LM message 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 1-4 fields represent octet counts. The octet count applies to all packets within the LM scope, and the octet count of a packet sent 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.

Block Number (8-bit): The Loss Measurement using Alternate-Marking method defined in [RFC8321] requires to color the data traffic. To be able to correlate the transmit and receive traffic counters of the matching color, the Block Number (or color) of the traffic counters is carried by the probe query and response messages for loss measurement. The Block Number can also be used to aggregate performance metrics collected.

HMAC: The probe message in authenticated mode includes a key Hashed Message Authentication Code (HMAC) [RFC2104] hash. Each probe query and response messages are authenticated by adding Sequence Number with Hashed Message Authentication Code (HMAC) TLV. It can use HMAC-SHA-256 truncated to 128 bits (similarly to the use of it in IPSec defined in [RFC4868]); hence the length of the HMAC field is 16 octets.

HMAC uses its own key and the mechanism to distribute the HMAC key is outside the scope of this document.

In authenticated mode, only the sequence number is encrypted, and the other payload fields are sent in clear text. The probe message may include Comp.MBZ (Must Be Zero) variable length field to align the packet on 16 octets boundary.

4. Probe Response Message

4.1. Loss Measurement Response Message Extensions

In this document, TWAMP Light probe response message formats are defined for loss measurement as shown in Figure 4 and Figure 5.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sequence Number                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Transmit Counter                       |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|X|B| Reserved  | Block Number  | MBZ                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Receive Counter                        |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sender Sequence Number                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sender Counter                         |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|X|B| Reserved  |Sender Block Nu|   MBZ                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Sender TTL   |                                               |
+-+-+-+-+-+-+-+-+                                               +
|                                                               |
.                                                               .
.                        Packet Padding                         .
.                                                               .
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4: TWAMP Light LM Probe Response Message - Unauthenticated Mode

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sequence Number                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        MBZ (12 octets)                        |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Transmit Counter                       |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|X|B| Reserved  | Block Number  | MBZ                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        MBZ (4 octets)                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
    |                        Receive Counter                        |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                        MBZ (8 octets)                         |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                        Sender Sequence Number                 |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                        MBZ (12 octets)                        |
    |                                                               |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                        Sender Counter                         |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |X|B| Reserved  |Sender Block Nu|   MBZ                         |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                        MBZ (4 octets)                         |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |  Sender TTL   |                                               |
    +-+-+-+-+-+-+-+-+                                               |
    |                        MBZ (15 octets)                        |
    |                                                               |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                        HMAC (16 octets)                       |
    |                                                               |
    |                                                               |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    .                                                               .
    .                        Packet Padding                         .
    .                                                               .
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 5: TWAMP Light LM Probe Response Message - Authenticated Mode

5. Security Considerations

The performance measurement is intended for deployment in well-managed private and service provider networks. As such, it assumes that a node involved in a measurement operation has previously verified the integrity of the path and the identity of the far-end reflector node.

If desired, attacks can be mitigated by performing basic validation and sanity checks, at the sender, of the counter or timestamp fields in received measurement response messages. The minimal state associated with these protocols also limits the extent of measurement disruption that can be caused by a corrupt or invalid message to a single query/response cycle.

Use of HMAC-SHA-256 in the authenticated mode protects the data integrity of the probe messages. Cryptographic measures may be enhanced by the correct configuration of access-control lists and firewalls.

6. IANA Considerations

This document does not require any IANA action.

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>.

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

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

[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

[RFC2104] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, DOI 10.17487/RFC2104, February 1997, <https://www.rfc-editor.org/info/rfc2104>.

[RFC4868] Kelly, S. and S. Frankel, "Using HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512 with IPsec", RFC 4868, DOI 10.17487/RFC4868, May 2007, <https://www.rfc-editor.org/info/rfc4868>.

[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>.

[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>.

[BBF.TR-390] "Performance Measurement from IP Edge to Customer Equipment using TWAMP Light", BBF TR-390, May 2017.

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 for reviewing this document and providing useful comments and suggestions. 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

ippm R. Geib, Ed.
Internet-Draft Deutsche Telekom
Intended status: Standards Track July 3, 2020
Expires: January 4, 2021

A Connectivity Monitoring Metric for IPPM
draft-geib-ippm-connectivity-monitoring-03

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

Copyright Notice

Table of Contents

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 or 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 processor- 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 change in network behaviour may then require 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 enables link-loss detection, if the change in end-to-end delay along a new route is differing from that of the original path.

A Segment Routing PMS which is part of an SR domain 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 segments. Basically, a number of overlaid measurement paths is set up. The delays of packets sent along each on of these paths is measured. Single changes in topology cause correlated changes in the measurement packet delay (or packet loss) of different measurement paths. By a suitable set up, the number of measurement paths may be limited to one per connection (or sub-path) to be monitored. In addition to information revealed by a commodity ICMP ping measurement, the metric and method specified here identify the location of a congested interface. To do so, tomography assumptions and methods are combined to first plan the overlaid SR measurement path set up and later on to evaluate the captured delay measurements.

This document specifies a type-p metric determining properties of an SR path which allows to monitor connectivity and congestion of interfaces and further allows to locate the path or interface which caused a change in the reported type-p metric. This document is focussed on the 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 [I-D.draft-ietf-isis-segment-routing-extensions], 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 availbale, [RFC8029] 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 [RFC8029] methods as a replacement of Adj-SIDs is out of scope of this document.

A 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 identify which link has failed. A round trip measurement also 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. If these form a partial overlay, any change in the network properties impacts more than a single loop's round trip time (or causes drops of packets of more than one loop). Randomly chosen loop paths including the interfaces or paths to be monitored may fail to produce unique result patterns. The approach picked here uses specified measurement loop and path overlay design. A centralised monitoring approach benefits from keeping the number of required measurement loops low. This improves scalability by minimising the number of measurement loops. This also keeps the number of required packets and results to be evaluated and correlated low.

An additional property of the measurement path set-up specified below is that it allows to estimate the packet round trip and the one way delay of a monitored link (or path). The delay along a single link is not perfectly symmetric. Packet processing causes small delay differences per interface and direction. These cause an error, which can't be quantified or removed by the specified method. Quantifying this error requires a different measurement set-up. As this will introduce additional measurements loops, packets and evaluations, the cost in terms of reduced scalability is not felt to be worth the benefit in measurement accuracy. IPPM however honors precision more than accuracy and the mentioned processing differences are relatively stable, resulting in relatively precise delay estimates.

An example SR domain is shown below. The PMS shown should monitor the connectivity of all 6 links between nodes L100 and L200 one one side and the connected nodes L050, L060 and L070 on the other side. The round trip times per measurement loop are assumed to exhibit unique delays.

```
   +---+   +----+     +----+
   |PMS|   |L100|-----|L050|
   +---+   +----+\   /+----+
     |    /    \  \_/______
     |   /      \  /      \+----+
  +----+/        \/_  +----|L060|
  |L300|         /  |/     +----+
  +----+\       /   /\_
         \     /   /   \
          \+----+ /   +----+
           |L200|-----|L070|
           +----+     +----+
```

Connectivity verification with a PMS

Figure 1

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

Monitoring the 6 links between Ln00 and L0m0 nodes requires 6 measurement loops, each of which has the following properties:

- o Each loop follows a single round trip from one Ln00 to one L0m0 (e.g., between L100 and L050).

- o Each loop passes two more links: one between that Ln00 and another L0m0 and from there to the other Ln00 (e.g., between L100 and L060 and then L060 to L200)

- o Every link is passed by a single round trip per measurement loop only once and only once unidirectional by two other loops, and the latter two pass along opposing directions (that's three loops passing each single link, e.g., one having a round trip L100 to L050 and back, a second passing L100 to L050 only and a third loop passing L050 to L100 only).

Note that any 6 links between two to six nodes can be monitored that way too (if multiple parallel links between two nodes are monitored, the differences in delay may require a sufficiently high clock resulotion, if applicable).

This 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. It is ommitted here for brevity):

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

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

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.

The measurement loops set up as shown have the following properties:

- o 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 (and changes the measured delay) of three loops. Traffic to Node-SIDs is rerouted.

- o If the loops are set up using Adj-SIDs only (and Node-SIDs only to send the packet from PMS to the loop starting point and from the loop end back to the PMS), any single complete loss of connectivity caused by a failing single link between any Ln00 and any L0m0 node terminates the traffic along three loops. The packets of these loops will be dropped, until the link gets back into service. Traffic to Adj-SIDs is not rerouted.

- Any congested single interface between any Ln00 and any L0m0 node only impacts the measured delay of two measurement loops.

- As an example, the formula for a single Round Trip Delay (RTD) is shown here 4 * RTD_L100-L050-L100 = 3 * M1 + M3 + M6 - M2 - M4 - M5

A closer look reveals that each single event of interest for the proposed metric, which are a loss of connectivity or a case of congestion, uniquely only impacts a single a-priori determinable set of measurement loops. If, e.g., connectivity is lost between L200 and L050, measurement loops (3), (4) and (6) indicate a change in the measured delay.

As a second example, if the interface L070 to L100 is congested, measurement loops (3) and (5) indicate a change in the measured delay. Without listing all events, all cases of single losses of connectivity or single events of congestion influence only delay measurements of a unique set of measurement loops.

A congestion event adding latency to two specific measurement loops allows calculation of the delay added by the queue at the congested interface. Thus, the resulting RTD increase can be assigned to a single interface.

3. Singleton Definition for Type-P-SR-Path-Connectivity-and-Congestion

3.1. Metric Name

Type-P-SR-Path-Connectivity-and-Congestion

3.2. Metric Parameters

- Src, the IP address of a source host

- Dst, the IP address of a destination host if IP routing is applicable; in the case of MPLS routing, a diagnostic address as specified by [RFC8029]

- T, a time

- lambda, a rate in reciprocal seconds

- o L, a packet length in bits. The packets of a Type P packet stream from which the sample Path-Connectivity-and-Congestion metric is taken MUST all be of the same length.

- o MLA, a Monitoring Loop Address information ensuring that a singleton passes a single sub-path_a to be monitored bidirectional, a sub-path_b to be monitored unidirectional and a sub-path_c to be monitored unidirectional, where sub-path_a, -_b and -_c MUST NOT be identical.

- o P, the specification of the packet type, over and above the source and destination addresses

- o DS, a constant time interval between two type-P packets

3.3. Metric Units

A sequence of consecutive time values.

3.4. Definition

A moving average of AV time values per measurement path is compared by a change point detection algorithm. The temporal packet spacing value DS represents the smallest period within which a change in connectivity or congestion may be detected.

A single loss of connectivity of a sub-path between two nodes affects three different measurement paths. Depending on the value chosen for DS, packet loss might occur (note that the moving average evaluation needs to span a longer period than convergence time; alternatively, packet-loss visible along the three measurement paths may serve as an evaluation criterium). After routing convergence the type-p packets along the three measurement paths show a change in delay.

A congestion of a single interface of a sub-path connecting two nodes affects two different measurement paths. The the type-p packets along the two congested measurement paths show an additional change in delay.

3.5. Discussion

Detection of a multiple losses of monitored sub-path connectivity or congestion of a multiple monitored sub-paths may be possible. These cases have not been investigated, but may occur in the case of Shared Risk Link Groups. Monitoring Shared Risk LinkGroups and sub-paths with multiple failures abd congestion is not within scope of this document.

3.6. Methodologies

For the given type-p, the methodology is as follows:

o The set of measurement paths MUST be routed in a way that each single loss of connectivity and each case of single interface congestion of one of the sub-paths passed by a type-p packet creates a unique pattern of type-p packets belonging to a subset of all configured measurement paths indicate a change in the measured delay. As a minimum, each sub-path to be monitored MUST be passed

o

  * by one measurement_path_1 and its type-p packet in bidirectional direction

  * by one measurement_path_2 and its type-p packet in "downlink" direction

  * by one measurement_path_3 and its type-p packet in "uplink" direction

o "Uplink" and "Downlink" have no architectural relevance. The terms are chosen to express, that the packets of measurement_path_2 and measuremnt_path_3 pass the monitored sub-path unidirectional in opposing direction. Measuremnt_path_1, measurement_path_2 and measurement_path_3 MUST NOT be identical.

o All measurement paths SHOULD terminate between identical sender and receiver interfaces. It is recommended to connect the sender and receiver as closely to the paths to be monitored as possible. Each intermediate sub-path between sender and receiver one one hand and sub-paths to be monitored is an additional source of errors requiring separate monitoring.

o Segment Routed domains supporting Node- and Adj-SIDs should enable the monitoring path set-up as specified. Other routing protocols may be used as well, but the monitoring path set up might be complex or impossible.

o Pre-compute how the two and three measurement path delay changes correlate to sub-path connectivity and congestion patterns. Absolute change valaues aren't required, a simultaneous change of two or three particular measurement paths is.

o Ensure that the temporal resolution of the measurement clock allows to reliably capture a unique delay value for each configured measurement path while sub-path connectivity is complete and no congestion is present.

- Synchronised clocks are not strictly required, as the metric is evaluating differences in delay. Changes in clock synchronisation SHOULD NOT be close to the time interval within which changes in connectivity or congestion should be monitored.

- At the Src host, select Src and Dst IP addresses, and address information to route the type-p packet along one of the configured measurement path. Form a test packet of Type-P with these addresses.

- Configure the Dst host access to receive the packet.

- At the Src host, place a timestamp, a sequence number and a unique identifier of the measurement path in the prepared Type-P packet, and send it towards Dst.

- Capture the one-way delay and determine packet-loss by the metrics specified by [RFC7679] and [RFC7680] respectively and store the result for the path.

- If two or three subpaths indicate a change in delay, report a change in connectivity or congestion status as pre-computed above.

- If two or three sub paths indicate a change in delay, report a change in connectivity or congestion status as pre-computed above.

Note that monitoring 6 sub paths requires setting up 6 monitoring paths as shown in the figure above.

3.7. Errors and Uncertainties

Sources of error are:

- Measurement paths whose delays don't indicate a change after sub-path connectivity changed.

- A timestamps whose resolution is missing or inacurrate at the delays measured for the different monitoring paths.

- Multiple occurrences of sub path connectivity and congestion.

- Loss of connectivity and congestion along sub-paths connecting the measurement device(s) with the sub-paths to be monitored.

3.8. Reporting the Metric

The metric reports loss of connectivity of monitored sub-path or congestion of an interface and identifies the sub-path and the direction of traffic in the case of congestion.

The temporal resolution of the detected events depends on the spacing interval of packets transmitted per measurement path. An identical sending interval is chosen for every measurement path. As a rule of thumb, an event is reliably detected if a sample consists of at least 5 probes indicating the same underlying change in behavior. Depending on the underlying event either two or three measurement paths are impacted. At least two consecutively received measurement packets per measurement path should suffice to indicate a change. The values chosen for an operational network will have to reflect scalability constraints of a PMS measurement interface. As an example, a PMS may work reliable if no more than one measurement packet is transmitted per millisecond. Further, measurement is configured so that the measurement packets return to the sender interface. Assume always groups of 6 links to be monitored as described above by 6 measurements paths. If one packet is sent per measurement path within 500 ms, up to 498 links can be monitored with a reliable temporal resolution of roughly one second per detected event.

Note that per group measurement packet spacing, measurement loop delay difference and latency caused by congestion impact the reporting interval. If each measurement path of a single 6 link monitoring group is addressed in consecutive milliseconds (within the 500 ms interval) and the sum of maximum physical delay of the per group measurement paths and latency possibly added by congestion is below 490 ms, the one second reports reliably capture 4 packets of two different measurement paths, if two measurement paths are congested, or 6 packets of three different measurement paths, if a link is lost.

A variety of reporting options exist, if scalability issues and network properties are respected.

4. Singleton Definition for Type-P-SR-Path-Round-Trip-Delay-Estimate

This section will be added in a later version, if there’s interest in picking up this work.

5. IANA Considerations

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

6. 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.

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>.

[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>.

[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>.

[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>.

7.2. Informative References

[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
Darmstadt 64295
Germany

Phone: +49 6151 5812747
Email: Ruediger.Geib@telekom.de

IPPM
Internet-Draft
Intended status: Standards Track
Expires: May 5, 2021

H. Song
Futurewei
B. Gafni
Mellanox Technologies, Inc.
T. Zhou
Z. Li
Huawei
F. Brockners
S. Bhandari
R. Sivakolundu
Cisco
T. Mizrahi, Ed.
Huawei Smart Platforms iLab

November 1, 2020

In-situ OAM Direct Exporting
draft-ietf-ippm-ioam-direct-export-02

Abstract

In-situ Operations, Administration, and Maintenance (IOAM) is used for recording and collecting operational and telemetry information. Specifically, IOAM allows telemetry data to be pushed into data packets while they traverse the network. This document introduces a new IOAM option type called the Direct Export (DEX) option, which is used as a trigger for IOAM data to be directly exported without being pushed into in-flight data packets.

Table of Contents

1. Introduction

IOAM [I-D.ietf-ippm-ioam-data] is used for monitoring traffic in the network, and for incorporating IOAM data fields into in-flight data packets.

IOAM makes use of four possible IOAM options, defined in [I-D.ietf-ippm-ioam-data]: Pre-allocated Trace Option, Incremental Trace Option, Proof of Transit (POT) Option, and Edge-to-Edge Option.

This document defines a new IOAM option type (also known as an IOAM type) called the Direct Export (DEX) option. This option is used as a trigger for IOAM nodes to export IOAM data to a receiving entity (or entities). A "receiving entity" in this context can be, for example, an external collector, analyzer, controller, decapsulating node, or a software module in one of the IOAM nodes.

This draft has evolved from combining some of the concepts of PBT-I from [I-D.song-ippm-postcard-based-telemetry] with immediate exporting from [I-D.ietf-ippm-ioam-flags].

2. Conventions

2.1. Requirement Language

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

2.2. Terminology

Abbreviations used in this document:

IOAM: In-situ Operations, Administration, and Maintenance

OAM: Operations, Administration, and Maintenance

DEX: Direct EXporting

3. The Direct Exporting (DEX) IOAM Option Type

3.1. Overview

The DEX option is used as a trigger for exporting telemetry data to a receiving entity (or entities).

This option is incorporated into data packets by an IOAM encapsulating node, and removed by an IOAM decapsulating node, as illustrated in Figure 1. The option can be read but not modified by transit nodes. Note: the terms IOAM encapsulating, decapsulating and transit nodes are as defined in [I-D.ietf-ippm-ioam-data].

```
                                      ^
                                      |Exported IOAM data
                                      |
                                      |
                                      |
                +--------------+------+-------+--------------+
                |              |              |              |
                |              |              |              |
  User      +---+----+     +---+----+     +---+----+     +---+----+
  packets   |Encapsu-|     | Transit|     | Transit|     |Decapsu-|
  --------->|lating  |====>| Node   |====>| Node   |====>|lating  |---->
            |Node    |     | A      |     | B      |     |Node    |
            +--------+     +--------+     +--------+     +--------+
            Insert DEX       Export         Export       Remove DEX
            option and     IOAM data      IOAM data      option and
            export data                                  export data
```

Figure 1: DEX Architecture

The DEX option is used as a trigger to export IOAM data. The trigger applies to transit nodes, the decapsulating node, and the encapsulating node:

- o An IOAM encapsulating node configured to incorporate the DEX option encapsulates the packet with the DEX option, and MAY export the requested IOAM data immediately. The IOAM encapsulating node is the only type of node allowed to push the DEX option.

- o A transit node that processes a packet with the DEX option MAY export the requested IOAM data.

- o An IOAM decapsulating node that processes a packet with the DEX option MAY export the requested IOAM data, and MUST decapsulate the IOAM header.

As in [I-D.ietf-ippm-ioam-data], the DEX option may be incorporated into all or a subset of the traffic that is forwarded by the encapsulating node. Moreover, IOAM nodes MAY export data for all traversing packets that carry the DEX option, or MAY selectively export data only for a subset of these packets.

The DEX option specifies which data fields should be exported, as specified in Section 3.2. The format and encapsulation of the packet that contains the exported data is not within the scope of the current document. For example, the export format can be based on [I-D.spiegel-ippm-ioam-rawexport].

A transit IOAM node that does not support the DEX option SHOULD ignore it. A decapsulating node that does not support the DEX option MUST remove it, along with any other IOAM options carried in the packet if such exist.

3.2. The DEX Option Format

The format of the DEX option is depicted in Figure 2. The length of the DEX option is either 8 octets or 16 octets, as the Flow ID and the Sequence Number fields (summing up to 8 octets) are optional. It is assumed that the lower layer protocol indicates the length of the DEX option, thus indicating whether the two optional fields are present.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Namespace-ID           |             Flags             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               IOAM-Trace-Type                 |   Reserved    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Flow ID (optional)                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Sequence Number  (Optional)                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: DEX Option Format

Namespace-ID A 16-bit identifier of the IOAM namespace, as defined in [I-D.ietf-ippm-ioam-data].

Flags A 16-bit field, comprised of 16 one-bit subfields. Flags are allocated by IANA, as defined in Section 4.2.

IOAM-Trace-Type A 24-bit identifier which specifies which data fields should be exported. The format of this field is as defined in [I-D.ietf-ippm-ioam-data]. Specifically, bit 23, which corresponds to the Checksum Complement data field, should be assigned to be zero by the IOAM encapsulating node, and ignored by transit and decapsulating nodes. The reason for this is that the Checksum Complement is intended for in-flight packet modifications and is not relevant for direct exporting.

Reserved This field SHOULD be ignored by the receiver.

Flow ID A 32-bit flow identifier. If the actual Flow ID is shorter than 32 bits, it is zero padded in its most significant bits. The field is set at the encapsulating node. The Flow ID can be uniformly assigned by a central controller or algorithmically generated by the encapsulating node. The latter approach cannot guarantee the uniqueness of Flow ID, yet the conflict probability is small due to the large Flow ID space. The Flow ID can be used to correlate the exported data of the same flow from multiple nodes and from multiple packets.

Sequence Number A 32-bit sequence number starting from 0 and increasing by 1 for each following monitored packet from the same flow at the encapsulating node. The Sequence Number, when combined with the Flow ID, provides a convenient approach to correlate the exported data from the same user packet.

4. IANA Considerations

4.1. IOAM Type

The "IOAM Type Registry" was defined in Section 7.2 of [I-D.ietf-ippm-ioam-data]. IANA is requested to allocate the following code point from the "IOAM Type Registry" as follows:

TBD-type IOAM Direct Export (DEX) Option Type

If possible, IANA is requested to allocate code point 4 (TBD-type).

4.2. IOAM DEX Flags

IANA is requested to define an "IOAM DEX Flags" registry. This registry includes 16 flag bits. Allocation should be performed based on the "RFC Required" procedure, as defined in [RFC8126].

5. Performance Considerations

The DEX option triggers exported packets to be exported to a receiving entity (or entities). In some cases this may impact the receiving entity's performance, or the performance along the paths leading to it.

Therefore, rate limiting may be enabled so as to ensure that direct exporting is used at a rate that does not significantly affect the network bandwidth, and does not overload the receiving entity (or the source node in the case of loopback). It should be possible to use each DEX on a subset of the data traffic, and to load balance the exported data among multiple receiving entities.

6. Security Considerations

The security considerations of IOAM in general are discussed in [I-D.ietf-ippm-ioam-data]. Specifically, an attacker may try to use the functionality that is defined in this document to attack the network.

An attacker may attempt to overload network devices by injecting synthetic packets that include the DEX option. Similarly, an on-path attacker may maliciously incorporate the DEX option into transit packets, or maliciously remove it from packets in which it is incorporated.

Forcing DEX, either in synthetic packets or in transit packets may overload the receiving entity (or entities). Since this mechanism affects multiple devices along the network path, it potentially amplifies the effect on the network bandwidth and on the receiving entity’s load.

In order to mitigate the attacks described above, it should be possible for IOAM-enabled devices to limit the exported IOAM data to a configurable rate.

IOAM is assumed to be deployed in a restricted administrative domain, thus limiting the scope of the threats above and their affect. This is a fundamental assumption with respect to the security aspects of IOAM, as further discussed in [I-D.ietf-ippm-ioam-data].

7. References

7.1. 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-10 (work in progress), July 2020.

[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>.

7.2. Informative References

[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 Flags", draft-ietf-ippm-ioam-flags-03 (work in progress), October 2020.

[I-D.song-ippm-postcard-based-telemetry]
Song, H., Zhou, T., Li, Z., Mirsky, G., Shin, J., and K. Lee, "Postcard-based On-Path Flow Data Telemetry using Packet Marking", draft-song-ippm-postcard-based-telemetry-08 (work in progress), October 2020.

[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-03 (work in progress), March 2020.

[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>.

Appendix A. Hop Limit and Hop Count in Direct Exporting

In order to help correlate and order the exported packets, it is possible to include the Hop_Lim/Node_ID data field in exported packets; if the IOAM-Trace-Type [I-D.ietf-ippm-ioam-data] has the Hop_Lim/Node_ID bit set, then exported packets include the Hop_Lim/Node_ID data field, which contains the TTL/Hop Limit value from a lower layer protocol.

An alternative approach was considered during the design of this document, according to which a 1-octet Hop Count field would be included in the DEX header (presumably by claiming some space from the Flags field). The Hop Limit would starts from 0 at the encapsulating node and be incremented by each IOAM transit node that supports the DEX option. In this approach the Hop Count field value would also be included in the exported packet.

