

ippm
Internet-Draft
Intended status: Standards Track
Expires: February 1, 2022

F. Brockners
Cisco
S. Bhandari
Thoughtspot
T. Mizrahi
Huawei
July 31, 2021

Integrity of In-situ OAM Data Fields
draft-brockners-ippm-ioam-data-integrity-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. IOAM deployments could require ensuring the integrity of IOAM data fields. This document specifies methods to ensure the integrity of IOAM data fields.

Status of This Memo

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

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

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

This Internet-Draft will expire on February 1, 2022.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

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

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Conventions	4
3. Threat Analysis	4
3.1. Modification: IOAM Data Fields	5
3.2. Modification: IOAM Option-Type Headers	5
3.3. Injection: IOAM Data Fields	6
3.4. Injection: IOAM Option-Type Headers	6
3.5. Replay	6
3.6. Management and Exporting	6
3.7. Delay	7
3.8. Threat Summary	7
4. Methods of providing integrity to IOAM data fields	8
4.1. Integrity Protected IOAM Option-Types	8
4.2. Subheader for Integrity Protected IOAM Option-Types	9
4.3. Space optimized symmetric key based signing of IOAM data	11
4.3.1. Overhead consideration	11
4.4. Space optimized asymmetric key based signing of trace data	12
4.4.1. Overhead consideration	12
5. IANA Considerations	12
5.1. IOAM Option-Type Registry	13
5.2. IOAM Integrity Protection Algorithm Suite Registry	13
6. Security Considerations	14
7. Acknowledgements	14
8. References	14
8.1. Normative References	14
8.2. Informative References	14
Authors' Addresses	16

1. Introduction

"In-situ" Operations, Administration, and Maintenance (IOAM) records OAM information within the packet while the packet traverses a particular network domain. The term "in-situ" refers to the fact that the OAM data is added to the data packets rather than is being sent within packets specifically dedicated to OAM. IOAM is to complement mechanisms such as Ping, Traceroute, or other active probing mechanisms. In terms of "active" or "passive" OAM, "in-situ" OAM can be considered a hybrid OAM type. "In-situ" mechanisms do not require extra packets to be sent. IOAM adds information to the already available data packets and therefore cannot be considered

passive. In terms of the classification given in [RFC7799] IOAM could be portrayed as Hybrid Type I. IOAM mechanisms can be leveraged where mechanisms using e.g., ICMP do not apply or do not offer the desired results, such as proving that a certain traffic flow takes a pre-defined path, SLA verification for the live data traffic, detailed statistics on traffic distribution paths in networks that distribute traffic across multiple paths, or scenarios in which probe traffic is potentially handled differently from regular data traffic by the network devices.

The current [I-D.ietf-ippm-ioam-data] assumes that IOAM is deployed in limited domains, where an operator has means to select, monitor, and control the access to all the networking devices, making the domain a trusted network. As such, IOAM tracing data is carried in the packets in clear and there are no protections against any node or middlebox tampering with the data. As a consequence, IOAM tracing data collected in an untrusted or semi-trusted environments cannot be trusted for critical operational decisions. Any rogue or unauthorized change to IOAM data fields in a user packet cannot be detected.

Recent discussions following the IETF last call on [I-D.ietf-ippm-ioam-data] revealed that there might be uses of IOAM where integrity protection of IOAM data fields is at least desirable, knowing that IOAM data fields integrity protection would incur extra effort in the data path of a device processing IOAM data fields. As such, the following additional considerations and requirements are to be taken into account in addition to addressing the problem of detectability of any integrity breach of the IOAM trace data collected:

1. IOAM trace data is processed by the data plane, hence viability of any method to prove integrity of the IOAM trace data must be feasible at data plane processing/forwarding rates (IOAM data might be applied to all traffic a router forwards).
2. IOAM trace data is carried within data packets. Additional space required to prove integrity of the data needs to be optimal, i.e. should not exceed the MTU or have adverse affect on packet processing.
3. Replay protection of older IOAM trace data should be possible. Without replay protection a rogue node can present the old IOAM trace data masking any ongoing network issues/activity making the IOAM trace data collection useless.

This document is to assist the IPPM working group in designing and specifying a solution for those deployments where the integrity of

IOAM data fields is a concern. This document proposes several methods to achieve integrity protection for IOAM data fields.

The discussion of the different methods to protect the integrity of IOAM data fields focuses mostly on protecting the integrity of IOAM Option-Types specified in [I-D.ietf-ippm-ioam-data], though the specified methods are not limited to these IOAM Option-Types. The methods could be applied to other IOAM Option-Types such as the DEX [I-D.ietf-ippm-ioam-direct-export] Option-Type.

2. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174]

Abbreviations used in this document:

IOAM: In-situ Operations, Administration, and Maintenance

MTU: Maximum Transmit Unit

OAM: Operations, Administration, and Maintenance

POT: Proof of Transit

SFC: Service Function Chain

3. Threat Analysis

This section presents a threat analysis of integrity-related threats in the context of IOAM. The threats that are discussed are assumed to be independent of the lower layer protocols; it is assumed that threats at other layers are handled by security mechanisms that are deployed at these layers.

This document is focused on integrity protection for IOAM data fields. Thus the threat analysis includes threats that are related to or result from compromising the integrity of IOAM data fields. Other security aspects such as confidentiality are not within the scope of this document.

Throughout the analysis there is a distinction between on-path and off-path attackers. As discussed in [I-D.ietf-detnet-security], on-path attackers are located in a position that allows interception and modification of in-flight protocol packets, whereas off-path attackers can only attack by generating protocol packets.

The analysis also includes the impact of each of the threats. Generally speaking, the impact of a successful attack on an OAM protocol [RFC7276] is a false illusion of nonexistent failures or preventing the detection of actual ones; in both cases, the attack may result in denial of service (DoS). Furthermore, creating the false illusion of a nonexistent issue may trigger unnecessary processing in some of the IOAM nodes along the path, and may cause more IOAM-related data to be exported to the management plane than is conventionally necessary. Beyond these general impacts, threat-specific impacts are discussed in each of the subsections below.

3.1. Modification: IOAM Data Fields

Threat

An attacker can maliciously modify the IOAM data fields of in-transit packets. The modification can either be applied to all packets or selectively applied to a subset of the en route packets. This threat is applicable to on-path attackers.

Impact

By systematically modifying the IOAM data fields of some or all of the in-transit packets an attacker can create a false picture of the paths in the network, the existence of faulty nodes and their location, and the network performance.

3.2. Modification: IOAM Option-Type Headers

Threat

An on-path attacker can modify IOAM data fields in one or more of the IOAM Option-Type headers in order to change or disrupt the behavior of nodes processing IOAM data fields along the path.

Impact

Changing the header of IOAM Option-Types may have several implications. An attacker can maliciously increase the processing overhead in nodes that process IOAM data fields and increase the on-the-wire overhead of IOAM data fields, for example by modifying the IOAM-Trace-Type field in the IOAM Trace-option header. An attacker can also prevent some of the nodes that process IOAM data fields from incorporating IOAM data fields by modifying the RemainingLen field.

3.3. Injection: IOAM Data Fields

Threat

An attacker can inject packets with IOAM Option-Types and IOAM data fields. This threat is applicable to both on-path and off-path attackers.

Impact

This attack and its impacts are similar to Section 3.1.

3.4. Injection: IOAM Option-Type Headers

Threat

An attacker can inject packets with IOAM Option-Type headers, thus manipulating other nodes that process IOAM data fields in the network. This threat is applicable to both on-path and off-path attackers.

Impact

This attack and its impacts are similar to Section 3.2.

3.5. Replay

Threat

An attacker can replay packets with IOAM data fields. Specifically, an attacker may replay a previously transmitted IOAM Option-Type with a new data packet, thus attaching old IOAM data fields to a fresh user packet. This threat is applicable to both on-path and off-path attackers.

Impact

As with previous threats, this threat may create a false image of a nonexistent failure, or may overload nodes which process IOAM data fields with unnecessary processing.

3.6. Management and Exporting

Threat

Attacks that compromise the integrity of IOAM data fields can be applied at the management plane, e.g., by manipulating network management packets. Furthermore, the integrity of IOAM data

fields that are exported to a receiving entity can also be compromised. Management plane attacks are not within the scope of this document; the network management protocol is expected to include inherent security capabilities. The integrity of exported data is also not within the scope of this document. It is expected that the specification of the export format will discuss the relevant security aspects.

Impact

Malicious manipulation of the management protocol can cause nodes that process IOAM data fields to malfunction, to be overloaded, or to incorporate unnecessary IOAM data fields into user packets. The impact of compromising the integrity of exported IOAM data fields is similar to the impacts of previous threats that were described in this section.

3.7. Delay

Threat

An on-path attacker may delay some or all of the in-transit packets that include IOAM data fields in order to create the false illusion of congestion. Delay attacks are well known in the context of deterministic networks [I-D.ietf-detnet-security] and synchronization [RFC7384], and may be somewhat mitigated in these environments by using redundant paths in a way that is resilient to an attack along one of the paths. This approach does not address the threat in the context of IOAM, as it does not meet the requirement to measure a specific path or to detect a problem along the path. It is noted that this threat is not within the scope of the threats that are mitigated in the scope of this document.

Impact

Since IOAM can be applied to a fraction of the traffic, an attacker can detect and delay only the packets that include IOAM data fields, thus preventing the authenticity of delay and load measurements.

3.8. Threat Summary

Threat	In scope	Out of scope
Modification: IOAM Data Fields	+	
Modification: IOAM Option-Type Headers	+	
Injection: IOAM Data Fields	+	
Injection: IOAM Option-Type Headers	+	
Replay	+	
Management and Exporting		+
Delay		+

Figure 1: Threat Analysis Summary

4. Methods of providing integrity to IOAM data fields

This section specifies additional IOAM Option-Types to carry data fields to provide for integrity protection. Methods for integrity protection can leverage symmetric or asymmetric key based signatures as described in the sub-sections below.

4.1. Integrity Protected IOAM Option-Types

Each of the IOAM Options defined in [I-D.ietf-ippm-ioam-data] are extended to include Integrity Protected (IP) IOAM Option-Types by allocating the following IOAM Option-Types in the IOAM Option-Type registry.

64 IOAM Pre-allocated Trace Integrity Protected Option-Type corresponds to IOAM Pre-allocated Trace Option-Type with integrity protection.

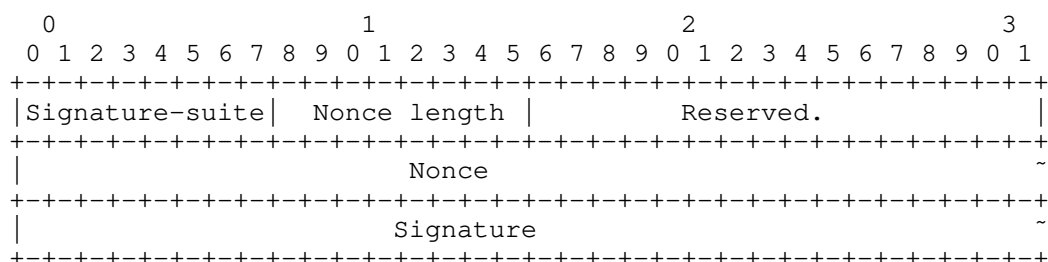
65 IOAM Incremental Trace Integrity Protected Option-Type corresponds to IOAM Incremental Trace Option-Type with integrity protection.

66 IOAM POT Integrity Protected Option-Type corresponds to IOAM POT Option-Type with integrity protection.

67 IOAM E2E Integrity Protected Option-Type corresponds to IOAM E2E Option-Type with integrity protection.

4.2. Subheader for Integrity Protected IOAM Option-Types

An integrity data sub-header is used in IOAM Integrity Protected Options. It is defined as follows:



Signature-suite: 8-bit unsigned integer. This field defines the algorithms used to compute the digest and the signature over the Option-Type header and data fields excluding the Signature field.

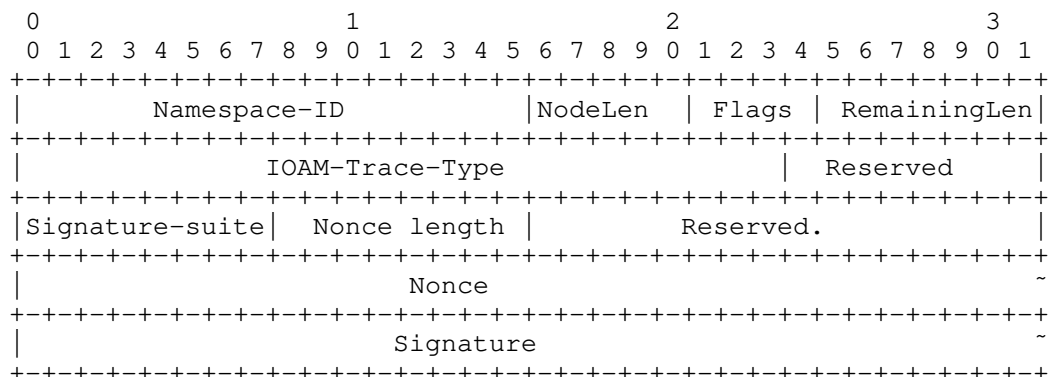
Nonce length: 8-bit unsigned integer. This field specifies the length of the Nonce field in octets.

Reserved: 16-bit Reserved field. MUST be set to zero upon transmission and ignored upon receipt.

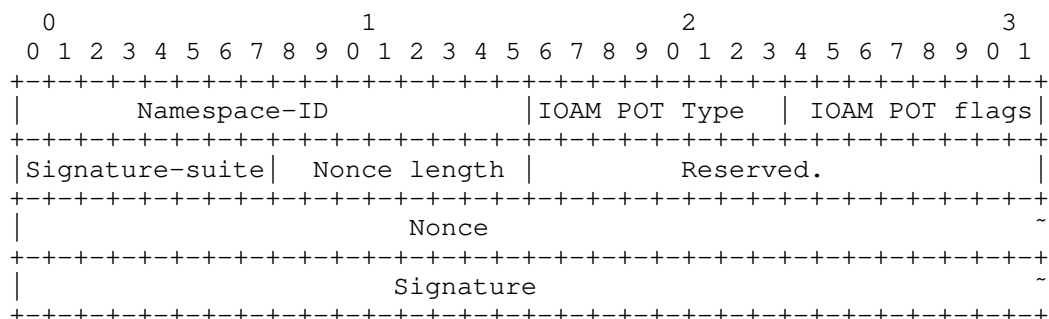
Nonce: Nonce is a variable length field with length specified in Nonce length.

Signature: Signature is the digital signature value generated by the method and algorithm specified by Signature-suite.

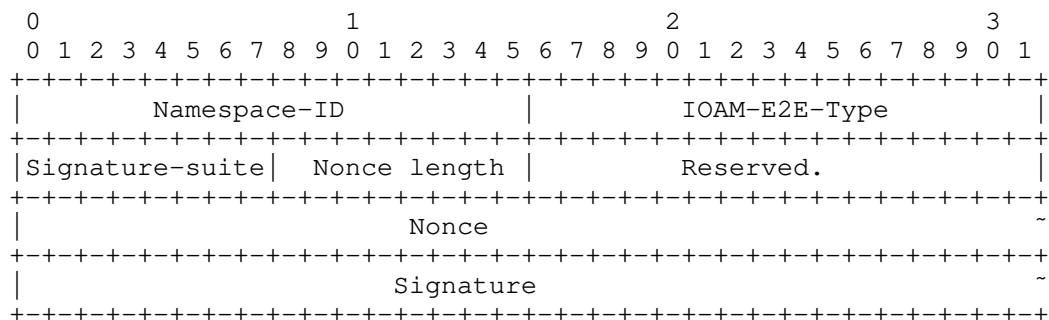
The Integrity sub-header follows the IOAM Option-Type header when the IOAM Option-Type is Integrity Protected Option. Pre-allocated and incremental Trace option headers are as defined in [I-D.ietf-ippm-ioam-data]. When the IOAM Option-Type is set to the IOAM Pre-allocated Trace Integrity Protected Option-Type or IOAM Incremental Trace Integrity Protected Option-Type then the Integrity Protection subheader follows the original IOAM Option Type header: :



IOAM POT option header as defined in [I-D.ietf-ippm-ioam-data] is followed by Integrity Protection subheader when IOAM Option Type is set to IOAM POT Integrity Protected Option-Type:



IOAM E2E option header as defined in [I-D.ietf-ippm-ioam-data] is followed by Integrity Protection subheader when IOAM Option Type is set to IOAM E2E Integrity Protected Option-Type:



4.3. Space optimized symmetric key based signing of IOAM data

This method assumes that symmetric keys have been distributed to the respective nodes as well as the Validator (the Validator receives all the keys). The details of the mechanisms of how keys are distributed are outside the scope of this document. The "Signature" field is populated as follows:

1. The first node creates a nonce and signature over the hash of IOAM Option excluding the Signature field, the nonce and its symmetric key. The nonce is included as a field in Integrity Protection sub-header of the corresponding IOAM Option. The resulting signature is included in the corresponding Signature field.
2. Transit nodes will update the Signature field by creating a signature of data where the data is [Signature || hash(node_data_list[x])] with its symmetric key in case of IOAM Trace Integrity Protected Options. Transit nodes updating IOAM POT Option will update the Signature field by creating a signature of data where the data is [Signature || hash(IOAM POT OPTION excluding Signature field)] with its symmetric key in case of IOAM POT Integrity Protected Option.
3. The Validator will iteratively recreate the Signature over the IOAM Option fields collected and matches the Signature field to validate the data integrity.

This method uses the following algorithms:

1. The algorithm to calculate the signature using symmetric key MUST be Advanced Encryption Standard (AES) AES-256. [AES] [NIST.800-38D].
2. The digest/hash algorithm used MUST be SHA-256 [SHS].

4.3.1. Overhead consideration

The Signature would consume 32 bytes with AES-256. With this method the Signature is carried only once for the entire packet. As there are dedicated options for carrying IOAM data with integrity protection, in case of performance concerns in calculating signature and validating it, these options can be used for a subset of the packets by using sampling of data to enable IOAM with integrity protection.

4.4. Space optimized asymmetric key based signing of trace data

This method assumes that asymmetric keys have been generated per IOAM node and the respective nodes can access their keys. The Validator receives all the public keys. The details of the mechanisms of how keys are generated and shared are outside the scope of this document. The "Signature" field is populated as follows:

1. The first node creates a nonce and signs over the hash of IOAM Option it populates excluding the Signature field in the option, the nonce and its private key. The resulting signature is included in the Signature field.
2. Transit nodes will update the Signature field by creating a signature of data where the data is [Signature || hash(node_data_list[x])] with its private key in case of IOAM Trace Integrity Protected Options. Transit nodes updating IOAM POT Option will update the Signature field by creating a signature of data where the data is [Signature || hash(IOAM POT OPTION excluding Signature field)] with its private key in case of IOAM POT Integrity Protected Option.
3. The Validator will iteratively recreate the Signature over the IOAM Option fields collected and matches the Signature field to validate the data integrity using public keys of the IOAM nodes.

This method uses the following algorithms:

1. The signature algorithm used MUST be the Elliptic Curve Digital Signature Algorithm (ECDSA) with curve P-256 [RFC6090] [DSS].
2. The digest/hash algorithm used MUST be SHA-256 [SHS].

4.4.1. Overhead consideration

The Signature consumes 32 bytes based on the SHA-256 ECDSA P-256 algorithm employed. With this method the Signature is only carried once for the entire packet. As there are dedicated options for carrying IOAM data with integrity protection, in case of performance concerns in calculating signature and validating it, these options can be used for a subset of the packets by using sampling of data to enable IOAM with integrity protection.

5. IANA Considerations

5.1. IOAM Option-Type Registry

The following code points are defined in this draft in "IOAM Option-Type Registry" :

64 IOAM Pre-allocated Trace Integrity Protected Option-Type

65 IOAM Incremental Trace Integrity Protected Option-Type

66 IOAM POT Integrity Protected Option-Type

67 IOAM E2E Integrity Protected Option-Type

5.2. IOAM Integrity Protection Algorithm Suite Registry

"IOAM Integrity Protection Algorithm Suite Registry" in the "In-Situ OAM (IOAM) Protocol Parameters" group. The one-octet "IOAM Integrity Protection Algorithm Suite Registry" identifiers assigned by IANA identify the digest algorithm and signature algorithm used in the Signature Suite Identifier field. IANA has registered the following algorithm suite identifiers for the digest algorithm and for the signature algorithm.

IOAM Integrity Protection Algorithm Suite Registry

Algorithm Suite Identifier	Digest Algorithm	Signature Algorithm	Specification Pointer
0x0	Reserved	Reserved	This document
0x1	SHA-256	ECDSA P-256	[SHS] [DSS] [RFC6090] This document
0x2	SHA-256	AES-256	[AES] [NIST.800-38D] This document
0xEF-0xFF	Unassigned	Unassigned	

Future assignments are to be made using the Standards Action process defined in [RFC8126]. Assignments consist of the one-octet algorithm suite identifier value and the associated digest algorithm name and signature algorithm name.

6. Security Considerations

This section will be completed in a future revision of this document.

7. Acknowledgements

The authors would like to thank Santhosh N, Rakesh Kandula, Saiprasad Muchala, Greg Mirsky, Benjamin Kaduk and Martin Duke for their comments and advice.

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>>.
- [RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

8.2. Informative References

- [AES] National Institute of Standards and Technology, "Advanced Encryption Standard (AES)", FIPS PUB 197, 2001, <<http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf>>.
- [DSS] National Institute of Standards and Technology, "Digital Signature Standard (DSS)", NIST FIPS Publication 186-4, DOI 10.6028/NIST.FIPS.186-4, 2013, <<http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf>>.
- [I-D.ietf-detnet-security] Grossman, E., Mizrahi, T., and A. J. Hacker, "Deterministic Networking (DetNet) Security Considerations", draft-ietf-detnet-security-16 (work in progress), March 2021.

- [I-D.ietf-ippm-ioam-data]
Brockners, F., Bhandari, S., and T. Mizrahi, "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data-14 (work in progress), June 2021.
- [I-D.ietf-ippm-ioam-direct-export]
Song, H., Gafni, B., Zhou, T., Li, Z., Brockners, F., Bhandari, S., Sivakolundu, R., and T. Mizrahi, "In-situ OAM Direct Exporting", draft-ietf-ippm-ioam-direct-export-05 (work in progress), July 2021.
- [NIST.800-38D]
National Institute of Standards and Technology,
"Recommendation for Block Cipher Modes of Operation:
Galois/Counter Mode (GCM) and GMAC", NIST Special
Publication 800-38D, 2001,
<<http://csrc.nist.gov/publications/nistpubs/800-38D/SP-800-38D.pdf>>.
- [RFC6090] McGrew, D., Igoe, K., and M. Salter, "Fundamental Elliptic Curve Cryptography Algorithms", RFC 6090,
DOI 10.17487/RFC6090, February 2011,
<<https://www.rfc-editor.org/info/rfc6090>>.
- [RFC7276] Mizrahi, T., Sprecher, N., Bellagamba, E., and Y. Weingarten, "An Overview of Operations, Administration, and Maintenance (OAM) Tools", RFC 7276,
DOI 10.17487/RFC7276, June 2014,
<<https://www.rfc-editor.org/info/rfc7276>>.
- [RFC7384] Mizrahi, T., "Security Requirements of Time Protocols in Packet Switched Networks", RFC 7384, DOI 10.17487/RFC7384, October 2014, <<https://www.rfc-editor.org/info/rfc7384>>.
- [RFC7799] Morton, A., "Active and Passive Metrics and Methods (with Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799, May 2016, <<https://www.rfc-editor.org/info/rfc7799>>.
- [SHS] National Institute of Standards and Technology, "Secure Hash Standard (SHS)", NIST FIPS Publication 180-4, DOI 10.6028/NIST.FIPS.180-4, 2015,
<<http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf>>.

Authors' Addresses

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

Email: fbrockne@cisco.com

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

Email: shwetha.bhandari@thoughtspot.com

Tal Mizrahi
Huawei
8-2 Matam
Haifa 3190501
Israel

Email: tal.mizrahi.phd@gmail.com

opsawg
Internet-Draft
Intended status: Best Current Practice
Expires: December 26, 2021

F. Brockners
Cisco
S. Bhandari, Ed.
Thoughtspot
D. Bernier
Bell Canada
T. Mizrahi, Ed.
Huawei
June 24, 2021

In-situ OAM Deployment
draft-brockners-opsawg-ioam-deployment-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 provides a framework for IOAM deployment and provides best current practices.

Status of This Memo

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

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

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

This Internet-Draft will expire on December 26, 2021.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

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

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Conventions	3
3. IOAM Deployment: Domains And Nodes	4
4. Types of IOAM	5
4.1. Per-hop Tracing IOAM	6
4.2. Proof of Transit IOAM	8
4.3. Edge to Edge IOAM	8
4.4. Direct Export IOAM	8
5. IOAM Encapsulations	8
5.1. IPv6	9
5.2. NSH	9
5.3. GRE	9
5.4. Geneve	10
5.5. Segment Routing	10
5.6. Segment Routing for IPv6	10
5.7. VXLAN-GPE	10
6. IOAM Data Export	10
7. IOAM Deployment Considerations	11
7.1. IOAM Namespaces	12
7.2. IOAM Layering	13
7.3. IOAM Trace Option Types	14
7.4. Traffic-sets That IOAM Is Applied To	16
7.5. IOAM Loopback Mode	16
7.6. IOAM Active Mode	16
7.7. Brown Field Deployments: IOAM Unaware Nodes	17
8. IOAM Manageability	17
9. IANA Considerations	18
10. Security Considerations	18
11. Acknowledgements	19
12. References	19
12.1. Normative References	19
12.2. Informative References	20
Authors' Addresses	23

1. Introduction

"In-situ" Operations, Administration, and Maintenance (IOAM) records OAM information within the packet while the packet traverses a particular network domain. The term "in-situ" refers to the fact that the OAM data is added to the data packets rather than is being

sent within packets specifically dedicated to OAM. IOAM is to complement mechanisms such as Ping, Traceroute, or other active probing mechanisms. In terms of "active" or "passive" OAM, "in-situ" OAM can be considered a hybrid OAM type. "In-situ" mechanisms do not require extra packets to be sent. IOAM adds information to the already available data packets and therefore cannot be considered passive. In terms of the classification given in [RFC7799] IOAM could be portrayed as Hybrid Type 1. IOAM mechanisms can be leveraged where mechanisms using e.g. ICMP do not apply or do not offer the desired results, such as proving that a certain traffic flow takes a pre-defined path, SLA verification for the live data traffic, detailed statistics on traffic distribution paths in networks that distribute traffic across multiple paths, or scenarios in which probe traffic is potentially handled differently from regular data traffic by the network devices.

2. Conventions

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

Abbreviations used in this document:

E2E	Edge to Edge
Geneve:	Generic Network Virtualization Encapsulation [I-D.ietf-nvo3-geneve]
GRE	Generic Routing Encapsulation
IOAM:	In-situ Operations, Administration, and Maintenance
MTU:	Maximum Transmit Unit
NSH:	Network Service Header [RFC8300]
OAM:	Operations, Administration, and Maintenance
POT:	Proof of Transit
SFC:	Service Function Chain
SID:	Segment Identifier
SR:	Segment Routing

VXLAN-GPE: Virtual eXtensible Local Area Network, Generic Protocol Extension [I-D.ietf-nvo3-vxlan-gpe]

3. IOAM Deployment: Domains And Nodes

IOAM is a network domain specific feature, with "network domain" being a set of network devices or entities within a single administration. IOAM is not targeted for a deployment on the global Internet. The part of the network which employs IOAM is referred to as the "IOAM-Domain". For example, an IOAM-domain can include an enterprise campus using physical connections between devices or an overlay network using virtual connections / tunnels for connectivity between said devices. An IOAM-domain is defined by its perimeter or edge. The operator of an IOAM-domain is expected to put provisions in place to ensure that packets which contain IOAM data fields do not leak beyond the edge of an IOAM domain, e.g. using for example packet filtering methods. The operator should consider the potential operational impact of IOAM to mechanisms such as ECMP processing (e.g. load-balancing schemes based on packet length could be impacted by the increased packet size due to IOAM), path MTU (i.e. ensure that the MTU of all links within a domain is sufficiently large to support the increased packet size due to IOAM) and ICMP message handling.

An IOAM-Domain consists of "IOAM encapsulating nodes", "IOAM decapsulating nodes" and "IOAM transit nodes". The role of a node (i.e. encapsulating, transit, decapsulating) is defined within an IOAM-Namespace (see below). A node can have different roles in different IOAM-Namespace.

An "IOAM encapsulating node" incorporates one or more IOAM-Option-Types into packets that IOAM is enabled for. If IOAM is enabled for a selected subset of the traffic, the IOAM encapsulating node is responsible for applying the IOAM functionality to the selected subset.

An "IOAM transit node" updates one or more of the IOAM-Data-Fields. If both the Pre-allocated and the Incremental Trace Option-Types are present in the packet, each IOAM transit node will update at most one of these Option-Types. A transit node does not add new IOAM-Option-Types to a packet, and does not change the IOAM-Data-Fields of an IOAM Edge-to-Edge Option-Type.

An "IOAM decapsulating node" removes IOAM-Option-Type(s) from packets.

The role of an IOAM-encapsulating, IOAM-transit or IOAM-decapsulating node is always performed within a specific IOAM-Namespace. This means that an IOAM node which is e.g. an IOAM-decapsulating node for

IOAM-Namespace "A" but not for IOAM-Namespace "B" will only remove the IOAM-Option-Types for IOAM-Namespace "A" from the packet. An IOAM decapsulating node situated at the edge of an IOAM domain removes all IOAM-Option-Types and associated encapsulation headers for all IOAM-Namespaces from the packet.

IOAM-Namespaces allow for a namespace-specific definition and interpretation of IOAM-Data-Fields. An interface-id could for example point to a physical interface (e.g., to understand which physical interface of an aggregated link is used when receiving or transmitting a packet) whereas in another case it could refer to a logical interface (e.g., in case of tunnels). Please refer to Section 7.1 for a discussion of IOAM-Namespaces.

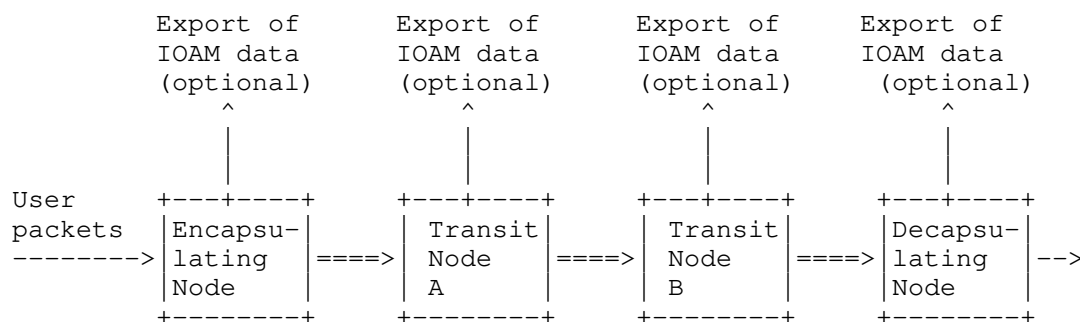


Figure 1: Roles of IOAM nodes

IOAM nodes which add or remove the IOAM-Data-Fields can also update the IOAM-Data-Fields at the same time. Or in other words, IOAM encapsulating or decapsulating nodes can also serve as IOAM transit nodes at the same time. Note that not every node in an IOAM domain needs to be an IOAM transit node. For example, a deployment might require that packets traverse a set of firewalls which support IOAM. In that case, only the set of firewall nodes would be IOAM transit nodes rather than all nodes.

4. Types of IOAM

IOAM supports different modes of operation, which are differentiated by the type of IOAM data fields being carried in the packet, the data being collected, the type of nodes which collect or update data as well as whether and how nodes export IOAM information.

- o Per-hop tracing: OAM information about each IOAM node a packet traverses is collected and stored within the user data packet as

the packet progresses through the IOAM domain. Potential uses of IOAM per-hop tracing include:

- * Optimization: Understand the different paths different packets traverse between a source and a sink in a network that uses load balancing such as Equal Cost Load Balancing (ECMP). This information could be used to tune the algorithm for ECMP for optimized network resource usage.
- * Operations/Troubleshooting: Understand which path a particular packet or set of packets take as well as what amount of jitter and delay different nodes in the path contribute to the overall end-to-end delay and jitter.
- o Proof-of-transit: Information that a verifier node can use to verify whether a packet has traversed all nodes that is supposed to traverse is stored within the user data packet. Proof-of-transit could for example be used to verify that a packet indeed passes through all services of a service function chain (e.g. verify whether a packet indeed traversed the set of firewalls that it is expected to traverse), or whether a packet indeed took the expected path.
- o Edge-to-edge: OAM information which is specific to the edges of an IOAM domain is collected and stored within the user data packet. Edge-to-Edge OAM could be used to gather operational information about a particular network domain, such as the delay and jitter incurred by that network domain or the traffic matrix of the network domain.
- o Direct export: OAM information about each IOAM node a packet traverses is collected and immediately exported to a collector. Direct export could be used in situations where per-hop tracing information is desired, but gathering the information within the packet - as with per-hop tracing - is not feasible. Rather than automatically correlating the per-hop tracing information, as done with per-hop tracing, direct export requires a collector to correlate the information from the individual nodes. In addition, all nodes enabled for direct export need to be capable to export the IOAM information to the collector.

4.1. Per-hop Tracing IOAM

"IOAM tracing data" is expected to be collected at every IOAM transit node that a packet traverses to ensure visibility into the entire path a packet takes within an IOAM-Domain. I.e., in a typical deployment all nodes in an IOAM-Domain would participate in IOAM and thus be IOAM transit nodes, IOAM encapsulating or IOAM decapsulating

nodes. If not all nodes within a domain are IOAM capable, IOAM tracing information (i.e., node data, see below) will only be collected on those nodes which are IOAM capable. Nodes which are not IOAM capable will forward the packet without any changes to the IOAM-Data-Fields. The maximum number of hops and the minimum path MTU of the IOAM domain is assumed to be known.

IOAM offers two different trace Option-Types, the "incremental" Option-Type as well as the "pre-allocated" Option-Type. For a discussion which of the two option types is the most suitable for an implementation and/or deployment, see Section 7.3.

Every node data entry holds information for a particular IOAM transit node that is traversed by a packet. The IOAM decapsulating node removes the IOAM-Option-Type(s) and processes and/or exports the associated data. All IOAM-Data-Fields are defined in the context of an IOAM-Namespace.

IOAM tracing can collect the following types of information:

- o Identification of the IOAM node. An IOAM node identifier can match to a device identifier or a particular control point or subsystem within a device.
- o Identification of the interface that a packet was received on, i.e. ingress interface.
- o Identification of the interface that a packet was sent out on, i.e. egress interface.
- o Time of day when the packet was processed by the node as well as the transit delay. Different definitions of processing time are feasible and expected, though it is important that all devices of an in-situ OAM domain follow the same definition.
- o Generic data: Format-free information where syntax and semantic of the information is defined by the operator in a specific deployment. For a specific IOAM-Namespace, all IOAM nodes should interpret the generic data the same way. Examples for generic IOAM data include geo-location information (location of the node at the time the packet was processed), buffer queue fill level or cache fill level at the time the packet was processed, or even a battery charge level.
- o Information to detect whether IOAM trace data was added at every hop or whether certain hops in the domain weren't IOAM transit nodes.

- o Data that relates to how the packet traversed a node (transit delay, buffer occupancy in case the packet was buffered, queue depth in case the packet was queued)

The Option-Types of incremental tracing and pre-allocated tracing are defined in [I-D.ietf-ippm-ioam-data].

4.2. Proof of Transit IOAM

IOAM Proof of Transit Option-Type is to support path or service function chain [RFC7665] verification use cases. Proof-of-transit uses methods like nested hashing or nested encryption of the IOAM data or mechanisms such as Shamir's Secret Sharing Schema (SSSS).

The IOAM Proof of Transit Option-Type consist of a fixed size "IOAM proof of transit option header" and "IOAM proof of transit option data fields". For details see [I-D.ietf-ippm-ioam-data].

4.3. Edge to Edge IOAM

The IOAM Edge-to-Edge Option-Type is to carry data that is added by the IOAM encapsulating node and interpreted by IOAM decapsulating node. The IOAM transit nodes may process the data but must not modify it.

The IOAM Edge-to-Edge Option-Type consist of a fixed size "IOAM Edge-to-Edge Option-Type header" and "IOAM Edge-to-Edge Option-Type data fields". For details see [I-D.ietf-ippm-ioam-data].

4.4. Direct Export IOAM

Direct Export is an IOAM mode of operation within which IOAM data to be directly exported to a collector rather than being collected within the data packets. The IOAM Direct Export Option-Type consist of a fixed size "IOAM direct export option header". Direct Export for IOAM is defined in [I-D.ietf-ippm-ioam-direct-export].

5. IOAM Encapsulations

IOAM data fields and associated data types for in-situ OAM are defined in [I-D.ietf-ippm-ioam-data]. The in-situ OAM data field can be transported by a variety of transport protocols, including NSH, Segment Routing, Geneve, IPv6, etc.

5.1. IPv6

IOAM encapsulation for IPv6 is defined in [I-D.ietf-ippm-ioam-ipv6-options]. IOAM deployment considerations for IPv6 networks are found in [I-D.ioametal-ippm-6man-ioam-ipv6-deployment].

5.2. NSH

IOAM encapsulation for NSH is defined in [I-D.ietf-sfc-ioam-nsh].

5.3. GRE

IOAM encapsulation for GRE is outlined as part of the "EtherType Protocol Identification of In-situ OAM Data" in [I-D.weis-ippm-ioam-eth], though no example protocol header stacks are provided in the document. When used with GRE, the IOAM-Option-Types (the below diagram uses "IOAM" as shorthand for IOAM-Option-Types) are sequenced in behind the GRE header that follows the "outer" IP header. Figure 2 shows two example protocol header stacks that use GRE along with IOAM.

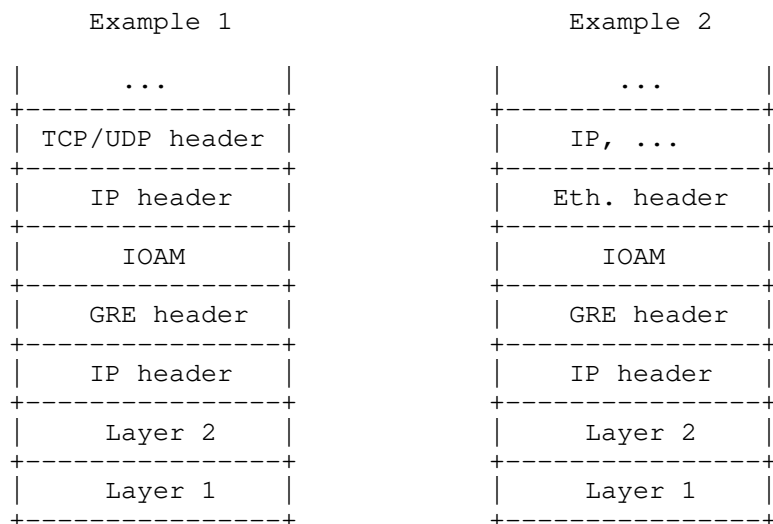


Figure 2: GRE with IOAM examples

5.4. Geneve

IOAM encapsulation for Geneve is defined in [I-D.brockners-ippm-ioam-geneve].

5.5. Segment Routing

IOAM encapsulation for Segment Routing is defined in [I-D.gandhi-spring-ioam-sr-mpls].

5.6. Segment Routing for IPv6

IOAM encapsulation for Segment Routing over IPv6 is defined in [I-D.ali-spring-ioam-srv6].

5.7. VXLAN-GPE

IOAM encapsulation for VXLAN-GPE is defined in [I-D.brockners-ippm-ioam-vxlan-gpe].

6. IOAM Data Export

IOAM nodes collect information for packets traversing a domain that supports IOAM. IOAM decapsulating nodes as well as IOAM transit nodes can choose to retrieve IOAM information from the packet, process the information further and export the information using e.g., IPFIX.

Raw data export of IOAM data using IPFIX is discussed in [I-D.spiegel-ippm-ioam-rawexport]. "Raw export of IOAM data" refers to a mode of operation where a node exports the IOAM data as it is received in the packet. The exporting node neither interprets, aggregates nor reformats the IOAM data before it is exported. Raw export of IOAM data is to support an operational model where the processing and interpretation of IOAM data is decoupled from the operation of encapsulating/updating/decapsulating IOAM data, which is also referred to as IOAM data-plane operation. The figure below shows the separation of concerns for IOAM export: Exporting IOAM data is performed by the "IOAM node" which performs IOAM data-plane operation, whereas the interpretation of IOAM data is performed by one or several IOAM data processing systems. The separation of concerns is to off-load interpretation, aggregation and formatting of IOAM data from the node which performs data-plane operations. In other words, a node which is focused on data-plane operations, i.e. forwarding of packets and handling IOAM data will not be tasked to also interpret the IOAM data, but can leave this task to another system or a set of systems. For scalability reasons, a single IOAM node could choose to export IOAM data to several IOAM data processing

systems. Similarly, there several monitoring systems or analytics systems can be used to further process the data received from the IOAM preprocessing systems. Figure 3 shows an overview of IOAM export, including IOAM data processing systems and monitoring/ analytics systems.

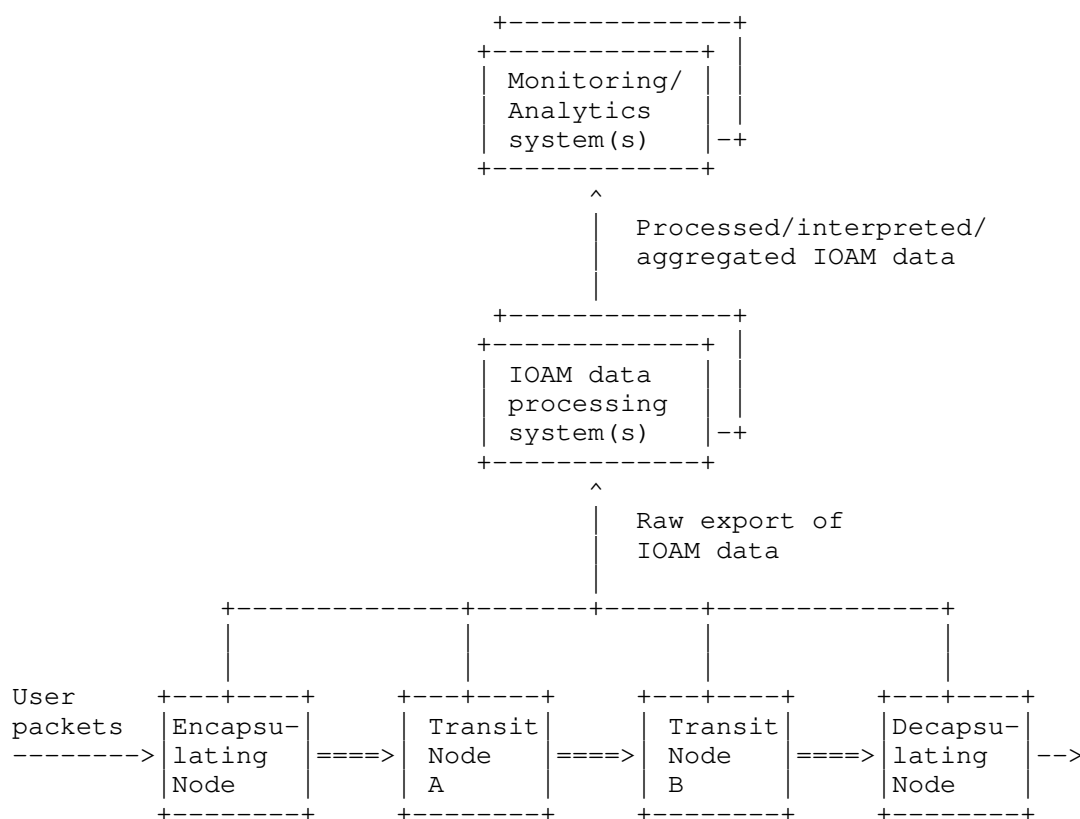


Figure 3: IOAM framework with data export

7. IOAM Deployment Considerations

This section discusses several aspects of an IOAM deployment, including IOAM Namespaces, IOAM Layering, traffic-sets that IOAM is applied to and IOAM loopback mode.

7.1. IOAM Namespaces

IOAM-Namespaces add further context to IOAM-Option-Types and associated IOAM-Data-Fields. IOAM-Namespaces support several different uses:

- o IOAM-Namespaces can be used by an operator to distinguish different operational domains. Devices at domain edges can filter on Namespace-IDs to provide for proper IOAM-Domain isolation.
- o IOAM-Namespaces provide additional context for IOAM-Data-Fields and thus ensure that IOAM-Data-Fields are unique and can be interpreted properly by management stations or network controllers. While, for example, the node identifier field does not need to be unique in a deployment (e.g. an operator may wish to use different node identifiers for different IOAM layers, even within the same device; or node identifiers might not be unique for other organizational reasons, such as after a merger of two formerly separated organizations), the combination of node_id and Namespace-ID should always be unique. Similarly, IOAM-Namespaces can be used to define how certain IOAM-Data-Fields are interpreted: IOAM offers three different timestamp format options. The Namespace-ID can be used to determine the timestamp format. IOAM-Data-Fields (e.g. buffer occupancy) which do not have a unit associated are to be interpreted within the context of a IOAM-Namespace.
- o IOAM-Namespaces can be used to identify different sets of devices (e.g., different types of devices) in a deployment: If an operator desires to insert different IOAM-Data-Fields based on the device, the devices could be grouped into multiple IOAM-Namespaces. This could be due to the fact that the IOAM feature set differs between different sets of devices, or it could be for reasons of optimized space usage in the packet header. It could also stem from hardware or operational limitations on the size of the trace data that can be added and processed, preventing collection of a full trace for a flow.
 - * Assigning different IOAM Namespace-IDs to different sets of nodes or network partitions and using the Namespace-ID as a selector at the IOAM encapsulating node, a full trace for a flow could be collected and constructed via partial traces in different packets of the same flow. Example: An operator could choose to group the devices of a domain into two IOAM-Namespaces, in a way that at average, only every second hop would be recorded by any device. To retrieve a full view of the deployment, the captured IOAM-Data-Fields of the two IOAM-Namespaces need to be correlated.