The main advantage of the Hop_Lim/Node_ID approach is that it provides information about the current hop count without requiring each transit node to modify the DEX option, thus simplifying the data plane functionality of Direct Exporting. The main advantage of the Hop Count approach that was considered is that it counts the number of IOAM-capable nodes without relying on the lower layer TTL, especially when the lower layer cannot prvide the accurate TTL information, e.g., Layer 2 Ethernet or hierarchical VPN. The Hop Count approach would also explicitly allow to detect a case where an IOAM-capable node fails to export packets. It would also be possible to use a flag to indicate an optional Hop Count field, which enables to control the tradeoff. On one hand it would address the use cases that the Hop_Lim/Node_ID cannot cover, and on the other hand it would not require transit switches to update the option if it was not supported or disabled. For the sake of simplicity the Hop Count approach was not pursued, and this field is not incorporated in the DEX header.

Authors' Addresses

Haoyu Song
Futurewei
2330 Central Expressway
Santa Clara 95050
USA

Email: haoyu.song@huawei.com

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

Email: gbarak@mellanox.com

Tianran Zhou
Huawei
156 Beiqing Rd.
Beijing 100095
China

Email: zhoutianran@huawei.com

Zhenbin Li
Huawei
156 Beiqing Rd.
Beijing 100095
China

Email: lizhenbin@huawei.com

Frank Brockners
Cisco Systems, Inc.
Hansaallee 249, 3rd Floor
DUESSELDORF, NORDRHEIN-WESTFALEN 40549
Germany

Email: fbrockne@cisco.com

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

Email: shwethab@cisco.com

Ramesh Sivakolundu
Cisco Systems, Inc.
170 West Tasman Dr.
SAN JOSE, CA 95134
U.S.A.

Email: sramesh@cisco.com

Tal Mizrahi (editor)
Huawei Smart Platforms iLab
8-2 Matam
Haifa 3190501
Israel

Email: tal.mizrahi.phd@gmail.com

IPPM
Internet-Draft
Intended status: Standards Track
Expires: April 29, 2021

T. Mizrahi
Huawei Smart Platforms iLab
F. Brockners
S. Bhandari
R. Sivakolundu
C. Pignataro
Cisco
A. Kfir
B. Gafni
Mellanox Technologies, Inc.
M. Spiegel
Barefoot Networks
J. Lemon
Broadcom
October 26, 2020

In-situ OAM Flags
draft-ietf-ippm-ioam-flags-03

Abstract

In-situ Operations, Administration, and Maintenance (IOAM) records operational and telemetry information in the packet while the packet traverses a path between two points in the network. This document presents new flags in the IOAM Trace Option headers. Specifically, the document defines the Loopback and Active flags.

Status of This Memo

Table of Contents

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.

IOAM data may be represented in one of four possible IOAM options: Pre-allocated Trace Option, Incremental Trace Option, Proof of Transit (POT) Option, and Edge-to-Edge Option. This document defines two new flags in the Pre-allocated and Incremental Trace options: the Loopback and Active flags.

2. Conventions

2.1. Requirement Language

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

2.2. Terminology

Abbreviations used in this document:

IOAM: In-situ Operations, Administration, and Maintenance

OAM: Operations, Administration, and Maintenance

3. New IOAM Trace Option Flags

This document defines two new flags in the Pre-allocated and Incremental Trace options:

Bit 1 "Loopback" (L-bit). Loopback mode is used to send a copy of a packet back towards the source, as further described in Section 4.

Bit 2 "Active" (A-bit). When set, this indicates that this is an active IOAM packet, where "active" is used in the sense defined in [RFC7799], rather than a data packet. The packet may be an IOAM probe packet, or a replicated data packet (the second and third use cases of Section 5).

4. Loopback in IOAM

Loopback is used for triggering each transit device along the path to loop back a copy of the data packet. 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 intended to provide an accelerated alternative to Traceroute, that allows the encapsulating node to receive responses from multiple transit nodes along the path in less then one round-trip-time, and by sending a single packet.

Loopback can be used only if a return path from transit nodes and destination nodes towards the source (encapsulating node) exists. Specifically, loopback is only applicable in encapsulations in which the identity of the encapsulating node is available in the encapsulation header. If an encapsulating node receives a looped back packet that was not originated from the current encapsulating node, the packet is dropped.

The encapsulating node either generates synthetic packets with an IOAM trace option that has the loopback flag set, or sets the loopack flag in a subset of the in-transit data packets. Loopback is used either proactively or on-demand, i.e., when a failure is detected. The encapsulating node also needs to ensure that sufficient space is available in the IOAM header for loopback operation, which includes transit nodes adding trace data on the original path and then again on the return path.

An IOAM trace option that has the loopback bit set MUST have the value '1' in the most significant bit of IOAM-Trace-Type, and '0' in the rest of the bits of IOAM-Trace-Type. Thus, every transit node that processes this trace option only adds a single data field, which is the Hop_Lim and node_id data field. The reason for allowing a single data field per hop is to minimize the impact of amplification attacks.

A loopback bit that is set indicates to the transit nodes processing this option that they are to create a copy of the received packet and send the copy back to the source of the packet. In this context the source is the IOAM encapsulating node, and it is assumed that the source address is available in the encapsulation header. Thus, the source address of the original packet is used as the destination address in the copied packet. The address of the node performing the copy operation is used as the source address. The IOAM transit node pushes the required data field *after* creating the copy of the packet, in order to allow any egress-dependent information to be set based on the egress of the copy rather than the original packet. The copy is also truncated, i.e., any payload that resides after the IOAM option(s) is removed before transmitting the looped back packet back towards the encapsulating node. The original packet continues towards its destination. The L-bit MUST be cleared in the copy of the packet that a node sends back towards the source.

On its way back towards the source, the copied packet is processed like any other packet with IOAM information, including adding any requested data at each transit node (assuming there is sufficient space).

Once the return packet reaches the IOAM domain boundary, IOAM decapsulation occurs as with any other packet containing IOAM information. Note that the looped back packet does not have the L-bit set. The IOAM encapsulating node that initiated the original loopback packet recognizes a received packet as an IOAM looped-back packet by checking the Node ID in the Hop_Lim/node_id field that corresponds to the first hop. If the Node ID matches the current IOAM node, it indicates that this is a looped back packet that was initiated by the current IOAM node, and processed accordingly. If there is no match in the Node ID, the packet is processed like a conventional IOAM-encapsulated packet.

Note that an IOAM encapsulating node may either be an endpoint (such as an IPv6 host), or a switch/router that pushes a tunnel encapsulation onto data packets. In both cases, the functionality that was described above avoids IOAM data leaks from the IOAM domain. Specificallly, if an IOAM looped-back packet reaches an IOAM boundary node that is not the IOAM node that initiated the loopback, the node does not process the packet as a loopback; the IOAM encapsulation is removed, and since the packet does not have any payload it is terminated. In either case, when the packet reaches the IOAM boundary its IOAM encapsulation is removed, preventing IOAM information from leaking out from the IOAM domain.

5. Active Measurement with IOAM

Active measurement methods [RFC7799] make use of synthetically generated packets in order to facilitate the measurement. This section presents use cases of active measurement using the IOAM Active flag.

The active flag indicates that a packet is used for active measurement. An IOAM decapsulating node that receives a packet with the Active flag set in one of its Trace options must terminate the packet. The active flag is intended to simplify the implementation of decapsulating nodes by indicating that the packet should not be forwarded further. It is not intended as a replacement for existing active OAM protocols, which may run in higher layers and make use of the active flag.

An example of an IOAM deployment scenario is illustrated in Figure 1. The figure depicts two endpoints, a source and a destination. The data traffic from the source to the destination is forwarded through a set of network devices, including an IOAM encapsulating node, which incorporates one or more IOAM options, a decapsulating node, which removes the IOAM options, optionally one or more transit nodes. The IOAM options are encapsulated in one of the IOAM encapsulation types, e.g., [I-D.ietf-sfc-ioam-nsh], or [I-D.ietf-ippm-ioam-ipv6-options].

```
 +---------+     +---------+     +---------+     +---------+     +---------+
 |         |     |  IOAM   |.....|  IOAM   |.....|  IOAM   |     |         |
 +---------+     +---------+     +---------+     +---------+     +---------+
 | L2/L3   |<===>| L2/L3   |<===>| L2/L3   |<===>| L2/L3   |<===>| L2/L3   |
 +---------+     +---------+     +---------+     +---------+     +---------+
   Source       Encapsulating    Transit      Decapsulating   Destination
                    Node           Node            Node

                 <------------  IOAM domain  ----------->
```

Figure 1: Network using IOAM.

This draft focuses on three possible use cases of active measurement using IOAM. These use cases are described using the example of Figure 1.

o Endpoint active measurement: synthetic probe packets are sent between the source and destination, traversing the IOAM domain. 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. Specifically, the active flag is not used in this case, and the IOAM layer needs not be aware that an active measurement mechanism is used at a higher layer.

o IOAM active measurement using probe packets within the IOAM domain: probe packets are generated and transmitted by the IOAM encapsulating node, and are expected to be terminated by the decapsulating node. IOAM data related to probe packets may be exported by one or more nodes along its path, by an exporting protocol that is outside the scope of this document (e.g., [I-D.spiegel-ippm-ioam-rawexport]). Probe packets include a Trace Option which has its Active flag set, indicating that the decapsulating node must terminate them.

o IOAM active measurement using replicated data packets: probe packets are created by the 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 that has its Active flag set, indicating that the decapsulating node should terminate it. It should be noted that the current document defines the role of the active flag in allowing the decapsulating node to terminate the packet, but the replication functionality in this context is outside the scope of this document.

6. IANA Considerations

IANA is requested to allocate the following bits in the "IOAM Trace Flags Registry" as follows:

Bit 1 "Loopback" (L-bit)

Bit 2 "Active" (A-bit)

Note that bit 0 is the most significant bit in the Flags Registry.

7. Performance Considerations

Each of the flags that are defined in this document may have performance implications. When using the loopback mechanism a copy of the data packet is sent back to the sender, thus generating more traffic than originally sent by the endpoints. Using active measurement with the active flag requires the use of synthetic (overhead) traffic.

Each of the mechanisms that use the flags above has a cost in terms of the network bandwidth, and may potentially load the node that analyzes the data. Therefore, it MUST be possible to use each of the mechanisms on a subset of the data traffic; an encapsulating node needs to be able to set the Loopback and Active flag selectively, in a way that considers the effect on the network performance. Similarly, transit and decapsulating nodes need to be able to selectively loop back packets with the Loopback flag, and to selectively export packets. Specifically, rate limiting can be enabled so as to ensure that the mechanisms are used at a rate that does not significantly affect the network bandwidth, and does not overload the receiving entity (or the source node in the case of loopback).

8. Security Considerations

The security considerations of IOAM in general are discussed in [I-D.ietf-ippm-ioam-data]. Specifically, an attacker may try to use the functionality that is defined in this document to attack the network.

An attacker may attempt to overload network devices by injecting synthetic packets that include an IOAM Trace Option with one or more of the flags defined in this document. Similarly, an on-path attacker may maliciously set one or more of the flags of transit packets.

- o Loopback flag: an attacker that sets this flag, either in synthetic packets or transit packet, can potentially cause an amplification, since each device along the path creates a copy of the data packet and sends it back to the source. The attacker can potentially leverage the loopback flag for a Distributed Denial of Service (DDoS) attack, as multiple devices send looped-back copies of a packet to a single source.

- o Active flag: the impact of synthetic packets with the active flag is no worse than synthetic data packets in which the Active flag is not set. By setting the active flag in en route packets an attacker can prevent these packets from reaching their destination, since the packet is terminated by the decapsulating device; however, note that an on-path attacker may achieve the same goal by changing the destination address of a packet. Another potential threat is amplification; if an attacker causes transit switches to replicate more packets than they are intended to replicate, either by setting the Active flag or by sending synthetic packets, then traffic is amplified, causing bandwidth degredation. As mentioned in Section 5, the specification of the replication mechanism is not within the scope of this document. A specification that defines the replication functionality should also address the security aspects of this mechanism.

In order to mitigate the attacks described above, as described in Section 7 it should be possible for IOAM-enabled devices to selectively apply the mechanisms that use the flags defined in this document to a subset of the traffic, and to limit the performance of synthetically generated packets to a configurable rate; specifically, network devices should be able to limit the rate of: (i) looped-back traffic (at transit nodes), (ii) replicated active packets (at encapsulating nodes), (iii) packets that are exported to a collector (from either encapsulating nodes or transit nodes), and (iv) synthetically generated packets (at encapsulating nodes).

Furthermore, as defined in Section 4, transit nodes that process a packet with the Loopback flag only add a single data field, and truncate any payload that follows the IOAM option(s), thus significanly limiting the possible impact of an amplification attack.

IOAM is assumed to be deployed in a restricted administrative domain, thus limiting the scope of the threats above and their affect. This is a fundamental assumtion with respect to the security aspects of IOAM, as further discussed in [I-D.ietf-ippm-ioam-data].

9. References

9.1. 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-10 (work in progress), July 2020.

[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>.

9.2. Informative References

[I-D.ietf-ippm-ioam-ipv6-options]
Bhandari, S., Brockners, F., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Kfir, A., Gafni, B., Lapukhov, P., Spiegel, M., Krishnan, S., Asati, R., and M. Smith, "In-situ OAM IPv6 Options", draft-ietf-ippm-ioam-ipv6-options-03 (work in progress), September 2020.

[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-04 (work in progress), June 2020.

[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-03 (work in progress), March 2020.

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

Authors' Addresses

Tal Mizrahi
Huawei Smart Platforms iLab
Israel

Email: tal.mizrahi.phd@gmail.com

Frank Brockners
Cisco Systems, Inc.
Hansaallee 249, 3rd Floor
DUESSELDORF, NORDRHEIN-WESTFALEN 40549
Germany

Email: fbrockne@cisco.com

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

Email: shwethab@cisco.com

Ramesh Sivakolundu
Cisco Systems, Inc.
170 West Tasman Dr.
SAN JOSE, CA 95134
U.S.A.

Email: sramesh@cisco.com

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

Email: cpignata@cisco.com

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

Email: avivk@mellanox.com

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

Email: gbarak@mellanox.com

Mickey Spiegel
Barefoot Networks
4750 Patrick Henry Drive
Santa Clara, CA 95054
US

Email: mspiegel@barefootnetworks.com

Jennifer Lemon
Broadcom
270 Innovation Drive
San Jose, CA 95134
US

Email: jennifer.lemon@broadcom.com

ippm
Internet-Draft
Intended status: Standards Track
Expires: May 5, 2021

S. Bhandari
Thoughtspot
F. Brockners
C. Pignataro
Cisco
H. Gredler
RtBrick Inc.
J. Leddy
Comcast
S. Youell
JMPC
T. Mizrahi
Huawei Network.IO Innovation Lab
A. Kfir
B. Gafni
Mellanox Technologies, Inc.
P. Lapukhov
Facebook
M. Spiegel
Barefoot Networks, an Intel company
S. Krishnan
Kaloom
R. Asati
Cisco
M. Smith
November 1, 2020

In-situ OAM IPv6 Options
draft-ietf-ippm-ioam-ipv6-options-04

Abstract

In-situ Operations, Administration, and Maintenance (IOAM) records operational and telemetry information in the packet while the packet traverses a path between two points in the network. This document outlines how IOAM data fields are encapsulated in IPv6.

Status of This Memo

Table of Contents

1. Introduction

In-situ Operations, Administration, and Maintenance (IOAM) records operational and telemetry information in the packet while the packet traverses a path between two points in the network. This document outlines how IOAM data fields are encapsulated in the IPv6 [RFC8200] and discusses deployment options for networks that use IPv6-encapsulated IOAM data fields. These options have distinct deployment considerations; for example, the IOAM domain can either be between hosts, or be between IOAM encapsulating and decapsulating network nodes that forward traffic, such as routers.

2. Conventions

2.1. Requirements Language

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

2.2. Abbreviations

Abbreviations used in this document:

E2E: Edge-to-Edge

IOAM: In-situ Operations, Administration, and Maintenance

ION: IOAM Overlay Network

OAM: Operations, Administration, and Maintenance

POT: Proof of Transit

3. In-situ OAM Metadata Transport in IPv6

In-situ OAM in IPv6 is used to enhance diagnostics of IPv6 networks. It complements other mechanisms designed to enhance diagnostics of IPv6 networks, such as the IPv6 Performance and Diagnostic Metrics Destination Option described in [RFC8250].

IOAM data fields can be encapsulated in "option data" fields using two types of extension headers in IPv6 packets - either Hop-by-Hop Options header or Destination options header. Deployments select one of these extension header types depending on how IOAM is used, as described in section 4 of [I-D.ietf-ippm-ioam-data]. Multiple options with the same Option Type MAY appear in the same Hop-by-Hop Options or Destination Options header, with distinct content.

In order for IOAM to work in IPv6 networks, IOAM MUST be explicitly enabled per interface on every node within the IOAM domain. Unless a particular interface is explicitly enabled (i.e., explicitly configured) for IOAM, a router MUST drop packets that contain extension headers carrying IOAM data-fields. This is the default behavior and is independent of whether the Hop-by-Hop options or Destination options are used to encode the IOAM data. This ensures that IOAM data does not unintentionally get forwarded outside the IOAM domain.

An IPv6 packet carrying IOAM data in an Extension header can have other extension headers, compliant with [RFC8200].

IPv6 Hop-by-Hop and Destination Option format for carrying in-situ OAM data fields:

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Option Type  |  Opt Data Len |   Reserved    |   IOAM Type   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|                                                               |  |
.                                                               .  I
.                                                               .  O
.                                                               .  A
.                                                               .  M
.                                                               .  .
.                          Option Data                          .  O
.                                                               .  P
.                                                               .  T
.                                                               .  I
.                                                               .  O
.                                                               .  N
|                                                               |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
```

Option Type: 8-bit option type identifier as defined inSection 6.

Opt Data Len: 8-bit unsigned integer. Length of this option, in octets, not including the first 2 octets.

Reserved: 8-bit field MUST be set to zero upon transmission and ignored upon reception.

IOAM Type: 8-bit field as defined in section 7.2 in [I-D.ietf-ippm-ioam-data].

Option Data: Variable-length field. Option-Type-specific data.

In-situ OAM Options are inserted as Option data as follows:

1. Pre-allocated Trace Option: The in-situ OAM Preallocated Trace option defined in [I-D.ietf-ippm-ioam-data] is represented as an IPv6 option in Hop-by-Hop extension header:

   Option Type: 001xxxxx 8-bit identifier of the IOAM type of option. xxxxx=TBD.

   IOAM Type: IOAM Pre-allocated Trace Option Type.

2. Incremental Trace Option: The in-situ OAM Incremental Trace option defined in [I-D.ietf-ippm-ioam-data] is represented as an IPv6 option in Hop-by-Hop extension header:

   Option Type: 001xxxxx 8-bit identifier of the IOAM type of option. xxxxx=TBD.

   IOAM Type: IOAM Incremental Trace Option Type.

3. Proof of Transit Option: The in-situ OAM POT option defined in [I-D.ietf-ippm-ioam-data] is represented as an IPv6 option in Hop-by-Hop extension header:

   Option Type: 001xxxxx 8-bit identifier of the IOAM type of option. xxxxx=TBD.

   IOAM Type: IOAM POT Option Type.

4. Edge to Edge Option: The in-situ OAM E2E option defined in [I-D.ietf-ippm-ioam-data] is represented as an IPv6 option in Destination extension header:

   Option Type: 000xxxxx 8-bit identifier of the IOAM type of option. xxxxx=TBD.

   IOAM Type: IOAM E2E Option Type.

All the in-situ OAM IPv6 options defined here have alignment requirements. Specifically, they all require 4n alignment. This ensures that fields specified in [I-D.ietf-ippm-ioam-data] are aligned at a multiple-of-4 offset from the start of the Hop-by-Hop and Destination Options header. In addition, to maintain IPv6 extension header 8-octet alignment and avoid the need to add or remove padding at every hop, the Trace-Type for Incremental Trace Option in IPv6 MUST be selected such that the IOAM node data length is a multiple of 8-octets.

4. IOAM Deployment In IPv6 Networks

4.1. Considerations for IOAM deployment in IPv6 networks

IOAM deployments in IPv6 networks should take the following considerations and requirements into account:

C1 It is desirable that the addition of IOAM data fields neither changes the way routers forward packets nor the forwarding decisions the routers take. Packets with added OAM information should follow the same path within the domain that an identical packet without OAM information would follow, even in the presence of ECMP. Such behavior is particularly important for deployments where IOAM data fields are only added "on-demand", e.g., to provide further insights in case of undesired network behavior for certain flows. Implementations of IOAM SHOULD ensure that ECMP behavior for packets with and without IOAM data fields is the same.

C2 Given that IOAM data fields increase the total size of a packet, the size of a packet including the IOAM data could exceed the PMTU. In particular, the incremental trace IOAM Hop-by-Hop (HbH) Option, which is intended to support hardware implementations of IOAM, changes Option Data Length en-route. Operators of an IOAM domain SHOULD ensure that the addition of OAM information does not lead to fragmentation of the packet, e.g., by configuring the MTU of transit routers and switches to a sufficiently high value. Careful control of the MTU in a network is one of the reasons why IOAM is considered a domain-specific feature (see also [I-D.ietf-ippm-ioam-data]). In addition, the PMTU tolerance range in the IOAM domain should be identified (e.g., through configuration) and IOAM encapsulation operations and/or IOAM data field insertion (in case of incremental tracing) should not be performed if it exceeds the packet size beyond PMTU.

C3 Packets with IOAM data or associated ICMP errors, should not arrive at destinations that have no knowledge of IOAM. For exmample, if IOAM is used in in transit devices, misleading ICMP errors due to addition and/or presence of OAM data in a packet could confuse the host that sent the packet if it did not insert the OAM information.

C4 OAM data leaks can affect the forwarding behavior and state of network elements outside an IOAM domain. IOAM domains SHOULD provide a mechanism to prevent data leaks or be able to ensure that if a leak occurs, network elements outside the domain are not affected (i.e., they continue to process other valid packets).

C5 The source that inserts and leaks the IOAM data needs to be easy to identify for the purpose of troubleshooting, due to the high complexity of troubleshooting a source that inserted the IOAM data and did not remove it when the packet traversed across an Autonomous System (AS). Such a troubleshooting process might require coordination between multiple operators, complex configuration verification, packet capture analysis, etc.

C6 Compliance with [RFC8200] requires OAM data to be encapsulated instead of header/option insertion directly into in-flight packets using the original IPv6 header.

4.2. IOAM domains bounded by hosts

For deployments where the IOAM domain is bounded by hosts, hosts will perform the operation of IOAM data field encapsulation and decapsulation. IOAM data is carried in IPv6 packets as Hop-by-Hop or Destination options as specified in this document.

4.3. IOAM domains bounded by network devices

For deployments where the IOAM domain is bounded by network devices, network devices such as routers form the edge of an IOAM domain. Network devices will perform the operation of IOAM data field encapsulation and decapsulation.

4.4. Deployment options

This section lists out possible deployment options that can be employed to meet the requirements listed in Section 4.1.

4.4.1. IPv6-in-IPv6 encapsulation

The "IPv6-in-IPv6" approach preserves the original IP packet and add an IPv6 header including IOAM data fields in an extension header in front of it, to forward traffic within and across an IOAM domain. The overlay network formed by the additional IPv6 header with the IOAM data fields included in an extension header is referred to as IOAM Overlay Network (ION) in this document.

The following steps should be taken to perform an IPv6-in-IPv6 approach:

1. The source address of the outer IPv6 header is that of the IOAM encapsulating node. The destination address of the outer IPv6 header is the same as the inner IPv6 destination address, i.e., the destination address of the packet does not change.

2. To simplify debugging in case of leaked IOAM data fields, consider a new IOAM E2E destination option to identify the Source IOAM domain (AS, v6 prefix). Insert this option into the IOAM destination options EH attached to the outer IPv6 header. This additional information would allow for easy identification of an AS operator that is the source of packets with leaked IOAM information. Note that leaked packets with IOAM data fields would only occur in case a router would be misconfigured.

3. All the IOAM options are defined with type "00" - skip over this option and continue processing the header. Presence of these options must not cause packet drops in network elements that do not understand the option. In addition, [I-D.ietf-6man-hbh-header-handling] should be considered.

4.4.2. IP-in-IPv6 encapsulation with ULA

The "IP-in-IPv6 encapsulation with ULA" [RFC4193] approach can be used to apply IOAM to either an IPv6 or an IPv4 network. In addition, it fulfills requirement C4 (avoid leaks) by using ULA for the ION. Similar to the IPv6-in-IPv6 encapsulation approach above, the original IP packet is preserved. An IPv6 header including IOAM data fields in an extension header is added in front of it, to forward traffic within and across the IOAM domain. IPv6 addresses for the ION, i.e. the outer IPv6 addresses are assigned from the ULA space. Addressing and routing in the ION are to be configured so that the IP-in-IPv6 encapsulated packets follow the same path as the original, non-encapsulated packet would have taken. This would create an internal IPv6 forwarding topology using the IOAM domain's interior ULA address space which is parallel with the forwarding topology that exists with the non-IOAM address space (the topology and address space that would be followed by packets that do not have supplemental IOAM information). Establishment and maintenance of the parallel IOAM ULA forwarding topology could be automated, e.g., similar to how LDP [RFC5036] is used in MPLS to establish and maintain an LSP forwarding topology that is parallel to the network's IGP forwarding topology.

Transit across the ION could leverage the transit approach for traffic between BGP border routers, as described in [RFC1772], "A.2.3 Encapsulation". Assuming that the operational guidelines specified in Section 4 of [RFC4193] are properly followed, the probability of leaks in this approach will be almost close to zero. If the packets do leak through IOAM egress device misconfiguration or partial IOAM egress device failure, the packets' ULA destination address is invalid outside of the IOAM domain. There is no exterior destination to be reached, and the packets will be dropped when they encounter either a router external to the IOAM domain that has a packet filter that drops packets with ULA destinations, or a router that does not have a default route.

4.4.3. x-in-IPv6 Encapsulation that is used Independently

In some cases it is desirable to monitor a domain that uses an overlay network that is deployed independently of the need for IOAM, e.g., an overlay network that runs Geneve-in-IPv6, or VXLAN-in-IPv6. In this case IOAM can be encapsulated in as an extension header in the tunnel (outer) IPv6 header. Thus, the tunnel encapsulating node is also the IOAM encapsulating node, and the tunnel end point is also the IOAM decapsulating node.

5. Security Considerations

This document describes the encapsulation of IOAM data fields in IPv6. Security considerations of the specific IOAM data fields for each case (i.e., Trace, Proof of Transit, and E2E) are described and defined in [I-D.ietf-ippm-ioam-data].

As this document describes new options for IPv6, these are similar to the security considerations of [RFC8200] and the weakness documented in [RFC8250].

6. IANA Considerations

This draft requests the following IPv6 Option Type assignments from the Destination Options and Hop-by-Hop Options sub-registry of Internet Protocol Version 6 (IPv6) Parameters.

http://www.iana.org/assignments/ipv6-parameters/ipv6-parameters.xhtml#ipv6-parameters-2

Hex Value	Binary Value act	chg	rest	Description	Reference
TBD_1_0	00	0	TBD_1	IOAM	[This draft]
TBD_1_1	00	1	TBD_1	IOAM	[This draft]

7. Acknowledgements

The authors would like to thank Tom Herbert, Eric Vyncke, Nalini Elkins, Srihari Raghavan, Ranganathan T S, Karthik Babu Harichandra Babu, Akshaya Nadahalli, Stefano Previdi, Hemant Singh, Erik Nordmark, LJ Wobker, Mark Smith, Andrew Yourtchenko and Justin Iurman for the comments and advice. For the IPv6 encapsulation, this document leverages concepts described in [I-D.kitamura-ipv6-record-route]. The authors would like to acknowledge the work done by the author Hiroshi Kitamura and people involved in writing it.

8. References

8.1. Normative References

[I-D.ietf-ippm-ioam-data]
Brockners, F., Bhandari, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Mozes, D., Lapukhov, P., Chang, R., and d. daniel.bernier@bell.ca, "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data-01 (work in progress), October 2017.

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

[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-6man-hbh-header-handling]
Baker, F. and R. Bonica, "IPv6 Hop-by-Hop Options Extension Header", March 2016.

[I-D.kitamura-ipv6-record-route]
Kitamura, H., "Record Route for IPv6 (PR6) Hop-by-Hop Option Extension", draft-kitamura-ipv6-record-route-00 (work in progress), November 2000.

[RFC1772] Rekhter, Y. and P. Gross, "Application of the Border Gateway Protocol in the Internet", RFC 1772, DOI 10.17487/RFC1772, March 1995, <https://www.rfc-editor.org/info/rfc1772>.

[RFC4193] Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses", RFC 4193, DOI 10.17487/RFC4193, October 2005, <https://www.rfc-editor.org/info/rfc4193>.

[RFC5036] Andersson, L., Ed., Minei, I., Ed., and B. Thomas, Ed., "LDP Specification", RFC 5036, DOI 10.17487/RFC5036, October 2007, <https://www.rfc-editor.org/info/rfc5036>.

[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>.

[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>.

Authors' Addresses

Shwetha Bhandari
Thoughtspot
3rd Floor, Indiqube Orion, 24th Main Rd, Garden Layout, HSR Layout
Bangalore, KARNATAKA 560 102
India

Email: shwetha.bhandari@thoughtspot.com

Frank Brockners
Cisco Systems, Inc.
Kaiserswerther Str. 115,
RATINGEN, NORDRHEIN-WESTFALEN 40880
Germany

Email: fbrockne@cisco.com

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

Email: cpignata@cisco.com

Hannes Gredler
RtBrick Inc.

Email: hannes@rtbrick.com

John Leddy
Comcast

Email: John_Leddy@cable.comcast.com

Stephen Youell
JP Morgan Chase
25 Bank Street
London E14 5JP
United Kingdom

Email: stephen.youell@jpmorgan.com

Tal Mizrahi
Huawei Network.IO Innovation Lab
Israel

Email: tal.mizrahi.phd@gmail.com

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

Email: avivk@mellanox.com

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

Email: gbarak@mellanox.com

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

Email: petr@fb.com

Mickey Spiegel
Barefoot Networks, an Intel company
4750 Patrick Henry Drive
Santa Clara, CA 95054
US

Email: mickey.spiegel@intel.com

Suresh Krishnan
Kaloom

Email: suresh@kaloom.com

Rajiv Asati
Cisco Systems, Inc.
7200 Kit Creek Road
Research Triangle Park, NC 27709
US

Email: rajiva@cisco.com

Mark Smith
PO BOX 521
HEIDELBERG, VIC 3084
AU

Email: markzzzsmith+id@gmail.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 10, 2021
G. Mirsky
X. Min
ZTE Corp.
W. Luo
Ericsson
October 7, 2020

Simple Two-way Active Measurement Protocol (STAMP) Data Model
draft-ietf-ippm-stamp-yang-06

Abstract

This document specifies the data model for implementations of Session-Sender and Session-Reflector for Simple Two-way Active Measurement Protocol (STAMP) mode using YANG.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

The Simple Two-way Active Measurement Protocol (STAMP) [RFC8762] can be used to measure performance parameters of IP networks such as latency, jitter, and packet loss by sending test packets and monitoring their experience in the network. The STAMP protocol [RFC8762] in unauthenticated mode is on-wire compatible with STAMP Light, discussed in Appendix I [RFC5357]. The STAMP Light is known to have many implementations though no common management framework being defined, thus leaving some aspects of test packet processing to interpretation. As one of the goals of STAMP is to support these variations, this document presents their analysis; describes common STAMP and STAMP model while allowing for STAMP extensions in the future. This document defines the STAMP data model and specifies it formally, using the YANG data modeling language [RFC7950].

This version of the interfaces data model conforms to the Network Management Datastore Architecture (NMDA) defined in [RFC8342].

1.1. Conventions used in this document

1.1.1. Requirements Language

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

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

2. Scope, Model, and Applicability

The scope of this document includes a model of the STAMP as defined in [RFC8762].