- * Assigning different IOAM Namespace-IDs to different sets of nodes or network partitions and using a separate instance of an IOAM-Option-Type for each Namespace-ID, a full trace for a flow could be collected and constructed via partial traces from each IOAM-Option-Type in each of the packets in the flow. Example: An operator could choose to group the devices of a domain into two IOAM-Namespace, in a way that each IOAM-Namespace is represented by one of two IOAM-Option-Types in the packet. Each node would record data only for the IOAM-Namespace that it belongs to, ignoring the other IOAM-Option-Type with a IOAM-Namespace to which it doesn't belong. To retrieve a full view of the deployment, the captured IOAM-Data-Fields of the two IOAM-Namespace need to be correlated.

7.2. IOAM Layering

If several encapsulation protocols (e.g., in case of tunneling) are stacked on top of each other, IOAM-Data-Fields could be present in different protocol fields at different layers. Layering allows operators to instrument the protocol layer they want to measure. The behavior follows the ships-in-the-night model, i.e. IOAM-Data-Fields in one layer are independent from IOAM-Data-Fields in another layer. Or in other words: Even though the term "layering" often implies some form of hierarchy and relationship, in IOAM, layers are independent from each other and don't assume any relationship among them. The different layers could, but do not have to share the same IOAM encapsulation mechanisms. Similarly, the semantics of the IOAM-Data-Fields, can, do not have to be associated to across different layers. For example, a node which inserts node-id information into two different layers could use "node-id=10" for one layer and "node-id=1000" for the second layer.

Figure 4 shows an example of IOAM layering. The figure shows a Geneve tunnel carried over IPv6 which starts at node A and ends at node D. IOAM information is encapsulated in IPv6 as well as in Geneve. At the IPv6 layer, node A is IOAM encapsulating node (into IPv6), node D is the IOAM decapsulating node and node B and node C are IOAM transit nodes. At the Geneve layer, node A is IOAM encapsulating node (into Geneve) and node D is IOAM decapsulating node (from Geneve). The use of IOAM at both layers as shown in the example here could be used to reveal which nodes of an underlay (here the IPv6 network) are traversed by tunneled packet in an overlay (here the Geneve network) - which assumes that the IOAM information encapsulated by nodes A and D into Geneve and IPv6 is associated to each other.

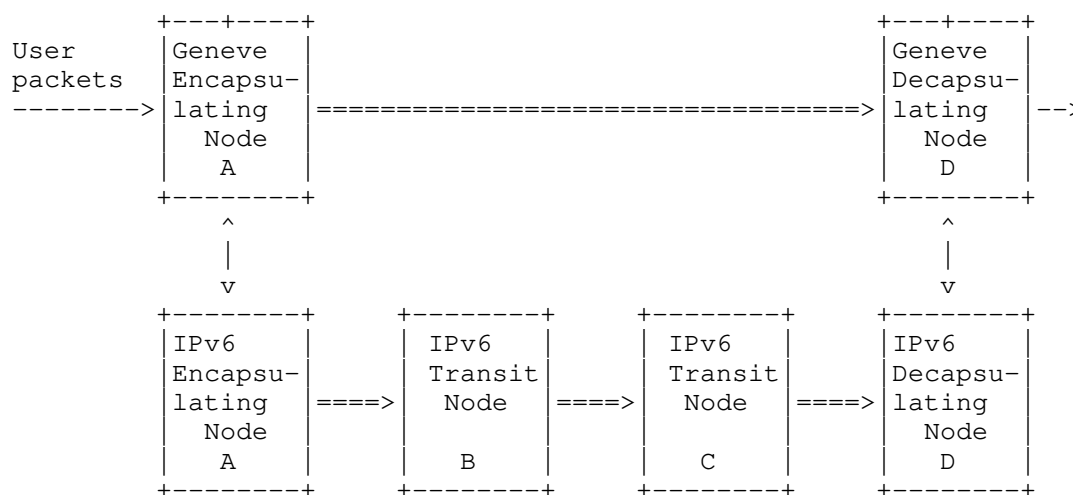


Figure 4: IOAM layering example

7.3. IOAM Trace Option Types

IOAM offers two different IOAM Option-Types for tracing:

"Incremental" Trace-Option-Type and "Pre-allocated" Trace-Option-Type. "Incremental" refers to a mode of operation where the packet is expanded at every IOAM node that adds IOAM-Data-Fields. "Pre-Allocated" describes a mode of operation where the IOAM encapsulating node allocates room for all IOAM-Data-Fields in the entire IOAM domain. More specifically:

Pre-allocated Trace-Option: This trace option is defined as a container of node data fields with pre-allocated space for each node to populate its information. This option is useful for implementations where it is efficient to allocate the space once and index into the array to populate the data during transit (e.g., software forwarders often fall into this class).

Incremental Trace-Option: This trace option is defined as a container of node data fields where each node allocates and pushes its node data immediately following the option header.

A deployment can choose to configure and support one or both of the IOAM Trace-Option-Types. The operator decides by means of configuration which Trace-Option-Type(s) will be used for a particular domain. Deployments can mix devices which include either the Incremental Trace-Option-Type or the Pre-allocated Trace-Option-Type, e.g. in case different types of packet forwarders and

associated different types of IOAM implementations exist in a deployment. As a result, both Option-Types can be present in a packet. IOAM decapsulating nodes remove both types of Trace-Option-Types from the packet.

The two different Option-Types cater to different packet forwarding infrastructures and are to allow an optimized implementation of IOAM tracing:

Pre-allocated Trace-Option: For some implementations of packet forwarders it is efficient to allocate the space for the maximum number of nodes that IOAM Trace Data-Fields should be collected from and insert/update information in the packet at flexible locations, based on a pointer retrieved from a field in the packet. The IOAM encapsulating node allocates an array of the size of the maximum number of nodes that IOAM Trace Data-Fields should be collected from. IOAM transit nodes index into the array to populate the data during transit. Software forwarders often fall into this class of packet forwarder implementations. The maximum number of nodes that IOAM information could be collected from is configured by the operator on the IOAM encapsulating node. The operator has to ensure that the packet with the pre-allocated array that carries the IOAM Data-Fields does not exceed the MTU of any of the links in the IOAM domain.

Incremental Trace-Option: Looking up a pointer contained in the packet and inserting/updating information at a flexible location in the packet as a result of the pointer lookup is costly for some forwarding infrastructures. Hardware-based packet forwarding infrastructures often fall into this category. Consequently, hardware-based packet forwarders could choose to support the incremental IOAM-Trace-Option-Type. The incremental IOAM-Trace-Option-Type eliminates the need for the IOAM transit nodes to read the full array in the Trace-Option-Type and allows packets to grow to the size of the MTU of the IOAM domain. IOAM transit nodes will expand the packet and insert the IOAM-Data-Fields as long as there is space available in the packet, i.e. as long as the size of the packet stays within the bounds of the MTU of any of the links in the IOAM domain. There is no need for the operator to configure the IOAM encapsulation node with the maximum number of nodes that IOAM information could be collected from. The operator has to ensure that the minimum MTU of any of the links in the IOAM domain is known to all IOAM transit nodes.

7.4. Traffic-sets That IOAM Is Applied To

IOAM can be deployed on all or only on subsets of the live user traffic, e.g. per interface, based on an access control list or flow specification defining a specific set of traffic, etc.

7.5. IOAM Loopback Mode

IOAM Loopback is used to trigger each transit device along the path of a packet to send a copy of the data packet back to the source. Loopback allows an IOAM encapsulating node to trace the path to a given destination, and to receive per-hop data about both the forward and the return path. Loopback is enabled by the encapsulating node setting the loopback flag. Looped-back packets use the source address of the original packet as destination address and the address of the node which performs the loopback operation as source address. Nodes which loop back a packet clear the loopback flag before sending the copy back towards the source. Loopback applies to IOAM deployments where the encapsulating node is either a host or the start of a tunnel: For details on IOAM loopback, please refer to [I-D.ietf-ippm-ioam-flags].

7.6. IOAM Active Mode

The IOAM active mode flag indicates that a packet is an active OAM packet as opposed to regular user data traffic. Active mode is expected to be used for active measurement using IOAM. Example use-cases include:

- o Endpoint detailed active measurement: Synthetic probe packets are sent between the source and destination, traversing the IOAM domain. These probe packets include a Trace Option-Type (i.e., either incremental or pre-allocated). Since the probe packets are sent between the endpoints, these packets are treated as data packets by the IOAM domain, and do not require special treatment at the IOAM layer. The encapsulating node can choose to set the Active flag, providing an explicit indication that these probe packets are meant for telemetry collection.
- o IOAM active measurement using probe packets: Probe packets are generated and transmitted by the IOAM encapsulating node, and are expected to be terminated by the decapsulating node. Probe packets include a Trace Option-Type (i.e., either incremental or pre-allocated) 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-Type that has its Active flag set, indicating that the decapsulating node should terminate it.

For details on IOAM active mode, please refer to [I-D.ietf-ippm-ioam-flags].

7.7. Brown Field Deployments: IOAM Unaware Nodes

A network can consist of a mix of IOAM aware and IOAM unaware nodes. The encapsulation of IOAM-Data-Fields into different protocols (see also Section 5) are defined such that data packets that include IOAM-Data-Fields do not get dropped by IOAM unaware nodes. For example, packets which contain the IOAM-Trace-Option-Types in IPv6 Hop by Hop extension headers are defined with bits to indicate "00 - skip over this option and continue processing the header". This will ensure that when a node that is IOAM unaware receives a packet with IOAM-Data-Fields included, does not drop the packet.

Deployments which leverage the IOAM-Trace-Option-Type(s) could benefit from the ability to detect the presence of IOAM unaware nodes, i.e. nodes which forward the packet but do not update/add IOAM-Data-Fields in IOAM-Trace-Option-Type(s). The node data that is defined as part of the IOAM-Trace-Option-Type(s) includes a Hop_Lim field associated to the node identifier to detect missed nodes, i.e. "holes" in the trace. Monitoring/Analytics system(s) could utilize this information to account for the presence of IOAM unaware nodes in the network.

8. IOAM Manageability

The YANG model for configuring IOAM in network nodes which support IOAM is defined in [I-D.zhou-ippm-ioam-yang].

A deployment can leverage IOAM profiles is to limit the scope of IOAM features, allowing simpler implementation, verification, and interoperability testing in the context of specific use cases that do not require the full functionality of IOAM. An IOAM profile defines a use case or a set of use cases for IOAM, and an associated set of rules that restrict the scope and features of the IOAM specification, thereby limiting it to a subset of the full functionality. IOAM profiles are defined in [I-D.mizrahi-ippm-ioam-profile].

9. IANA Considerations

This document does not request any IANA actions.

10. Security Considerations

As discussed in [RFC7276], a successful attack on an OAM protocol in general, and specifically on IOAM, can prevent the detection of failures or anomalies, or create a false illusion of nonexistent ones.

The Proof of Transit Option-Type (Section 4.2) is used for verifying the path of data packets. The security considerations of POT are further discussed in [I-D.ietf-sfc-proof-of-transit].

Security considerations related to the use of IOAM flags, in particular the loopback flag are found in [I-D.ietf-ippm-ioam-flags].

IOAM data can be subject to eavesdropping. Although the confidentiality of the user data is not at risk in this context, the IOAM data elements can be used for network reconnaissance, allowing attackers to collect information about network paths, performance, queue states, buffer occupancy and other information. Recon is an improbable security threat in an IOAM deployment that is within a confined physical domain. However, in deployments that are not confined to a single LAN, but span multiple inter-connected sites (for example, using an overlay network), the inter-site links can be secured (e.g., by IPsec) in order to avoid external eavesdropping. Another possible mitigation approach is to use the "direct exporting" mode [I-D.ietf-ippm-ioam-direct-export]. In this case the IOAM related trace information would not be available in the customer data packets, but would trigger exporting of (secured) packet-related IOAM information at every node. IOAM data export and securing IOAM data export is outside the scope of this document.

IOAM can be used as a means for implementing Denial of Service (DoS) attacks, or for amplifying them. For example, a malicious attacker can add an IOAM header to packets or modify an IOAM header in en route packets in order to consume the resources of network devices that take part in IOAM or collectors that analyze the IOAM data. Another example is a packet length attack, in which an attacker pushes headers associated with IOAM Option-Types into data packets, causing these packets to be increased beyond the MTU size, resulting in fragmentation or in packet drops. Such DoS attacks can be mitigated by deploying IOAM in confined administrative domains, and by defining performance limits on IOAM encapsulation and IOAM exporting. By limiting the rate and/or percentage of packets that

are subject to IOAM encapsulation and the rate of exported traffic, it is possible to confine the impact of such attacks.

Since IOAM options may include timestamps, if network devices use synchronization protocols then any attack on the time protocol [RFC7384] can compromise the integrity of the timestamp-related data fields. Synchronization attacks can be mitigated by combining a secured time distribution scheme, e.g., [RFC8915], and by using redundant clock sources [RFC5905] and/or redundant network paths for the time distribution protocol [RFC8039].

At the management plane, attacks may be implemented by misconfiguring or by maliciously configuring IOAM-enabled nodes in a way that enables other attacks. Thus, IOAM configuration should be secured in a way that authenticates authorized users and verifies the integrity of configuration procedures.

Notably, IOAM is expected to be deployed in specific network domains, thus confining the potential attack vectors to within the network domain. Indeed, in order to limit the scope of threats to within the current network domain the network operator is expected to enforce policies that prevent IOAM traffic from leaking outside of the IOAM domain, and prevent IOAM data from outside the domain to be processed and used within the domain. Note that the Immediate Export mode (reference to be added in a future revision) can mitigate the potential threat of IOAM data leaking through data packets.

11. Acknowledgements

The authors would like to thank Tal Mizrahi, Eric Vyncke, Nalini Elkins, Srihari Raghavan, Ranganathan T S, Barak Gafni, Karthik Babu Harichandra Babu, Akshaya Nadahalli, LJ Wobker, Erik Nordmark, Vengada Prasad Govindan, Andrew Yourtchenko, Aviv Kfir, Tianran Zhou, Zhenbin (Robin), Joe Clarke, Al Morton, Tom Herbet, Haoyu song, and Mickey Spiegel for the comments and advice on IOAM.

12. References

12.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

12.2. Informative References

[I-D.ali-spring-ioam-srv6]

Ali, Z., Gandhi, R., Filsfils, C., Brockners, F., Nainar, N., Pignataro, C., Li, C., Chen, M., and G. Dawra, "Segment Routing Header encapsulation for In-situ OAM Data", draft-ali-spring-ioam-srv6-03 (work in progress), November 2020.

[I-D.brockners-ippm-ioam-geneve]

Brockners, F., Bhandari, S., Govindan, V. P., Pignataro, C., Nainar, N. K., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Lapukhov, P., Gafni, B., Kfir, A., and M. Spiegel, "Geneve encapsulation for In-situ OAM Data", draft-brockners-ippm-ioam-geneve-05 (work in progress), November 2020.

[I-D.brockners-ippm-ioam-vxlan-gpe]

Brockners, F., Bhandari, S., Govindan, V. P., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Kfir, A., Gafni, B., Lapukhov, P., and M. Spiegel, "VXLAN-GPE Encapsulation for In-situ OAM Data", draft-brockners-ippm-ioam-vxlan-gpe-03 (work in progress), November 2019.

[I-D.gandhi-spring-ioam-sr-mpls]

Gandhi, R., Ali, Z., Filsfils, C., Brockners, F., Wen, B., and V. Kozak, "Segment Routing with MPLS Data Plane Encapsulation for In-situ OAM Data", draft-gandhi-spring-ioam-sr-mpls-02 (work in progress), August 2019.

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

Brockners, F., Bhandari, S., and T. Mizrahi, "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data-12 (work in progress), February 2021.

[I-D.ietf-ippm-ioam-direct-export]

Song, H., Gafni, B., Zhou, T., Li, Z., Brockners, F., Bhandari, S., Sivakolundu, R., and T. Mizrahi, "In-situ OAM Direct Exporting", draft-ietf-ippm-ioam-direct-export-03 (work in progress), February 2021.

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

Mizrahi, T., Brockners, F., Bhandari, S., Sivakolundu, R., Pignataro, C., Kfir, A., Gafni, B., Spiegel, M., and J. Lemon, "In-situ OAM Flags", draft-ietf-ippm-ioam-flags-04 (work in progress), February 2021.

- [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-05 (work in progress), February 2021.
- [I-D.ietf-nvo3-geneve]
Gross, J., Ganga, I., and T. Sridhar, "Geneve: Generic Network Virtualization Encapsulation", draft-ietf-nvo3-geneve-16 (work in progress), March 2020.
- [I-D.ietf-nvo3-vxlan-gpe]
(Editor), F. M., (editor), L. K., and U. E. (editor), "Generic Protocol Extension for VXLAN (VXLAN-GPE)", draft-ietf-nvo3-vxlan-gpe-11 (work in progress), March 2021.
- [I-D.ietf-sfc-ioam-nsh]
Brockners, F. and S. Bhandari, "Network Service Header (NSH) Encapsulation for In-situ OAM (IOAM) Data", draft-ietf-sfc-ioam-nsh-05 (work in progress), December 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-08 (work in progress), November 2020.
- [I-D.ioametal-ippm-6man-ioam-ipv6-deployment]
Bhandari, S., Brockners, F., Mizrahi, T., Kfir, A., Gafni, B., Spiegel, M., Krishnan, S., and M. Smith, "Deployment Considerations for In-situ OAM with IPv6 Options", draft-ioametal-ippm-6man-ioam-ipv6-deployment-03 (work in progress), March 2020.
- [I-D.mizrahi-ippm-ioam-profile]
Mizrahi, T., Brockners, F., Bhandari, S., Sivakolundu, R., Pignataro, C., Kfir, A., Gafni, B., Spiegel, M., Zhou, T., and J. Lemon, "In Situ OAM Profiles", draft-mizrahi-ippm-ioam-profile-04 (work in progress), February 2021.
- [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-04 (work in progress), November 2020.

- [I-D.weis-ippm-ioam-eth]
Weis, B., Brockners, F., Hill, C., Bhandari, S., Govindan, V. P., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Kfir, A., Gafni, B., Lapukhov, P., and M. Spiegel, "EtherType Protocol Identification of In-situ OAM Data", draft-weis-ippm-ioam-eth-04 (work in progress), May 2020.
- [I-D.zhou-ippm-ioam-yang]
Zhou, T., Guichard, J., Brockners, F., and S. Raghavan, "A YANG Data Model for In-Situ OAM", draft-zhou-ippm-ioam-yang-08 (work in progress), July 2020.
- [RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <<https://www.rfc-editor.org/info/rfc5905>>.
- [RFC7276] Mizrahi, T., Sprecher, N., Bellagamba, E., and Y. Weingarten, "An Overview of Operations, Administration, and Maintenance (OAM) Tools", RFC 7276, DOI 10.17487/RFC7276, June 2014, <<https://www.rfc-editor.org/info/rfc7276>>.
- [RFC7384] Mizrahi, T., "Security Requirements of Time Protocols in Packet Switched Networks", RFC 7384, DOI 10.17487/RFC7384, October 2014, <<https://www.rfc-editor.org/info/rfc7384>>.
- [RFC7665] Halpern, J., Ed. and C. Pignataro, Ed., "Service Function Chaining (SFC) Architecture", RFC 7665, DOI 10.17487/RFC7665, October 2015, <<https://www.rfc-editor.org/info/rfc7665>>.
- [RFC7799] Morton, A., "Active and Passive Metrics and Methods (with Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799, May 2016, <<https://www.rfc-editor.org/info/rfc7799>>.
- [RFC8039] Shpiner, A., Tse, R., Schelp, C., and T. Mizrahi, "Multipath Time Synchronization", RFC 8039, DOI 10.17487/RFC8039, December 2016, <<https://www.rfc-editor.org/info/rfc8039>>.
- [RFC8300] Quinn, P., Ed., Elzur, U., Ed., and C. Pignataro, Ed., "Network Service Header (NSH)", RFC 8300, DOI 10.17487/RFC8300, January 2018, <<https://www.rfc-editor.org/info/rfc8300>>.

[RFC8915] Franke, D., Sibold, D., Teichel, K., Dansarie, M., and R. Sundblad, "Network Time Security for the Network Time Protocol", RFC 8915, DOI 10.17487/RFC8915, September 2020, <<https://www.rfc-editor.org/info/rfc8915>>.

Authors' Addresses

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

Email: fbrockne@cisco.com

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

Email: shwetha.bhandari@thoughtspot.com

Daniel Bernier
Bell Canada
Canada

Email: daniel.bernier@bell.ca

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

Email: tal.mizrahi.phd@gmail.com

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

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

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

Abstract

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

Status of This Memo

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

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

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

This Internet-Draft will expire on 29 August 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

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

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

1. Introduction

1.1. Current Performance and Diagnostic Metrics (PDM)

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

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

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

1.2. PDMv2 Introduction

PDMv2 introduces confidential, integrity and authentication.

TBD

2. Conventions used in this document

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

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

3. Terminology

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

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

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

4. Protocol Flow

The protocol will proceed in 3 steps.

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

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

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

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

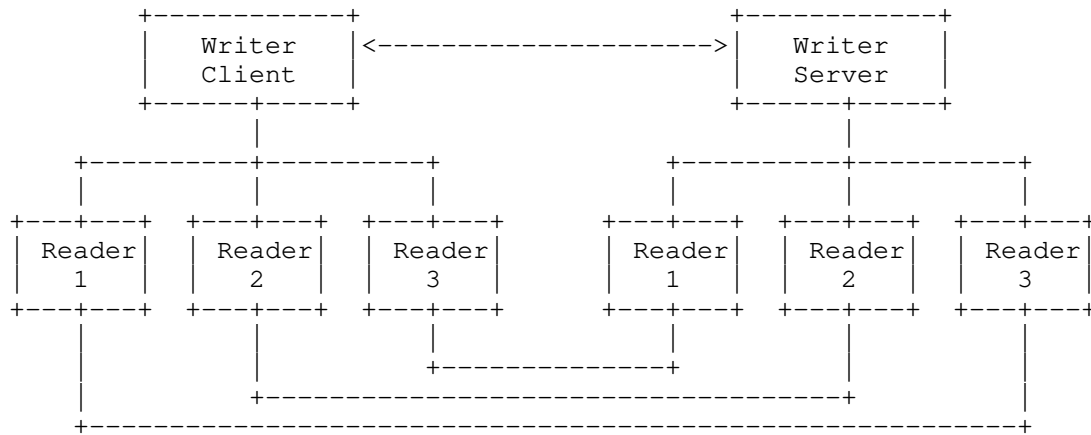
4.1. Registration Phase

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

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

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

4.1.2. Diagram of Registration Flow



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

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

They use HPKE KEM to negotiate a "SharedSecret".