```
     o-----------------------------------------------------------o
     |                       Config client                       |
     o-----------------------------------------------------------o
           ||                                            ||
           ||                 NETCONF/RESTCONF           ||
           ||                                            ||
 o-----------------------o                 o--------------------------o
 |     Config server     |                 |      Config server       |
 |                       |                 |                          |
 +-----------------------+                 +--------------------------+
 | STAMP Session-Sender  | <--- STAMP--->  | STAMP Session-Reflector  |
 +-----------------------+                 +--------------------------+
```

Figure 1: STAMP Reference Model

2.1. Data Model Parameters

This section describes containers within the STAMP data model.

2.1.1. STAMP-Sender

The stamp-session-sender container holds items that are related to the configuration of the stamp Session-Sender logical entity.

The stamp-session-sender-state container holds information about the state of the particular STAMP test session.

RPCs stamp-sender-start and stamp-sender-stop respectively start and stop the referenced session by the session-id of the STAMP.

2.1.1.1. Controls for Test Session and Performance Metric Calculation

The data model supports several scenarios for a STAMP Session-Sender to execute test sessions and calculate performance metrics:

> The test mode in which the test packets are sent unbound in time as defined by the parameter 'interval' in the stamp-session-sender container frequency is referred to as continuous mode.

Performance metrics in the continuous mode are calculated at a period defined by the parameter 'measurement-interval'.

The test mode that has a specific number of the test packets configured for the test session in the 'number-of-packets' parameter is referred to as a periodic mode. The STAMP-Sender MAY repeat the test session with the same parameters. The 'repeat' parameter defines the number of tests and the 'repeat-interval' - the interval between the consecutive tests. The performance metrics are calculated after each test session when the interval defined by the 'session-timeout' expires.

2.1.2. STAMP-Reflector

The stamp-session-reflector container holds items that are related to the configuration of the STAMP Session-Reflector logical entity.

The stamp-session-refl-state container holds Session-Reflector state data for the particular STAMP test session.

3. Data Model

Creating STAMP data model presents a number of challenges and among them is the identification of a test-session at Session-Reflector. A Session-Reflector MAY require only as little as its IP and UDP port number in received STAMP-Test packet to spawn new test session. More so, to test processing of Class-of-Service along the same route in Equal Cost Multi-Path environment Session-Sender may perform STAMP test sessions concurrently using the same source IP address, source UDP port number, destination IP address, and destination UDP port number. Thus the only parameter that can be used to differentiate these test sessions would be DSCP value. The DSCP field may get re-marked along the path, and without the use of [RFC7750] that will go undetected, but by using five-tuple instead of four-tuple as a key, we can ensure that STAMP test packets that are considered as different test sessions follow the same path even in ECMP environments.

3.1. Tree Diagrams

This section presents a simplified graphical representation of the STAMP data model using a YANG tree diagram [RFC8340].

```
module: ietf-stamp
    +--rw stamp
    |  +--rw stamp-session-sender {session-sender}?
    |  |  +--rw sender-enable?   boolean
    |  |  +--rw test-session* [session-id]
    |  |     +--rw session-id                        uint32
    |  |     +--rw test-session-enable?              boolean
    |  |     +--rw number-of-packets?                union
    |  |     +--rw packet-padding-size?              uint32
    |  |     +--rw interval?                         uint32
    |  |     +--rw session-timeout?                  uint32
    |  |     +--rw measurement-interval?             uint32
    |  |     +--rw repeat?                           union
    |  |     +--rw repeat-interval?                  uint32
    |  |     +--rw dscp-value?                       inet:dscp
    |  |     +--rw test-session-reflector-mode?   session-reflector-mode
    |  |     +--rw sender-ip                         inet:ip-address
    |  |     +--rw sender-udp-port                   inet:port-number
    |  |     +--rw reflector-ip                      inet:ip-address
    |  |     +--rw reflector-udp-port?               inet:port-number
    |  |     +--rw sender-timestamp-format?          timestamp-format
    |  |     +--rw security! {stamp-security}?
    |  |     |  +--rw key-chain?   kc:key-chain-ref
    |  |     +--rw first-percentile?                 percentile
    |  |     +--rw second-percentile?                percentile
    |  |     +--rw third-percentile?                 percentile
    |  +--rw stamp-session-reflector {session-reflector}?
    |     +--rw reflector-enable?       boolean
    |     +--rw ref-wait?               uint32
    |     +--rw reflector-mode-state?   session-reflector-mode
    |     +--rw test-session* [session-id]
    |        +--rw session-id                       uint32
    |        +--rw dscp-handling-mode?              session-dscp-mode
    |        +--rw dscp-value?                      inet:dscp
    |        +--rw sender-ip?                       union
    |        +--rw sender-udp-port?                 union
    |        +--rw reflector-ip?                    union
    |        +--rw reflector-udp-port?              inet:port-number
    |        +--rw reflector-timestamp-format?   timestamp-format
    |        +--rw security! {stamp-security}?
    |           +--rw key-chain?   kc:key-chain-ref
```

Figure 2: STAMP Configuration Tree Diagram

```
  module: ietf-stamp
      +--ro stamp-state
```

```
     +--ro stamp-session-sender-state {session-sender}?
     |  +--ro test-session-state* [session-id]
     |     +--ro session-id              uint32
     |     +--ro sender-session-state?   enumeration
     |     +--ro current-stats
     |     |  +--ro start-time                     yang:date-and-time
     |     |  +--ro packet-padding-size?          uint32
     |     |  +--ro interval?                     uint32
     |     |  +--ro duplicate-packets?            uint32
     |     |  +--ro reordered-packets?            uint32
     |     |  +--ro sender-timestamp-format?      timestamp-format
     |     |  +--ro reflector-timestamp-format?   timestamp-format
     |     |  +--ro dscp?                         inet:dscp
     |     |  +--ro two-way-delay
     |     |  |  +--ro delay
     |     |  |  |  +--ro min?   yang:gauge64
     |     |  |  |  +--ro max?   yang:gauge64
     |     |  |  |  +--ro avg?   yang:gauge64
     |     |  |  +--ro delay-variation
     |     |  |     +--ro min?   yang:gauge32
     |     |  |     +--ro max?   yang:gauge32
     |     |  |     +--ro avg?   yang:gauge32
     |     |  +--ro one-way-delay-far-end
     |     |  |  +--ro delay
     |     |  |  |  +--ro min?   yang:gauge64
     |     |  |  |  +--ro max?   yang:gauge64
     |     |  |  |  +--ro avg?   yang:gauge64
     |     |  |  +--ro delay-variation
     |     |  |     +--ro min?   yang:gauge32
     |     |  |     +--ro max?   yang:gauge32
     |     |  |     +--ro avg?   yang:gauge32
     |     |  +--ro one-way-delay-near-end
     |     |  |  +--ro delay
     |     |  |  |  +--ro min?   yang:gauge64
     |     |  |  |  +--ro max?   yang:gauge64
     |     |  |  |  +--ro avg?   yang:gauge64
     |     |  |  +--ro delay-variation
     |     |  |     +--ro min?   yang:gauge32
     |     |  |     +--ro max?   yang:gauge32
     |     |  |     +--ro avg?   yang:gauge32
     |     |  +--ro low-percentile
     |     |  |  +--ro delay-percentile
     |     |  |  |  +--ro rtt-delay?        yang:gauge64
     |     |  |  |  +--ro near-end-delay?   yang:gauge64
     |     |  |  |  +--ro far-end-delay?    yang:gauge64
     |     |  |  +--ro delay-variation-percentile
     |     |  |     +--ro rtt-delay-variation?        yang:gauge32
     |     |  |     +--ro near-end-delay-variation?   yang:gauge32
```

```
         |     |  |     +--ro far-end-delay-variation?    yang:gauge32
         |     |  +--ro mid-percentile
         |     |  |  +--ro delay-percentile
         |     |  |  |  +--ro rtt-delay?        yang:gauge64
         |     |  |  |  +--ro near-end-delay?   yang:gauge64
         |     |  |  |  +--ro far-end-delay?    yang:gauge64
         |     |  |  +--ro delay-variation-percentile
         |     |  |     +--ro rtt-delay-variation?        yang:gauge32
         |     |  |     +--ro near-end-delay-variation?   yang:gauge32
         |     |  |     +--ro far-end-delay-variation?    yang:gauge32
         |     |  +--ro high-percentile
         |     |  |  +--ro delay-percentile
         |     |  |  |  +--ro rtt-delay?        yang:gauge64
         |     |  |  |  +--ro near-end-delay?   yang:gauge64
         |     |  |  |  +--ro far-end-delay?    yang:gauge64
         |     |  |  +--ro delay-variation-percentile
         |     |  |     +--ro rtt-delay-variation?        yang:gauge32
         |     |  |     +--ro near-end-delay-variation?   yang:gauge32
         |     |  |     +--ro far-end-delay-variation?    yang:gauge32
         |     |  +--ro two-way-loss
         |     |  |  +--ro loss-count?         int32
         |     |  |  +--ro loss-ratio?         percentage
         |     |  |  +--ro loss-burst-max?     int32
         |     |  |  +--ro loss-burst-min?     int32
         |     |  |  +--ro loss-burst-count?   int32
         |     |  +--ro one-way-loss-far-end
         |     |  |  +--ro loss-count?         int32
         |     |  |  +--ro loss-ratio?         percentage
         |     |  |  +--ro loss-burst-max?     int32
         |     |  |  +--ro loss-burst-min?     int32
         |     |  |  +--ro loss-burst-count?   int32
         |     |  +--ro one-way-loss-near-end
         |     |  |  +--ro loss-count?         int32
         |     |  |  +--ro loss-ratio?         percentage
         |     |  |  +--ro loss-burst-max?     int32
         |     |  |  +--ro loss-burst-min?     int32
         |     |  |  +--ro loss-burst-count?   int32
         |     |  +--ro sender-ip                     inet:ip-address
         |     |  +--ro sender-udp-port               inet:port-number
         |     |  +--ro reflector-ip                  inet:ip-address
         |     |  +--ro reflector-udp-port?           inet:port-number
         |     |  +--ro sent-packets?                 uint32
         |     |  +--ro rcv-packets?                  uint32
         |     |  +--ro sent-packets-error?           uint32
         |     |  +--ro rcv-packets-error?            uint32
         |     |  +--ro last-sent-seq?                uint32
         |     |  +--ro last-rcv-seq?                 uint32
         |     +--ro history-stats* [session-id]
```

```
     |         +--ro session-id                     uint32
     |         +--ro end-time                       yang:date-and-time
     |         +--ro packet-padding-size?           uint32
     |         +--ro interval?                      uint32
     |         +--ro duplicate-packets?             uint32
     |         +--ro reordered-packets?             uint32
     |         +--ro sender-timestamp-format?       timestamp-format
     |         +--ro reflector-timestamp-format?    timestamp-format
     |         +--ro dscp?                          inet:dscp
     |         +--ro two-way-delay
     |         |  +--ro delay
     |         |  |  +--ro min?   yang:gauge64
     |         |  |  +--ro max?   yang:gauge64
     |         |  |  +--ro avg?   yang:gauge64
     |         |  +--ro delay-variation
     |         |     +--ro min?   yang:gauge32
     |         |     +--ro max?   yang:gauge32
     |         |     +--ro avg?   yang:gauge32
     |         +--ro one-way-delay-far-end
     |         |  +--ro delay
     |         |  |  +--ro min?   yang:gauge64
     |         |  |  +--ro max?   yang:gauge64
     |         |  |  +--ro avg?   yang:gauge64
     |         |  +--ro delay-variation
     |         |     +--ro min?   yang:gauge32
     |         |     +--ro max?   yang:gauge32
     |         |     +--ro avg?   yang:gauge32
     |         +--ro one-way-delay-near-end
     |         |  +--ro delay
     |         |  |  +--ro min?   yang:gauge64
     |         |  |  +--ro max?   yang:gauge64
     |         |  |  +--ro avg?   yang:gauge64
     |         |  +--ro delay-variation
     |         |     +--ro min?   yang:gauge32
     |         |     +--ro max?   yang:gauge32
     |         |     +--ro avg?   yang:gauge32
     |         +--ro low-percentile
     |         |  +--ro delay-percentile
     |         |  |  +--ro rtt-delay?        yang:gauge64
     |         |  |  +--ro near-end-delay?   yang:gauge64
     |         |  |  +--ro far-end-delay?    yang:gauge64
     |         |  +--ro delay-variation-percentile
     |         |     +--ro rtt-delay-variation?        yang:gauge32
     |         |     +--ro near-end-delay-variation?   yang:gauge32
     |         |     +--ro far-end-delay-variation?    yang:gauge32
     |         +--ro mid-percentile
     |         |  +--ro delay-percentile
     |         |  |  +--ro rtt-delay?        yang:gauge64
```

```
         |        |  |  +--ro near-end-delay?   yang:gauge64
         |        |  |  +--ro far-end-delay?    yang:gauge64
         |        |  +--ro delay-variation-percentile
         |        |     +--ro rtt-delay-variation?        yang:gauge32
         |        |     +--ro near-end-delay-variation?   yang:gauge32
         |        |     +--ro far-end-delay-variation?    yang:gauge32
         |        +--ro high-percentile
         |        |  +--ro delay-percentile
         |        |  |  +--ro rtt-delay?        yang:gauge64
         |        |  |  +--ro near-end-delay?   yang:gauge64
         |        |  |  +--ro far-end-delay?    yang:gauge64
         |        |  +--ro delay-variation-percentile
         |        |     +--ro rtt-delay-variation?        yang:gauge32
         |        |     +--ro near-end-delay-variation?   yang:gauge32
         |        |     +--ro far-end-delay-variation?    yang:gauge32
         |        +--ro two-way-loss
         |        |  +--ro loss-count?         int32
         |        |  +--ro loss-ratio?         percentage
         |        |  +--ro loss-burst-max?     int32
         |        |  +--ro loss-burst-min?     int32
         |        |  +--ro loss-burst-count?   int32
         |        +--ro one-way-loss-far-end
         |        |  +--ro loss-count?         int32
         |        |  +--ro loss-ratio?         percentage
         |        |  +--ro loss-burst-max?     int32
         |        |  +--ro loss-burst-min?     int32
         |        |  +--ro loss-burst-count?   int32
         |        +--ro one-way-loss-near-end
         |        |  +--ro loss-count?         int32
         |        |  +--ro loss-ratio?         percentage
         |        |  +--ro loss-burst-max?     int32
         |        |  +--ro loss-burst-min?     int32
         |        |  +--ro loss-burst-count?   int32
         |        +--ro sender-ip                     inet:ip-address
         |        +--ro sender-udp-port               inet:port-number
         |        +--ro reflector-ip                  inet:ip-address
         |        +--ro reflector-udp-port?           inet:port-number
         |        +--ro sent-packets?                 uint32
         |        +--ro rcv-packets?                  uint32
         |        +--ro sent-packets-error?           uint32
         |        +--ro rcv-packets-error?            uint32
         |        +--ro last-sent-seq?                uint32
         |        +--ro last-rcv-seq?                 uint32
         +--ro stamp-session-refl-state {session-reflector}?
            +--ro reflector-light-admin-status?   boolean
            +--ro test-session-state* [session-id]
               +--ro session-id                    uint32
               +--ro reflector-timestamp-format?   timestamp-format
```

```
             +--ro sender-ip                    inet:ip-address
             +--ro sender-udp-port              inet:port-number
             +--ro reflector-ip                 inet:ip-address
             +--ro reflector-udp-port?          inet:port-number
             +--ro sent-packets?                uint32
             +--ro rcv-packets?                 uint32
             +--ro sent-packets-error?          uint32
             +--ro rcv-packets-error?           uint32
             +--ro last-sent-seq?               uint32
             +--ro last-rcv-seq?                uint32
```

Figure 3: STAMP State Tree Diagram

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

Figure 4: STAMP RPC Tree Diagram

3.2. YANG Module

```
   <CODE BEGINS> file "ietf-stamp@2020-10-07.yang"

   module ietf-stamp {
     yang-version 1.1;
     namespace "urn:ietf:params:xml:ns:yang:ietf-stamp";
     //namespace need to be assigned by IANA
     prefix "ietf-stamp";

   import ietf-inet-types {
   prefix inet;
   reference "RFC 6991: Common YANG Types.";
   }
   import ietf-yang-types {
        prefix yang;
   reference "RFC 6991: Common YANG Types.";
   }
   import ietf-key-chain {
   prefix kc;
   reference "RFC 8177: YANG Data Model for Key Chains.";
   }
```

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

     contact
       "WG Web: http://tools.ietf.org/wg/ippm/
        WG List: ippm@ietf.org

        Editor: Greg Mirsky
             gregimirsky@gmail.com
        Editor: Xiao Min
             xiao.min2@zte.com.cn
        Editor: Wei S Luo
             wei.s.luo@ericsson.com";

     description
       "This YANG module specifies a vendor-independent model
        for the Simple Two-way Active Measurement Protocol (STAMP).

        The data model covers two STAMP logical entities -
        Session-Sender and Session-Reflector; characteristics
        of the STAMP test session, as well as measured and
        calculated performance metrics.

           Copyright (c) 2020 IETF Trust and the persons identified as
           the document authors.  All rights reserved.
           Redistribution and use in source and binary forms, with or
           without modification, is permitted pursuant to, and subject
           to the license terms contained in, the Simplified BSD
           License set forth in Section 4.c of the IETF Trust's Legal
           Provisions Relating to IETF Documents
           (http://trustee.ietf.org/license-info).

           This version of this YANG module is part of RFC XXXX; see
           the RFC itself for full legal notices.";

     revision "2020-10-07" {
       description
         "Initial Revision. Base STAMP specification is covered";
       reference
         "RFC XXXX: STAMP YANG Data Model.";
     }

   /*
    * Typedefs
    */
   typedef session-reflector-mode {
     type enumeration {
       enum stateful {
```

```
         description
           "When the Session-Reflector is stateful,
           i.e. is aware of STAMP-Test session state.";
         }
         enum stateless {
           description
             "When the Session-Reflector is stateless,
             i.e. is not aware of the state of
             STAMP-Test session.";
         }
       }
       description "State of the Session-Reflector";
   }

   typedef session-dscp-mode {
     type enumeration {
       enum copy-received-value {
         description
           "Use DSCP value copied from received
           STAMP test packet of the test session.";
       }
       enum use-configured-value {
         description
           "Use DSCP value configured for this
           test session on the Session-Reflector.";
       }
     }
     description
       "DSCP handling mode by Session-Reflector.";
   }

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

   typedef percentage {
```

```
     type decimal64 {
       fraction-digits 5;
     }
     description "Percentage";
   }

   typedef percentile {
     type decimal64 {
       fraction-digits 5;
     }
     description
       "Percentile is a measure used in statistics
       indicating the value below which a given
       percentage of observations in a group of
       observations fall.";
   }


   /*
    * Feature definitions.
    */
   feature session-sender {
     description
       "This feature relates to the device functions as the
       STAMP Session-Sender";
   }

   feature session-reflector {
     description
       "This feature relates to the device functions as the
       STAMP Session-Reflector";
   }

   feature stamp-security {
     description "Secure STAMP supported";
   }

   /*
    * Reusable node groups
    */

   grouping maintenance-statistics {
     description "Maintenance statistics grouping";
     leaf sent-packets {
       type uint32;
       description "Packets sent";
     }
     leaf rcv-packets {
```

```
       type uint32;
       description "Packets received";
     }
     leaf sent-packets-error {
       type uint32;
       description "Packets sent error";
     }
     leaf rcv-packets-error {
       type uint32;
       description "Packets received error";
     }
     leaf last-sent-seq {
       type uint32;
       description "Last sent sequence number";
     }
     leaf last-rcv-seq {
       type uint32;
       description "Last received sequence number";
     }
   }

   grouping test-session-statistics {
     description
       "Performance metrics calculated for
       a STAMP test session.";

     leaf packet-padding-size {
       type uint32;
       description
         "Size of the Packet Padding. Suggested to run
         Path MTU Discovery to avoid packet fragmentation
         in IPv4 and packet blackholing in IPv6";
     }

     leaf interval  {
       type uint32;
       units microseconds;
       description
               "Time interval between transmission of two
         consecutive packets in the test session";
       }

       leaf duplicate-packets  {
         type uint32;
         description "Duplicate packets";
       }

       leaf reordered-packets  {
```

```
          type uint32;
          description "Reordered packets";
        }

        leaf sender-timestamp-format {
          type timestamp-format;
          description "Sender Timestamp format";
        }

        leaf reflector-timestamp-format {
          type timestamp-format;
          description "Reflector Timestamp format";
        }

        leaf dscp {
          type inet:dscp;
          description
            "The DSCP value that was placed in the header of
            STAMP UDP test packets by the Session-Sender.";
        }

        container two-way-delay {
          description
            "two way delay result of the test session";
          uses delay-statistics;
        }

        container one-way-delay-far-end {
          description
            "one way delay far-end of the test session";
          uses delay-statistics;
        }

        container one-way-delay-near-end {
          description
            "one way delay near-end of the test session";
          uses delay-statistics;
        }

            container low-percentile {
              when "/stamp/stamp-session-sender/"
                +"test-session[session-id]/"
                            +"first-percentile != '0.00'" {
                   description
                  "Only valid if the
                  the first-percentile is not NULL";
              }
              description
```

```
            "Low percentile report";
          uses time-percentile-report;
        }

             container mid-percentile {
          when "/stamp/stamp-session-sender/"
            +"test-session[session-id]/"
            +"second-percentile != '0.00'" {
            description
              "Only valid if the
              the first-percentile is not NULL";
          }
          description
            "Mid percentile report";
          uses time-percentile-report;
        }

        container high-percentile {
          when "/stamp/stamp-session-sender/"
            +"test-session[session-id]/"
            +"third-percentile != '0.00'" {
            description
              "Only valid if the
              the first-percentile is not NULL";
          }
          description
            "High percentile report";
          uses time-percentile-report;
        }

        container two-way-loss {
          description
            "two way loss count and ratio result of
            the test session";
          uses packet-loss-statistics;
        }

        container one-way-loss-far-end {
          when "/stamp/stamp-session-sender/"
            +"test-session[session-id]/"
            +"test-session-reflector-mode = 'stateful'" {
            description
              "One-way statistic is only valid if the
              session-reflector is in stateful mode.";
          }
          description
            "one way loss count and ratio far-end of
            the test session";
```

```
            uses packet-loss-statistics;
          }

          container one-way-loss-near-end {
            when "/stamp/stamp-session-sender/"
              +"test-session[session-id]/"
              +"test-session-reflector-mode = 'stateful'" {
              description
                "One-way statistic is only valid if the
                session-reflector is in stateful mode.";
            }
            description
              "one way loss count and ratio near-end of
              the test session";
            uses packet-loss-statistics;
          }
          uses session-parameters;
          uses maintenance-statistics;
  }

  grouping stamp-session-percentile {
    description "Percentile grouping";
    leaf first-percentile {
      type percentile;
      default 95.00;
      description
        "First percentile to report";
    }
    leaf second-percentile {
      type percentile;
      default 99.00;
      description
        "Second percentile to report";
    }
    leaf third-percentile {
      type percentile;
      default 99.90;
      description
        "Third percentile to report";
    }
  }

  grouping delay-statistics {
    description "Delay statistics grouping";
    container delay {
    description "Packets transmitted delay";
      leaf min {
        type yang:gauge64;
```

```
          units nanoseconds;
          description
            "Min of Packets transmitted delay";
        }
        leaf max {
          type yang:gauge64;
          units nanoseconds;
          description
            "Max of Packets transmitted delay";
        }
        leaf avg {
        type yang:gauge64;
        units nanoseconds;
        description
          "Avg of Packets transmitted delay";
        }
      }

      container delay-variation {
        description
          "Packets transmitted delay variation";
        leaf min {
          type yang:gauge32;
          units nanoseconds;
          description
            "Min of Packets transmitted
            delay variation";
        }
        leaf max {
          type yang:gauge32;
          units nanoseconds;
          description
            "Max of Packets transmitted
            delay variation";
        }
        leaf avg {
          type yang:gauge32;
          units nanoseconds;
                    description
            "Avg of Packets transmitted
            delay variation";
        }
      }
    }

    grouping time-percentile-report {
      description "Delay percentile report grouping";
      container delay-percentile {
```

```
        description
          "Report round-trip, near- and far-end delay";
        leaf rtt-delay {
          type yang:gauge64;
          units nanoseconds;
          description
            "Percentile of round-trip delay";
        }
        leaf near-end-delay {
          type yang:gauge64;
            units nanoseconds;
            description
              "Percentile of near-end delay";
        }
        leaf far-end-delay {
          type yang:gauge64;
          units nanoseconds;
          description
            "Percentile of far-end delay";
        }
      }

      container delay-variation-percentile {
        description
          "Report round-trip, near- and far-end delay variation";
        leaf rtt-delay-variation {
          type yang:gauge32;
          units nanoseconds;
          description
            "Percentile of round-trip delay-variation";
        }
        leaf near-end-delay-variation {
          type yang:gauge32;
          units nanoseconds;
          description
            "Percentile of near-end delay variation";
        }
        leaf far-end-delay-variation {
          type yang:gauge32;
          units nanoseconds;
          description
            "Percentile of far-end delay-variation";
        }
      }
    }

    grouping packet-loss-statistics {
      description
```

```
        "Grouping for Packet Loss statistics";
      leaf loss-count {
        type int32;
        description
          "Number of lost packets
          during the test interval.";
      }
      leaf loss-ratio {
        type percentage;
        description
          "Ratio of packets lost to packets
          sent during the test interval.";
      }
      leaf loss-burst-max {
        type int32;
        description
          "Maximum number of consecutively
          lost packets during the test interval.";
      }
      leaf loss-burst-min {
        type int32;
        description
          "Minimum number of consecutively
                  lost packets during the test interval.";
      }
      leaf loss-burst-count {
        type int32;
        description
        "Number of occasions with packet
        loss during the test interval.";
      }
    }

    grouping session-parameters {
      description
        "Parameters Session-Sender";
      leaf sender-ip {
        type inet:ip-address;
        mandatory true;
        description "Sender IP address";
      }
      leaf sender-udp-port {
        type inet:port-number {
          range "49152..65535";
        }
        mandatory true;
        description "Sender UDP port number";
      }
```

```
        leaf reflector-ip {
        type inet:ip-address;
        mandatory true;
        description "Reflector IP address";
      }
      leaf reflector-udp-port {
        type inet:port-number{
          range "862 | 1024..49151 | 49152..65535";
        }
        default 862;
        description "Reflector UDP port number";
      }
    }

    grouping session-security {
      description
        "Grouping for STAMP security and related parameters";
      container security {
        if-feature stamp-security;
        presence "Enables secure STAMP";
        description
          "Parameters for STAMP authentication";
        leaf key-chain {
          type kc:key-chain-ref;
          description "Name of key-chain";
        }
      }
    }

    /*
     * Configuration Data
     */
    container stamp {
      description
        "Top level container for STAMP configuration";

      container stamp-session-sender {
        if-feature session-sender;
        description "STAMP Session-Sender container";

        leaf sender-enable {
          type boolean;
          default "true";
          description
            "Whether this network element is enabled to
            act as STAMP Session-Sender";
        }
```

```
       list test-session {
         key "session-id";
         unique "sender-ip sender-udp-port reflector-ip"
           +" reflector-udp-port dscp-value";
         description
           "This structure is a container of test session
           managed objects";

         leaf session-id {
           type uint32;
           description "Session ID";
         }

         leaf test-session-enable {
           type boolean;
           default "true";
           description
             "Whether this STAMP Test session is enabled";
         }

         leaf number-of-packets {
           type union {
             type uint32 {
               range 1..4294967294 {
               description
                 "The overall number of UDP test packet
                 to be transmitted by the sender for this
                 test session";
               }
             }
             type enumeration {
               enum forever {
                 description
                   "Indicates that the test session SHALL
                   be run *forever*.";
               }
             }
           }
           default 10;
           description
             "This value determines if the STAMP-Test session is
             bound by number of test packets or not.";
         }

         leaf packet-padding-size {
           type uint32;
           default 30;
           description
```

```
            "Size of the Packet Padding. Suggested to run
            Path MTU Discovery to avoid packet fragmentation in
            IPv4 and packet blackholing in IPv6";
         }

        leaf interval  {
          type uint32;
          units microseconds;
          description
            "Time interval between transmission of two
            consecutive packets in the test session in
            microseconds";
        }

        leaf session-timeout {
          when "../number-of-packets != 'forever'" {
            description
              "Test session timeout only valid if the
              test mode is periodic.";
          }
          type uint32;
          units "seconds";
          default 900;
          description
            "The timeout value for the Session-Sender to
            collect outstanding reflected packets.";
        }

        leaf measurement-interval {
          when "../number-of-packets = 'forever'" {
            description
              "Valid only when the test to run forever,
              i.e. continuously.";
          }
          type uint32;
          units "seconds";
          default 60;
          description
            "Interval to calculate performance metric when
            the test mode is 'continuous'.";
        }

        leaf repeat {
          type union {
            type uint32 {
              range 0..4294967294;
            }
            type enumeration {
```

```
              enum forever {
                description
                  "Indicates that the test session SHALL
                  be repeated *forever* using the
                  information in repeat-interval
                  parameter, and SHALL NOT decrement
                  the value.";
              }
            }
          }
          default 0;
          description
            "This value determines if the STAMP-Test session must
            be repeated. When a test session has completed, the
            repeat parameter is checked. The default value
            of 0 indicates that the session MUST NOT be repeated.
            If the repeat value is 1 through 4,294,967,294
            then the test session SHALL be repeated using the
            information in repeat-interval parameter.
            The implementation MUST decrement the value of repeat
            after determining a repeated session is expected.";
        }

        leaf repeat-interval {
          when "../repeat != '0'";
          type uint32;
          units seconds;
          default 0;
          description
            "This parameter determines the timing of repeated
            STAMP-Test sessions when repeat is more than 0.";
        }

        leaf dscp-value {
          type inet:dscp;
          default 0;
          description
            "DSCP value to be set in the test packet.";
        }

        leaf test-session-reflector-mode {
          type session-reflector-mode;
          default "stateless";
          description
            "The mode of STAMP-Reflector for the test session.";
        }

        uses session-parameters;
```

```
         leaf sender-timestamp-format {
           type timestamp-format;
           default ntp-format;
           description "Sender Timestamp format";
         }
         uses session-security;
         uses stamp-session-percentile;
       }
     }

     container stamp-session-reflector {
       if-feature session-reflector;
       description
         "STAMP Session-Reflector container";
       leaf reflector-enable {
         type boolean;
         default "true";
         description
           "Whether this network element is enabled to
           act as STAMP Session-Reflector";
       }

       leaf ref-wait {
         type uint32 {
           range 1..604800;
         }
         units seconds;
         default 900;
             description
           "REFWAIT(STAMP test session timeout in seconds),
           the default value is 900";
       }

       leaf reflector-mode-state {
         type session-reflector-mode;
               default stateless;
         description
           "The state of the mode of the STAMP
           Session-Reflector";
       }

       list test-session {
         key "session-id";
         unique "sender-ip sender-udp-port reflector-ip"
         +" reflector-udp-port";
         description
           "This structure is a container of test session
           managed objects";
```

```
         leaf session-id {
           type uint32;
           description "Session ID";
         }

         leaf dscp-handling-mode {
           type session-dscp-mode;
           default copy-received-value;
           description
             "Session-Reflector handling of DSCP:
             - use value copied from received STAMP-Test packet;
             - use value explicitly configured";
         }

         leaf dscp-value {
           when "../dscp-handling-mode = 'use-configured-value'";
           type inet:dscp;
           default 0;
           description
           "DSCP value to be set in the reflected packet
           if dscp-handling-mode is set to use-configured-value.";
         }

         leaf sender-ip {
           type union {
             type inet:ip-address;
             type enumeration {
               enum any {
                 description
                   "Indicates that the Session-Reflector
                   accepts STAMP test packets from
                   any Session-Sender";
               }
             }
           }
           default any;
           description
             "This value determines whether specific
             IPv4/IPv6 address of the Session-Sender
             or the wildcard, i.e. any address";
         }

         leaf sender-udp-port {
           type union {
             type inet:port-number {
               range "49152..65535";
             }
             type enumeration {
```

```
               enum any {
                 description
                   "Indicates that the Session-Reflector
                   accepts STAMP test packets from
                   any Session-Sender";
               }
             }
           }
           default any;
           description
             "This value determines whether specific
             port number of the Session-Sender
             or the wildcard, i.e. any";
         }

         leaf reflector-ip {
           type union {
             type inet:ip-address;
             type enumeration {
               enum any {
                 description
                   "Indicates that the Session-Reflector
                   accepts STAMP test packets on
                   any of its interfaces";
               }
             }
           }
           default any;
           description
             "This value determines whether specific
             IPv4/IPv6 address of the Session-Reflector
             or the wildcard, i.e. any address";
         }

         leaf reflector-udp-port {
           type inet:port-number{
             range "862 | 1024..49151 | 49152..65535";
           }
           default 862;
           description "Reflector UDP port number";
         }

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

```
        }
      }
    }

     /*
     * Operational state data nodes
     */
    container stamp-state {
      config false;
      description
        "Top level container for STAMP state data";

      container stamp-session-sender-state {
        if-feature session-sender;
        description
          "Session-Sender container for state data";
        list test-session-state{
          key "session-id";
          description
            "This structure is a container of test session
            managed objects";

          leaf session-id {
            type uint32;
            description "Session ID";
          }

          leaf sender-session-state {
            type enumeration {
              enum active {
                description "Test session is active";
              }
              enum ready {
                description "Test session is idle";
              }
            }
            description
              "State of the particular STAMP test
              session at the sender";
          }

          container current-stats {
            description
              "This container contains the results for the current
              Measurement Interval in a Measurement session ";
            leaf start-time {
              type yang:date-and-time;
              mandatory true;
```

```
            description
              "The time that the current Measurement Interval started";
          }

          uses test-session-statistics;

        }

        list history-stats {
          key session-id;
          description
            "This container contains the results for the history
            Measurement Interval in a Measurement session ";
          leaf session-id {
            type uint32;
            description
              "The identifier for the Measurement Interval
              within this session";
          }

          leaf end-time {
            type yang:date-and-time;
            mandatory true;
            description
              "The time that the Measurement Interval ended";
          }

          uses test-session-statistics;
        }
      }
    }

    container stamp-session-refl-state {
      if-feature session-reflector;
      description
        "STAMP Session-Reflector container for
        state data";
      leaf reflector-light-admin-status {
        type boolean;
        description
          "Whether this network element is enabled to
          act as STAMP Session-Reflector";
      }

      list test-session-state {
        key "session-id";
        description
          "This structure is a container of test session
```

```
           managed objects";

         leaf session-id {
           type uint32;
           description "Session ID";
         }

         leaf reflector-timestamp-format {
           type timestamp-format;
           description "Reflector Timestamp format";
         }
         uses session-parameters;
         uses maintenance-statistics;

       }
     }
   }

   rpc stamp-sender-start {
     description
       "start the configured sender session";
     input {
       leaf session-id {
         type uint32;
         mandatory true;
         description
           "The STAMP session to be started";
       }
     }
   }

   rpc stamp-sender-stop {
     description
       "stop the configured sender session";
     input {
       leaf session-id {
         type uint32;
         mandatory true;
         description
           "The session to be stopped";
       }
     }
   }
   }

    <CODE ENDS>
```

4. IANA Considerations

This document registers a URI in the IETF XML registry [RFC3688]. Following the format in [RFC3688], the following registration is requested to be made.

URI: urn:ietf:params:xml:ns:yang:ietf-stamp

Registrant Contact: The IPPM WG of the IETF.

XML: N/A, the requested URI is an XML namespace.

This document registers a YANG module in the YANG Module Names registry [RFC7950].

name: ietf-stamp

namespace: urn:ietf:params:xml:ns:yang:ietf-stamp

prefix: stamp

reference: RFC XXXX

5. Security Considerations

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8446].

The NETCONF access control model [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a pre-configured subset of all available NETCONF or RESTCONF protocol operations and content.

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

TBD

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

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

TBD

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

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

TBD

6. Acknowledgments

Authors recognize and appreciate valuable comments provided by Adrian Pan and Henrik Nydell.

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>.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <https://www.rfc-editor.org/info/rfc3688>.

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

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <https://www.rfc-editor.org/info/rfc6242>.

[RFC7750] Hedin, J., Mirsky, G., and S. Baillargeon, "Differentiated Service Code Point and Explicit Congestion Notification Monitoring in the Two-Way Active Measurement Protocol (TWAMP)", RFC 7750, DOI 10.17487/RFC7750, February 2016, <https://www.rfc-editor.org/info/rfc7750>.

[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>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

[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>.

[RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <https://www.rfc-editor.org/info/rfc8341>.

[RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <https://www.rfc-editor.org/info/rfc8342>.

[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>.

[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>.

7.2. Informative References

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <https://www.rfc-editor.org/info/rfc8340>.

Appendix A. Example of STAMP Session Configuration

Figure 5 shows a configuration example of a STAMP-Sender.


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


Figure 5: XML instance of STAMP Session-Sender configuration

```
<?xml version="1.0" encoding="utf-8"?>
<data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
 <stamp xmlns="urn:ietf:params:xml:ns:yang:ietf-stamp">
     <stamp-session-reflector>
       <session-enable>enable</session-enable>
       <ref-wait/> <!-- use default 900 seconds -->
       <!-- use default 'stateless' -->
       <reflector-mode-state/>
       <session-id></session-id>
       <!-- use default 'copy-received-value' -->
       <dscp-handling-mode/>
       <!-- not used because of dscp-hanling-mode
             being 'copy-received-value' -->
       <dscp-value/>
       <sender-ip/> <!-- use default 'any' -->
       <sender-udp-port/>  <!-- use default 'any' -->
       <reflector-ip/> <!-- use default 'any' -->
       <reflector-udp-port/>  <!-- use default 862 -->
       <reflector-timestamp-format/>
       <!-- No authentication -->
     </stamp-session-reflector>
  </stamp>
</data>
```


Figure 6: XML instance of STAMP Session-Reflector configuration

Authors' Addresses

Greg Mirsky
ZTE Corp.

Email: gregimirsky@gmail.com

Xiao Min
ZTE Corp.

Email: xiao.min2@zte.com.cn

Wei S Luo
Ericsson

Email: wei.s.luo@ericsson.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 6, 2021

Z. Li
China Mobile
M. Chen
Huawei
G. Mirsky
ZTE Corp.

November 02, 2020

Performance Measurement on LAG
draft-li-ippm-pm-on-lag-03

Abstract

This document defines extensions to One-way Active Measurement Protocol (OWAMP), Two-way Active Measurement Protocol (TWAMP), and Simple Two-Way Active Measurement Protocol (STAMP) to implement performance measurement on every member link of a Link Aggregation Group (LAG). With the measured metrics of each member links of a LAG, it enables operators to enforce performance metric based traffic steering policy among the member links.

Requirements Language

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

Status of This Memo

Table of Contents

1. Problem Statement

Link Aggregation Group (LAG), as defined in [IEEE802.1AX], provides mechanisms to combine multiple physical links into a single logical link. This logical link provides higher bandwidth and better resiliency, because if one of the physical member links fails, the aggregate logical link can continue to forward traffic over the remaining operational physical member links.

Normally, when forwarding traffic over a LAG, a hash based or the like mechanism is used to load balance the traffic among member links of the LAG. In some cases, the link delays of the member links are different because the member links are over different transport paths. To provide low delay service to time sensitive traffic, we have to know the link delay of each member link of a LAG and then steer traffic accordingly. This requires a solution that could measure the performance metrics of each member link of a LAG.

However, when using One-way Active Measurement Protocol (OWAMP) [RFC4656], Two-way Active Measurement Protocol (TWAMP) [RFC5357], or Simple Two-Way Active Measurement Protocol (STAMP) [RFC8762] to measure the performance of a LAG, the LAG is treated as a single logical link/path. The measured metrics reflect the performance of one member link or an average of some/all member links of the LAG.

In addition, for LAG, using passive or hybrid methods (like alternative marking[RFC8321] or iOAM [I-D.ietf-ippm-ioam-data]) can only monitor the link crossed by traffic. Means the measured metrics only reflect the performance of some member links or an average of some/all member links of the LAG as well. Therefore, in order to measure every link of a LAG, using active methods would be more appropriate.

This document defines extensions to OWAMP [RFC4656], TWAMP [RFC5357] or STAMP [RFC8762] to implement performance measurement on every member link of a LAG.

2. Micro Session on LAG

This document intends to address the scenario (e.g., Figure 1) where two hosts (A and B) are directly connected by a LAG (e.g., the LAG is consisted by three links). The purpose is to measure the performance of each link of the LAG.