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

The "SharedSecret" is shared securely:

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

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

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

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

The Kri SHOULD be initialized to zero.

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

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

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

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

5. Security Goals

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

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

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

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

5.1. Security Goals for Confidentiality

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

5.2. Security Goals for Integrity

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

5.3. Security Goals for Authentication

TBD

5.4. Cryptographic Algorithm

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

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

6. PDMv2 Destination Options

6.1. Destinations Option Header

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

6.2. Metrics information in PDMv2

The IPv6 PDMv2 destination option contains the following base fields:

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

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

The 5-tuple consists of:

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

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

- a) Link-Local
- b) Global Unicast

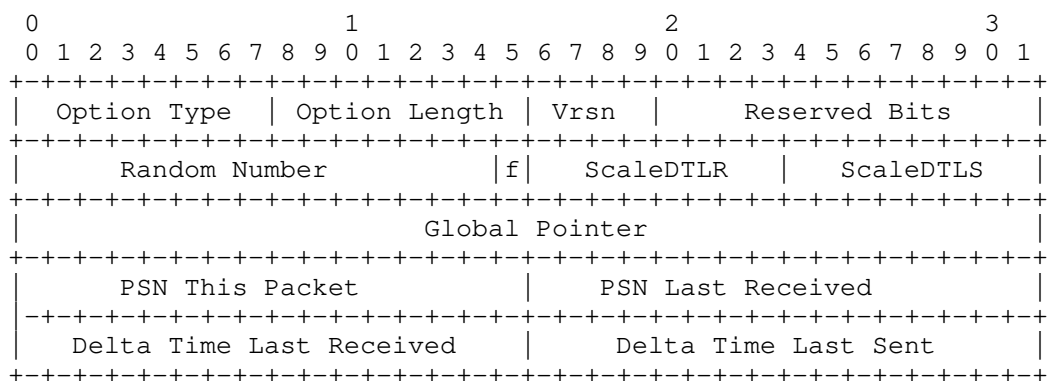
The Global Pointer is treated as a common entity over all the 5-tuples with the same SADDR type. It is initialised to the value 1 and increments for every packet sent. Global Pointer provides a measure of the amount of IPv6 traffic sent by the PDMv2 node.

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

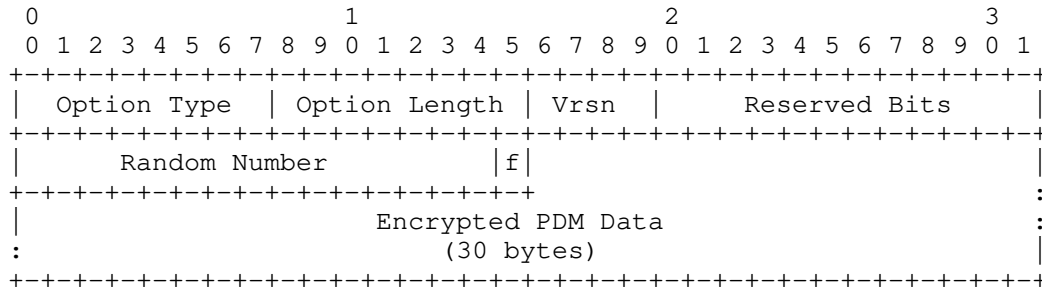
6.3. PDMv2 Layout

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

Following is the representation of the unencrypted PDMv2 header:



Following is the representation of the encrypted PDMv2 header:



Option Type

0x0F

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

Option Length

0x12: Unencrypted PDM

0x22: Encrypted PDM

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

Version Number

0x2

4-bit unsigned number.

Reserved Bits

12-bits.

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

Random Number

15-bit unsigned number.

TBD

Flag Bit

1-bit field.

TBD

Scale Delta Time Last Received (SCALEDTLR)

8-bit unsigned number.

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

Scale Delta Time Last Sent (SCALEDTLS)

8-bit unsigned number.

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

Global Pointer

32-bit unsigned number.

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

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

Packet Sequence Number This Packet (PSNTP)

16-bit unsigned number.

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

Packet Sequence Number Last Recieved (PSNLR)

16-bit unsigned number.

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

Delta Time Last Received (DELTATLR)

16-bit unsigned integer.

The value is set according to the scale in SCALEDTLR.

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

Delta Time Last Sent (DELTATLS)

16-bit unsigned integer.

The value is set according to the scale in SCAEDTLS.

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

7. Security Considerations

TBD

8. Privacy Considerations

TBD

9. IANA Considerations

This memo includes no request to IANA.

10. Contributors

TBD

11. References

11.1. References

11.2. Normative References

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

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

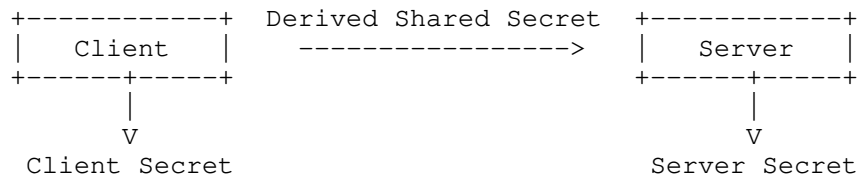
11.3. Informative References

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

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

A.1. One Client / One Server

Let's start with one client and one server.



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

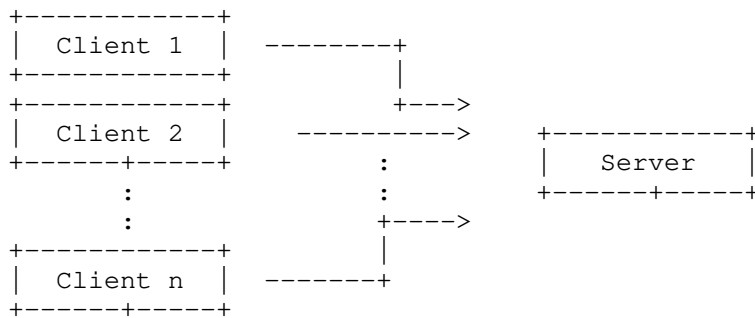
What is stored at the Client and Server to be able to encrypt and decrypt packets? The shared secret or private key.

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

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

A.2. Multiple Clients / One Server

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



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

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

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

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

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

A.3. Multiple Clients / Multiple Servers

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

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

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

Appendix B. Sample Implementation of Registration

B.1. Overall summary

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

B.2. High level flow

The following steps describe the protocol flow:

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

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

B.3. Commands used

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

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

Appendix C. Change Log

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

Appendix D. Open Issues

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

Authors' Addresses

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

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

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

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

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

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

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

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

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

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

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

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

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

Abstract

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

Status of This Memo

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

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

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

This Internet-Draft will expire on 8 August 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

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

1. Introduction

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

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

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

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

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

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

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

2. Conventions Used in This Document

2.1. Requirements Language

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

2.2. Abbreviations

AMM: Alternate-Marking Method.

DLM: Direct Loss Measurement.

HMAC: Hashed Message Authentication Code.

MBZ: Must be Zero.

PM: Performance Measurement.

SHA: Secure Hash Algorithm.

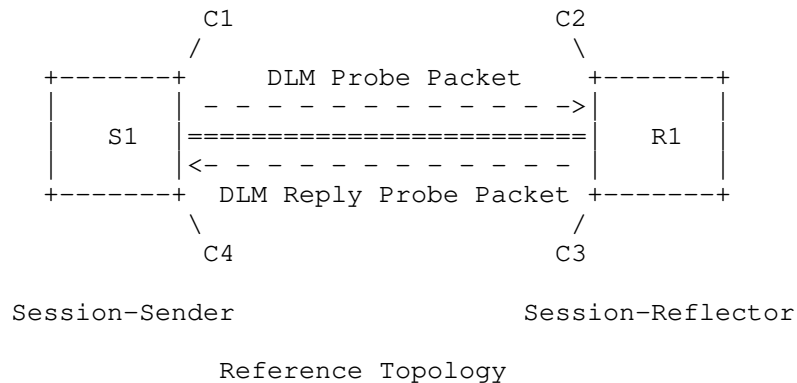
SSID: Sender Session Identifier.

STAMP: Simple Two-Way Active Measurement Protocol.

TTL: Time To Live.

2.3. Reference Topology

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



3. Overview

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

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

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

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

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

4. Session-Sender Direct Loss Measurement Probe Packet

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

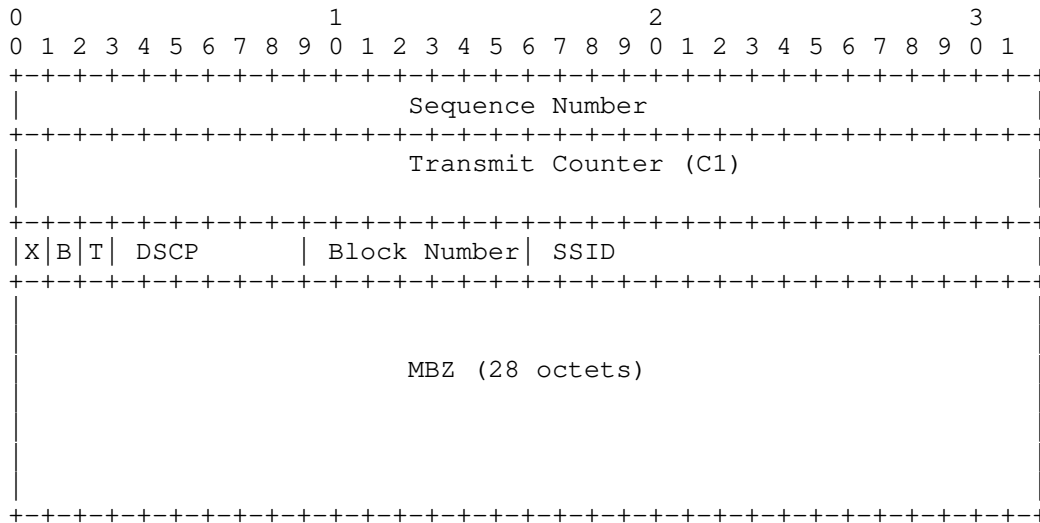


Figure 1: Session-Sender Direct Loss Measurement Probe Packet - Unauthenticated Mode

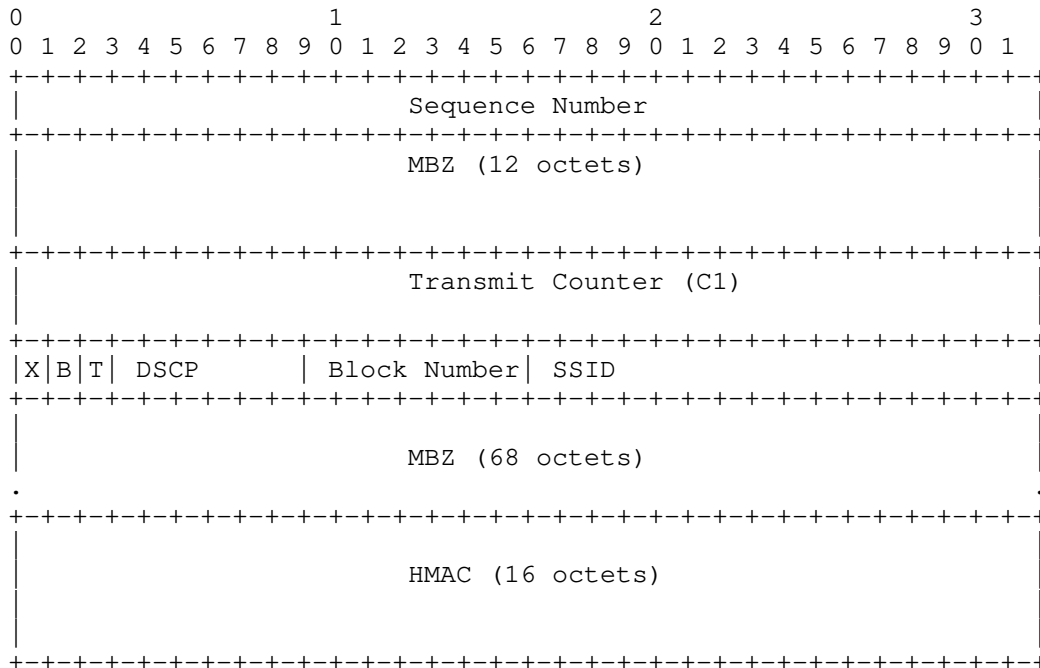


Figure 2: Session-Sender Direct Loss Measurement Probe Packet - Authenticated Mode

Fields are defined as the following:

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

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

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

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

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

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

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

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

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

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

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

5. Session-Reflector Direct Loss Measurement Probe Packet

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

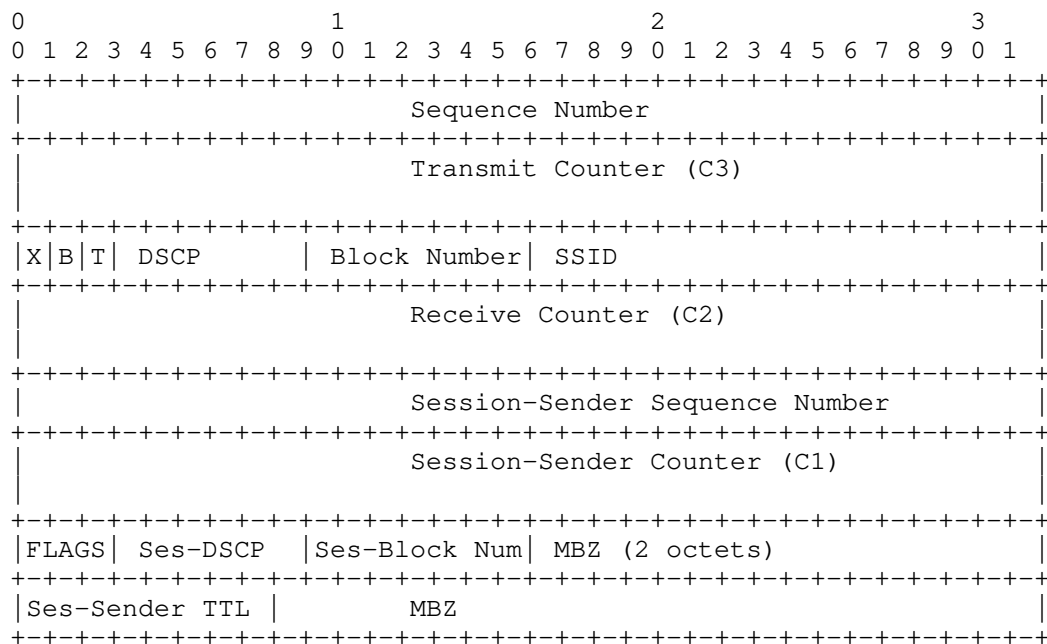


Figure 3: Session-Reflector Direct Loss Measurement Probe Packet - Unauthenticated Mode

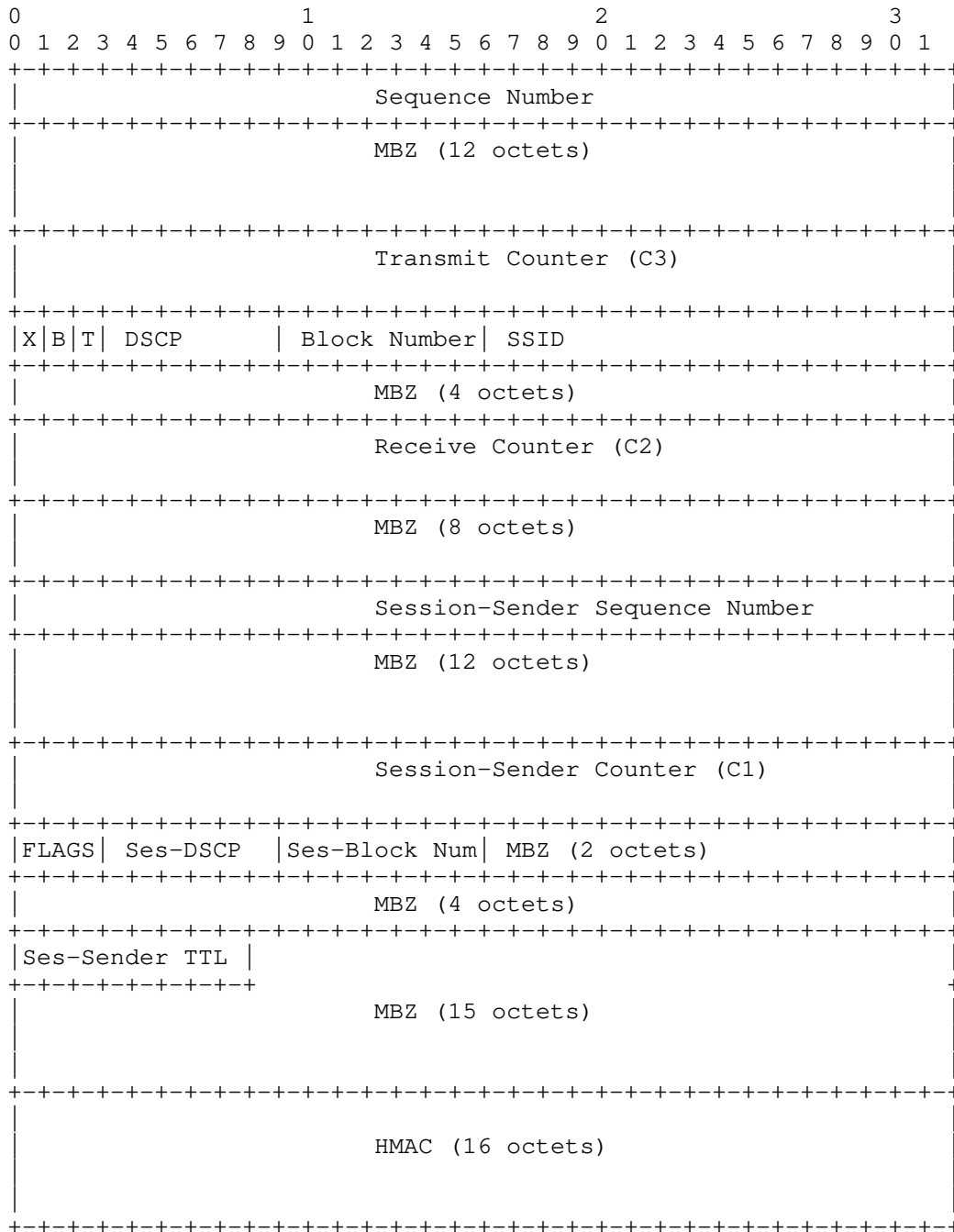


Figure 4: Session-Reflector Direct Loss Measurement Probe Packet -
Authenticated Mode

Fields are defined as the following:

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

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

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

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

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

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

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

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

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

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

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

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

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

6. Data Loss Calculation

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

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

The Total Transmit and Receive Loss are calculated as follows:

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

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

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

7. Optional Extensions

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

8. Integrity Protection and Confidentiality Protection

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

9. Operational Considerations

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

10. Security Considerations

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

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

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

11. IANA Considerations

This document has no IANA actions.

12. References

12.1. Normative References

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

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

12.2. Informative References

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

Acknowledgments

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

Authors' Addresses

Rakesh Gandhi (editor)
Cisco Systems, Inc.
Canada

Email: rgandhi@cisco.com

Clarence Filsfils
Cisco Systems, Inc.

Email: cfilsfil@cisco.com

Daniel Voyer
Bell Canada

Email: daniel.voyer@bell.ca

Mach(Guoyi) Chen
Huawei

Email: mach.chen@huawei.com

Bart Janssens
Colt

Email: Bart.Janssens@colt.net

Stefano Salsano
Universita di Roma "Tor Vergata"
Italy

Internet-Draft Simple Direct Loss Measurement Procedure February 2022

Email: stefano.salsano@uniroma2.it

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

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

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

Abstract

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

Requirements Language

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

Status of This Memo

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

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

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

This Internet-Draft will expire on 28 April 2022.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction	2
2. Terminology	3
3. Problem Statement	3
3.1. Frequent and Dynamic Change of Flows	3
3.1.1. Tidal Effect	4
3.1.2. UPF Expansion	4
3.2. Enterprise Service Demand	4
3.3. Large Scale Network Monitor Deployment and Maintenance	4
3.4. Service Flow Path Change	5
4. Requirement	5
4.1. Ingress Flow Learning	5
4.2. Egress Flow Learning	5
4.3. Hop-by-Hop Flow Learning	6
4.4. Auto Flow Aging	6
4.5. Flow Learning Policy	6
5. IANA Considerations	6
6. Security Considerations	6
7. References	6
7.1. Normative References	6
7.2. Informative References	7
Authors' Addresses	7

1. Introduction

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

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

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

2. Terminology

OAM: Operations, Administration, and Maintenance

SLA: Service Level Agreement

NFV: Network Function Virtualization

UNI: User-Network-Interface

CN: Core Network

3. Problem Statement

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

3.1. Frequent and Dynamic Change of Flows

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

3.1.1. Tidal Effect

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

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

3.1.2. UPF Expansion

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

3.2. Enterprise Service Demand

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

3.3. Large Scale Network Monitor Deployment and Maintenance

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

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

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

3.4. Service Flow Path Change

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

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

4. Requirement

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

4.1. Ingress Flow Learning

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

4.2. Egress Flow Learning

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

4.3. Hop-by-Hop Flow Learning

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

4.4. Auto Flow Aging

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

4.5. Flow Learning Policy

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

5. IANA Considerations

This document has no request to IANA

6. Security Considerations

TBD

7. References

7.1. Normative References

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

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

7.2. Informative References

[I-D.ietf-6man-ipv6-alt-mark]
Fioccola, G., Zhou, T., Cociglio, M., Qin, F., and R. Pang, "IPv6 Application of the Alternate Marking Method", Work in Progress, Internet-Draft, draft-ietf-6man-ipv6-alt-mark-12, 22 October 2021, <<https://www.ietf.org/archive/id/draft-ietf-6man-ipv6-alt-mark-12.txt>>.

[I-D.ietf-mpls-inband-pm-encapsulation]
Cheng, W., Min, X., Zhou, T., Dong, X., and Y. Peleg, "Encapsulation For MPLS Performance Measurement with Alternate Marking Method", Work in Progress, Internet-Draft, draft-ietf-mpls-inband-pm-encapsulation-02, 25 October 2021, <<https://www.ietf.org/archive/id/draft-ietf-mpls-inband-pm-encapsulation-02.txt>>.

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

Authors' Addresses

Liuyan Han
China Mobile
Beijing
China

Email: hanliuyan@chinamobile.com

Minxue Wang
China Mobile
Beijing
China

Email: wangminxue@chinamobile.com

Fan Yang
Huawei Technologies
Beijing
China

Email: shirley.yangfan@huawei.com

Jinming Huang
Huawei Technologies
Dongguan
China

Email: zhangshengli4@huawei.com

IPPM
Internet-Draft
Intended status: Standards Track
Expires: April 16, 2022

H. Song
Futurewei
B. Gafni
Nvidia
T. Zhou
Z. Li
Huawei
F. Brockners
Cisco
S. Bhandari, Ed.
Thoughtspot
R. Sivakolundu
Cisco
T. Mizrahi, Ed.
Huawei
October 13, 2021

In-situ OAM Direct Exporting
draft-ietf-ippm-ioam-direct-export-07

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 or locally aggregated without being pushed into in-flight data packets. The exporting method and format are outside the scope of this document.

Status of This Memo

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

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

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

This Internet-Draft will expire on April 16, 2022.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction	2
2. Conventions	3
2.1. Requirement Language	3
2.2. Terminology	3
3. The Direct Exporting (DEX) IOAM Option Type	3
3.1. Overview	3
3.1.1. DEX Packet Selection	5
3.1.2. Responding to the DEX Trigger	5
3.2. The DEX Option Format	6
4. IANA Considerations	8
4.1. IOAM Type	8
4.2. IOAM DEX Flags	8
4.3. IOAM DEX Extension-Flags	9
5. Performance Considerations	9
6. Security Considerations	10
7. Acknowledgments	11
8. References	11
8.1. Normative References	11
8.2. Informative References	11
Appendix A. Hop Limit in Direct Exporting	12
Authors' Addresses	12

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 locally aggregate and process IOAM data, and/or to export it to a receiving entity (or entities). Throughout the document this functionality is referred to as collection and/or exporting. 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.

Note that even though the IOAM Option-Type is called "Direct Export", it depends on the deployment whether the receipt of a packet with DEX option type leads to the creation of another packet. Some deployments might simply use the packet with the DEX option type to trigger local processing of OAM data. The functionality of this local processing is not within the scope of this document.

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", "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. 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 collecting IOAM data locally or for exporting it to a receiving entity (or entities). Specifically, the DEX option can be used as a trigger for collecting IOAM data by an IOAM node and locally aggregating it; thus, this

aggregated data can be periodically pushed to a receiving entity, or pulled by a receiving entity on-demand.

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].

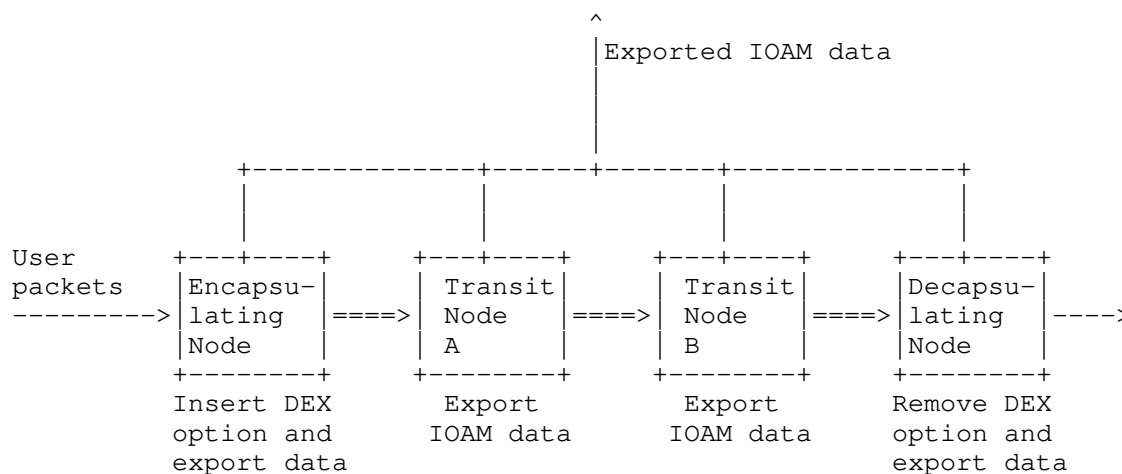


Figure 1: DEX Architecture

The DEX option is used as a trigger to collect and/or 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 (possibly a subset of) the packets it forwards with the DEX option, and MAY export and/or collect the requested IOAM data immediately. Only IOAM encapsulating nodes are allowed to add the DEX option type to a packet.
- o A transit node that processes a packet with the DEX option MAY export and/or collect the requested IOAM data.
- o An IOAM decapsulating node that processes a packet with the DEX option MAY export and/or collect the requested IOAM data, and MUST decapsulate the IOAM header.

As in [I-D.ietf-ippm-ioam-data], the DEX option can be incorporated into all or a subset of the traffic that is forwarded by the encapsulating node, as further discussed in Section 3.1.1 below. Moreover, IOAM nodes respond to the DEX trigger by exporting and/or collection IOAM data either for all traversing packets that carry the DEX option, or selectively only for a subset of these packets, as further discussed in Section 3.1.2 below.

3.1.1. DEX Packet Selection

If an IOAM encapsulating node incorporates the DEX option into all the traffic it forwards it may lead to an excessive amount of exported data, which may overload the network and the receiving entity. Therefore, an IOAM encapsulating node that supports the DEX option MUST support the ability to incorporate the DEX option selectively into a subset of the packets that are forwarded by it.

Various methods of packet selection and sampling have been previously defined, such as [RFC7014] and [RFC5475]. Similar techniques can be applied by an IOAM encapsulating node to apply DEX to a subset of the forwarded traffic.

The subset of traffic that is forwarded or transmitted with a DEX option SHOULD NOT exceed $1/N$ of the interface capacity on any of the IOAM encapsulating node's interfaces. It is noted that this requirement applies to the total traffic that incorporates a DEX option, including traffic that is forwarded by the IOAM encapsulating node and probe packets that are generated by the IOAM encapsulating node. In this context N is a parameter that can be configurable by network operators. If there is an upper bound, M , on the number of IOAM transit nodes in any path in the network, then it is recommended to use an N such that $N \gg M$. The rationale is that a packet that includes a DEX option may trigger an exported packet from each IOAM transit node along the path for a total of M exported packets. Thus, if $N \gg M$ then the number of exported packets is significantly lower than the number of data packets forwarded by the IOAM encapsulating node. If there is no prior knowledge about the network topology or size, it is recommended to use $N > 100$.

3.1.2. Responding to the DEX Trigger

The DEX option specifies which data fields should be exported and/or collected, as specified in Section 3.2. As mentioned above, the data can be locally collected, and optionally can be aggregated and exported to a receiving entity, either proactively or on-demand. If IOAM data is exported, 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].

An IOAM node that performs DEX-triggered exporting MUST support the ability to limit the rate of the exported packets. The rate of exported packets SHOULD be limited so that the number of exported packets is significantly lower than the number of packets that are forwarded by the device. The exported data rate SHOULD NOT exceed $1/N$ of the interface capacity on any of the IOAM node's interfaces. It is recommended to use $N > 100$. Depending on the IOAM node's architecture considerations, the export rate may be limited to a lower number in order to avoid loading the IOAM node. An IOAM node MAY maintain a counter or a set of counters that count the events in which the IOAM node receives a packet with the DEX option type and does not collect and/or export data due to the rate limits.

Exported packets SHOULD NOT be exported over a path or a tunnel that is subject to IOAM direct exporting. Furthermore, IOAM encapsulating nodes that can identify a packet as an IOAM exported packet MUST NOT push a DEX option into such a packet. This requirement is intended to prevent nested exporting and/or exporting loops.

A transit or decapsulating IOAM node that receives an unknown IOAM option type ignores it (as defined in [I-D.ietf-ippm-ioam-data]), and specifically nodes that do not support the DEX option ignore it. Note that as per [I-D.ietf-ippm-ioam-data] a decapsulating node removes the IOAM encapsulation and all its IOAM options, and specifically in the case where one of these options is a (possibly unknown) DEX option.

3.2. The DEX Option Format

The format of the DEX option is depicted in Figure 2. The length of the DEX option is at least 8 octets. The DEX option MAY include one or more optional fields. The existence of the optional fields is indicated by the corresponding flags in the Extension-Flags field. Two optional fields are defined in this document, the Flow ID and the Sequence Number fields. Every optional field MUST be exactly 4 octets long. Thus, the Extension-Flags field explicitly indicates the length of the DEX option. Defining a new optional field requires an allocation of a corresponding flag in the Extension-Flags field, as specified in Section 4.2.

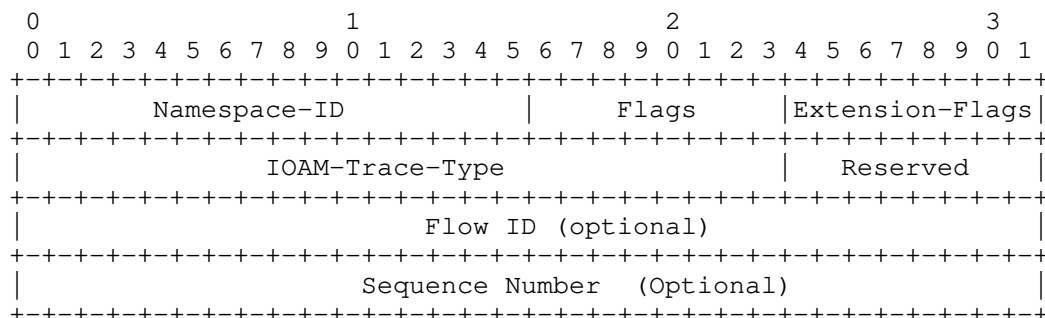


Figure 2: DEX Option Format

Namespace-ID	A 16-bit identifier of the IOAM namespace, as defined in [I-D.ietf-ippm-ioam-data].
Flags	An 8-bit field, comprised of 8 one-bit subfields. Flags are allocated by IANA, as defined in Section 4.2.
Extension-Flags	An 8-bit field, comprised of 8 one-bit subfields. Extension-Flags are allocated by IANA, as defined in Section 4.3. Every bit in the Extension-Flag field that is set to 1 indicates the existence of a corresponding optional 4-octet field. An IOAM node that receives a DEX option with an unknown flag set to 1 MUST ignore the corresponding optional field.
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, the bit that 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.
Optional fields	The optional fields, if present, reside after the Reserved field. The order of the optional fields is according to the respective bits that are enabled in the Extension-Flags field. Each optional field is 4 octets long.

Flow ID An optional 32-bit field representing the 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 An optional 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 8 flag bits. Allocation is based on the "RFC Required" procedure, as defined in [RFC8126].

New registration requests MUST use the following template:

Bit: Desired bit to be allocated in the 8 bit Flags field of the DEX option.

Description: Brief description of the newly registered bit.

Reference: Reference to the document that defines the new bit.

4.3. IOAM DEX Extension-Flags

IANA is requested to define an "IOAM DEX Extension-Flags" registry. This registry includes 8 flag bits. Bit 0 (the most significant bit) and bit 1 in the registry are allocated by this document, and described in Section 3.2. Allocation of the other bits should be performed based on the "RFC Required" procedure, as defined in [RFC8126].

Bit 0 "Flow ID [RFC XXXX] [RFC Editor: please replace with the RFC number of the current document]"

Bit 1 "Sequence Number [RFC XXXX] [RFC Editor: please replace with the RFC number of the current document]"

New registration requests MUST use the following template:

Bit: Desired bit to be allocated in the 8 bit Extension-Flags field of the DEX option.

Description: Brief description of the newly registered bit.

Reference: Reference to the document that defines the new bit.

5. Performance Considerations

The DEX option triggers IOAM data to be collected and/or 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, the performance impact of these exported packets is limited by taking two measures: at the encapsulating nodes, by selective DEX encapsulation (Section 3.1.1), and at the transit nodes, by limiting exporting rate (Section 3.1.2). These two measures ensure that direct exporting is used at a rate that does not significantly affect the network bandwidth, and does not overload the receiving entity. Moreover, it is possible to load balance the exported data among multiple receiving entities, although the exporting method is not within the scope of this document.

It should be noted that in some networks DEX data may be exported over an out-of-band network, in which a large volume of exported traffic does not compromise user traffic. In this case an operator may choose to disable the exporting rate limiting.

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.

The amplification effect of DEX may be worse in wide area networks in which there are multiple IOAM domains. For example, if DEX is used in IOAM domain 1 for exporting IOAM data to a receiving entity, then the exported packets of domain 1 can be forwarded through IOAM domain 2, in which they are subject to DEX. The exported packets of domain 2 may in turn be forwarded through another IOAM domain (or through domain 1), and theoretically this recursive amplification may continue infinitely.

In order to mitigate the attacks described above, the following requirements (Section 3) have been defined:

- o Selective DEX (Section 3.1.1) is applied by IOAM encapsulating nodes in order to limit the potential impact of DEX attacks to a small fraction of the traffic.
- o Rate limiting of exported traffic (Section 3.1.2) is applied by IOAM nodes in order to prevent overloading attacks and in order to significantly limit the scale of amplification attacks.
- o IOAM encapsulating nodes are required to avoid pushing the DEX option into IOAM exported packets (Section 3.1.2), thus preventing some of the amplification and export loop scenarios.

Although the exporting method is not within the scope of this document, any exporting method MUST secure the exported data from the IOAM node to the receiving entity. Specifically, an IOAM node that performs DEX exporting MUST send the exported data to a pre-configured trusted receiving entity. Furthermore, an IOAM node MUST

gain explicit consent to export data to a receiving entity before starting to send exported data.

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. Acknowledgments

The authors thank Martin Duke, Tommy Pauly, Greg Mirsky, and other members of the IPPM working group for many helpful comments.

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-15 (work in progress), October 2021.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC5475] Zseby, T., Molina, M., Duffield, N., Niccolini, S., and F. Raspall, "Sampling and Filtering Techniques for IP Packet Selection", RFC 5475, DOI 10.17487/RFC5475, March 2009, <<https://www.rfc-editor.org/info/rfc5475>>.
- [RFC7014] D'Antonio, S., Zseby, T., Henke, C., and L. Peluso, "Flow Selection Techniques", RFC 7014, DOI 10.17487/RFC7014, September 2013, <<https://www.rfc-editor.org/info/rfc7014>>.
- [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-flags]
Mizrahi, T., Brockners, F., Bhandari, S., Sivakolundu, R., Pignataro, C., Kfir, A., Gafni, B., Spiegel, M., and J. Lemon, "In-situ OAM Loopback and Active Flags", draft-ietf-ippm-ioam-flags-06 (work in progress), August 2021.

[I-D.song-ippm-postcard-based-telemetry]

Song, H., Mirsky, G., Filsfils, C., Abdelsalam, A., Zhou, T., Li, Z., Shin, J., and K. Lee, "Postcard-based On-Path Flow Data Telemetry using Packet Marking", draft-song-ippm-postcard-based-telemetry-10 (work in progress), July 2021.

[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-05 (work in progress), July 2021.

[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 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 start 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.

Authors' Addresses

Haoyu Song
Futurewei
2330 Central Expressway
Santa Clara 95050
USA

Email: haoyu.song@futurewei.com

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

Email: gbarak@nvidia.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 (editor)
Thoughtspot
3rd Floor, Indiqube Orion, 24th Main Rd, Garden Layout, HSR Layout
Bangalore, KARNATAKA 560 102
India

Email: shwetha.bhandari@thoughtspot.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
8-2 Matam
Haifa 3190501
Israel

Email: tal.mizrahi.phd@gmail.com

IPPM
Internet-Draft
Intended status: Standards Track
Expires: April 16, 2022

T. Mizrahi
Huawei
F. Brockners
Cisco
S. Bhandari, Ed.
Thoughtspot
R. Sivakolundu
C. Pignataro
Cisco
A. Kfir
B. Gafni
Nvidia
M. Spiegel
Barefoot Networks, an Intel company
J. Lemon
Broadcom
October 13, 2021

In-situ OAM Loopback and Active Flags
draft-ietf-ippm-ioam-flags-07

Abstract

In-situ Operations, Administration, and Maintenance (IOAM) collects operational and telemetry information in packets while they traverse a path between two points in the network. This document defines two new flags in the IOAM Trace Option headers, specifically the Loopback and Active flags.

Status of This Memo

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

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

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

This Internet-Draft will expire on April 16, 2022.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction	2
2. Conventions	3
2.1. Requirements Language	3
2.2. Terminology	3
3. New IOAM Trace Option Flags	3
4. Loopback in IOAM	3
4.1. Loopback: Encapsulating Node Functionality	4
4.1.1. Loopback Packet Selection	5
4.2. Receiving and Processing Loopback	6
4.3. Loopback on the Return Path	7
4.4. Terminating a Looped Back Packet	7
5. Active Measurement with IOAM	7
6. IANA Considerations	9
7. Performance Considerations	9
8. Security Considerations	10
9. Acknowledgments	11
10. References	11
10.1. Normative References	11
10.2. Informative References	12
Authors' Addresses	12

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.

The Loopback flag is used to request that each transit device along the path loops back a truncated copy of the data packet to the sender. The Active flag indicates that a packet is used for active measurement. The term active measurement in the context of this document is as defined in [RFC7799].

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. 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). When set, the Loopback flag triggers sending a copy of a packet back towards the source, as further described in Section 4.

Bit 2 "Active" (A-bit). When set, the Active flag indicates that a packet is an active measurement packet rather than a data packet, where "active" is used in the sense defined in [RFC7799]. 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

The Loopback flag is used to request that each transit device along the path loops back a truncated copy of the data packet to the sender. 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 than one round-trip-time, and by sending a single packet.

As illustrated in Figure 1, an IOAM encapsulating node can push an IOAM encapsulation that includes the Loopback flag onto some or all of the packets it forwards. The IOAM transit node and the decapsulating node both create copies of the packet and loop them back to the encapsulating node. The decapsulating node also terminates the IOAM encapsulation, and then forwards the packet towards the destination. The two IOAM looped back copies are terminated by the encapsulating node.

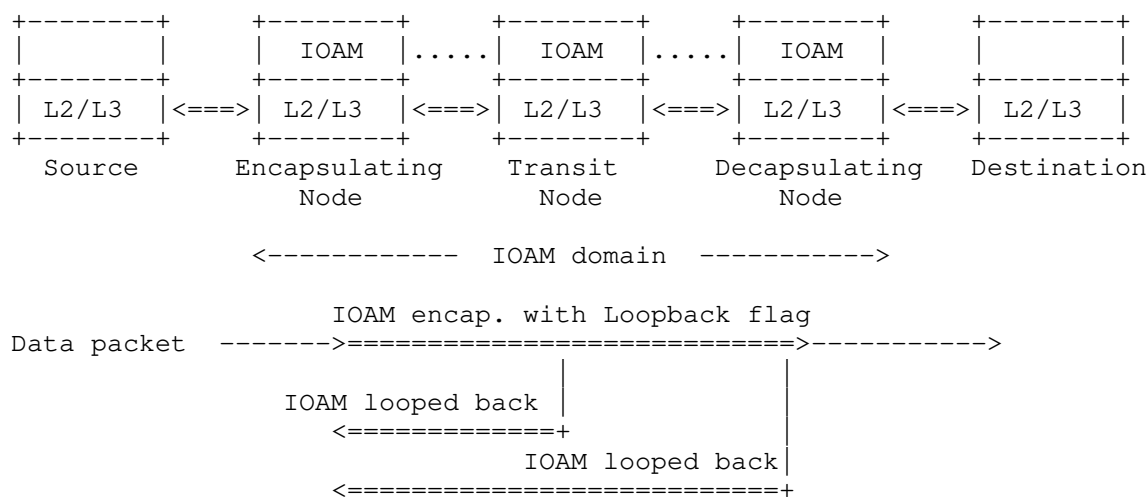


Figure 1: Loopback in IOAM.

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.

4.1. Loopback: Encapsulating Node Functionality

The encapsulating node either generates synthetic packets with an IOAM trace option that has the Loopback flag set, or sets the loopback 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 flag 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. A transit node that receives a packet with an IOAM trace option that has the Loopback flag set and the IOAM-Trace-Type is not equal to '1' in the most significant bit and '0' in the rest of the bits, MUST NOT loop back a copy of the packet. The reason for allowing a single data field per hop is to minimize the impact of amplification attacks.

IOAM encapsulating nodes MUST NOT push an IOAM encapsulation with the Loopback flag onto data packets that already include an IOAM encapsulation. This requirement is intended to prevent IOAM Loopback nesting, where looped back packets may be subject to loopback in a nested IOAM domain.

4.1.1. Loopback Packet Selection

If an IOAM encapsulating node incorporates the Loopback flag into all the traffic it forwards it may lead to an excessive amount of looped back packets, which may overload the network and the encapsulating node. Therefore, an IOAM encapsulating node that supports the Loopback flag MUST support the ability to incorporate the Loopback flag selectively into a subset of the packets that are forwarded by it.

Various methods of packet selection and sampling have been previously defined, such as [RFC7014] and [RFC5475]. Similar techniques can be applied by an IOAM encapsulating node to apply Loopback to a subset of the forwarded traffic.

The subset of traffic that is forwarded or transmitted with a Loopback flag SHOULD NOT exceed $1/N$ of the interface capacity on any of the IOAM encapsulating node's interfaces. It is noted that this requirement applies to the total traffic that incorporates a Loopback flag, including traffic that is forwarded by the IOAM encapsulating node and probe packets that are generated by the IOAM encapsulating node. In this context N is a parameter that can be configurable by network operators. If there is an upper bound, M , on the number of IOAM transit nodes in any path in the network, then it is recommended to use an N such that $N \gg M$. The rationale is that a packet that

includes the Loopback flag triggers a looped back packet from each IOAM transit node along the path for a total of M looped back packets. Thus, if $N \gg M$ then the number of looped back packets is significantly lower than the number of data packets forwarded by the IOAM encapsulating node. If there is no prior knowledge about the network topology or size, it is recommended to use $N > 100$.

The loopback flag MUST NOT be set if it is not guaranteed that there is a return path from each of the IOAM transit and IOAM decapsulating nodes, or if the encapsulating node's identity is not available in the encapsulation header.

4.2. Receiving and Processing Loopback

A Loopback flag 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. If the address of the encapsulating node is not available in the encapsulation header, then the transit/decapsulating node does not loop back a copy of the original 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.

An IOAM node that supports the reception and processing of the Loopback flag MUST support the ability to limit the rate of the looped back packets. The rate of looped back packets SHOULD be limited so that the number of looped back packets is significantly lower than the number of packets that are forwarded by the device. The looped back data rate SHOULD NOT exceed $1/N$ of the interface capacity on any of the IOAM node's interfaces. It is recommended to use $N > 100$. Depending on the IOAM node's architecture considerations, the loopback response rate may be limited to a lower number in order to avoid loading the IOAM node.