```
+---+                        +---+
|   |------------------------|   |
| A |------------------------| B |
|   |------------------------|   |
+---+                        +---+
```

Figure 1: PM for LAG

To measure performance metrics of every member link of a LAG, multiple sessions (one session for each member link) need to be established between the two hosts that are connected by the LAG. These sessions are called micro sessions in the remainder of this document.

All micro sessions of a LAG share the same Sender Address, Receiver Address. As for the Sender Port and Receiver Port, the micro sessions may share the same Sender Port and Receiver Port pair, or each micro session is configured with different Sender Port and Receiver Port pair. But from simplifying operation point of view, the former is recommended.

In addition, with micro sessions, there needs a way to correlate a session with a member link. For example, when receives a Control or Test packet, the Server/Reflector/Receiver needs to know from which member link the packet is received, and then correlate the packet with a micro session. This is different from the existing OWAMP [RFC4656], TWAMP [RFC5357], or STAMP [RFC8762].

This document defines new command types to indicate that a session is a micro session, the details are described in Section 3 and 4 of this document. For a micro session, on receiving of a Control/Test packet, the receiver uses the receiving link to correlate the packet with a particular session. In addition, Test packets may need to carry the member link information for validation checking. For example, when a Session-Sender receives a Test packet, it may need to check whether the Test packet is from the expected member link.

3. Mirco OWAMP Session

This document assumes that the OWAMP Server and the OWAMP Receiver of an OWAMP micro session are at the same host.

3.1. Micro OWAMP-Control

To support micro OWAMP session, a new command, which is referred to as Request-OW-Micro-Session (TBD1), is defined in this document. The Request-OW-Micro-Session command is based on the OWAMP Request-Session command, and uses the message format as described in Section 3.5 of OWAMP [RFC4656]. Test session creation of micro OWAMP session follows the same procedure as defined in Section 3.5 of OWAMP [RFC4656] with the following additions:

When a OWAMP Server receives a Request-OW-Micro-Session command, if the Session is accepted, the OWAMP Server MUST build an association between the session and the member link from which the Request-Session message is received.

3.2. Micro OWAMP-Test

Micro OWAMP-Test reuses the OWAMP-Test packet format and procedures as defined in Section 4 of OWAMP [RFC4656] with the following additions:

The micro OWAMP Sender MUST send the micro OWAMP-Test packets over the member link with which the session is associated. When receives a Test packet, the micro OWAMP receiver MUST use the member link from which the Test packet is received to correlate the micro OWAMP session. If there is no such a session, the Test packet MUST be discarded.

4. Mirco TWAMP Session

As above, this document assumes that the TWAMP Server and the TWAMP Session-Reflector of a micro OWAMP session are at the same host.

4.1. Micro TWAMP-Control

To support micro TWAMP session, a new command, which is referred to as Request-TW-Micro-Session (TBD2), is defined in this document. The Request-TW-Micro-Session command is based on the TWAMP Request-Session command, and uses the message format as described in Section 3.5 of TWAMP [RFC5357]. Test session creation of micro TWAMP session follows the same procedure as defined in Section 3.5 of TWAMP [RFC5357] with the following additions:

When a micro TWAMP Server receives a Request-TW-Micro-Session command, if the micro TWAMP Session is accepted, the micro TWAMP Server MUST build an association between the session and the member link from which the Request-Session message is received.

4.2. Micro TWAMP-Test

The micro TWAMP-Test protocol is based on the TWAMP-Test protocol [RFC5357] with the following extensions.

4.2.1. Sender Behavior

In addition to inheriting the TWAMP sender behavior as defined Section 4.1 of [RFC5357], the micro TWAMP Session-Sender MUST send the micro TWAMP-Test packets over the member link with which the session is associated.

When sending Test packet, the micro TWAMP Session-Sender MUST put the Sender member link identifier that is associated with the micro TWAMP session in the Sender Member Link ID. If the Session-Sender knows the Reflector member link identifier, it MUST put it in the Reflector Member Link ID fields (see Figure 2 and Figure 3). Otherwise, the Reflector Member Link ID field MUST be set to zero.

The Sender member link identifier is used by the Session-Sender to check whether a reflected Test packet is received from the member link that associates to the correct micro TWAMP session. Therefore, it is carried in the Sender Member Link ID field of a Test packet and sent to the Session-Reflector. Then it will be sent back by the Session-Reflector with the reflected Test packet.

The Reflector member link identifier carried in the Reflector Member Link ID field is used by the Session-Receiver to check whether a Test packet is received from the member link that associates to the correct micro TWAMP session. Means that the Session-Sender has to learns the Reflector member link identifier. Once the Session-Sender learns the Reflector member link identifier, it MUST put the identifier in the Reflector Member Link ID field (see Figure 2 or Figure 3) of the Test packets that will be sent to the Session-Reflector. The Reflector member link identifier can be obtained from pre-configuration or learned through control plane or data plane (e.g., learned from a reflected Test packet). How to abtain/learn the Reflector member link identifier is out of the scope of this document.

When receives a reflected Test packet, the micro TWAMP Session-Sender MUST use the receiving member link to correlate the reflected Test packet to a micro TWAMP session. If there is no such a session, the reflected Test packet MUST be discarded. If a matched session exists, the Session-Sender MUST use the identifier carried in the Sender Member Link ID field to validate whether the reflected Test packet is correctly transmitted over the expected member link. If the validation is failed, the Test packet MUST be discarded.

4.2.1.1. Packet Format and Content

The micro TWAMP Session-Sender packet format is based on the TWAMP Session-Sender packet format as defined in Section 4.1.2 of [RFC5357]. In addition, in order to carry the LAG member link identifier, two new fields (Sender and Reflector Member Link ID) are added. The formats are as below:

For unauthenticated mode:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sequence Number                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Timestamp                            |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Error Estimate         |             MBZ               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Sender Member Link ID      |    Reflector Member Link ID   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
.                         Packet Padding                        .
.                                                               .
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: Session-Sender Packet format in Unauthenticated Mode

For authenticated mode:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sequence Number                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                        MBZ (12 octets)                        |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Timestamp                            |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Error Estimate         |             MBZ               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Sender Member Link ID      |   Reflector Member Link ID    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                       HMAC (16 octets)                        |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
.                        Packet Padding                         .
.                                                               .
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: Session-Sender Packet Format in Authenticated Mode

Except for the Sender/Reflector Member Link ID field, all the other fields are the same as defined in Section 4.1.2 of TWAMP [RFC5357], which is originally defined in Section 4.1.2 of OWAMP [RFC4656]. Therefore, it follows the same procedure and guidelines as defined in Section 4.1.2 of TWAMP [RFC5357].

Sender Member Link ID (2-octets in length): it is defined to carry the LAG member link identifier of the Sender side. The value of the Sender Member Link ID MUST be unique at the Session-Sender.

Reflector Member Link ID (2-octets in length): it is defined to carry the LAG member link identifier of the Reflector side. The value of the Reflector Member ID MUST be unique at the Session-Reflector.

4.2.2. Reflector Behavior

The micro TWAMP Session-Reflector inherits the behaviors of a TWAMP Session-Reflector as defined in Section 4.2 of [RFC5357].

In addition, when receives a Test packet, the micro TWAMP Session-Reflector MUST use the receiving member link to correlate the Test packet to a micro TWAMP session. If there is no such a session, the Test packet MUST be discarded. If Reflector Member Link ID is not zero, the Reflector MUST use the Reflector member link identifier to check whether it associates with the receiving member link. If it does not, the Test packet MUST be discarded.

When sends a response to the received Test packet, the micro TWAMP Session-Sender MUST copy the Sender member link identifier from the received Test packet and put it in the Sender Member Link ID field of the reflected Test packet (see Figure 4 and Figure 5). In addition, the micro TWAMP Session-Reflector MUST fill the Reflector Member Link ID field (see Figure 2 or Figure 3) of the reflected Test packet with the member link identifier that are associated with the micro TWAMP session.

4.2.2.1. Packet Format and Content

The micro TWAMP Session-Reflector packet format is based on the TWAMP Session-Reflector packet format as defined in Section 4.2.1 of [RFC5357]. In addition, in order to carry the LAG member link identifier, two new fields (Sender and Reflector Member Link ID) are added. The formats are as below:

For unauthenticated mode:

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                        Sequence Number                        |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                          Timestamp                            |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |         Error Estimate        |           MBZ                 |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                      Receive Timestamp                        |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                    Sender Sequence Number                     |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                      Sender Timestamp                         |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |      Sender Error Estimate    |    Sender Member Link ID      |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |  Sender TTL   |      MBZ      |   Reflector Member Link ID    |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    .                                                               .
    .                         Packet Padding                        .
    .                                                               .
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4: Session-Reflector Packet Format in Unauthenticated Mode

For authenticated and encrypted modes:

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

```
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                        Receive Timestamp                      |
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                        MBZ (8 octets)                         |
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                        Sender Sequence Number                 |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                        MBZ (12 octets)                        |
      |                                                               |
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                      Sender Timestamp                         |
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |      Sender Error Estimate    |                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               +
      |                        MBZ (6 octets)                         |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |  Sender TTL   |                                               |
      +-+-+-+-+-+-+-+-+                                               +
      |                                                               |
      |                                                               |
      |                        MBZ (15 octets)                        |
      +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      |                        HMAC (16 octets)                       |
      |                                                               |
      |                                                               |
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                                                               |
      .                         Packet Padding                        .
      .                                                               .
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 5: Session-Reflector Packet Format in Authenticated Mode

Except for the Sender/Reflector Member Link ID field, all the other fields are the same as defined in Section 4.2.1 of TWAMP [RFC5357]. Therefore, it follows the same procedure and guidelines as defined in Section 4.2.1 of TWAMP [RFC5357].

Sender Member Link ID (2-octets in length): it is defined to carry the LAG member link identifier of the Sender side. The value of the Sender Member Link ID MUST be unique at the Session-Sender.

Reflector Member Link ID (2-octets in length): it is defined to carry the LAG member link identifier of the Reflector side. The value of the Reflector Member ID MUST be unique at the Session-Reflector.

5. Mirco STAMP Session

5.1. Micro STAMP-Test

The micro STAMP-Test protocol is based on the STAMP-Test protocol [RFC8762] and [I-D.ietf-ippm-stamp-option-tlv] with the following extensions.

5.1.1. Session-Sender Packet Format

The micro STAMP Session-Sender Test packet formats are based on the STAMP Session-Sender Test packet formats and with some extensions, two new fields (Sender and Reflector Member Link ID) are added. The formats are as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sequence Number                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Timestamp                            |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Error Estimate        |             SSID              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Sender Member Link ID      |   Reflector Member Link ID    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                        MBZ  (24 octets)                       |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 6: Session-Sender Test Packet in Unauthenticated Mode

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Sequence Number                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                       MBZ (12 octets)                         |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Timestamp                             |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Error Estimate         |            SSID               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Sender Member Link ID      |   Reflector Member Link ID    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         MBZ (64 octets)                       |
~                                                               ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                       HMAC (16 octets)                        |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 7: Session-Sender Test Packet in Authenticated Mode

Except for the Sender/Reflector Member Link ID fields, all the other fields are as defined in STAMP [RFC8762] and [I-D.ietf-ippm-stamp-option-tlv].

Sender Member Link ID (2-octets in length): it is defined to carry the LAG member link identifier of the Sender side. The value of the Sender Member Link ID MUST be unique at the Session-Sender.

Reflector Member Link ID (2-octets in length): it is defined to carry the LAG member link identifier of the Reflector side. The value of the Reflector Member ID MUST be unique at the Session-Reflector.

5.1.2. Session-Reflector Packet Format

The micro STAMP Session-Reflector Test packet formats are based on the STAMP Session-Reflector Test packet formats with some minor extensions, two new fields (Sender and Reflector Member Link ID) are added. The formats are as follows:

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                        Sequence Number                        |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                          Timestamp                            |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |         Error Estimate        |           SSID                |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                          Receive Timestamp                    |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                 Session-Sender Sequence Number                |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                  Session-Sender Timestamp                     |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    | Session-Sender Error Estimate |    Sender Member Link ID      |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |Ses-Sender TTL |      MBZ      |   Reflector Member Link ID    |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 8: Session-Reflector Test Packet in Unauthenticated Mode

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sequence Number                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        MBZ (12 octets)                        |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Timestamp                            |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Error Estimate        |             SSID              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Sender Member Link ID    |   Reflector Member Link ID    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Receive Timestamp                      |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        MBZ (8 octets)                         |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Session-Sender Sequence Number                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        MBZ (12 octets)                        |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Session-Sender Timestamp                      |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Session-Sender Error Estimate |                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               +
|                          MBZ (6 octets)                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Ses-Sender TTL |                                               |
+-+-+-+-+-+-+-+-+                                               +
|                                                               |
|                        MBZ (15 octets)                        |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        HMAC (16 octets)                       |
|                                                               |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 9: Session-Reflector Test Packet in Authenticated Mode

Except for the Sender/Reflector Member Link ID fields, all the other fields are as defined in STAMP [RFC8762] and [I-D.ietf-ippm-stamp-option-tlv].

Sender Member Link ID (2-octets in length): it is defined to carry the LAG member link identifier of the Sender side. The value of the Sender Member Link ID MUST be unique at the Session-Sender.

Reflector Member Link ID (2-octets in length): it is defined to carry the LAG member link identifier of the Reflector side. The value of the Reflector Member ID MUST be unique at the Session-Reflector.

5.1.3. Micro STAMP-Test Procedures

The micro STAMP-Test reuses the procedures as defined in Section 4 of STAMP [RFC8762] with the following additions:

The micro STAMP Session-Sender MUST send the micro STAMP-Test packets over the member link with which the session is associated.

The configuration and management of the mapping between a micro STAMP session and the Sender/Reflector member link identifiers are outside the scope of this document.

When sending a Test packet, the micro STAMP Session-Sender MUST set the Sender Member Link ID field with the member link identifier that is associated with the micro STAMP session. If the Session-Sender knows the Reflector member link identifier, it MUST put it in the Reflector Member Link ID field (see Figure 6 or Figure 7). Otherwise, the Reflector Member Link ID field MUST be set to zero.

The Sender member link identifier is used by the Session-Sender to check whether a reflected Test packet is received from the member link that associates with the correct micro STAMP session. The Reflector member link identifier is used by the Session-Receiver to check whether a Test packet is received from the member link that associates with the correct micro STAMP session.

The Reflector member link identifier can be obtained from pre-configuration or learned through data plane (e.g., learned from a reflected Test packet). How to abtain/learn the Reflector member link identifier is out of the scope of this document.

When receives a Test packet, the micro STAMP Session-Reflector MUST use the receiving member link to correlate the Test packet to a micro STAMP session. If there is no such a micro STAMP session, the Test packet MUST be discarded. If the Reflector Member Link ID is not zero, the micro STAMP Session-Reflector MUST use the Reflector member link identifier to check whether it associates with the micro STAMP session. If it does not, the Test packet MUST be discarded and no reflected Test packet will be sent back the Session-Sender. If all validation passed, the Session-Reflector sends a reflected Test packet to the Session-Sender. The micro STAMP Session-Reflector MUST put the Sender and Reflector member link identifiers that are associated with the micro STAMP session in the Sender Member Link ID and Reflector Member Link ID fields (see Figure 8 and Figure 9) respectively. The Sender member link identifier is copied from the received Test packet.

When receives a reflected Test packet, the micro STAMP Session-Sender MUST use the receiving member link to correlate the reflected Test Packet to a micro STAMP session. If there is no such a session, the reflected Test packet MUST be discarded. If a matched micro STAMP session exists, the Session-Sender MUST use the identifier carried in the Sender Member Link ID field to check whether it associates with the session. If the checking failed, the Test packet MUST be discarded.

6. IANA Considerations

6.1. Mico OWAMP-Control Command

This document requires the IANA to allocate the following command type from OWAMP-Control Command Number Registry.

Value	Description	Semantics Definition
TBD1	Request-OW-Micro-Session	This document, Section 3.1

6.2. Mico TWAMP-Control Command

This document requires the IANA to allocate the following command type from TWAMP-Control Command Number Registry.

Value	Description	Semantics Definition
TBD1	Request-TW-Micro-Session	This document, Section 4.1

7. Security Considerations

The security considerations in [RFC4656], [RFC5357], [RFC8762] apply to this document.

8. Acknowledgements

The authors would like to thank Min Xiao, Fang Xin for the valuable comments to this work.

9. References

9.1. Normative References

[I-D.ietf-ippm-stamp-option-tlv]
Mirsky, G., Min, X., Nydell, H., Foote, R., Masputra, A., and E. Ruffini, "Simple Two-way Active Measurement Protocol Optional Extensions", draft-ietf-ippm-stamp-option-tlv-09 (work in progress), August 2020.

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

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

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

[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>.

9.2. Informative 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-10 (work in progress), July 2020.

[IEEE802.1AX]
IEEE Std. 802.1AX, "IEEE Standard for Local and metropolitan area networks - Link Aggregation", November 2008.

[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

Zhenqiang Li
China Mobile

Email: li_zhenqiang@hotmail.com

Mach(Guoyi) Chen
Huawei

Email: mach.chen@huawei.com

Greg Mirsky
ZTE Corp.

Email: gregimirsky@gmail.com

IPPM M. Cociglio
Internet-Draft Telecom Italia
Intended status: Informational A. Ferrieux
Expires: 6 May 2021 Orange Labs
G. Fioccola
Huawei Technologies
I. Lubashev
Akamai Technologies
F. Bulgarella
Telecom Italia
I. Hamchaoui
Orange Labs
M. Nilo
Telecom Italia
R. Sisto
Politecnico di Torino
D. Tikhonov
LiteSpeed Technologies
2 November 2020

Explicit Flow Measurements Techniques
draft-mdt-ippm-explicit-flow-measurements-00

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.

Discussion Venues

This note is to be removed before publishing as an RFC.

Discussion of this document takes place on the IPPM Working Group mailing list (ippm@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/ippm/.

Source for this draft and an issue tracker can be found at https://github.com/igorlord/draft-xxx-ippm-flow-measurements.

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 May 2021.

Copyright Notice

Table of Contents

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 latecy 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 an additional single-bit delay signal that can be used together with the spin bit 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, different from a two-bit VEC, 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.

If enabled, delay bit has to be used in addition to the spin bit. 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 second 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 in every spin period, 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 Delay bit mechanism: the first bit is the spin bit and the second one is the delay bit.

```
        +--------+   --  --  --  --  --   +--------+
        |        |      ----------->      |        |
        | Client |                        | Server |
        |        |     <-----------       |        |
        +--------+   --  --  --  --  --   +--------+

        (a) No traffic at beginning.

        +--------+   00  00  01  --  --   +--------+
        |        |      ----------->      |        |
        | Client |                        | Server |
        |        |     <-----------       |        |
        +--------+   --  --  --  --  --   +--------+

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

        +--------+   00  00  00  00  00   +--------+
        |        |      ----------->      |        |
        | Client |                        | Server |
        |        |     <-----------       |        |
        +--------+   --  --  01  00  00   +--------+
```

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

         +--------+   10  10  11  00  00   +---------+
         |        |       ----------->     |         |
         | Client |                        |  Server |
         |        |      <-----------      |         |
         +--------+   00  00  00  00  00   +---------+

         (d) The Client inverts the spin bit and
          reflects the Delay Sample.

         +--------+   10  10  10  10  10   +---------+
         |        |       ----------->     |         |
         | Client |                        |  Server |
         |        |      <-----------      |         |
         +--------+   00  00  11  10  10   +---------+

         (e) The Server reflects the Delay Sample.

         +--------+   00  00  01  10  10   +---------+
         |        |       ----------->     |         |
         | Client |                        |  Server |
         |        |      <-----------      |         |
         +--------+   10  10  10  10  10   +---------+

         (f) The client reverts the spin
          bit and reflects the Delay Sample.
```

Figure 1: Spin bit and Delay bit

3.2.1. Generation Phase

Only client is actively involved in the generation phase.

When connection starts and spin bit is set to 0, the client initializes the delay bit of the first packet to 1, so it becomes the delay sample for that marking period. Only this packet is marked with the delay bit set to 1 for this round trip period; the other ones will carry the spin bit, while the delay bit will be set to 0.

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 algorithm provides a procedure named "recovery process" to regenerate the delay sample and to inform a possible observer of the problem so the measurement can be restarted.

3.2.1.1. The Recovery Process

Absent packet loss or reordering, every spin period ends with a delay sample inside. If that does not happen and a spin period terminates without a delay sample inside, the client waits a further spin period; then, it creates a new delay sample by setting the delay bit to 1 on the first outgoing packet of the subsequent period.

The spin period with all delay bits set to 0 informs observers that there was a problem and delay measurements for this flow should be reset till the next delay sample is received.

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 slightly different.

* Server side reflection: when a delay sample arrives, the server marks the first packet in the opposite direction as the delay sample, if the outgoing packet has the same spin bit value as the delay sample. Otherwise, the delay sample is ignored.

* Client side reflection: when a delay sample arrives, the client marks the first packet in the opposite direction as the delay sample, if the outgoing packet has the opposite spin bit value then the delay sample. Otherwise, the delay sample is ignored.

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.

Note that reflection takes place for the delay sample regardless of its position within the spin period, as long as it stays within its original spin period.

A time threshold for the retransmission of the delay sample is used to eliminate 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. Two Bits Delay Measurement: Spin Bit + 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 in spin bit analysis, such as heuristic validation of the goodness of an edge signal.

3.2.3.1. 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 identify spin periods in the packet flow and assign delay samples to spin periods. If a spin period is missing a delay sample, the measurement needs to be restarted from the subsequent delay sample. Hence, measurements must take into account delay samples belonging to adjacent spin periods.

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

          (a) client-server RTT

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

          (b) server-client RTT
```

Figure 2: Round-trip time (both direction)

3.2.3.2. 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 reflected delay sample observed in the opposite direction.

As with RTT measurement, the observer must identify spin periods in the packet flow and assign delay samples to spin periods. If a spin period is missing a delay sample, the measurement needs to be restarted from the subsequent delay sample. So a measurement, to be valid, must take into account delay samples belonging to adjacent periods, for the upstream component, or to the same period for the downstream component.

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.

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

      (a) client-observer half-RTT

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

      (b) observer-server half-RTT
```

Figure 3: Half Round-trip time (both direction)

3.2.3.3. 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.

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

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

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

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

Figure 4: Intra-domain Round-trip time (client-observer: upstream)

3.2.4. Observer's Algorithm and Edge Rejection Interval

To provide a formal description of the observer behavior, we define a "matching delay sample" to be a delay sample with the spin bit value that matched the spin bit value of then-current spin period.

Upon detecting a matching delay sample, if a matching delay sample was also detected in the previous period, then the two delay samples can be used to calculate RTT measurement.

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 spin bit or connection handshake):

* If matching delay samples have been detected in both directions in the current spin period, they can be used to measure the observer-server half-RTT.

* If a matching delay sample has been detected in client-to-observer direction, AND a matching delay sample had been detected in observer-to-client direction in the previous spin period, they can be used to measure the observer-client half-RTT.

The described observer behavior depends on the ability to accurately identify current spin periods and to reject spurious spin edges, caused by packet reordering. Failure to do so will lead to many missed measurement opportunities and will decrease the amount of usable delay samples available to the observer.

To implement spurious edge rejection, every time a spin bit edge is detected, the observer starts a new spin period and begins a time interval during which it rejects spin edges (e.g. 5ms). This guarantees protection against spurious edges due to packets that have been reordered by less than the time interval. The mechanism only works for intervals smaller than the RTT of the observed connection; if RTT is smaller than the edge rejection interval, the observer cannot measure the RTT.

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.

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

(a) client-server RTPL

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

(b) server-client RTPL

Figure 5: 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.

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

(a) client-observer half-RTPL

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

(b) observer-server half-RTPL

Figure 6: Half Round-trip packet loss (both direction)

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

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

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

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

Figure 7: 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):

```
      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
```

Figure 8: 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, "lowPacketRate/(3*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*packetsPerAck)", where the value of "packetsPerAck" can vary by protocol, by implementation, and by network conditions.

4.2. 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 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-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)
```

Figure 9: 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 reordring, 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 sQuare 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)

Figure 10: 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-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)

Figure 11: 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)".

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

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

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

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

Figure 12: 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)".

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

                (a) in client-server channel (dloss_up)

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

                (b) in server-client channel (dloss_down)
```

Figure 13: 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.

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 alone or combined with the delay bit. A unidirectional or bidirectional observer can be used.

Method	# of bits	Available Delay Metrics		Impairments Resiliency
		UNIDIR Observer	BIDIR Observer	
S: Spin Bit	1	RTT	x2 Half RTT	lower
SD: Spin Bit + Delay Bit	2	RTT	x2 Half RTT	higher

x2 Same metric for both directions.

Figure 14: Delay Comparison

For the Loss measurement, each row in the table of Figure 15 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 UNIDIR Observer	 BIDIR Observer	Proto	Measurement Aspects 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 15: 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 the delay bit 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 loss bit (L) 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|L|K|P|P|
+-+-+-+-+-+-+-+-+
```