4.3. Loopback on the Return Path

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).

4.4. Terminating a Looped Back Packet

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 and IOAM-Namespace match 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. Specifically, 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 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 2. 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].

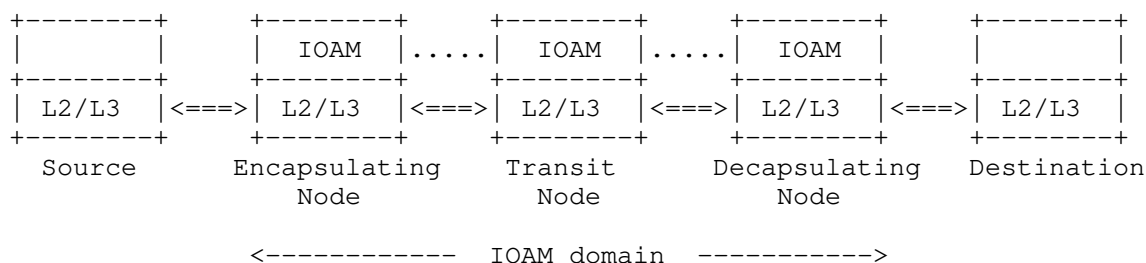


Figure 2: Network using IOAM.

This draft focuses on three possible use cases of active measurement using IOAM. These use cases are described using the example of Figure 2.

- 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.

If the volume of traffic that incorporates the Active flag is large, it may overload the network and the IOAM node(s) that process the active measurement packet. Thus, the rate of the traffic that includes the Active flag rate SHOULD NOT exceed $1/N$ of the interface capacity on any of the IOAM node's interfaces. It is recommended to use $N > 100$. Depending on the IOAM node's architecture considerations, the rate of Active-enabled IOAM packets may be limited to a lower number in order to avoid loading the IOAM node.

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, as further discussed in Section 4.1.1 and Section 5.

Transit and decapsulating nodes that support Loopback need to be able to limit the looped back packets (Section 4.2) 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 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.

IOAM is assumed to be deployed in a restricted administrative domain, thus limiting the scope of the threats above and their effect. This is a fundamental assumption with respect to the security aspects of IOAM, as further discussed in [I-D.ietf-ippm-ioam-data]. However, even given this limited scope, security threats should still be considered and mitigated. Specifically, 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 degradation. 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.

Some of the security threats that were discussed in this document may be worse in a wide area network in which there are nested IOAM domains. For example, if there are two nested IOAM domains that use loopback, then a looped-back copy in the outer IOAM domain may be forwarded through another (inner) IOAM domain and may be subject to loopback in that (inner) IOAM domain, causing the amplification to be worse than in the conventional case.

In order to mitigate the performance-related 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, IOAM nodes should be able to:

- o Limit the rate of IOAM packets with the Loopback flag (IOAM encapsulating nodes), as discussed in Section 4.1.1.
- o Limit the rate of looped back packets (IOAM transit and decapsulating nodes), as discussed in Section 4.2.
- o Limit the rate of IOAM packets with the Active flag (IOAM encapsulating nodes), as discussed in Section 5.

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 significantly limiting the possible impact of an amplification attack.

9. Acknowledgments

The authors thank Martin Duke, Tommy Pauly, Greg Mirsky, and other members of the IPPM working group for many helpful comments.

10. References

10.1. 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-15 (work in progress), October 2021.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.

- [RFC5475] Zseby, T., Molina, M., Duffield, N., Niccolini, S., and F. Raspall, "Sampling and Filtering Techniques for IP Packet Selection", RFC 5475, DOI 10.17487/RFC5475, March 2009, <<https://www.rfc-editor.org/info/rfc5475>>.
- [RFC7014] D'Antonio, S., Zseby, T., Henke, C., and L. Peluso, "Flow Selection Techniques", RFC 7014, DOI 10.17487/RFC7014, September 2013, <<https://www.rfc-editor.org/info/rfc7014>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

10.2. Informative References

- [I-D.ietf-ippm-ioam-ipv6-options]
Bhandari, S. and F. Brockners, "In-situ OAM IPv6 Options", draft-ietf-ippm-ioam-ipv6-options-06 (work in progress), July 2021.
- [I-D.ietf-sfc-ioam-nsh]
Brockners, F. and S. Bhandari, "Network Service Header (NSH) Encapsulation for In-situ OAM (IOAM) Data", draft-ietf-sfc-ioam-nsh-06 (work in progress), July 2021.
- [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-05 (work in progress), July 2021.
- [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
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 (editor)
Thoughtspot
3rd Floor, Indiqube Orion, 24th Main Rd, Garden Layout, HSR Layout
Bangalore, KARNATAKA 560 102
India

Email: shwetha.bhandari@thoughtspot.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
Nvidia

Email: avivk@nvidia.com

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

Email: gbarak@nvidia.com

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

Email: mickey.spiegel@intel.com

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

Email: jennifer.lemon@broadcom.com

IPPM Working Group
Internet-Draft
Intended status: Standards Track
Expires: 6 August 2022

R. Gandhi, Ed.
C. Filsfils
Cisco Systems, Inc.
D. Voyer
Bell Canada
M. Chen
Huawei
B. Janssens
Colt
R. Foote
Nokia
2 February 2022

Simple TWAMP (STAMP) Extensions for Segment Routing Networks
draft-ietf-ippm-stamp-srpm-03

Abstract

Segment Routing (SR) leverages the source routing paradigm. SR is applicable to both Multiprotocol Label Switching (SR-MPLS) and IPv6 (SRv6) forwarding planes. This document specifies RFC 8762 (Simple Two-Way Active Measurement Protocol (STAMP)) extensions for SR networks, for both SR-MPLS and SRv6 forwarding planes by augmenting the optional extensions defined in RFC 8972.

Status of This Memo

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

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

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

This Internet-Draft will expire on 6 August 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction	2
2. Conventions Used in This Document	3
2.1. Requirements Language	3
2.2. Abbreviations	3
2.3. Reference Topology	4
3. Destination Node Address TLV	4
4. Return Path TLV	6
4.1. Return Path Sub-TLVs	7
4.1.1. Return Path Control Code Sub-TLV	7
4.1.2. Return Address Sub-TLV	8
4.1.3. Return Segment List Sub-TLVs	8
5. Security Considerations	11
6. IANA Considerations	12
7. References	13
7.1. Normative References	13
7.2. Informative References	14
Acknowledgments	15
Authors' Addresses	15

1. Introduction

Segment Routing (SR) leverages the source routing paradigm for Software Defined Networks (SDNs). SR is applicable to both Multiprotocol Label Switching (SR-MPLS) and IPv6 (SRv6) forwarding planes [RFC8402]. SR Policies as defined in [I-D.ietf-spring-segment-routing-policy] are used to steer traffic through a specific, user-defined paths using a stack of Segments. A comprehensive SR Performance Measurement (PM) toolset is one of the essential requirements to measure network performance to provide Service Level Agreements (SLAs).

The Simple Two-Way Active Measurement Protocol (STAMP) provides capabilities for the measurement of various performance metrics in IP networks [RFC8762] without the use of a control channel to pre-signal session parameters. [RFC8972] defines optional extensions, in the form of TLVs, for STAMP. Note that the YANG data model defined in [I-D.ietf-ippm-stamp-yang] can be used to provision the STAMP Session-Sender and STAMP Session-Reflector.

The STAMP test packets are transmitted along an IP path between a Session-Sender and a Session-Reflector to measure performance delay and packet loss along that IP path. It may be desired in SR networks that the same path (same set of links and nodes) between the Session-Sender and Session-Reflector is used for the STAMP test packets in both directions. This is achieved by using the STAMP [RFC8762] extensions for SR-MPLS and SRv6 networks specified in this document by augmenting the optional extensions defined in [RFC8972].

2. Conventions Used in This Document

2.1. Requirements Language

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

2.2. Abbreviations

MPLS: Multiprotocol Label Switching.

PM: Performance Measurement.

SID: Segment ID.

SL: Segment List.

SR: Segment Routing.

SR-MPLS: Segment Routing with MPLS forwarding plane.

SRv6: Segment Routing with IPv6 forwarding plane.

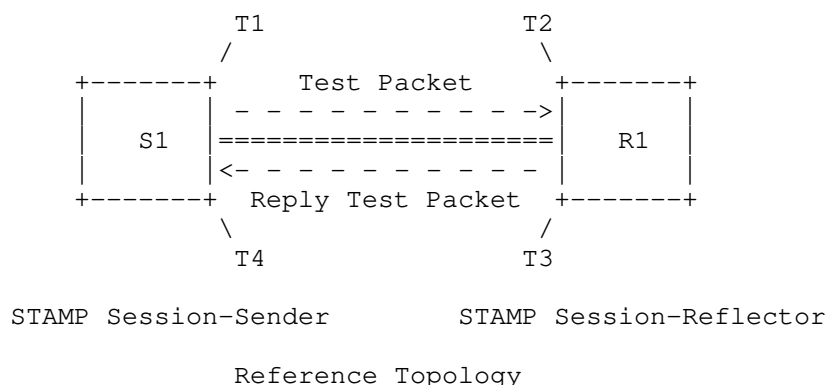
SSID: STAMP Session Identifier.

STAMP: Simple Two-Way Active Measurement Protocol.

2.3. Reference Topology

In the reference topology shown below, the STAMP Session-Sender S1 initiates a STAMP test packet and the STAMP Session-Reflector R1 transmits a reply STAMP test packet. The reply test packet may be transmitted to the Session-Sender S1 on the same path (same set of links and nodes) or a different path in the reverse direction from the path taken towards the Session-Reflector R1.

The nodes S1 and R1 may be connected via a link or an SR path [RFC8402]. The link may be a physical interface, virtual link, or Link Aggregation Group (LAG) [IEEE802.1AX], or LAG member link. The SR path may be an SR Policy [I-D.ietf-spring-segment-routing-policy] on node S1 (called head-end) with destination to node R1 (called tail-end).



3. Destination Node Address TLV

The Session-Sender may need to transmit test packets to the Session-Reflector with a different destination address not matching an address on the Session-Reflector e.g. when the STAMP test packet is encapsulated by a tunneling protocol or an MPLS Segment List with IPv4 address from 127/8 range or Segment Routing Header (SRH) with IPv6 address `::1/128`. For testing ECMPs, the Session-Sender may select different IPv4 addresses from 127/8 range or select different Flow Label values for IPv6. When using IPv4 destination address from 127/8 range, the STAMP test packet may not reach the intended Session-Reflector in an error condition, an un-intended node may transmit reply test packets resulting in reporting of invalid measurement metrics.

[RFC8972] defines STAMP test packets that can include one or more optional TLVs. In this document, Destination Node Address TLV (Type TBA1) is defined for STAMP test packet [RFC8972] and has the following format shown in Figure 1:

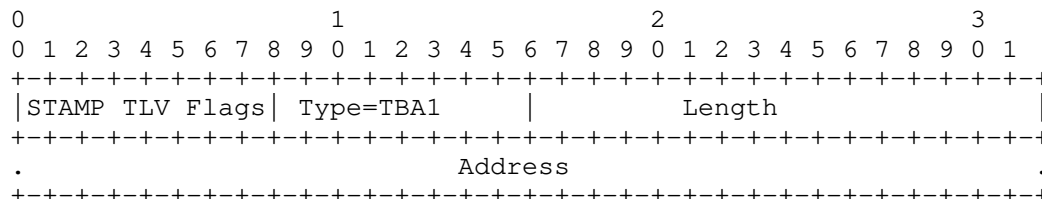


Figure 1: Destination Node Address TLV Format

The Length field is used to decide the Address Family of the Address.

The STAMP TLV Flags are set using the procedures described in [RFC8972].

The Destination Node Address TLV is optional. The Destination Node Address TLV indicates the address of the intended Session-Reflector node of the test packet. When Session-Sender test packet destination address is different than the actual Session-Reflector address, the actual Session-Reflector address SHOULD be transmitted to the Session-Reflector by including a Destination Node Address TLV.

D (Wrong Destination Flag): A one-bit flag at position TBA3.

D (Wrong Destination with Reply Required): A Session-Sender MUST set the D flag to 0 before transmitting an extended STAMP test packet when test packet reply is required. A Session-Reflector that supports this TLV, MUST set the D flag to 1 if the Session-Reflector determined that it is not the intended Destination as identified in the Destination Node Address TLV. Otherwise, the Session-Reflector MUST set the D flag in the reply test packet to 0.

D (Wrong Destination with No Reply Required): A Session-Sender MUST set the D flag to 1 before transmitting an extended STAMP test packet when test packet reply is not required. A Session-Reflector that supports this TLV, MUST NOT reply and MUST drop the packet if the Session-Reflector determined that it is not the intended Destination as identified in the Destination Node Address TLV.

4. Return Path TLV

For end-to-end SR paths, the Session-Reflector may need to transmit the reply test packet on a specific return path. The Session-Sender can request this in the test packet to the Session-Reflector using a Return Path TLV. With this TLV carried in the Session-Sender test packet, signaling and maintaining dynamic SR network state for the STAMP sessions on the Session-Reflector are avoided.

For links, the Session-Reflector may need to transmit the reply test packet on the same incoming link in the reverse direction. The Session-Sender can request this in the test packet to the Session-Reflector using a Return Path TLV.

[RFC8972] defines STAMP test packets that can include one or more optional TLVs. In this document, the TLV Type (value TBA2) is defined for the Return Path TLV that carries the return path for the Session-Sender test packet. The format of the Return Path TLV is shown in Figure 2:

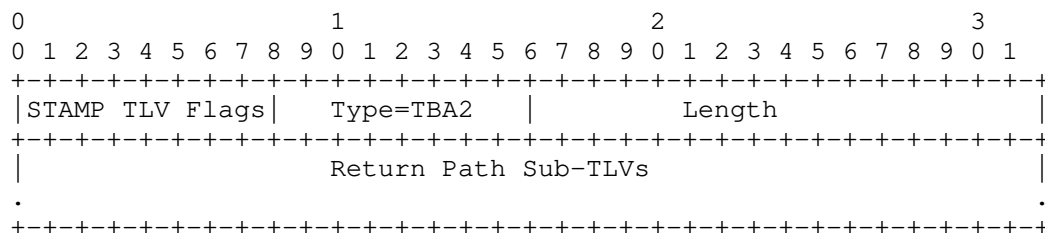


Figure 2: Return Path TLV

The STAMP TLV Flags are set using the procedures described in [RFC8972].

The Return Path TLV is optional. The Session-Sender MUST only insert one Return Path TLV in the STAMP test packet. The Session-Reflector that supports this TLV, MUST only process the first Return Path TLV in the test packet and ignore other Return Path TLVs if present, and it MUST NOT add Return Path TLV in the reply test packet. The Session-Reflector that supports this TLV MUST reply using the Return Path received in the Session-Sender test packet. Otherwise, the procedure defined in [RFC8762] is followed by the Session-Reflector.

4.1. Return Path Sub-TLVs

The Return Path TLV contains one or more Sub-TLVs to carry the information for the requested return path. A Return Path Sub-TLV can carry Return Path Control Code, Return Path IP Address or Return Path Segment List.

The STAMP Sub-TLV Flags are set using the procedures described in [RFC8972].

When Return Path Sub-TLV is present in the Session-Sender test packet, the Session-Reflector that supports this TLV, MUST transmit reply test packet using the return path information specified in the Return Path Sub-TLV.

A Return Path TLV MUST NOT contain both Control Code Sub-TLV as well as Return Address or Return Segment List Sub-TLV.

4.1.1. Return Path Control Code Sub-TLV

The format of the Return Path Control Code Sub-TLV is shown in Figure 3. The Type of the Return Path Control Code Sub-TLV is defined as following:

- * Type (value 1): Return Path Control Code. The Session-Sender can request the Session-Reflector to transmit the reply test packet based on the flags defined in the Control Code field.

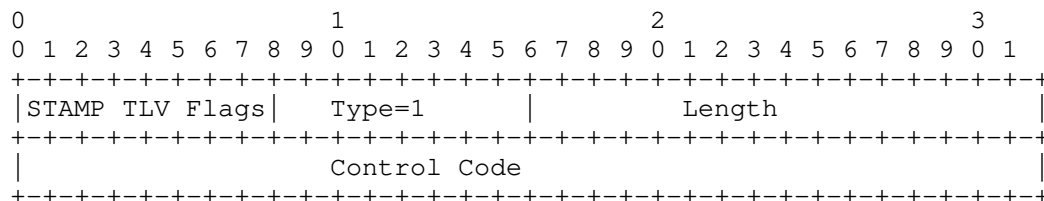


Figure 3: Control Code Sub-TLV in Return Path TLV

Control Code Flags (32-bit): Defined as follows.

0x0: No Reply Requested.

0x1: Reply Requested on the Same Link.

When Control Code flag is set to 0x0 in the Session-Sender test packet, the Session-Reflector does not transmit reply test packet to the Session-Sender and terminates the STAMP test packet. Only the one-way measurement is applicable in this case. Optionally, the

Session-Reflector may locally stream performance metrics via telemetry using the information from the received test packet. All other Return Path Sub-TLVs MUST be ignored in this case.

When Control Code flag is set to 0x1 in the Session-Sender test packet, the Session-Reflector transmits the reply test packet over the same incoming link where the test packet is received in the reverse direction towards the Session-Sender. All other Return Path Sub-TLVs MUST be ignored in this case.

4.1.2. Return Address Sub-TLV

The STAMP reply test packet may be transmitted to the Session-Sender to a different destination address on the Session-Sender using Return Path TLV. For this, the Session-Sender can specify in the test packet the receiving destination node address for the Session-Reflector reply test packet. When transmitting the STAMP test packet to a different destination address, the Session-Sender MUST follow the procedure defined in Section 4.3 of [RFC8762].

The format of the Return Address Sub-TLV is shown in Figure 4. The Address Family field indicates the type of the address, and it SHALL be set to one of the assigned values in the "IANA Address Family Numbers" registry. The Type of the Return Address Sub-TLV is defined as following:

- * Type (value 2): Return Address. Destination node address of the Session-Reflector reply test packet different than the Source Address in the Session-Sender test packet.

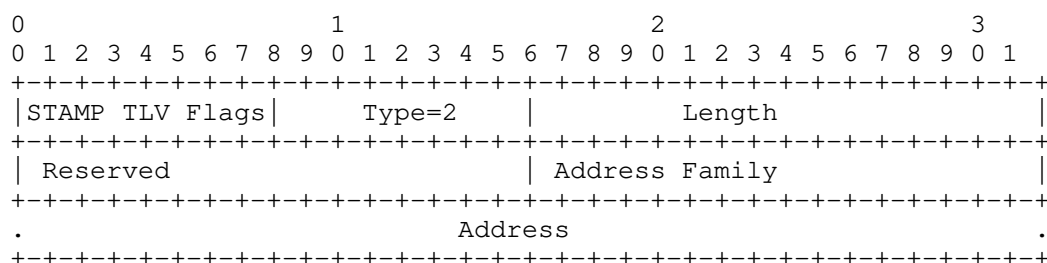


Figure 4: Return Address Sub-TLV in Return Path TLV

4.1.3. Return Segment List Sub-TLVs

The format of the Segment List Sub-TLVs in the Return Path TLV is shown in Figures 5, 6, and 7. The segment entries MUST be in network order. The Segment List Sub-TLV can be one of the following Types:

- * Type (value 3): SR-MPLS Label Stack of the Return Path
- * Type (value 4): SRv6 Segment List of the Return Path
- * Type (value 5): Structured SRv6 Segment List of the Return Path

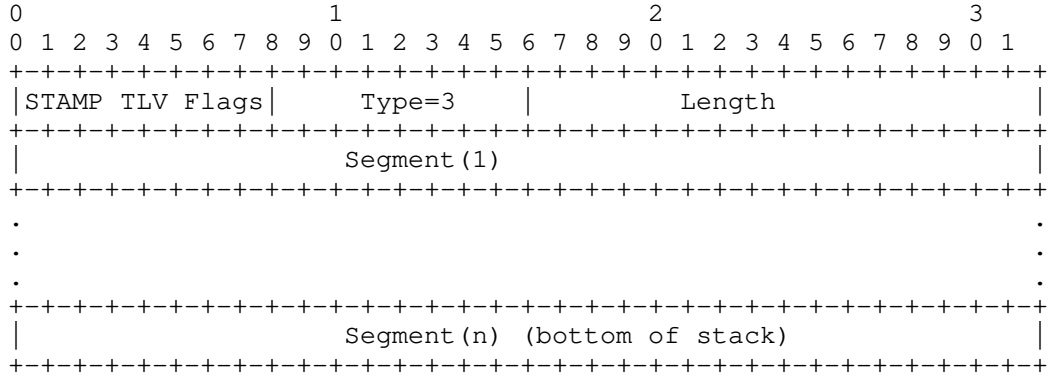


Figure 5: SR-MPLS Segment List Sub-TLV in Return Path TLV

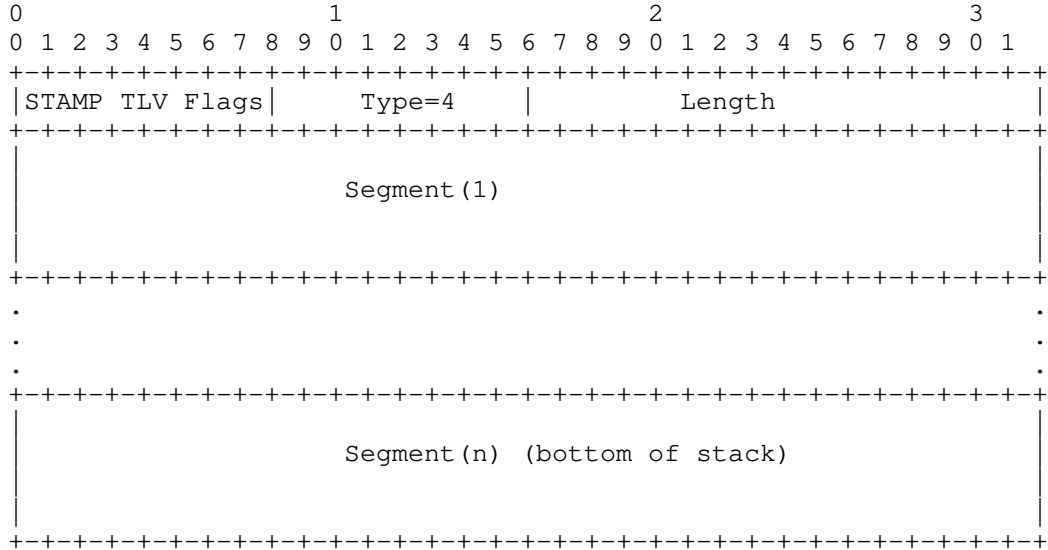


Figure 6: SRv6 Segment List Sub-TLV in Return Path TLV

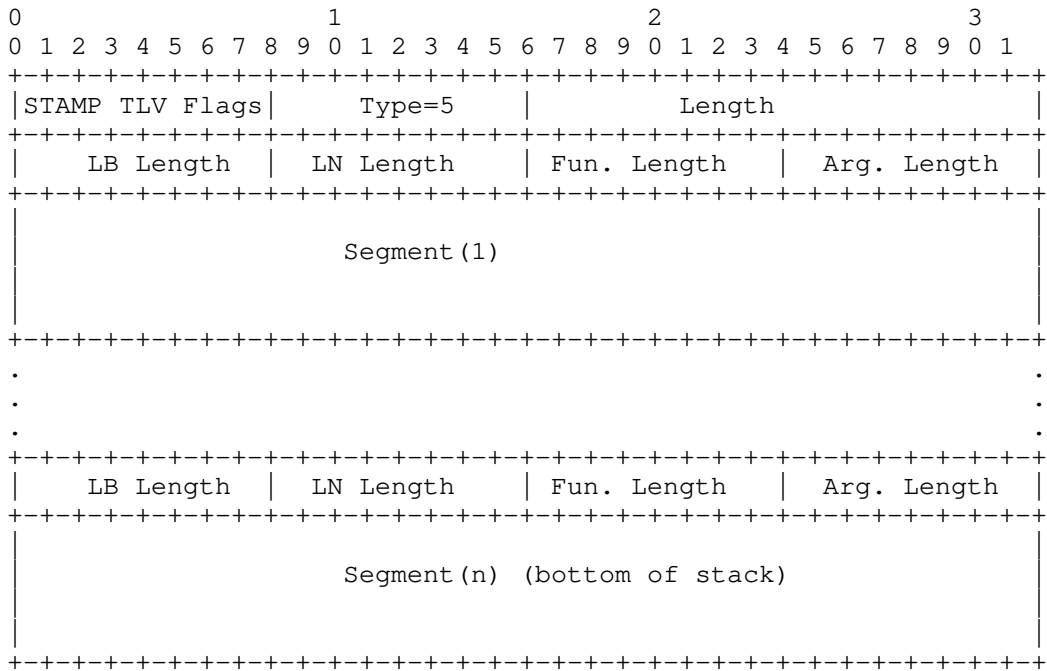


Figure 7: Structured SRv6 Segment List Sub-TLV in Return Path TLV

An SR-MPLS Label Stack Sub-TLV may carry only Binding SID [I-D.ietf-pce-binding-label-sid] of the Return SR-MPLS Policy.