Figure 16: 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|
+-+-+-+-+-+-+-+-+
```

Figure 17: Scheme 2A

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

Figure 18: Scheme 2B

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, and eventually bits 5 and 6 to carry additional information, like the delay bit and the round-trip loss bit, 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", Work in Progress, Internet-Draft, draft-ietf-tcpm-accurate-ecn-12, 28 October 2020, <http://www.ietf.org/internet-drafts/draft-ietf-tcpm-accurate-ecn-12.txt>.

[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", DOI 10.1145/3340301.3341127, Proceedings of the Applied Networking Research Workshop, July 2019, <https://doi.org/10.1145/3340301.3341127>.

[I-D.trammell-ippm-spin] Trammell, B., "An Explicit Transport-Layer Signal for Hybrid RTT Measurement", Work in Progress, Internet-Draft, draft-trammell-ippm-spin-00, 9 January 2019, <http://www.ietf.org/internet-drafts/draft-trammell-ippm-spin-00.txt>.

[I-D.trammell-tsvwg-spin] Trammell, B., "A Transport-Independent Explicit Signal for Hybrid RTT Measurement", Work in Progress, Internet-Draft, draft-trammell-tsvwg-spin-00, 2 July 2018, <http://www.ietf.org/internet-drafts/draft-trammell-tsvwg-spin-00.txt>.

[IPv6AltMark] 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-02, 13 October 2020, <http://www.ietf.org/internet-drafts/draft-ietf-6man-ipv6-alt-mark-02.txt>.

[QUIC-TRANSPORT]
Iyengar, J. and M. Thomson, "QUIC: A UDP-Based Multiplexed and Secure Transport", Work in Progress, Internet-Draft, draft-ietf-quic-transport-32, 20 October 2020, <http://www.ietf.org/internet-drafts/draft-ietf-quic-transport-32.txt>.

[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., Even, R., Fioccola, G., Fossati, T., Ihlar, M., Morton, A., and S. Emile, "Adding Explicit Passive Measurability of Two-Way Latency to the QUIC Transport Protocol", Work in Progress, Internet-Draft, draft-trammell-quic-spin-03, 14 May 2018, <http://www.ietf.org/internet-drafts/draft-trammell-quic-spin-03.txt>.

[TRANSPORT-ENCRYPT]
Fairhurst, G. and C. Perkins, "Considerations around Transport Header Confidentiality, Network Operations, and the Evolution of Internet Transport Protocols", Work in Progress, Internet-Draft, draft-ietf-tsvwg-transport-encrypt-17, 8 September 2020, <http://www.ietf.org/internet-drafts/draft-ietf-tsvwg-transport-encrypt-17.txt>.

[UDP-OPTIONS]
Touch, J., "Transport Options for UDP", Work in Progress, Internet-Draft, draft-ietf-tsvwg-udp-options-08, 12 September 2019, <http://www.ietf.org/internet-drafts/draft-ietf-tsvwg-udp-options-08.txt>.

[UDP-SURPLUS]
Herbert, T., "UDP Surplus Header", Work in Progress, Internet-Draft, draft-herbert-udp-space-hdr-01, 8 July 2019, <http://www.ietf.org/internet-drafts/draft-herbert-udp-space-hdr-01.txt>.

Authors' Addresses

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

Email: mauro.cociglio@telecomitalia.it

Alexandre Ferrieux
Orange Labs

Email: alexandre.ferrieux@orange.com

Giuseppe Fioccola
Huawei Technologies
Riesstrasse, 25
80992 Munich
Germany

Email: giuseppe.fioccola@huawei.com

Igor Lubashev
Akamai Technologies

Email: ilubashe@akamai.com

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

Email: fabio.bulgarella@guest.telecomitalia.it

Isabelle Hamchaoui
Orange Labs

Email: isabelle.hamchaoui@orange.com

Massimo Nilo
Telecom Italia

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 G. Mirsky
Internet-Draft ZTE Corp.
Intended status: Standards Track W. Lingqiang
Expires: April 29, 2021 G. Zhui
ZTE Corporation
H. Song
Futurewei Technologies
October 26, 2020

Hybrid Two-Step Performance Measurement Method
draft-mirsky-ippm-hybrid-two-step-06

Abstract

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

Table of Contents

1. Introduction

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

Several technical methods have been proposed to enable the collection of network state information instantaneous to the packet processing, among them [P4.INT] and [I-D.ietf-ippm-ioam-data]. The instantaneous, i.e., in the data packet itself, collection of telemetry information simplifies the process of attribution of telemetry information to the particular monitored flow. On the other hand, this collection method impacts the data packets, potentially changing their treatment by the networking nodes. Also, the amount of information the instantaneous method collects might be incomplete because of the limited space it can be allotted. Other proposals defined methods to collect telemetry information in a separate packet from each node traversed by the monitored data flow. Examples of this approach to collecting telemetry information are [I-D.ietf-ippm-ioam-direct-export] and [I-D.song-ippm-postcard-based-telemetry]. These methods allow data collection from any arbitrary path and avoid directly impacting data packets. On the other hand, the correlation of data and the monitored flow requires that each packet with telemetry information also includes characteristic information about the monitored flow.

This document introduces Hybrid Two-Step (HTS) as a new method of telemetry collection that improvers accuracy of a measurement by separating the act of measuring or calculating the performance metric from the collecting and transporting this information while minimizing the overhead of the generated load in a network. HTS method extends the two-step mode of Residence Time Measurement (RTM) defined in [RFC8169] to on-path network state collection and transport. HTS allows the collection of telemetry information from any arbitrary path, does not change data packets of the monitored flow and makes the process of attribution of telemetry to the data flow simple.

2. Conventions used in this document

2.1. Terminology

RTM Residence Time Measurement

ECMP Equal Cost Multipath

MTU Maximum Transmission Unit

HTS Hybrid Two-Step

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

2.2. Requirements Language

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

3. Problem Overview

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

- the resolution and accuracy of each measurement;

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

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

Depending on the implementation, it may be a challenge to compute the difference between message arrival and departure times and - on the fly - add the necessary residence time information to the same message. And that task may become even more challenging if the packet is encrypted. Recording the departure of a packet time in the same packet may be decremental to the accuracy of the measurement, because the departure time includes the variable time component (such as that associated with buffering and queuing of the packet). A similar problem may lower the quality of, for example, information that characterizes utilization of the egress interface. If unable to obtain the data consistently, without variable delays for additional processing, information may not accurately reflect the egress interface state. To mitigate this problem [RFC8169] defined an RTM two-step mode.

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

4. Theory of Operation

The HTS method consists of the two phases:

- o performing a measurement or obtaining network state information, one or more than one type, on a node;

- o collecting and transporting the measurement.

HTS uses HTS Trigger carried in a data packet or a specially constructed test packet. For example, an HTS Trigger could be a packet that includes iOAM Namespace-ID and IOAM-Trace-Type information [I-D.ietf-ippm-ioam-data] or a packet in the flow to which the Alternate-Marking method [RFC8321] is applied. Nature of the HTS Trigger is a transport network layer-specific, and its description is outside the scope of this document. The packet that includes the HTS Trigger in this document is also referred to as the trigger packet.

The HTS method uses the HTS Follow-up packet, in this document also referred to as the follow-up packet, to collect measurement and network state data from the nodes. The node that creates the HTS Trigger also generates the HTS Follow-up packet. The follow-up packet contains characteristic information, copied from the trigger packet, sufficient for participating HTS nodes to associate it with the original packet. The exact composition of the characteristic information is specific for each transport network, and its definition is outside the scope of this document. The follow-up packet also uses the same encapsulation as the data packet. If not payload but only network information used to load-balance flows in equal cost multipath (ECMP), use of the network encapsulation identical to the trigger packet should guarantee that the follow-up packet remains in-band, i.e., traverses the same set of network elements, with the original data packet with the HTS Trigger. Only one outstanding follow-up packet MUST be on the node for the given path. That means that if the node receives an HTS Trigger for the flow on which it still waits for the follow-up packet to the previous HTS Trigger, the node will originate the follow-up packet to transport the former set of the network state data and transmit it before it sends the follow-up packet with the latest collection of network state information.

4.1. Operation of the HTS Ingress Node

A node that originates the HTS Trigger is referred to as HTS ingress node. As stated, the ingress node originates the follow-up packet. The follow-up packet has the transport network encapsulation identical with the trigger packet followed by the HTS shim and one or more telemetry information elements encoded as Type-Length-Value {TLV}. Figure 1 displays the example of the follow-up packet format.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                      Transport Network                        ~
|                        Encapsulation                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Ver|HTS Shim Len|     Flags     |        Sequence Number        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Telemetry Data Profile                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                     Telemetry Data TLVs                       ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: Follow-up Packet Format

Fields of the HTS shim are as follows:

Version (Ver) is the two-bits long field. It specifies the version of the HTS shim format. This document defines the format for the 0b00 value of the field.

HTS Shim Length is the six bits-long field. It defines the length of the HTS shim in bytes. The minimal value of the field is four bytes.

```
 0
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|F|  Reserved   |
+-+-+-+-+-+-+-+-+
```

Figure 2: Flags Field Format

Flags is eight-bits long field. The format of the Flags field displayed in Figure 2.

Full (F) flag MUST be set to zero by the node originating the HTS follow-up packet and MUST be set to one by the node that does not add its telemetry data to avoid exceeding MTU size.

The node originating the follow-up packet MUST zero the Reserved field and ignore it on the receipt.

Sequence Number is 16 bits-long field. The zero-based value of the field reflects the place of the HTS follow-up packet in the sequence of the HTS follow-up packets originated in response to the same HTS trigger. The ingress node MUST set the value of the field to zero.

Telemetry Data Profile is the optional variable length field of bit-size flags. Each flag indicates requested type of telemetry data to be collected at the each HTS node. The increment of the field is four bytes with a minimum length of zero. For example, IOAM-Trace-Type information defined in [I-D.ietf-ippm-ioam-data] can be used in the Telemetry Data Profile field.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Type             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~                             Value                             ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: Telemetry Data TLV Format

Telemetry Data TLV is a variable-length field. Multiple TLVs MAY be placed in an HTS packet. Additional TLVs may be enclosed within a given TLV, subject to the semantics of the (outer) TLV in question. Figure 3 presentes the format of a Telementry Data TLV, where fields are defined as the following:

Type - two-octet-long field that characterizes the interpretation of the Value field.

Length - two-octet-long field equal to the length of the Value field in octets.

Value - a variable-length field. Its interpretation and encoding is determined by the value of the Type field. IOAM data fields, defined in [I-D.ietf-ippm-ioam-data], MAY be carried in the Value field.

All multibyte fields defined in this specification are in network byte order.

4.2. Operation of the HTS Intermediate Node

Upon receiving the trigger packet the HTS intermediate node MUST:

- copy the transport information;
- start the HTS Follow-up Timer for the obtained flow.

Upon receiving the follow-up packet the HTS intermediate node MUST:

- verify that the matching transport information exists and the Full flag is cleared, then stop the associated HTS Follow-up timer;
- collect telemetry data requested in the Telemetry Data Profile field or defined by the local HTS policy;
- if adding the collected telemetry would not exceed MTU, then append data into Telemetry Data TLVs field and transmit the follow-up packet;
- otherwise, set the value of the Full flag to one and transmit the received a follow-up packet;
- originate the new follow-up packet using the same transport information. The value of the Sequence Number field in the HTS shim MUST be set to the value of the field in the received follow-up packet incremented by one. Copy collected telemetry data and transmit the packet.

If the HTS Follow-up Timer expires, the intermediate node MUST:

- originate the follow-up packet using transport information associated with the expired timer;

- o initialize the HTS shim by setting Version field to 0b00 and Sequence Number field to 0. Values of HTS Shim Length and Telemetry Data Profile fields MAY be set according to the local policy.

- o copy telemetry information into Telemetry Data TLVs field and transmit the packet.

If the intermediate node receives a "late" follow-up packet, i.e., a packet to which the node has no associated HTS Follow-up timer, the node MUST forward the "late" packet.

4.3. Operation of the HTS Egress Node

Upon receiving the trigger packet the HTS egress node MUST:

- o copy the transport information;

- o start the HTS Collection timer for the obtained flow.

When the egress node receives the follow-up packet for the known flow, i.e., the flow to which the Collection timer is running, the node MUST:

- o copy telemetry information;

- o restart the corresponding Collection timer.

When the Collection timer expires the egress relays the collected telemetry information for processing and analysis to a local or remote agent.

4.4. Considerations for HTS Timers

This specification defines two timers - HTS Follow-up and HTS Collection. Because for the particular flow there MUST be not more than one HTS Trigger, values of HTS timers bounded by the rate of the trigger generation for that flow.

4.5. Deploying HTS in a Multicast Network

Previous sections discussed the operation of HTS in a unicast network. Multicast services are important, and the ability to collect telemetry information is an invaluable component in delivering a high quality of experience. While the replication of data packets is necessary, replication of HTS follow-up packets is not. Replication of multicast data packets down a multicast tree may be set based on multicast routing information or explicit information included in the special header, as, for example, in Bit-Indexed Explicit Replication [RFC8296]. A replicating node processes HTS packet as defined below:

- o the first transmitted multicast packet MUST be followed by the received corresponding HTS packet as described in Section 4.2;
- o each consecutively transmitted copy of the original multicast packet MUST be followed by the new HTS packet originated by the replicating node that acts as a intermediate HTS node when the HTS Follow-up timer expired.

As a result, there are no duplicate copies of Telemetry Data TLV for the same pair of ingress and egress interfaces. At the same time, all ingress/egress pairs traversed by the given multicast packet reflected in their respective Telemetry Data TLV. Consequently, a centralized controller would be able to reconstruct and analyze the state of the particular multicast distribution tree based on HTS packets collected from egress nodes.

5. IANA Considerations

TBD

6. Security Considerations

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

The ability to potentially authenticate and/or encrypt the network state data for scenarios both with and without the participation of intermediate nodes that participate in HTS measurement is left for further study.

While it is possible for a supposed compromised node to intercept and modify the network state information in the follow-up packet, this is an issue that exists for nodes in general - for all data that to be carried over the particular networking technology - and is therefore the basis for an additional presumed trust model associated with an existing network.

7. Acknowledgments

Authors express their gratitude and appreciation to Joel Halpern for the most helpful and insightful discussion on the applicability of HTS in a Service Function Chaining domain.

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>.

[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-ippm-ioam-data] Brockners, F., Bhandari, S., and T. Mizrahi, "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data-10 (work in progress), July 2020.

[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-01 (work in progress), August 2020.

[I-D.song-ippm-postcard-based-telemetry] Song, H., Zhou, T., Li, Z., Shin, J., and K. Lee, "Postcard-based On-Path Flow Data Telemetry", draft-song-ippm-postcard-based-telemetry-07 (work in progress), April 2020.

[P4.INT] "In-band Network Telemetry (INT)", P4.org Specification, October 2017.

[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>.

[RFC8169] Mirsky, G., Ruffini, S., Gray, E., Drake, J., Bryant, S., and A. Vainshtein, "Residence Time Measurement in MPLS Networks", RFC 8169, DOI 10.17487/RFC8169, May 2017, <https://www.rfc-editor.org/info/rfc8169>.

[RFC8296] Wijnands, IJ., Ed., Rosen, E., Ed., Dolganow, A., Tantsura, J., Aldrin, S., and I. Meilik, "Encapsulation for Bit Index Explicit Replication (BIER) in MPLS and Non-MPLS Networks", RFC 8296, DOI 10.17487/RFC8296, January 2018, <https://www.rfc-editor.org/info/rfc8296>.

[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

Greg Mirsky
ZTE Corp.

Email: gregimirsky@gmail.com

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

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

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

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

Haoyu Song
Futurewei Technologies
2330 Central Expressway
Santa Clara
USA

Email: hsong@futurewei.com

IP Performance Measurement Group
Internet-Draft
Intended status: Standards Track
Expires: May 19, 2021

Y. Wang
T. Zhou
Huawei
H. Yang
China Mobile
C. Liu
China Unicom

November 15, 2020

Simple Two-way Active Measurement Protocol Extensions for Hop-by-Hop OAM Data Collection

draft-wang-ippm-stamp-hbh-extensions-02

Abstract

This document defines optional TLVs which are carried in Simple Two-way Active Measurement Protocol (STAMP) test packets to enhance the STAMP base functions. Such extensions to STAMP enable OAM data measurement and collection at every node and link along a STAMP test packet's delivery path without maintaining a state for each configured STAMP-Test session at every devices.

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].

Copyright Notice

Table of Contents

1. Introduction

Simple Two-way Active Measurement Protocol (STAMP) [RFC8762] enables the measurement of both one-way and round-trip performance metrics, such as delay, delay variation, and packet loss. In the STAMP session, the bidirectional packet flow is transmitted between STAMP Session-Sender and STAMP Session-Reflector. The STAMP Session-Reflector receives test packets transmitted from Session-Sender and acts according to the configuration. However, the performance of intermediate nodes and links that STAMP test packets traverse are invisible. In addition, the STAMP instance must be configured at every intermediate node to measure the performance per node and link that test packets traverse, which increases the complexity of OAM in large-scale networks.

STAMP Extensions have defined several optional TLVs to enhance the STAMP base functions. These optional TLVs are defined as updates of the STAMP Optional Extensions [I-D.ietf-ippm-stamp-option-tlv]. This document extents optional TLVs to STAMP, which enables performance measurement at every intermediate node and link along a STAMP test packet's delivery path, such as measurement of delay, delay variation, packet loss, and record of route information. The following sections describe the use of TLVs for STAMP that extend STAMP capability beyond its base specification.

2. Terminology

Following are abbreviations used in this document:

STAMP: Simple Two-way Active Measurement Protocol.

IOAM: In-situ OAM.

HbH: Hop-by-Hop.

3. TLV Extensions to STAMP

3.1. IOAM Tracing Data TLV

STAMP Session-Sender MAY place the IOAM Tracing Data TLV in Session-Sender test packets to record the IOAM tracing data at every IOAM capable node along the Session-Sender test packet's forward-delivery path. As STAMP uses symmetrical packets, the Session-Sender MUST set the Length value as a multiple of 4 octets according to the number of nodes and the IOAM-Trace-Type (i.e. a 24-bit identifier which specifies which data types are used in the node data list [I-D.ietf-ippm-ioam-data]). And the node-data-copied-list fields MUST be set to zero upon Session-Sender test packets transmission and ignored upon receipt.

The IOAM Tracing Data TLV has the following format:

```
 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
+-------------------------------+-------------------------------+
|   IOAM-Tracing-Data Type      |            Length             |
+---------------------------------------------------------------+
|                    node data copied list [0]                  |
+---------------------------------------------------------------+
|                    node data copied list [1]                  |
+---------------------------------------------------------------+
~                                ...                            ~
+---------------------------------------------------------------+
|                    node data copied list [n]                  |
+---------------------------------------------------------------+
```

Fig. 1 IOAM Tracing Data TLV Format

where fields are defined as the following:

o IOAM-Tracing-Data Type: To be assigned by IANA.

o Length: A 2-octet field that indicates the length of the value field in octets and equal to a multiple of 4 octets dependent on the number of nodes and IOAM-Trace-Type bits.

o node data copied list [0..n]: A variable-length field, which record the copied content of each node data element determined by the IOAM-Trace-Type. The order of packing the data fields in each node data element follows the bit order of the IOAM-Trace-Type field (see section 4.4.1 of [I-D.ietf-ippm-ioam-data]). The last node data element in this list is the node data of the first IOAM trace capable node in the path.

In an IOAM domain, the STAMP Session-Sender and the STAMP Session-Reflector MAY be configured as the IOAM encapsulating node and the IOAM decapsulating node. The STAMP Session-Sender (i.e. the IOAM encapsulating node) generates the test packet with the IOAM Tracing Data TLV. For applying the IOAM Trace-Option functionalities to the Session-Sender test packet, the Session-Sender must inserts the "trace option header" and allocate an node-data-list array [I-D.ietf-ippm-ioam-data] into "option data" fields of Hop-by-Hop Options header in IPv6 packets [I-D.ietf-ippm-ioam-ipv6-options], and sets the corresponding bits in the IOAM-Trace-Type. Also, the STAMP Session-Sender allocates a node-data-copied-list array in the optional IOAM Tracing Data TLV to store OAM data retrieved from every IOAM transit node along the Session-Sender test packet's delivery path.

When the STAMP Session-Reflector (i.e. the IOAM decapsulating node) received the STAMP Session-Sender test packet with the IOAM-Tracing-Data TLV, it MUST copy the node-data-list array into the node-data-copied-list array carried in the Session-Reflector test packet before transmission and MUST remove the IOAM-Data-Fields. Hence, carrying IOAM-Tracing-Data TLV in STAMP test packets enables OAM data collection and measurement at every node and link.

Also the STAMP Session-Reflector MAY be configured as IOAM encapsulating node to apply the IOAM Trace-Option functionalities to the Session-Reflector test packet. Hence, OAM data collection and measurement can be also enabled at every node and link along the Session-Reflector test packet's backward delivery path. When the reflected packet arrives at the Session-Sender, it can be either locally processed or sent to the centralized controller.

In addition to IOAM, other telemetry data (e.g. alternate marking) could be transmitted by STAMP optional TLV extensions. The draft will keep the option open for future augmentations.

3.2. Forward HbH Delay TLV

STAMP Session-Sender MAY place the Forward HbH Delay TLV in Session-Sender test packets to record the ingress timestamp and the egress timestamp at every intermediate nodes along the Session-Sender test packet's forward path. The Session-Sender MUST set the Length value according to the number of explicitly listed intermediate nodes in the forward path and the timestamp formats. There are several methods to synchronize the clock, e.g., Network Time Protocol (NTP) [RFC5905] and IEEE 1588v2 Precision Time Protocol (PTP) [IEEE.1588.2008]. For example, if a 64-bit timestamp format defined in NTP is used, the Length value MUST be set as a multiple of 16 octets. The Timestamp Tuple list [1..n] fields MUST be set to zero upon Session-Sender test packets transmission.

The Forward HbH Delay TLV has the following format:

```
 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
+-------------------------------+---------------+---------------+
|   Forward HbH Delay Type      |    Length     |   Node Left   |
+-------------------------------+---------------+---------------+
|                                                               |
|                   Timestamp Tuple list [1]                    |
|                                                               |
|                                                               |
+---------------------------------------------------------------+
~                              ...                              ~
+---------------------------------------------------------------+
|                                                               |
|                   Timestamp Tuple list [n]                    |
|                                                               |
|                                                               |
+---------------------------------------------------------------+
```

Fig. 2 Forward HbH Delay TLV Format

where fields are defined as the following:

- o Forward HbH Delay Type: To be assigned by IANA.

- o Length: A 8-bit field that indicates the length of the value portion in octets and MUST be a multiple of 16 octets according to the number of explicitly listed intermediate nodes in the forward path.

- o Node Left: A 8-bit unsigned integer, which indicates the number of intermediate nodes remaining. That is, number of explicitly listed intermediate nodes still to be visited before reaching the destination node in the forward path. The Node Left field is set to n-1, where n is the number of intermediate nodes.

- o Timestamp Tuple list [1..n]: A variable-length field, which record the timestamp when the Session-Sender test packet is received at the ingress of the n-th intermediate node Ingress Timestamp [n] and the timestamp when the Session-Sender test packet is sent at egress of the n-th intermediate node Egress Timestamp [n]. For example, if a 64-bit timestamp format defined in NTP is used, the length of each Timestamp tuple (Ingress Timestamp [n], Egress Timestamp [n]) must be 16 octets. The Timestamp Tuple list is encoded starting from the last intermediate node which is explicitly listed. That is, the first element of the Timestamp Tuple list [1] records the timestamps when the Session-Sender test packet received and forwarded at the last intermediate node of a explicit path, the second element records the penultimate

Timestamp Tuple when the Session-Sender test packet received and forwarded at the penultimate intermediate node of a explicit path, and so on.

In the following reference topology, Node N1, N2, N3, N4 and N5 are SRv6 capable nodes. Node N1 is the STAMP Session-Sender and Node N5 is the STAMP Session-Reflector. T1 is the Timestamp taken by the Session-Sender (i.e. N1) at the start of transmitting the test packet. T2 is the Receive Timestamp when the test packet was received by the Session-Reflector (i.e. N5). T3 is the Timestamp taken by the Session-Reflector at the start of transmitting the test packet. T4 is the Receive Timestamp when the test packet was received by the Session-Sender. Timestamp tuples (t1,t2), (t3,t4) and (t5,t6) are the timestamps when the test packet received and transmitted by sequence of intermediate nodes in the forward path. Timestamp Tuples (t7,t8), (t9,t10) and (t11,t12) are the timestamps when the test packet received and transmitted by sequence of intermediate nodes in the backward path.

```
======          ======          ======          ======         ======
|    | T1--->t1 |    | t2--->t3 |    | t4--->t5 |    | t6--->T2|    |
| N1 |==========| N2 |==========| N3 |==========| N4 |=========| N5 |
|    | T4<---t12|    |t11<---t10|    | t9<---t8 |    | t7<---T3|    |
======          ======          ======          ======         ======
```

Fig. 3 Reference Topology

The STAMP Session-Sender (i.e. Node N1) generates the STAMP test packet with the Forward HbH Delay TLV. When an intermediate node receives the STAMP test packet, the node punts the packet to control plane and fills the Ingress Timestamp [n] filed in the Timestamp Tuple list [n]. Then the time taken by the intermediate node transmitting the test packet is recorded in to Egress Timestamp [n] field. The mechanism of timestamping and punting packet to control plane is outside the scope of this specification.

When the STAMP Session-Reflector received the test packet with the Forward HbH Delay TLV, it MUST copy the Forward HbH Delay TLV into the Session-Reflector test packet before its transmission. Using Forward HbH Delay TLV in STAMP testing enables delay measurement per link in the forward path.

3.3. Backward HbH Delay TLV

STAMP Session-Sender MAY place the Backward HbH Delay TLV in Session-Sender test packets to record the ingress timestamp and egress timestamp when Session-Reflector test packets are received and sent at every intermediate nodes in the backward path. The Session-Sender MUST set the Length value according to the number of explicitly listed intermediate nodes in the backward path and the timestamp formats. There are several methods to synchronize the clock, e.g., Network Time Protocol (NTP) [RFC5905] and IEEE 1588v2 Precision Time Protocol (PTP) [IEEE.1588.2008]. For example, if a 64-bit timestamp format defined in NTP is used, the Length value MUST be set as a multiple of 16 octets. The Timestamp Tuple list [1..n] fields MUST be set to zero upon Session-Sender test packets transmission and ignored upon receipt.

The Backward HbH Delay TLV has the following format:

```
 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
+-------------------------------+---------------+---------------+
|   Backward HbH Delay Type     |    Length     |   Node Left   |
+-------------------------------+---------------+---------------+
|                                                               |
|                   Timestamp Tuple list [1]                    |
|                                                               |
|                                                               |
+---------------------------------------------------------------+
~                              ...                              ~
+---------------------------------------------------------------+
|                                                               |
|                   Timestamp Tuple list [n]                    |
|                                                               |
|                                                               |
+---------------------------------------------------------------+
```

Fig. 4 Backward HbH Delay TLV Format

where fields are defined as the following:

o Backward HbH Delay Type: To be assigned by IANA.

o Length: A 8-bit field that indicates the length of the value portion in octets and will be a multiple of 16 octets dependent on the number of explicitly listed intermediate nodes in the backward path.

o Node Left: A 8-bit unsigned integer, which indicates the number of intermediate nodes remaining. That is, number of explicitly listed intermediate nodes still to be visited before reaching the destination node in the backward path. The Node Left field is set to n-1, where n is the number of intermediate nodes.

o Timestamp Tuple list [1..n]: A variable-length field, which record the timestamp when the reflected test packet is received at the ingress of the n-th intermediate node and the timestamp when the reflected test packet is sent at egress of the n-th intermediate node. For example, if a 64-bit timestamp format defined in NTP is used, the length of each Timestamp tuple (Ingress Timestamp [n], Egress Timestamp [n]) must be 16 octets. The Timestamp Tuple list is encoded starting from the last intermediate node which is explicitly listed. That is, the first element of the Timestamp Tuple list [1] records the timestamps when the reflected test packet received and forwarded at the last intermediate node of a explicit path, the second element records the penultimate Timestamp Tuple when the reflected test packet received and forwarded at the penultimate intermediate node of a explicit path, and so on.

When the STAMP Session-Reflector received the Session-Sender test packet with the Backward HbH Delay TLV, it MUST copy the Backward HbH Delay TLV into the Session-Reflector test packet.

When an intermediate node receives the reflected test packet, the node sends the packet to control plane and fills the Ingress Timestamp [n] filed of Backward HbH Delay TLV. Then the time taken by the intermediate node transmitting the test packet is recorded in to Egress Timestamp [n] field of Backward HbH Delay TLV. Using Backward HbH Delay TLV in STAMP testing enables delay measurement per link in the backward path.

3.4. HbH Packet Loss TLV

STAMP Session-Sender MAY place the HbH Packet Loss TLV in Session-Sender test packets to record the number of Session-Sender test packets received at and transmitted by every intermediate nodes along the forward path. The Session-Sender MUST set the Length value according to the number of explicitly listed intermediate nodes in the forward path. A Counter Tuple is composed of a 64-bit Receive Counter field and a 64-bit Transmit Counter field. The Counter Tuple list [1..n] fields MUST be set to zero upon Session-Sender test packets transmission.

The HbH Packet Loss TLV has the following format:

```
 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
+-------------------------------+---------------+---------------+
|     HbH Packet Loss Type      |    Length     |   Node Left   |
+-------------------------------+---------------+---------------+
|                                                               |
|                    Counter Tuple list [1]                     |
|                                                               |
|                                                               |
+---------------------------------------------------------------+
~                              ...                              ~
+---------------------------------------------------------------+
|                                                               |
|                    Counter Tuple list [n]                     |
|                                                               |
|                                                               |
+---------------------------------------------------------------+
```

Fig. 5 HbH Packet Loss TLV Format

where fields are defined as the following:

- o HbH Packet Loss Type: To be assigned by IANA.

- o Length: A 8-bit field that indicates the length of the value portion in octets and will be a multiple of 16 octets dependent on the number of explicitly listed intermediate nodes in the forward path.

- o Node Left: A 8-bit unsigned integer, which indicates the number of intermediate nodes remaining. That is, number of explicitly listed intermediate nodes still to be visited before reaching the destination node in the forward path. The Node Left field is set to n-1, where n is the number of intermediate nodes.

- o Counter Tuple list [1..n]: A variable-length field, which record the Receive Counter and the Transmit Counter when the test packet is received at and transmitted by the n-th intermediate node. The Counter Tuple list is encoded starting from the last intermediate node which is explicitly listed. That is, the first element of the Counter Tuple list [1] records the Receive Counter and the Transmit Counter when the test packet is received at and transmitted by the last intermediate node of a explicit path, the second element records the penultimate Counter Tuple when the test packet received and forwarded at the penultimate intermediate node of a explicit path, and so on.

The STAMP Session-Sender generates the STAMP test packet with the HbH Packet Loss TLV. When an intermediate node receives the STAMP test packet, the node punts the packet to control plane and writes the Receive Counter and the Transmit Counter at the Counter Tuple list [n] in the Session-Sender test packet. The mechanism of punting packet to control plane is outside the scope of this specification.

When the STAMP Session-Reflector received the test packet with the HbH Packet Loss TLV, it MUST copy the HbH Packet Loss TLV into the Session-Reflector test packet before its transmission. Using HbH Packet Loss TLV in STAMP testing enables packet loss measurement per node/link in the forward path.

3.5. HbH Bandwidth Utilization TLV

STAMP Session-Sender MAY place the HbH Bandwidth Utilization TLV in Session-Sender test packets to record the ingress and egress bandwidth utilization at every intermediate nodes along the forward path. The Session-Sender MUST set the Length value according to the number of explicitly listed intermediate nodes in the forward path. A BW Utilization Tuple is composed of a 32-bit ingress bandwidth utilization field and a 32-bit egress bandwidth utilization field. The BW Utilization Tuple list [1..n] fields MUST be set to zero upon Session-Sender test packets transmission.

The HbH Bandwidth Utilization TLV has the following format:

```
 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
+-------------------------------+---------------+---------------+
|    HbH BW Utilization Type    |     Length    |   Node Left   |
+-------------------------------+---------------+---------------+
|                BW Utilization Tuple list [1]                  |
|                                                               |
+---------------------------------------------------------------+
~                              ...                              ~
+---------------------------------------------------------------+
|                BW Utilization Tuple list [n]                  |
|                                                               |
+---------------------------------------------------------------+
```

Fig. 6 HbH Bandwidth Utilization TLV Format

where fields are defined as the following:

o HbH BW Utilization Type: To be assigned by IANA.

- o Length: A 8-bit field that indicates the length of the value portion in octets and will be a multiple of 8 octets dependent on the number of explicitly listed intermediate nodes in the forward path.

- o Node Left: A 8-bit unsigned integer, which indicates the number of intermediate nodes remaining. That is, number of explicitly listed intermediate nodes still to be visited before reaching the destination node in the forward path. The Node Left field is set to n-1, where n is the number of intermediate nodes.

- o BW Utilization Tuple list [1..n]: A variable-length field, which record the ingress and egress bandwidth utilization when the test packet is received at and transmitted by the n-th intermediate node. The BW Utilization Tuple list is encoded starting from the last intermediate node which is explicitly listed. That is, the first element of the BW Utilization Tuple list [1] records the ingress and the egress bandwidth utilization when the test packet is received at and transmitted by the last intermediate node of a explicit path, the second element records the penultimate BW Utilization Tuple when the test packet received at and transmitted by the penultimate intermediate node of a explicit path, and so on.

The STAMP Session-Sender generates the STAMP test packet with the HbH BW Utilization TLV. When an intermediate node receives the STAMP test packet, the node punts the packet to control plane and writes the ingress and egress bandwidth utilization at the BW Utilization Tuple list [n] in the Session-Sender test packet. The mechanism of punting packet to control plane is outside the scope of this specification.

When the STAMP Session-Reflector received the test packet with the HbH BW Utilization TLV, it MUST copy the HbH BW Utilization TLV into the Session-Reflector test packet before its transmission. The HbH BW Utilization TLV carried in STAMP test packet is usable to detect and troubleshoot the link congestion in the forward path.

3.6. HbH Timestamp Information TLV

STAMP Session-Sender MAY place the HbH Timestamp Information TLV in Session-Sender test packets to query the ingress and egress Timestamp Information at every intermediate nodes along the forward path. The Timestamp Information includes the source of clock synchronization and the method of timestamp obtainment. There are several types of clock synchronization source, e.g., NTP, PTP. The method of timestamp obtainment may be from control plane (e.g. NTP) or from data plane (e.g. PTP). A Timestamp Info Tuple is composed of a

8-bit ingress clock source field, a 8-bit ingress timestamp obtainment field, a 8-bit egress clock source field, and a 8-bit egress timestamp obtainment field. The Session-Sender MUST set the Length value according to the number of explicitly listed intermediate nodes in the forward path. The Timestamp Info Tuple list [1..n] fields MUST be set to zero upon Session-Sender test packets transmission.

The HbH Timestamp Information TLV has the following format:

```
 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
+-------------------------------+---------------+---------------+
|    HbH Timestamp Info Type    |     Length    |   Node Left   |
+-------------------------------+---------------+---------------+
|                Timestamp Info Tuple list [1]                  |
+---------------------------------------------------------------+
~                              ...                              ~
+---------------------------------------------------------------+
|                Timestamp Info Tuple list [n]                  |
+---------------------------------------------------------------+
```

Fig. 6 HbH Timestamp Information TLV Format

where fields are defined as the following:

o HbH Timestamp Info Type: To be assigned by IANA.

o Length: A 8-bit field that indicates the length of the value portion in octets and will be a multiple of 4 octets dependent on the number of explicitly listed intermediate nodes in the forward path.

o Node Left: A 8-bit unsigned integer, which indicates the number of intermediate nodes remaining. That is, number of explicitly listed intermediate nodes still to be visited before reaching the destination node in the forward path. The Node Left field is set to n-1, where n is the number of intermediate nodes.

o Timestamp Info Tuple list [1..n]: A variable-length field, which record the source of clock synchronization and the method of timestamp obtainment at the ingress and egress when the test packet is received at and transmitted by the n-th intermediate node. The Timestamp Info Tuple list is encoded starting from the last intermediate node which is explicitly listed. That is, the first element of the Timestamp Info Tuple list [1] records the source of clock synchronization and the method of timestamp obtainment at the ingress and egress when the test packet is received at and transmitted by the last intermediate node of a explicit path, the second element records the penultimate Timestamp Info Tuple when the test packet received at and transmitted by the penultimate intermediate node of a explicit path, and so on.

The STAMP Session-Sender generates the STAMP test packet with the HbH Timestamp Information TLV. When an intermediate node receives the STAMP test packet, the node punts the packet to control plane and writes the source of clock synchronization and the method of timestamp obtainment at the Timestamp Info Tuple list [n] in the Session-Sender test packet. The mechanism of punting packet to control plane is outside the scope of this specification.

When the STAMP Session-Reflector received the test packet with the HbH Timestamp Information TLV, it MUST copy the HbH Timestamp Information TLV into the Session-Reflector test packet before its transmission. The HbH Timestamp Information TLV carried in STAMP test packet is usable to query timestamp information from every nodes in the forward path.

Note that the source of clock synchronization, NTP or PTP, is part of configuration of the Session-Sender/Reflector or a particular test session [RFC8762]. This draft recommends every intermediate nodes to be configured to use the same source of clock synchronization.

4. IANA Considerations

IANA is requested to allocate values for the following TLV Type from the "STAMP TLV Type" registry [I-D.ietf-ippm-stamp-option-tlv].

Code Point	Description	Reference
TBA1	IOAM Tracing Data TLV	This document
TBA2	Forward HbH Delay TLV	This document
TBA3	Backward HbH Delay TLV	This document
TBA4	HbH Packet Loss TLV	This document
TBA5	HbH BW Utilization TLV	This document
TBA6	HbH Timestamp Information TLV	This document

5. Security Considerations

This document extensions new optional TLVs to STAMP. It does not introduce any new security risks to STAMP.

6. Acknowledgements

The authors would like to thank Hongwei Yang, Giuseppe Fioccola and Chang Liu for the reviews and comments.

7. References

7.1. Normative References

[I-D.ietf-ippm-ioam-data]
"Data Fields for In-situ OAM", <https://datatracker.ietf.org/doc/draft-ietf-ippm-ioam-data/>.

[I-D.ietf-ippm-ioam-ipv6-options]
"In-situ OAM IPv6 Options", <https://datatracker.ietf.org/doc/draft-ietf-ippm-ioam-ipv6-options/>.

[I-D.ietf-ippm-stamp-option-tlv]
"Simple Two-way Active Measurement Protocol Optional Extensions", <https://datatracker.ietf.org/doc/draft-ietf-ippm-stamp-option-tlv/>.

[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>.

[RFC8762] "Simple Two-Way Active Measurement Protocol", <https://datatracker.ietf.org/doc/rfc8762/>.

7.2. Informative References

[IEEE.1588.2008]
"IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems", <https://ieeexplore.ieee.org/document/4579760>.

[RFC5905] "Network Time Protocol Version 4: Protocol and Algorithms Specification", <https://www.rfc-editor.org/info/rfc5905>.

Authors' Addresses

Yali Wang
Huawei
156 Beijing Rd., Haidian District
Beijing
China

Email: wangyali11@huawei.com

Tianran Zhou
Huawei
156 Beijing Rd., Haidian District
Beijing
China

Email: zhoutianran@huawei.com

Hongwei Yang
China Mobile
Beijing
China

Email: yanghongwei@chinamobile.com

Chang Liu
China Unicom
Beijing
China

Email: liuc131@chinaunicom.cn

IPPM Working Group
Internet-Draft
Intended status: Standards Track
Expires: March 8, 2021

X. Min
G. Mirsky
ZTE Corp.
L. Bo
China Telecom

September 4, 2020

Echo Request/Reply for Enabled In-situ OAM Capabilities
draft-xiao-ippm-ioam-conf-state-07

Abstract

This document describes an extension to the echo request/reply mechanisms used in IPv6, MPLS and SFC environments, which can be used within an IOAM domain, allowing the IOAM encapsulating node to acquire the enabled IOAM capabilities of each IOAM transit node and/or IOAM decapsulating node.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

The Data Fields for In-situ OAM (IOAM) [I-D.ietf-ippm-ioam-data] defines data fields for IOAM which records OAM information within the packet while the packet traverses a particular network domain, which is called an IOAM domain. IOAM can be used to complement OAM mechanisms based on, e.g., ICMP or other types of probe packets, and IOAM mechanisms can be leveraged where mechanisms using, e.g., ICMP do not apply or do not offer the desired results.

As specified in [I-D.ietf-ippm-ioam-data], within the IOAM-domain, the IOAM data may be updated by network nodes that the packet traverses. The device which adds an IOAM data container to the packet to capture IOAM data is called the "IOAM encapsulating node", whereas the device which removes the IOAM data container is referred to as the "IOAM decapsulating node". Nodes within the domain which are aware of IOAM data and read and/or write or process the IOAM data are called "IOAM transit nodes". Both the IOAM encapsulating node and the decapsulating node are referred to as domain edge devices, which can be hosts or network devices.

In order to add accurate IOAM data container to the packet, the IOAM encapsulating node needs to know the enabled IOAM capabilities at the

IOAM transit nodes and/or the IOAM decapsulating node as a whole, e.g., how many IOAM transit nodes will add tracing data and what kinds of data fields will be added.

This document describes an extension to the echo request/reply mechanisms used in IPv6, MPLS and SFC environments, which can be used within an IOAM domain, allowing the IOAM encapsulating node to acquire the enabled IOAM capabilities of each IOAM transit node and/or IOAM decapsulating node.

The following documents contain references to the echo request/reply mechanisms used in IPv6, MPLS and SFC environments:

o [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")

o [RFC8029] ("Detecting Multiprotocol Label Switched (MPLS) Data-Plane Failures")

o [I-D.ietf-sfc-multi-layer-oam] ("Active OAM for Service Function Chains in Networks")

This feature described in this document is assumedly applied to explicit path (strict or loose), because the precondition for this feature to work is that the echo request reaches each IOAM transit node as live traffic traverses.

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

E2E: Edge to Edge

ICMP: Internet Control Message Protocol

IOAM: In-situ Operations, Administration, and Maintenance

LSP: Label Switched Path

MPLS: Multi-Protocol Label Switching

MBZ: Must Be Zero

MTU: Maximum Transmission Unit

NTP: Network Time Protocol

OAM: Operations, Administration, and Maintenance

POSIX: Portable Operating System Interface

POT: Proof of Transit

PTP: Precision Time Protocol

SFC: Service Function Chain

TTL: Time to Live

3. IOAM Capabilities Formats

3.1. IOAM Capabilities TLV in Echo Request

In echo request IOAM Capabilities uses TLV (Type-Length-Value tuple) which have the following format:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Type = IOAM Capabilities   |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Namespace-IDs Length     |              MBZ              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.                     List of Namespace-IDs                     .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: IOAM Capabilities TLV in Echo Request

When this TLV is present in the echo request sent by an IOAM encapsulating node, it means that the IOAM encapsulating node requests the receiving node to reply with its enabled IOAM capabilities. If there is no IOAM capability to be reported by the receiving node, then this TLV SHOULD be ignored by the receiving node, which means the receiving node SHOULD send echo reply without IOAM capabilities or no echo reply, in the light of whether the echo request includes other TLV than IOAM Capabilities TLV. List of Namespace-IDs MAY be included in this TLV of echo request, it means that the IOAM encapsulating node requests only the IOAM capabilities which matches one of the Namespace-IDs. The Namespace-ID has the same definition as what's specified in [I-D.ietf-ippm-ioam-data].

Type is set to the value which indicates that it's an IOAM Capabilities TLV.

Length is the length of the TLV's Value field in octets, Namespace-IDs Length is the Length of the List of Namespace-IDs field in octets.

Value field of this TLV is zero padded to align to a 4-octet boundary.

3.2. IOAM Capabilities TLV in Echo Reply

In echo reply IOAM Capabilities uses TLV which have the following format:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Type = IOAM Capabilities   |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Sub-TLVs Length        |               MBZ             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.                       List of Sub-TLVs                        .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: IOAM Capabilities TLV in Echo Reply

When this TLV is present in the echo reply sent by an IOAM transit node and/or an IOAM decapsulating node, it means that IOAM function is enabled at this node and this TLV contains the enabled IOAM capabilities of the sender. List of Sub-TLVs which contain the IOAM capabilities SHOULD be included in this TLV of the echo reply. Note that the IOAM encapsulating node or the IOAM decapsulating node can also be an IOAM transit node.

Type is set to the value which indicates that it's an IOAM Capabilities TLV.

Length is the length of the TLV's Value field in octets, Sub-TLVs Length is the length of the List of Sub-TLVs field in octets.

Value field of this TLV or any Sub-TLV is zero padded to align to a 4-octet boundary. Based on the data fields for IOAM specified in [I-D.ietf-ippm-ioam-data], five kinds of Sub-TLVs are defined in this document, and in an IOAM Capabilities TLV the same kind of Sub-TLV can appear more times than one with different Namespace-ID. Note that the IOAM encapsulating node may receive both IOAM Pre-allocated Tracing Capabilities sub-TLV and IOAM Incremental Tracing Capabilities sub-TLV in the process of traceroute, which means both pre-allocated tracing node and incremental tracing node are on the same path, or some node supports both pre-allocated tracing and incremental tracing, the behavior of the IOAM encapsulating node in this scenario is outside the scope of this document.

3.2.1. IOAM Pre-allocated Tracing Capabilities sub-TLV

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Sub-type = Pre-allocated trace |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                IOAM-Trace-Type                |    Reserved   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Namespace-ID          |           Egress_MTU          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Egress_if_id (short or wide format)       ......             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: IOAM Pre-allocated Tracing Capabilities Sub-TLV

When this sub-TLV is present in the IOAM Capabilities TLV, it means that the sending node is an IOAM transit node and IOAM tracing function is enabled at this IOAM transit node.

Sub-type is set to the value which indicates that it's an IOAM Pre-allocated Tracing Capabilities sub-TLV.

Length is the length of the sub-TLV's Value field in octets, if Egress_if_id is in the short format which is 16 bits long, it MUST be set to 10, and if Egress_if_id is in the wide format which is 32 bits long, it MUST be set to 12.

IOAM-Trace-Type field has the same definition as what's specified in section 4.4 of [I-D.ietf-ippm-ioam-data].

Reserved field is reserved for future use and MUST be set to zero.

Namespace-ID field has the same definition as what's specified in section 4.4 of [I-D.ietf-ippm-ioam-data], it should be one of the Namespace-IDs listed in the IOAM Capabilities TLV of echo request.

Egress_MTU field has 16 bits and specifies the MTU of the egress direction out of which the sending node would forward the received echo request, it should be the MTU of the egress interface or the MTU between the sending node and the downstream IOAM transit node.

Egress_if_id field has 16 bits (in short format) or 32 bits (in wide format) and specifies the identifier of the egress interface out of which the sending node would forward the received echo request.

3.2.2. IOAM Incremental Tracing Capabilities sub-TLV

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Sub-type = Incremental trace |            Length              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               IOAM-Trace-Type                 |   Reserved    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Namespace-ID          |           Egress_MTU          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Egress_if_id (short or wide format)       ......             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4: IOAM Incremental Tracing Capabilities Sub-TLV

When this sub-TLV is present in the IOAM Capabilities TLV, it means that the sending node is an IOAM transit node and IOAM tracing function is enabled at this IOAM transit node.

Sub-type is set to the value which indicates that it's an IOAM Incremental Tracing Capabilities sub-TLV.

Length is the length of the sub-TLV's Value field in octets, if Egress_if_id is in the short format which is 16 bits long, it MUST be set to 10, and if Egress_if_id is in the wide format which is 32 bits long, it MUST be set to 12.

IOAM-Trace-Type field has the same definition as what's specified in section 4.4 of [I-D.ietf-ippm-ioam-data].

Reserved field is reserved for future use and MUST be set to zero.

Namespace-ID field has the same definition as what's specified in section 4.4 of [I-D.ietf-ippm-ioam-data], it should be one of the Namespace-IDs listed in the IOAM Capabilities TLV of echo request.

Egress_MTU field has 16 bits and specifies the MTU of the egress direction out of which the sending node would forward the received echo request, it should be the MTU of the egress interface or the MTU between the sending node and the downstream IOAM transit node.

Egress_if_id field has 16 bits (in short format) or 32 bits (in wide format) and specifies the identifier of the egress interface out of which the sending node would forward the received echo request.

3.2.3. IOAM Proof of Transit Capabilities sub-TLV

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Sub-type = POT Capabilities  |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Namespace-ID           | IOAM-POT-Type |P|SoR|Reserved |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 5: IOAM Proof of Transit Capabilities Sub-TLV

When this sub-TLV is present in the IOAM Capabilities TLV, it means that the sending node is an IOAM transit node and IOAM proof of transit function is enabled at this IOAM transit node.

Sub-type is set to the value which indicates that it's an IOAM Proof of Transit Capabilities sub-TLV.

Length is the length of the sub-TLV's Value field in octets, and MUST be set to 4.

Namespace-ID field has the same definition as what's specified in section 4.5 of [I-D.ietf-ippm-ioam-data], it should be one of the Namespace-IDs listed in the IOAM Capabilities TLV of echo request.

IOAM-POT-Type field and P bit have the same definition as what's specified in section 4.5 of [I-D.ietf-ippm-ioam-data]. If the IOAM encapsulating node receives IOAM-POT-Type and/or P bit values from an IOAM transit node that are different from its own, then the IOAM encapsulating node MAY choose to abandon the proof of transit function or to select one kind of IOAM-POT-Type and P bit, it's based on the policy applied to the IOAM encapsulating node.

SoR field has two bits which means the size of "Random" and "Cumulative" data, which are specified in section 4.5 of [I-D.ietf-ippm-ioam-data]. This document defines SoR as follow:

0b00 means 64-bit "Random" and 64-bit "Cumulative" data.

0b01~0b11: Reserved for future standardization

Reserved field is reserved for future use and MUST be set to zero.

3.2.4. IOAM Edge-to-Edge Capabilities sub-TLV

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Sub-type = E2E Capabilities  |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Namespace-ID           |         IOAM-E2E-Type         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|TSF|TSL|       Reserved        |              MBZ              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 6: IOAM Edge-to-Edge Capabilities Sub-TLV

When this sub-TLV is present in the IOAM Capabilities TLV, it means that the sending node is an IOAM decapsulating node and IOAM edge-to-edge function is enabled at this IOAM decapsulating node. That is to say, if the IOAM encapsulating node receives this sub-TLV, the IOAM encapsulating node can determine that the node which sends this sub-TLV is an IOAM decapsulating node.

Sub-type is set to the value which indicates that it's an IOAM Edge-to-Edge Capabilities sub-TLV.

Length is the length of the sub-TLV's Value field in octets, and MUST be set to 8.

Namespace-ID field has the same definition as what's specified in section 4.6 of [I-D.ietf-ippm-ioam-data], it should be one of the Namespace-IDs listed in the IOAM Capabilities TLV of echo request.

IOAM-E2E-Type field has the same definition as what's specified in section 4.6 of [I-D.ietf-ippm-ioam-data].

TSF field specifies the timestamp format used by the sending node. This document defines TSF as follow:

0b00: PTP timestamp format

0b01: NTP timestamp format

0b10: POSIX timestamp format

0b11: Reserved for future standardization

TSL field specifies the timestamp length used by the sending node. This document defines TSL as follow:

When TSF field is set to 0b00 which indicates PTP timestamp format:

0b00: 64-bit PTPv1 timestamp as defined in IEEE1588-2008 [IEEE1588v2]

0b01: 80-bit PTPv2 timestamp as defined in IEEE1588-2008 [IEEE1588v2]

0b10~0b11: Reserved for future standardization

When TSF field is set to 0b01 which indicates NTP timestamp format:

0b00: 32-bit NTP timestamp as defined in NTPv4 [RFC5905]

0b01: 64-bit NTP timestamp as defined in NTPv4 [RFC5905]

0b10: 128-bit NTP timestamp as defined in NTPv4 [RFC5905]

0b11: Reserved for future standardization

When TSF field is set to 0b10 or 0b11, the TSL field would be ignored.

Reserved field is reserved for future use and MUST be set to zero.

3.2.5. IOAM DEX Capabilities sub-TLV

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Sub-type = DEX Capabilities  |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               IOAM-Trace-Type                 |   Reserved    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Namespace-ID          |           Reserved            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 7: IOAM DEX Capabilities Sub-TLV

When this sub-TLV is present in the IOAM Capabilities TLV, it means that the sending node is an IOAM transit node and IOAM DEX function is enabled at this IOAM transit node.

Sub-type is set to the value which indicates that it's an IOAM DEX Capabilities sub-TLV.

Length is the length of the sub-TLV's Value field in octets, and MUST be set to 8.

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 3.2 of [I-D.ietf-ippm-ioam-direct-export], it should be one of the Namespace-IDs listed in the IOAM Capabilities TLV of echo request.

Reserved field is reserved for future use and MUST be set to zero.

3.2.6. IOAM End-of-Domain sub-TLV

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Sub-type = End of Domain    |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Namespace-ID           |              MBZ              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 8: IOAM End of Domain Sub-TLV

When this sub-TLV is present in the IOAM Capabilities TLV, it means that the sending node is an IOAM decapsulating node. That is to say, if the IOAM encapsulating node receives this sub-TLV, the IOAM encapsulating node can determine that the node which sends this sub-TLV is an IOAM decapsulating node. When the IOAM Edge-to-Edge Capabilities sub-TLV is present in the IOAM Capabilities TLV sent by the IOAM decapsulating node, the IOAM End-of-Domain sub-TLV doesn't need to be present in the same IOAM Capabilities TLV, otherwise the End-of-Domain sub-TLV MUST be present in the IOAM Capabilities TLV sent by the IOAM decapsulating node. Since both the IOAM Edge-to-Edge Capabilities sub-TLV and the IOAM End-of-Domain sub-TLV can be used to indicate that the sending node is an IOAM decapsulating node, it's recommended to include only the IOAM Edge-to-Edge Capabilities sub-TLV if IOAM edge-to-edge function is enabled at this IOAM decapsulating node.

Length is the length of the sub-TLV's Value field in octets, and MUST be set to 4.

Namespace-ID field has the same definition as what's specified in section 4.6 of [I-D.ietf-ippm-ioam-data], it should be one of the Namespace-IDs listed in the IOAM Capabilities TLV of echo request.

4. Operational Guide

Once the IOAM encapsulating node is triggered to acquire the enabled IOAM capabilities of each IOAM transit node and/or IOAM decapsulating node, the IOAM encapsulating node will send a batch of echo requests that include the IOAM Capabilities TLV, first with TTL equal to 1 to reach the nearest node which may be an IOAM transit node or not, then with TTL equal to 2 to reach the second nearest node which also may be an IOAM transit node or not, on the analogy of this to increase 1 to TTL every time the IOAM encapsulating node sends a new echo request, until the IOAM encapsulating node receives echo reply sent by the IOAM decapsulating node, which should contain the IOAM Capabilities TLV including the IOAM Edge-to-Edge Capabilities sub-TLV or the IOAM End-of-Domain sub-TLV. Alternatively, if the IOAM encapsulating node knows exactly all the IOAM transit nodes and/or IOAM decapsulating node beforehand, once the IOAM encapsulating node is triggered to acquire the enabled IOAM capabilities, it can send echo request to each IOAM transit node and/or 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 even the live user traffic, and the specific triggering mechanisms are outside the scope of this document.

Each IOAM transit node and/or IOAM decapsulating node that receives an echo request containing the IOAM Capabilities TLV will send an echo reply to the IOAM encapsulating node, and within the echo reply, there should be an IOAM Capabilities TLV containing one or more sub-TLVs. The IOAM Capabilities TLV contained in the echo request would be ignored by the receiving node that is unaware of IOAM.

5. Security Considerations

Knowledge of the state of the IOAM domain may be considered confidential. Implementations SHOULD provide a means of filtering the addresses to which echo request/reply may be sent.

6. IANA Considerations

This document has no IANA actions.

7. Acknowledgements

The authors would like to acknowledge Tianran Zhou for his careful review and helpful comments.

The authors appreciate the f2f discussion with Frank Brockners on this document.

8. 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-10 (work in progress), July 2020.

[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-01 (work in progress), August 2020.

[I-D.ietf-sfc-multi-layer-oam]
Mirsky, G., Meng, W., Khasnabish, B., and C. Wang, "Active OAM for Service Function Chains in Networks", draft-ietf-sfc-multi-layer-oam-06 (work in progress), June 2020.

[IEEE1588v2]
Institute of Electrical and Electronics Engineers, "IEEE Std 1588-2008 - IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems", IEEE Std 1588-2008, 2008, <http://standards.ieee.org/findstds/standard/1588-2008.html>.

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

[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>.

[RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <https://www.rfc-editor.org/info/rfc5905>.

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

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

[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>.

Authors' Addresses

Xiao Min
ZTE Corp.
Nanjing
China

Phone: +86 25 88013062
Email: xiao.min2@zte.com.cn

Greg Mirsky
ZTE Corp.
USA

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: January 13, 2021

T. Zhou, Ed.
G. Fioccola
Huawei
S. Lee
LG U+
M. Cociglio
Telecom Italia
W. Li
Huawei

July 12, 2020

Enhanced Alternate Marking Method
draft-zhou-ippm-enhanced-alternate-marking-05

Abstract

This document extends the IPv6 alternate marking option to provide the enhanced capabilities.

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

Copyright Notice

Table of Contents

1. Introduction

The Alternate Marking [RFC8321] and Multipoint Alternate Marking [I-D.ietf-ippm-multipoint-alt-mark] define the Alternate Marking technique that is an 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.

AltMark Option [I-D.ietf-6man-ipv6-alt-mark] applies the Alternate Marking Method for IPv6 protocol, and defines Extension Header Option to encode Alternate Marking Method for both Hop-by-Hop Options Header and Destination Options Header.

While the AltMark Option implement the basic alternate marking method, this document defines the extended data fields for the AltMark Option and provides the enhanced capabilities.

It is worth mentioning that the enhanced capabilities are intended for further use and are optional.

2. Data Fields Format

The following figure shows the data fields format for enhanced alternate marking. This data is expected to be encapsulated to specific transports.

```
 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
+---------------------------------------+-+-+-+-+---------------+
|           FlowMonID                   |L|D|R|R|  NextHeader   |
+---------------------------------------+-+-+-+-+---------------+
|           FlowMonID Ext               |R|R|F|R| Len   |   R   |
+---------------------------------------+-+-+-+-+---------------+
|           MetaInfo            |             R                 |
+---------------------------------------------------------------+
```

where:

o FlowMonID - Flow Monitoring Identification is the same as defined in AltMark Option [I-D.ietf-6man-ipv6-alt-mark].

o L and D - Loss Flag and Delay Flag are the same as defined in AltMark Option [I-D.ietf-6man-ipv6-alt-mark].

o NextHeader - Identify whether to carry the extended data fields.

o FlowMonID Ext - 20 bits unsigned integer. This used to extend the FlowMonID to reduce the conflict when random allocation is applied

o R - Reserved for further use. This bit MUST be set to zero.

o F - Flow direction identification. F = 1, indicate the flow direction is forward.

o Len - Length. It indicates the length of extension headers.

o MetaInfo - A 16 bits Bitmap to indicate more meta data attached for the enhanced function.

3. Enhanced Alternate Marking capabilities

The extended data fields presented in the previous section can be used for several uses. Some possible applications can be:

1. shortest marking periods of single marking method for thicker packet loss measurements.

2. more dense delay measurements than double marking method (down to each packet).

3. increase the entropy of flow monitoring identifier by extending the size of FlowMonID.

4. and so on.

4. Security Considerations

TBD

5. IANA Considerations

This document has no request to IANA.

6. References

6.1. Normative References

[I-D.ietf-ippm-multipoint-alt-mark]
Fioccola, G., Cociglio, M., Sapio, A., and R. Sisto, "Multipoint Alternate Marking method for passive and hybrid performance monitoring", draft-ietf-ippm-multipoint-alt-mark-09 (work in progress), March 2020.

[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>.

[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>.

6.2. 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", draft-ietf-6man-ipv6-alt-mark-01 (work in progress), June 2020.

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

Shinyoung Lee
LG U+
71, Magokjungang 8-ro, Gangseo-gu
Seoul
Republic of Korea

Email: leesy@lguplus.co.kr

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

Email: mauro.cociglio@telecomitalia.it

Weidong Li
Huawei
156 Beiqing Rd.
Beijing 100095
China

Email: poly.li@huawei.com

IPPM
Internet-Draft
Intended status: Standards Track
Expires: January 31, 2021

T. Zhou, Ed.
Huawei
J. Guichard
Futurewei
F. Brockners
S. Raghavan
Cisco Systems

July 30, 2020

A YANG Data Model for In-Situ OAM
draft-zhou-ippm-ioam-yang-08

Abstract

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

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

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

This document defines a data model for IOAM capabilities using the YANG data modeling language [RFC7950]. This YANG model supports all the five IOAM options, which are Incremental Tracing Option, Pre-allocated Tracing Option, Direct Export Option[I-D.ietf-ippm-ioam-direct-export], Proof of Transit(PoT) Option, and Edge-to-Edge Option.

2. Conventions used in this document

The keywords "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

BCP14, [RFC2119], [RFC8174] when, and only when, they appear in all capitals, as shown here.

The following terms are defined in [RFC7950] and are used in this specification:

- augment
- data model
- data node

The terminology for describing YANG data models is found in [RFC7950].

2.1. Tree Diagrams

Tree diagrams used in this document follow the notation defined in [RFC8340].

3. Design of the IOAM YANG Data Model

3.1. Profiles

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

```
module: ietf-ioam
   +--rw ioam
      +--rw ioam-profiles
         +--rw admin-config
         |  +--rw enabled?   boolean
         +--rw ioam-profile* [profile-name]
            +--rw profile-name                    string
            +--rw filter
            |  +--rw filter-type?   ioam-filter-type
            |  +--rw acl-name?      -> /acl:acls/acl/name
            +--rw protocol-type?                  ioam-protocol-type
            +--rw incremental-tracing-profile {incremental-trace}?
            |  ...
            +--rw preallocated-tracing-profile {preallocated-trace}?
            |  ...
            +--rw direct-export-profile {direct-export}?
            |  ...
            +--rw pot-profile {proof-of-transit}?
            |  ...
            +--rw e2e-profile {edge-to-edge}?
               ...
```

The "enabled" is an administrative configuration. When it is set to true, IOAM configuration is enabled for the system. Meanwhile, the IOAM data-plane functionality is enabled.

The "filter" is used to identify a flow, where the IOAM profile can apply. There may be multiple filter types. ACL [RFC8519] is the default one.

The IOAM data can be encapsulated into multiple protocols, e.g., IPv6 [I-D.ietf-ippm-ioam-ipv6-options] and NSH [I-D.ietf-sfc-ioam-nsh]. The "protocol-type" is used to indicate where the IOAM is applied. For example, if the "protocol-type" is IPv6, the IOAM ingress node will encapsulate the associated flow with the IPv6-IOAM [I-D.ietf-ippm-ioam-ipv6-options] format.

IOAM data includes five encapsulation types, i.e., incremental tracing data, preallocated tracing data, direct export data, prove of transit data and end to end data. In practice, multiple IOAM data types can be encapsulated into the same IOAM header. The "ioam-profile" contains a set of sub-profiles, each of which relates to one encapsulation type. The configured object may not support all the sub-profiles. The supported sub-profiles are indicated by 5 defined features, i.e., "incremental-trace", "preallocated-trace", "direct export", "proof-of-transit", "edge-to-edge".

3.2. Preallocated Tracing Profile

The IOAM tracing data is expected to be collected at every node that a packet traverses to ensure visibility into the entire path a packet takes within an IOAM domain. The preallocated tracing option will create pre-allocated space for each node to populate its information . The "preallocated-tracing-profile" contains the detailed information for the preallocated tracing data. The information includes:

o enabled: indicates whether the preallocated tracing profile is enabled.

o node-action: indicates the operation (e.g., encapsulate IOAM header, transit the IOAM data, or decapsulate IOAM header) applied to the dedicated flow.

o use-namespace: indicate the namespace used for the trace types.

o trace-type: indicates the per-hop data to be captured by the IOAM enabled nodes and included in the node data list.

o Loopback mode is used to send a copy of a packet back towards the source.

o Active mode indicates that a packet is used for active measurement.

```
+--rw preallocated-tracing-profile {preallocated-trace}?
   +--rw enabled?                boolean
   +--rw node-action?            ioam-node-action
   +--rw trace-types
   |  +--rw use-namespace?   ioam-namespace
   |  +--rw trace-type*      ioam-trace-type
   +--rw enable-loopback-mode?   boolean
   +--rw enable-active-mode?   boolean
```

3.3. Incremental Tracing Profile

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

```
+--rw incremental-tracing-profile {incremental-trace}?
   +--rw enabled?               boolean
   +--rw node-action?           ioam-node-action
   +--rw trace-types
   |  +--rw use-namespace?   ioam-namespace
   |  +--rw trace-type*   ioam-trace-type
   +--rw enable-loopback-mode?   boolean
   +--rw enable-active-mode?   boolean
   +--rw max-length?            uint32
```

3.4. Direct Export Profile

The direct export option is used as a trigger for IOAM nodes to export IOAM data to a receiving entity (or entities). The "direct-export-profile" contains the detailed information for the direct export data. The detailed information is the same as the Preallocated Tracing Profile, but with one more optional variable, "flow-id", which is used to correlate the exported data of the same flow from multiple nodes and from multiple packets.

```
+--rw direct-export-profile {direct-export}?
   +--rw enabled?               boolean
   +--rw node-action?           ioam-node-action
   +--rw trace-types
   |  +--rw use-namespace?   ioam-namespace
   |  +--rw trace-type*      ioam-trace-type
   +--rw enable-loopback-mode?   boolean
   +--rw enable-active-mode?   boolean
   +--rw flow-id?            uint32
```

3.5. Proof of Transit Profile

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

```
+--rw pot-profile {proof-of-transit}?
   +--rw enabled?               boolean
   +--rw active-profile-index?   pot:profile-index-range
   +--rw pot-profile-list* [pot-profile-index]
      +--rw pot-profile-index    profile-index-range
      +--rw prime-number         uint64
      +--rw secret-share         uint64
      +--rw public-polynomial    uint64
      +--rw lpc                  uint64
      +--rw validator?           boolean
      +--rw validator-key?       uint64
      +--rw bitmask?             uint64
          +--rw opot-masks
         +--rw downstream-mask*   uint64
         +--rw upstream-mask*     uint64
```

3.6. Edge to Edge Profile

The IOAM edge to edge option is to carry data that is added by the IOAM encapsulating node and interpreted by IOAM decapsulating node. The "e2e-profile" contains the detailed information for the edge to edge data. The detailed information includes:

- o enabled: indicates whether the edge to edge profile is enabled.

- o node-action is the same semantic as in Section 2.2.

- o use-namespace: indicate the namespace used for the edge to edge types.

- o e2e-type indicates data to be carried from the ingress IOAM node to the egress IOAM node.

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

4. IOAM YANG Module

```
<CODE BEGINS> file "ietf-ioam@2020-07-13.yang"
module ietf-ioam {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-ioam";
  prefix "ioam";
```

```
   import ietf-pot-profile {
     prefix "pot";
     reference "draft-ietf-sfc-proof-of-transit";
   }

   import ietf-access-control-list {
     prefix "acl";
     reference
       "RFC 8519: YANG Data Model for Network Access Control
        Lists (ACLs)";
   }

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

   contact
     "WG Web: <http://tools.ietf.org/wg/ippm>
      WG List: <ippm@ietf.org>
      Editor: zhoutianran@huawei.com
      Editor: james.n.guichard@futurewei.com
      Editor: fbrockne@cisco.com
      Editor: srihari@cisco.com";

   description
     "This YANG module specifies a vendor-independent data
      model for the In Situ OAM (IOAM).

     Copyright (c) 2020 IETF Trust and the persons identified as
     authors of the code.  All rights reserved.

     Redistribution and use in source and binary forms, with or
     without modification, is permitted pursuant to, and subject
     to the license terms contained in, the Simplified BSD License
     set forth in Section 4.c of the IETF Trust's Legal Provisions
     Relating to IETF Documents
     (http://trustee.ietf.org/license-info).

     This version of this YANG module is part of RFC XXXX; see the
     RFC itself for full legal notices.";

   revision 2020-07-13 {
     description "Initial revision.";
     reference "draft-zhou-ippm-ioam-yang";
   }

  /*
   * FEATURES
   */
```

```
   feature incremental-trace
   {
     description
       "This feature indicated that the incremental tracing option is
        supported";
     reference "draft-ietf-ippm-ioam-data";
   }

   feature preallocated-trace
   {
     description
       "This feature indicated that the preallocated tracing option is
        supported";
     reference "draft-ietf-ippm-ioam-data";
   }

   feature direct-export
   {
     description
       "This feature indicated that the direct export option is
        supported";
     reference "ietf-ippm-ioam-direct-export";
   }

   feature proof-of-transit
   {
     description
       "This feature indicated that the proof of transit option is
        supported";
     reference "draft-ietf-ippm-ioam-data";
   }

   feature edge-to-edge
   {
     description
       "This feature indicated that the edge to edge option is
        supported";
     reference "draft-ietf-ippm-ioam-data";
   }

  /*
   * IDENTITIES
   */
   identity base-filter {
     description
       "Base identity to represent a filter. A filter is used to
       specify the flow to apply the IOAM profile. ";
   }
```

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

   identity base-protocol {
     description
       "Base identity to represent the carrier protocol. It's used to
        indicate what layer and protocol the IOAM data is embedded.";
   }

   identity ipv6-protocol {
     base base-protocol;
     description
       "The described IOAM data is embedded in IPv6 protocol.";
     reference "ietf-ippm-ioam-ipv6-options";
   }

   identity nsh-protocol  {
     base base-protocol;
     description
       "The described IOAM data is embedded in NSH.";
     reference "ietf-sfc-ioam-nsh";
   }

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

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

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

   identity action-decapsulate {
     base base-node-action;
     description
       "indicate the node is to decapsulate the IOAM packet";
```

```
   }

   identity base-trace-type {
     description
       "Base identity to represent trace types";
   }

   identity trace-hop-lim-node-id {
     base base-trace-type;
     description
       "indicates presence of Hop_Lim and node_id in the
        node data.";
   }

   identity trace-if-id {
     base base-trace-type;
     description
       "indicates presence of ingress_if_id and egress_if_id in the
        node data.";
   }

   identity trace-timestamp-seconds {
     base base-trace-type;
     description
       "indicates presence of time stamp seconds in the node data.";
   }

   identity trace-timestamp-nanoseconds {
     base base-trace-type;
     description
       "indicates presence of time stamp nanoseconds in the node data.";
   }

   identity trace-transit-delay {
     base base-trace-type;
     description
       "indicates presence of transit delay in the node data.";
   }

   identity trace-namespace-data {
     base base-trace-type;
     description
       "indicates presence of namespace specific data (short format)
        in the node data.";
   }

   identity trace-queue-depth {
     base base-trace-type;
```

```
     description
       "indicates presence of queue depth in the node data.";
   }

   identity trace-opaque-state-snapshot {
     base base-trace-type;
     description
       "indicates presence of variable length Opaque State Snapshot
        field.";
   }

   identity trace-hop-lim-node-id-wide {
     base base-trace-type;
     description
       "indicates presence of Hop_Lim and node_id wide in the
        node data.";
   }

   identity trace-if-id-wide {
     base base-trace-type;
     description
       "indicates presence of ingress_if_id and egress_if_id wide in
        the node data.";
   }

   identity trace-namespace-data-wide {
     base base-trace-type;
     description
       "indicates presence of namespace specific data in wide format
        in the node data.";
   }

   identity trace-buffer-occupancy {
     base base-trace-type;
     description
       "indicates presence of buffer occupancy in the node data.";
   }

   identity trace-checksum-complement {
     base base-trace-type;
     description
       "indicates presence of the Checksum Complement node data.";
   }

   identity base-pot-type {
     description
       "Base identity to represent Proof of Transit(PoT) types";
   }
```

```
   identity pot-bytes-16 {
     base base-pot-type;
     description
       "POT data is a 16 Octet field.";
   }

   identity base-e2e-type {
     description
       "Base identity to represent e2e types";
   }

   identity e2e-seq-num-64 {
     base base-e2e-type;
     description
       "indicates presence of a 64-bit sequence number";
   }

   identity e2e-seq-num-32 {
     base base-e2e-type;
     description
       "indicates presence of a 32-bit sequence number";
   }

   identity e2e-timestamp-seconds {
     base base-e2e-type;
     description
       "indicates presence of timestamp seconds for the
        transmission of the frame";
   }

   identity e2e-timestamp-subseconds {
     base base-e2e-type;
     description
       "indicates presence of timestamp subseconds for the
        transmission of the frame";
   }

   identity base-namespace {
     description
       "Base identity to represent the namespace";
   }

   identity namespace-ietf {
     base base-namespace;
     description
       "namespace that specified in IETF.";
   }
```

```
  /*
   * TYPE DEFINITIONS
   */

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

   typedef ioam-protocol-type {
     type identityref {
       base base-protocol;
     }
     description
       "Specifies a known type of carrier protocol for the IOAM data.";
   }

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

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

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

   typedef ioam-e2e-type {
     type identityref {
       base base-e2e-type;
     }
```

```
     description
       "Specifies a known e2e type.";
   }

   typedef ioam-namespace {
     type identityref {
       base base-namespace;
     }
     description
       "Specifies the supported namespace.";
   }

  /*
   * GROUP DEFINITIONS
   */

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

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

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

   grouping encap-tracing {
     description
       "A grouping for the generic configuration for
        tracing profile.";

     container trace-types {
       description
         "the list of trace types for encapsulate";

       leaf use-namespace {
         type ioam-namespace;
         description
           "the namespace used for the encapsulation";
       }
```

```
       leaf-list trace-type {
         type ioam-trace-type;
         description
           "The trace type is only defined at the encapsulation node.";
       }
     }

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

     leaf enable-active-mode {
       type boolean;
       default false;
       description
         "Active mode indicates that a packet is used for active
          measurement. An IOAM decapsulating node that receives a
          packet with the Active flag set in one of its Trace options
          must terminate the packet.";
     }
   }

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

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

     uses encap-tracing {
       when "node-action = 'ioam:action-encapsulate'";
     }

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

```
    }
  }

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


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

    uses encap-tracing {
      when "node-action = 'ioam:action-encapsulate'";
    }
  }

  grouping ioam-direct-export-profile {
    description
      "A grouping for direct export profile.";

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

    uses encap-tracing {
      when "node-action = 'ioam:action-encapsulate'";
    }

    leaf flow-id {
      when "../node-action = 'ioam:action-encapsulate'";
      type uint32;
      description
        "flow-id is used to correlate the exported data of the same
         flow from multiple nodes and from multiple packets.";
    }
  }

  grouping ioam-e2e-profile {
    description
      "A grouping for end to end profile.";

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

```
      }

      container e2e-types {
        when "../node-action = 'ioam:action-encapsulate'";
        description
          "the list of e2e types for encapsulate";

        leaf use-namespace {
          type ioam-namespace;
          description
            "the namespace used for the encapsulation";
        }

        leaf-list e2e-type {
          type ioam-e2e-type;
          description
            "The e2e type is only defined at the encapsulation node.";
        }
      }
    }

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

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

   /*
    * DATA NODES
    */

    container ioam {
      description "IOAM top level container";

      container ioam-profiles {
        description
          "Contains a list of IOAM profiles.";

        container admin-config {
          description
            "Contains all the administrative configurations related to
```

```
             the IOAM functionalities and all the IOAM profiles.";

          uses ioam-admin-config;
        }

        list ioam-profile {
          key "profile-name";
          ordered-by user;
          description
            "A list of IOAM profiles that configured on the node.";

          leaf profile-name {
            type string;
            mandatory true;
            description
              "Unique identifier for each IOAM profile";
          }

          container filter {
            uses ioam-filter;
            description
              "The filter which is used to indicate the flow to apply
              IOAM.";
          }

          leaf protocol-type {
            type ioam-protocol-type;
            description
              "This item is used to indicate the carrier protocol where
              the IOAM is applied.";
          }

          container incremental-tracing-profile {
            if-feature incremental-trace;
            description
              "describe the profile for incremental tracing option";

            leaf enabled {
              type boolean;
              default false;
              description
                "When true, apply incremental tracing option to the
                 specified flow identified by the filter.";
            }

            uses ioam-incremental-tracing-profile;
          }
```

```
         container preallocated-tracing-profile {
           if-feature preallocated-trace;
           description
             "describe the profile for preallocated tracing option";

           leaf enabled {
             type boolean;
             default false;
             description
               "When true, apply preallocated tracing option to the
                specified flow identified by the following filter.";
           }

           uses ioam-preallocated-tracing-profile;
         }

         container direct-export-profile {
           if-feature direct-export;
           description
             "describe the profile for direct-export option";

           leaf enabled {
             type boolean;
             default false;
             description
               "When true, apply direct-export option to the
                specified flow identified by the following filter.";
           }

           uses ioam-direct-export-profile;
         }

         container pot-profile {
           if-feature proof-of-transit;
           description
             "describe the profile for PoT option";

           leaf enabled {
             type boolean;
             default false;
             description
               "When true, apply Proof of Transit option to the
                specified flow identified by the following filter.";
           }

           leaf active-profile-index {
             type pot:profile-index-range;
             description
```

```
               "Proof of transit profile index that is currently
                active. Will be set in the first hop of the path
                or chain. Other nodes will not use this field.";
          }

          uses pot:pot-profile;
        }

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

          leaf enabled {
            type boolean;
            default false;
            description
              "When true, apply End to end option to the
               specified flow identified by the following filter.";
          }

          uses ioam-e2e-profile;
        }
      }
    }
  }
}
<CODE ENDS>
```

5. Security Considerations

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC5246].

The NETCONF access control model [RFC6536] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

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

o /ioam/ioam-profiles/admin-config

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

o /ioam/ioam-profiles/ioam-profile

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

6. IANA Considerations

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

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

```
URI: urn:ietf:params:xml:ns:yang:ietf-ioam
Registrant Contact: The IESG.
XML: N/A; the requested URI is an XML namespace.
```

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

```
name: ietf-ioam
namespace: urn:ietf:params:xml:ns:yang:ietf-ioam
prefix: ioam
reference: RFC XXXX
```

7. Acknowledgements

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

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", draft-ietf-ippm-ioam-data-10 (work in progress), July 2020.

[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-00 (work in progress), February 2020.

[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-06 (work in progress), June 2020.

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

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <https://www.rfc-editor.org/info/rfc3688>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <https://www.rfc-editor.org/info/rfc5246>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <https://www.rfc-editor.org/info/rfc6242>.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, DOI 10.17487/RFC6536, March 2012, <https://www.rfc-editor.org/info/rfc6536>.

[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>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

[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>.

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <https://www.rfc-editor.org/info/rfc8340>.

[RFC8519] Jethanandani, M., Agarwal, S., Huang, L., and D. Blair, "YANG Data Model for Network Access Control Lists (ACLs)", RFC 8519, DOI 10.17487/RFC8519, March 2019, <https://www.rfc-editor.org/info/rfc8519>.

8.2. Informative References

[I-D.ietf-ippm-ioam-ipv6-options]
Bhandari, S., Brockners, F., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Kfir, A., Gafni, B., Lapukhov, P., Spiegel, M., Krishnan, S., and R. Asati, "In-situ OAM IPv6 Options", draft-ietf-ippm-ioam-ipv6-options-02 (work in progress), July 2020.

[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-04 (work in progress), June 2020.

Authors' Addresses

Tianran Zhou
Huawei
156 Beiqing Rd.
Beijing 100095
China

Email: zhoutianran@huawei.com

Jim Guichard
Futurewei
United States of America

Email: james.n.guichard@futurewei.com

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

Email: fbrockne@cisco.com

Srihari Raghavan
Cisco Systems
Tril Infopark Sez, Ramanujan IT City
Neville Block, 2nd floor, Old Mahabalipuram Road
Chennai, Tamil Nadu 600113
India

Email: srihari@cisco.com