An SRv6 Segment List Sub-TLV and Structured SRv6 Segment List Sub-TLV may carry only Binding SID [I-D.ietf-pce-binding-label-sid] of the Return SRv6 Policy.

A Structured SRv6 Segment List Sub-TLV is used carry the structure and behavior for SRv6 SIDs [RFC8986] used in the Return SRv6 path as shown in Figure 7. The structure is intended for informational use by the control and management planes. The fields in the structure of the Sub-TLV are defined as follows [RFC8986]:

- * LB Length: 1 octet. SRv6 SID Locator Block (LB) length in bits.
- * LN Length: 1 octet. SRv6 SID Locator Node (LN) length in bits.
- * Fun. Length: 1 octet. SRv6 SID Function length in bits.
- * Arg. Length: 1 octet. SRv6 SID Arguments length in bits.

In Structured SRv6 Segment List Sub-TLV, the sum of all four sizes MUST be less than or equal to 128 bits. If the sum of all four sizes is larger than 128 bits, the Sub-TLV MUST be ignored by the Session-Reflector.

The Session-Sender MUST only insert one Segment List Return Path Sub-TLV in the test packet. The Session-Reflector MUST only process the first Segment List Return Path Sub-TLV in the test packet and ignore other Segment List Return Path Sub-TLVs if present.

Note that in addition to P2P SR paths, the Return Segment List Sub-TLV is also applicable to P2MP SR paths. For example, for P2MP SR paths, it may only carry the Node Segment Identifier of the Session-Sender in order for the reply test packet to follow an SR path to the Session-Sender.

5. Security Considerations

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

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

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

STAMP uses the well-known UDP port number that could become a target of denial of service (DoS) or could be used to aid man-in-the-middle (MITM) attacks. Thus, the security considerations and measures to mitigate the risk of the attack documented in Section 6 of [RFC8545] equally apply to the STAMP extensions in this document.

The STAMP extensions defined in this document may be used for potential "proxying" attacks. For example, a Session-Sender may specify a return path that has a destination different from that of the Session-Sender. But normally, such attacks will not happen in an SR domain where the Session-Senders and Session-Reflectors belong to the same domain. In order to prevent using the extension defined in this document for proxying any possible attacks, the return path has destination to the same node where the forward path is from. The

Session-Reflector may drop the Session-Sender test packet when it cannot determine whether the Return Path has the destination to the Session-Sender. That means, the Session-Sender should choose a proper source address according to the specified Return Path to help the Session-Reflector to make that decision.

6. IANA Considerations

IANA has created the "STAMP TLV Types" registry for [RFC8972]. IANA is requested to allocate a value for the Destination Address TLV Type and a value for the Return Path TLV Type from the IETF Review TLV range of the same registry.

Value	Description	Reference
TBA1	Destination Node Address TLV	This document
TBA2	Return Path TLV	This document

Table 1: STAMP TLV Types

IANA is requested to create a sub-registry for "Return Path Sub-TLV Type". All code points in the range 1 through 175 in this registry shall be allocated according to the "IETF Review" procedure as specified in [RFC8126]. Code points in the range 176 through 239 in this registry shall be allocated according to the "First Come First Served" procedure as specified in [RFC8126]. Remaining code points are allocated according to Table 2:

Value	Description	Reference
1 - 175	IETF Review	This document
176 - 239	First Come First Served	This document
240 - 251	Experimental Use	This document
252 - 254	Private Use	This document

Table 2: Return Path Sub-TLV Type Registry

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

Type	Description	Reference
0	Reserved	This document
1	Return Path Control Code	This document
2	Return Address	This document
3	SR-MPLS Label Stack of the Return Path	This document
4	SRv6 Segment List of the Return Path	This document
5	Structured SRv6 Segment List of the Return Path	This document
255	Reserved	This document

Table 3: Return Path Sub-TLV Types

IANA has created the "STAMP TLV Flags" subregistry. IANA is requested to allocate the following bit position in the "STAMP TLV Flags" subregistry.

Bit Position	Symbol	Description	Reference
TBA3	D	Wrong Destination	This document

Table 4: STAMP TLV Flags

7. References

7.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

- [RFC8762] Mirsky, G., Jun, G., Nydell, H., and R. Foote, "Simple Two-Way Active Measurement Protocol", RFC 8762, DOI 10.17487/RFC8762, March 2020, <<https://www.rfc-editor.org/info/rfc8762>>.
- [RFC8972] Mirsky, G., Min, X., Nydell, H., Foote, R., Masputra, A., and E. Ruffini, "Simple Two-Way Active Measurement Protocol Optional Extensions", RFC 8972, DOI 10.17487/RFC8972, January 2021, <<https://www.rfc-editor.org/info/rfc8972>>.
- [RFC8986] Filsfils, C., Ed., Camarillo, P., Ed., Leddy, J., Voyer, D., Matsushima, S., and Z. Li, "Segment Routing over IPv6 (SRv6) Network Programming", RFC 8986, DOI 10.17487/RFC8986, February 2021, <<https://www.rfc-editor.org/info/rfc8986>>.

7.2. Informative References

- [RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <<https://www.rfc-editor.org/info/rfc8402>>.
- [RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.
- [RFC8545] Morton, A., Ed. and G. Mirsky, Ed., "Well-Known Port Assignments for the One-Way Active Measurement Protocol (OWAMP) and the Two-Way Active Measurement Protocol (TWAMP)", RFC 8545, DOI 10.17487/RFC8545, March 2019, <<https://www.rfc-editor.org/info/rfc8545>>.
- [RFC8799] Carpenter, B. and B. Liu, "Limited Domains and Internet Protocols", RFC 8799, DOI 10.17487/RFC8799, July 2020, <<https://www.rfc-editor.org/info/rfc8799>>.
- [I-D.ietf-spring-segment-routing-policy] Filsfils, C., Talaulikar, K., Voyer, D., Bogdanov, A., and P. Mattes, "Segment Routing Policy Architecture", Work in Progress, Internet-Draft, draft-ietf-spring-segment-routing-policy-14, 25 October 2021, <<https://www.ietf.org/archive/id/draft-ietf-spring-segment-routing-policy-14.txt>>.

[I-D.ietf-pce-binding-label-sid]
Sivabalan, S., Filsfils, C., Tantsura, J., Previdi, S.,
and C. L. (editor), "Carrying Binding Label/Segment
Identifier in PCE-based Networks.", Work in Progress,
Internet-Draft, draft-ietf-pce-binding-label-sid-12, 24
January 2022, <<https://www.ietf.org/archive/id/draft-ietf-pce-binding-label-sid-12.txt>>.

[I-D.ietf-ippm-stamp-yang]
Mirsky, G., Min, X., and W. S. Luo, "Simple Two-way Active
Measurement Protocol (STAMP) Data Model", Work in
Progress, Internet-Draft, draft-ietf-ippm-stamp-yang-09,
12 July 2021, <<https://www.ietf.org/archive/id/draft-ietf-ippm-stamp-yang-09.txt>>.

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

Acknowledgments

The authors would like to thank Thierry Couture for the discussions
on the use-cases for Performance Measurement in Segment Routing. The
authors would also like to thank Greg Mirsky, Mike Koldychev, Gyan
Mishra, Tianran Zhou, Al Mortons, Reshad Rahman, Zhenqiang Li, Frank
Brockners, and Cheng Li for providing comments and suggestions.

Authors' Addresses

Rakesh Gandhi (editor)
Cisco Systems, Inc.
Canada

Email: rgandhi@cisco.com

Clarence Filsfils
Cisco Systems, Inc.

Email: cfilsfil@cisco.com

Daniel Voyer
Bell Canada

Email: daniel.voyer@bell.ca

Mach (Guoyi) Chen
Huawei

Email: mach.chen@huawei.com

Bart Janssens
Colt

Email: Bart.Janssens@colt.net

Richard Foote
Nokia

Email: footer.foote@nokia.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: 13 January 2022

G. Mirsky
X. Min
ZTE Corp.
W.S. Luo
Ericsson
12 July 2021

Simple Two-way Active Measurement Protocol (STAMP) Data Model
draft-ietf-ippm-stamp-yang-09

Abstract

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

Status of This Memo

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

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

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

This Internet-Draft will expire on 13 January 2022.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction	2
1.1. Conventions used in this document	2
1.1.1. Requirements Language	3
2. Scope, Model, and Applicability	3
2.1. Data Model Parameters	3
2.1.1. STAMP-Sender	3
2.1.2. STAMP-Reflector	4
3. Data Model	4
3.1. Tree Diagrams	5
3.2. YANG Module	10
4. IANA Considerations	31
5. Security Considerations	32
6. Acknowledgments	33
7. References	33
7.1. Normative References	33
7.2. Informative References	34
Appendix A. Example of STAMP Session Configuration	35
Authors' Addresses	36

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 TWAMP Light, discussed in Appendix I [RFC5357]. The TWAMP 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 the data model of the base STAMP specification. The defined STAMP data model can be augmented to include STAMP extensions, for example, described in [RFC8972]. 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] and Section 3 [RFC8972].

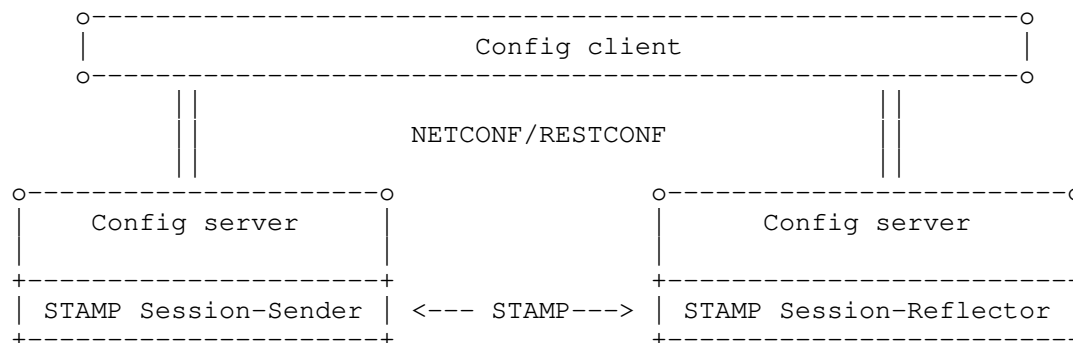


Figure 1: STAMP Reference Model

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 stamp-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 the STAMP data model presents several challenges, and among them is the identification of a test-session at Session-Reflector. A Session-Reflector MAY require only as little as the STAMP Session Identifier (SSID) and the source IP address in received STAMP-Test packet to spawn a 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 Class of Service TLV (Section 4.4 [RFC8972]) that will go undetected, but by using SSID and the source IP address 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 sender-test-session* [stamp-session-id]
|   |   |   +--rw test-session-enable?    boolean
|   |   |   +--rw number-of-packets?      union
|   |   |   +--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 session-sender-udp-port inet:port-number
|   |   |   +--rw stamp-session-id        uint32
|   |   |   +--rw session-reflector-ip    inet:ip-address
|   |   |   +--rw session-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 reflector-test-session* [stamp-session-id]
|   |   |   +--rw stamp-session-id        union
|   |   |   +--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* [stamp-session-id]
        +--ro stamp-session-id          uint32
        +--ro sender-session-state?     enumeration
      +--ro current-stats
        +--ro start-time                yang:date-and-time
        +--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 session-sender-udp-port    inet:port-number
|--ro session-reflector-ip      inet:ip-address
|--ro session-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* [stamp-session-id]
+--ro stamp-session-id            uint32
+--ro end-time                    yang:date-and-time
+--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 session-sender-udp-port    inet:port-number
+--ro session-reflector-ip       inet:ip-address
+--ro session-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* [stamp-session-id]
  +---ro stamp-session-id          uint32
  +---ro reflector-timestamp-format? timestamp-format
  +---ro sender-ip                  inet:ip-address
  +---ro session-sender-udp-port    inet:port-number
  +---ro session-reflector-ip       inet:ip-address
  +---ro session-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 stamp-session-id    uint32
+---x stamp-sender-stop
    +---w input
        +---w stamp-stamp-session-id    uint32

```

Figure 4: STAMP RPC Tree Diagram

3.2. YANG Module

```

<CODE BEGINS> file "ietf-stamp@2021-07-12.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) 2021 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 "2021-07-10" {

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";
reference
  "RFC 8762 Simple Two-way Active
   Measurement Protocol (STAMP) Section 4.";
}

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.";
```



```
reference
  "RFC 8762 Simple Two-way Active
  Measurement Protocol (STAMP) Section 4.2.1.";
}

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";
  reference
    "RFC 8762 Simple Two-way Active
    Measurement Protocol (STAMP) Section 4.2.";
}

feature session-reflector {
  description
    "This feature relates to the device functions as the
    STAMP Session-Reflector";
  reference
    "RFC 8762 Simple Two-way Active
    Measurement Protocol (STAMP) Section 4.3.";
}

feature stamp-security {
  description "Secure STAMP supported";
  reference
```

```
    "RFC 8762 Simple Two-way Active
      Measurement Protocol (STAMP) Section 4.4.";
  }

  /*
   * 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 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/"
    +"sender-test-session[stamp-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/"
    +"sender-test-session[stamp-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/"
    +"sender-test-session[stamp-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/"
    +"sender-test-session[stamp-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/"
      +"sender-test-session[stamp-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 session-sender-udp-port {
```



```
    type inet:port-number {
      range "49152..65535";
    }
    mandatory true;
    description "Sender UDP port number";
    reference
      "RFC 8762 Simple Two-Way Active
      Measurement Protocol Section 4.1.";
  }
  leaf stamp-session-id {
    type uint32;
    description
      "A STAMP test session identifier
      assigned by the Session-Sender.";
    reference
      "RFC 8972 Simple Two-Way Active
      Measurement Protocol Optional
      Extensions Section 3.";
  }
  leaf session-reflector-ip {
    type inet:ip-address;
    mandatory true;
    description "Reflector IP address";
  }
  leaf session-reflector-udp-port {
    type inet:port-number{
      range "862 | 1024..49151 | 49152..65535";
    }
    default 862;
    description
      "Reflector UDP port number";
    reference
      "RFC 8762 Simple Two-Way Active
      Measurement Protocol Section 4.1.";
  }
}

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";
    }
  }
}
```

```
    }
  }
  reference
    "RFC 8762 Simple Two-Way Active
    Measurement Protocol Section 4.4.";
}

/*
 * 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";
      reference
        "RFC 8762 Simple Two-Way Active
        Measurement Protocol Section 4.2.";
    }

    list sender-test-session {
      key "stamp-session-id";
      unique "stamp-session-id";
      description
        "This structure is a container of test session
        managed objects";

      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 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 reflector-test-session {
    key "session-index";
    unique "sender-ip stamp-session-id";
    description
        "This structure is a container of test session
        managed objects";

    leaf session-index {
        type uint32;
        description "Session index";
    }

    leaf stamp-session-id {
        type union {
            type uint32;
            type enumeration {
                enum any {
                    description
                        "Indicates that the Session-Reflector
                        accepts STAMP test packets from
                        a Session-Sender with any SSID
                        value";
                }
            }
        }
        description
            "This value determines whether specific
            SSID of the Session-Sender
            or the wildcard, i.e. any SSID accepted";
        reference
            "RFC 8972 Simple Two-Way Active
            Measurement Protocol Optional
            Extensions Section 3.";
    }

    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";
        reference
            "RFC 8762 Simple Two-Way Active
            Measurement Protocol Section 4.1.";
    }

    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-index";
      description
        "This structure is a container of test session
        managed objects";

      leaf session-index {
        type uint32;
        description "Session index";
      }

      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-index;
  description
    "This container contains the results for the history
    Measurement Interval in a Measurement session ";
  leaf session-index {
    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-index";
    description
      "This structure is a container of test session
      managed objects";

    leaf session-index {
      type uint32;
      description "Session index";
    }

    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 stamp-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 stamp-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>>.
- [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>>.
- [RFC8972] Mirsky, G., Min, X., Nydell, H., Foote, R., Masputra, A., and E. Ruffini, "Simple Two-Way Active Measurement Protocol Optional Extensions", RFC 8972, DOI 10.17487/RFC8972, January 2021, <<https://www.rfc-editor.org/info/rfc8972>>.

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>
      <stamp-session-id>10</stamp-session-id>
      <test-session-enable>enable<test-session-enable>
      <number-of-packets>forever</number-of-packets>
      <interval>10</interval> <!-- 10 microseconds -->
      <measurement-interval/> <!-- use default 60 seconds -->
      <!-- use default 0 repetitions,
            i.e. do not repeat this session -->
      <repeat/>
      <dscp-value/> <!-- use default 0 (CS0) -->
      <!-- use default 'stateless' -->
      <test-session-reflector-mode/>
      <sender-ip></sender-ip>
      <session-sender-udp-port></session-sender-udp-port>
      <session-reflector-ip></session-reflector-ip>
      <session-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/>
      <stamp-session-id/> <!-- use default 'any' -->
      <!-- 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, gregory.mirsky@ztetx.com

Xiao Min
ZTE Corp.

Email: xiao.min2@zte.com.cn

Wei S Luo
Ericsson

Email: wei.s.luo@ericsson.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: 14 August 2022

Y. Li
H. Yang
T. Sun
China Mobile
10 February 2022

One-way Delay Measurement Based on Deterministic Networking
draft-li-ippm-deterministic-owd-measurement-01

Abstract

One-way delay is a key indicator to measure network quality. Some applications are one-way transmission in the network, such as some high-definition video services, and are very sensitive to one-way delay. Excessive delay will affect user experience greatly. To some extent, the network can't even be used, so it is very important to accurately measure the network transmission delay. The current one-way delay measurement method has problems such as high complexity and low measurement accuracy. In order to solve the problem of high-precision one-way delay measurement, a one-way delay measurement method based on deterministic networking is proposed in this document. The method takes advantage of the delay characteristics of the deterministic networking and does not depend on precise time synchronization. The method realizes the one-way delay measurement of any service flow between any network elements. Its technical advantages are: the network does not need to send measurement packets, can test all traffic types, does not change network status, does not change the format of traffic packets, and does not require network elements to support time synchronization protocols.

Status of This Memo

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

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

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

This Internet-Draft will expire on 14 August 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction	2
2. Conventions Used in This Document	3
2.1. Terminology	3
2.2. Requirements Language	4
3. One-way Delay Measurement Method Based on Deterministic Networking	4
4. Procedures of the One-way Delay Measurement Method	7
5. Security Considerations	8
6. IANA Considerations	8
7. Normative References	8
Authors' Addresses	9

1. Introduction

One-way transmission delay is a key indicator to measure network quality. Some applications are based on one-way transmission in the network, such as some high-definition video services, and are very sensitive to one-way delay. Excessive one-way delay will affect user experience dramatically, so it is very important to accurately measure the one-way transmission delay of the network.

There are several kinds of methods to measure one-way delay. The first kind of methods is active measurement. A sender will send measurement protocol messages, such as Two-Way Active Measurement Protocol (TWAMP) [RFC8186] messages, to the network to measure the one-way delay of the sender and receiver. The advantage of active measurement is that it is flexible in application. The disadvantage is that the measurement messages cannot measure the delay of real services, and the measurement of one-way delay requires sender and receiver to support time synchronization protocol, such as NTP [RFC5905] and PTP [IEEE.1588.2008]. The first kind of methods is passive measurement. The passive measurement devices will calculate

network delay by collecting actual business traffic. The advantage of passive measurement is that it can measure the one-way delay of real services. The disadvantage is that two passive measurement devices need to be deployed, and the two devices require time synchronization, which is difficult to implement. The third kind of methods is hybrid measurement. Hybrid measurement is a combination of active and passive measurements, that is, inserting some fields or flags in the service message to realize the delay measurement of the actual service. The disadvantage is that the message format of the actual service is changed, which will affect the forwarding behavior of the service and have observer effect. The network element needs to be able to recognize and forward the modified service message, and time synchronization of the network element is also required.

The above-mentioned one-way delay measurement methods have the following shortcomings. Firstly, if the measurement message is injected into actual network, it will occupy network bandwidth resources and interfere with the actual service flow, so the measured delay is not the delay of the actual service. Secondly, the measurement equipment or network elements need to support time synchronization protocols, which is difficult to implement and costly.

To address the following shortcomings of existing methods, this document presents the following technical solution. A high-precision one-way delay measurement method is proposed, which can be used to measure the one-way delay of actual service packets, without sending measurement messages, without changing the actual network status, without changing service messages, and without the need for network elements to support time synchronization protocols.

2. Conventions Used in This Document

2.1. Terminology

NTP Network Time Protocol

PTP Precision Time Protocol

TWAMP Two-Way Active Measurement Protocol

SLA Service Level Agreement

2.2. Requirements Language

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

3. One-way Delay Measurement Method Based on Deterministic Networking

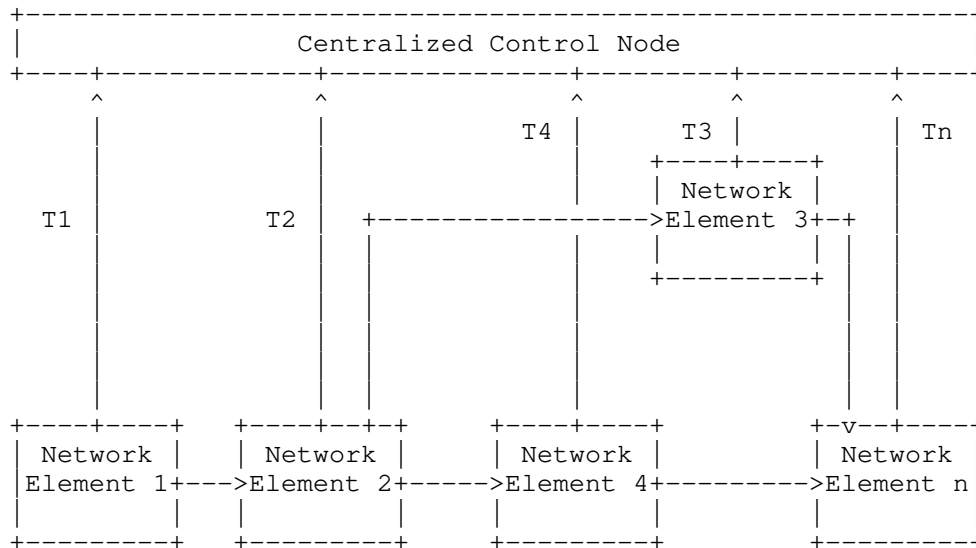


Figure 1: Figure 1: A schematic diagram of the network topology structure

A schematic diagram of the network topology structure to describe the proposed method is shown in Figure 1. The network may be a SDN (Software Defined Network) or a traditional network. Whether it is SDN or traditional network, there is a centralized control node (or called a centralized management unit) for collecting network information sent by network elements and sending control information to the network. Taking SDN as an example, the centralized control node can be a SDN controller. For traditional networks, the centralized control node can be a network management system. The information from the network element to the centralized control node generally passes through the management network. In our solution, the management network from each network element to the centralized control node is required to use a delay deterministic network. As an example, the delay deterministic network may be a time sensitive network (TSN) or a deterministic Internet (Deterministic Internet

Network, DIP) [RFC8655], etc. Through the delay deterministic network, the transmission delay of the network element information from the network element to the centralized control node can be guaranteed to be fixed. $T_1 \sim T_n$ in Figure 1 represent the network element information delay from the network element to the centralized control node of network element 1 to n respectively.

As shown in Figure 1, suppose network traffic of a real service flow passes through network element 1, network element 2, ..., network element n in turn, and the time when network traffic passes through the network element is recorded as t_1, t_2, \dots, t_n . The timestamp maybe the ingress timestamp of network traffic entering the network element or the egress timestamp of network traffic flowing out of the network element after the forwarding is completed. Each network element transmits the flow information to the centralized control node through the delay deterministic network when real traffic passes, and the transmission delays of each network element to transmit the flow information to the centralized control node through the delay deterministic network are denoted as T_1, T_2, \dots, T_n , respectively. The timestamps when the centralized control node receives the flow information of each network element are t_1', t_2', \dots, t_n' .

Taking the calculation of the one-way transmission delay of traffic from network element 1 to network element 2 as an example, the one-way transmission delay can be calculated in the following way. Firstly, because the clocks of network element 1 and network element 2 are not synchronized, suppose the time deviation between the two is Δt . Then the one-way transmission delay of traffic from network element 1 to network element 2 satisfies the following formula (1). Among them, Delay represents the one-way transmission delay of traffic from network element 1 to network element 2.

Formula (1): $\text{Delay} = t_2 - t_1 - \Delta t$

Secondly, because the clocks between network element 1 and the centralized control node are not synchronized, assuming that the time deviation between the two is $\Delta t'$, the time for the traffic information collected from the network element 1 to reach the centralized control node through the delay deterministic network satisfies the following formula (2).

Formula (2): $t_1' = t_1 + T_1 + \Delta t'$

Thirdly, the clocks between network element 2 and the centralized control node are not synchronized, and the time deviation between network element 2 and the centralized control node is $\text{delta_t}' - \text{delta_t}$. The time $t2'$ for the collected traffic to reach the centralized control node satisfies the following formula (3).

Formula (3): $t2' = t2 + T2 + \text{delta_t}' - \text{delta_t}$

Forthly, subtracting the formula (2) from the above formula (3), we can obtain the following formula (4).

Formula (4): $t2 - t1 - \text{delta_t} = t2' - t1' + T1 - T2$

Fifthly, substituting the above formula (4) into the above formula (1), the following formula (5) can be obtained.

Formula (5): $\text{Delay} = t2' - t1' + T1 - T2$

So far, the one-way transmission delay of traffic from network element 1 to network element 2 is obtained. Taking the calculation of one-way transmission delay of traffic from network element 1 to network element 3 as an example, the one-way transmission delay can be calculated in the following way: I) Referring to the above formula (5), the one-way transmission delay of traffic from network element 1 to network element 2 is: $\text{Delay}_{12} = t2' - t1' + T1 - T2$. II) Referring to the above formula (5), the one-way transmission delay of traffic from network element 2 to network element 3 is: $\text{Delay}_{23} = t3' - t2' + T2 - T3$. III) The one-way transmission delay of traffic from network element 1 to network element 3 is: $\text{Delay}_{13} = \text{Delay}_{12} + \text{Delay}_{23} = t2' - t1' + T1 - T2 + t3' - t2' + T2 - T3 = t3' - t1' + T1 - T3$. It can be seen that the one-way transmission delay between any two network elements can be calculated similarly to the above formula (5). For example, taking network element m and network element n as an example, the transmission delay of traffic from network element m to network element n is: $\text{Delay} = tn' - tm' + Tm - Tn$, where tn' and tm' are the time when the traffic information of network element m and network n reaches the centralized control node, and Tm and Tn are transmission delay of the traffic information from network element m and network element n to the centralized control node respectively through delay deterministic network.

4. Procedures of the One-way Delay Measurement Method

In this section, the procedures of the proposed one-way delay measurement method will be elaborated. Assume there are two network element. It is determined that the time when the centralized control node receives the first flow information is the first time, and the time when the second flow information is received by the centralized control node is determined to be the second time. The first flow information is sent to the centralized control node via delay deterministic network, and the second flow information is also sent to the centralized control node via delay deterministic network. The procedures of the one-way delay measurement method is shown in Figure 2.

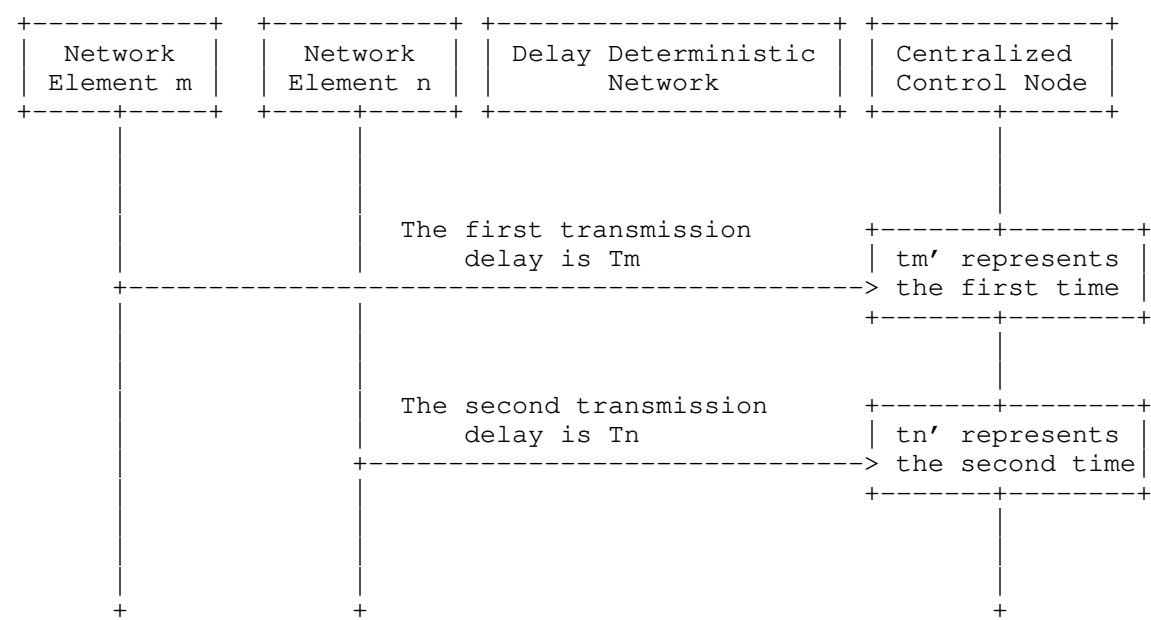


Figure 2: Figure 2: Procedures of the one-way delay measurement method

The transmission delay of traffic from the first network element to the second network element can be determined based on the first time, the second time, the first transmission delay, and the second transmission delay.

The first traffic information is sent by the first network element to the centralized control node via a delay deterministic network at the moment when the traffic passes through the first network element. And the time when the traffic passes through the first network

element refers to the moment when traffic enters the first network element or the time when traffic flows out of the first network element.

The second traffic information is sent by the second network element to the centralized control node via a delay deterministic network at the moment when the traffic passes through the second network element. And the time when the traffic passes through the second network element refers to the moment when traffic enters the second network element or the time when traffic flows out of the second network element.

It is determined that the transmission delay of the first traffic information from the first network element to the centralized control node is the first transmission delay, and it is determined that the transmission delay of the second traffic information from the second network element to the centralized control node is the second transmission delay. The transmission delay of traffic from the first network element to the second network element can be determined based on the following formula: $\text{Delay} = t_n' - t_m' + T_m - T_n$. Wherein, t_n' represents the second time, t_m' represents the first time, T_m represents the first transmission delay, T_n represents the second transmission delay, and Delay represents transmission delay of the traffic from the first network element to the second network element. In the above method, the delay deterministic network is used to ensure that the first transmission delay and the second transmission delay are fixed delays.

5. Security Considerations

TBD.

6. IANA Considerations

TBD.

7. Normative References

[IEEE.1588.2008]

IEEE, "IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems", July 2008.

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

- [RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <<https://www.rfc-editor.org/info/rfc5905>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8186] Mirsky, G. and I. Meilik, "Support of the IEEE 1588 Timestamp Format in a Two-Way Active Measurement Protocol (TWAMP)", RFC 8186, DOI 10.17487/RFC8186, June 2017, <<https://www.rfc-editor.org/info/rfc8186>>.
- [RFC8655] Finn, N., Thubert, P., Varga, B., and J. Farkas, "Deterministic Networking Architecture", RFC 8655, DOI 10.17487/RFC8655, October 2019, <<https://www.rfc-editor.org/info/rfc8655>>.

Authors' Addresses

Yang Li
China Mobile
Beijing
100053
China

Email: liyangzn@chinamobile.com

Hongwei Yang
China Mobile
Beijing
100053
China

Email: yanghongwei@chinamobile.com

Tao Sun
China Mobile
Beijing
100053
China

Email: suntao@chinamobile.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: 8 September 2022

Z. Li
China Mobile
T. Zhou
Huawei
J. Guo
ZTE Corp.
G. Mirsky
Ericsson
R. Gandhi
Cisco
7 March 2022

One-way/Two-way Active Measurement Protocol Extensions for Performance
Measurement on LAG
draft-li-ippm-otwamp-on-lag-03

Abstract

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

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

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

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

This Internet-Draft will expire on 8 September 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction	2
2. Micro Session on LAG	3
3. Mirco OWAMP Session	4
3.1. Micro OWAMP-Control	4
3.2. Micro OWAMP-Test	4
4. Mirco TWAMP Session	5
4.1. Micro TWAMP-Control	5
4.2. Micro TWAMP-Test	5
4.2.1. Sender Packet Format and Content	5
4.2.2. Sender Behavior	7
4.2.3. Reflector Packet Format and Content	8
4.2.4. Reflector Behavior	11
5. IANA Considerations	11
5.1. Mico OWAMP-Control Command	11
5.2. Mico TWAMP-Control Command	11
6. Security Considerations	12
7. Acknowledgements	12
8. References	12
8.1. Normative References	12
8.2. Informative References	12
Authors' Addresses	13

1. Introduction

Link Aggregation Group (LAG), as defined in [IEEE802.1AX], provides mechanisms to combine multiple physical links into a single logical link. This logical link offers 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.

Usually, when forwarding traffic over LAG, the hash-based mechanism is used to load balance the traffic across the LAG member links. Link delay of each member link varies because of different transport paths. To provide low latency service for time sensitive traffic, we need to explicitly steer the traffic across the LAG member links based on the link delay, loss and so on. That requires a solution to measure the performance metrics of every member link of a LAG. Hence the measured performance metrics can work together with layer 2 bundle member link attributes advertisement [RFC8668] for traffic steering.

OWAMP [RFC4656] and TWAMP [RFC5357] are two active measurement methods according to the classification given in [RFC7799], which can complement passive and hybrid methods. With both methods, running a single test session over the aggregation without the knowledge of each member link would make it impossible to measure the performance of a given physical member link. The measured metrics can only reflect the performance of one member link or an average of some/all member links of the LAG.

This document extends OWAMP and TWAMP to implement performance measurement on every member link of a LAG. The proposed method could also potentially apply to layer 3 ECMP (Equal Cost Multi-Path), e.g., with SR-Policy [I-D.ietf-spring-segment-routing-policy].

2. Micro Session on LAG

This document intends to address the scenario (e.g., Figure 1) where a LAG (e.g., the LAG includes four member links) directly connects two nodes (A and B). The goal is to measure the performance of each link of the LAG.

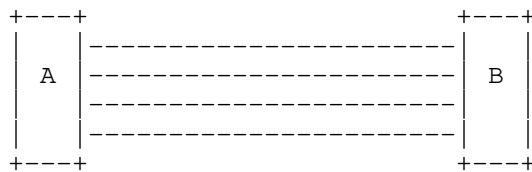


Figure 1: PM for LAG

To measure the performance metrics of every member link of a LAG, multiple sessions (one session for each member link) need to be established between the two end points 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 IP Address and Receiver IP Address. As for the UDP layer, the micro sessions may share the same Sender Port and Receiver Port pair, or each micro session is configured with a different Sender Port and Receiver Port pair. But from the operational point of view, the former is simpler and is RECOMMENDED.

The micro sessions need to associate with the corresponding member links. For example, when the Server/Reflector/Receiver receives a Test packet, it needs to know from which member link the packet is received, and correlate it with a micro session.

This document defines new command types to indicate the set of micro sessions of a LAG. The details are described in Sections 3 and 4 of this document. Upon receiving a Test packet, the receiver uses the receiving link's identifier to correlate the packet to a particular micro session. In addition, Test packets MAY carry the member link information for validation check. For example, when a micro Session-Sender receives a reflected 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 end point.

3.1. Micro OWAMP-Control

To support the micro OWAMP session, a new command, Request-OW-Micro-Sessions (TBD1), is defined in this document. The Request-OW-Micro-Sessions 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 an OWAMP Server receives a Request-OW-Micro-Sessions command, if the request is accepted, the OWAMP Server MUST build a set of micro sessions for all the member links of the LAG from which the Request-OW-Micro-Sessions 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 end point.

4.1. Micro TWAMP-Control

To support the micro TWAMP session, a new command, Request-TW-Micro-Sessions (TBD2), is defined in this document. The Request-TW-Micro-Sessions 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 TWAMP Server receives a Request-TW-Micro-Sessions command, if the request is accepted, the TWAMP Server MUST build a set of micro sessions for all the member links of the LAG from which the Request-TW-Micro-Sessions 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 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]. Two new fields (Sender Micro-session ID and Reflector Micro-session ID) are added to carry the LAG member link identifiers.

For unauthenticated mode, the format is as below:

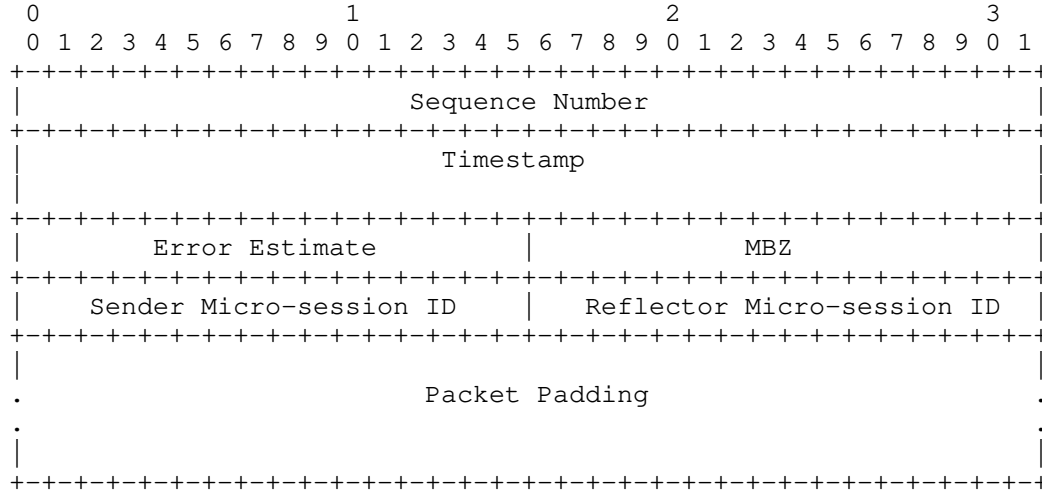


Figure 2: Micro Session-Sender Packet format in Unauthenticated Mode

For authenticated mode, the format is as below:

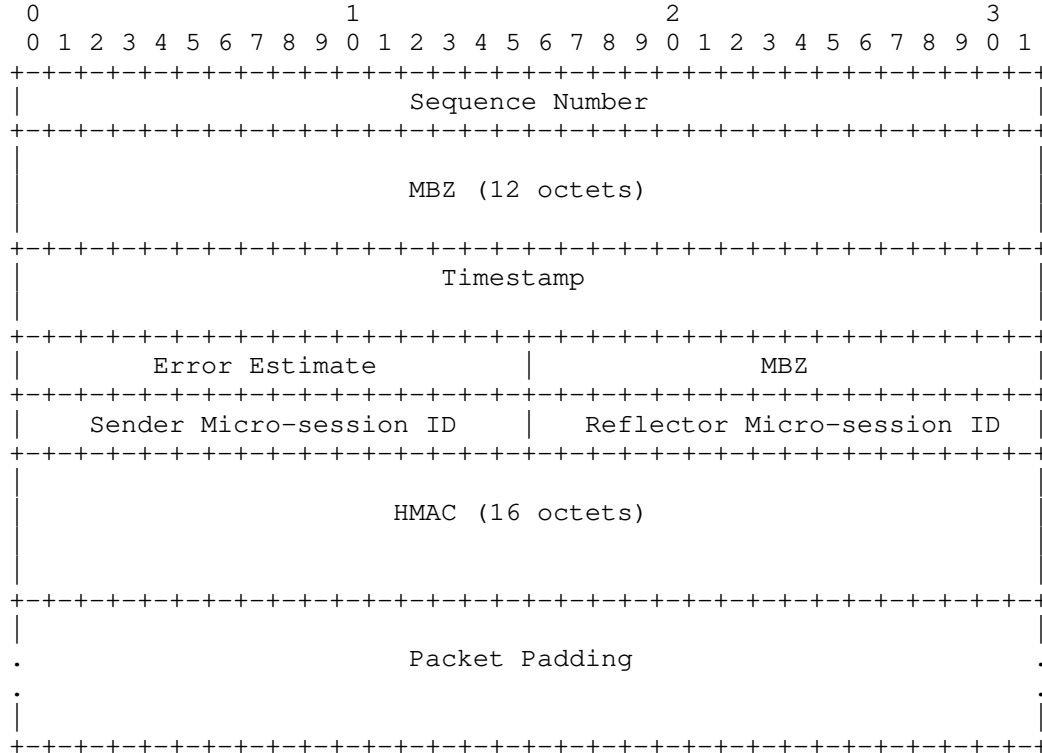


Figure 3: Micro Session-Sender Packet Format in Authenticated Mode

Except for the Sender/Reflector Micro-session ID field, all the other fields are the same as defined in Section 4.1.2 of TWAMP [RFC5357], which is 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 Micro-session ID (2-octets in length): it is defined to carry the Micro-session identifier of the Sender side. The value of the Sender Micro-session ID MUST be unique at the Session-Sender.
- * Reflector Micro-session ID (2-octets in length): it is defined to carry the Micro-session identifier of the Reflector side. The value of the Reflector Micro-session ID MUST be unique at the Session-Reflector.

4.2.2. Sender Behavior

The micro TWAMP Session-Sender inherits the behaviors of the TWAMP Session-Reflector as defined in Section 4.1 of [RFC5357]. In addition, the micro TWAMP Session-Sender MUST send the micro TWAMP-Test packets over the member link with which the session is associated.

When sending the 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 Micro-session ID. If the Session-Sender knows the Reflector member link identifier, the Reflector Micro-session ID field (see Figure 2 and Figure 3) MUST be set. Otherwise, the Reflector Micro-session ID field MUST be zero.

A Test packet with Sender member link identifier is sent to the Session-Reflector, and then is reflected with the same Sender member link identifier. So the Session-Sender can use the Sender member link identifier to check whether a reflected Test packet is received from the member link associated with the correct micro TWAMP session.

The Reflector member link identifier carried in the Reflector Micro-session ID field is used by the Session-Receiver to check whether a Test packet is received from the member link associated with the correct micro TWAMP session. It means that the Session-Sender has to learn the Reflector member link identifier. Once the Session-Sender knows the Reflector member link identifier, it MUST put the identifier in the Reflector Micro-session 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 from the data plane (e.g., the reflected Test packet). How to obtain/learn the Reflector member link identifier is out of the scope of this document.

When receiving 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 micro Session-Sender MUST use the Sender Micro-session ID to validate whether the reflected Test packet is correctly transmitted over the expected member link. If the validation fails, the Test packet MUST be discarded. The micro Session-Sender MUST use the Reflector Micro-session ID to validate the Reflector's behavior. If the validation fails, the Test packet MUST be discarded.

4.2.3. Reflector 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]. Two new fields (Sender and Reflector Micro-session ID) are added to carry the LAG member link identifiers.

For unauthenticated mode, the format is as below:

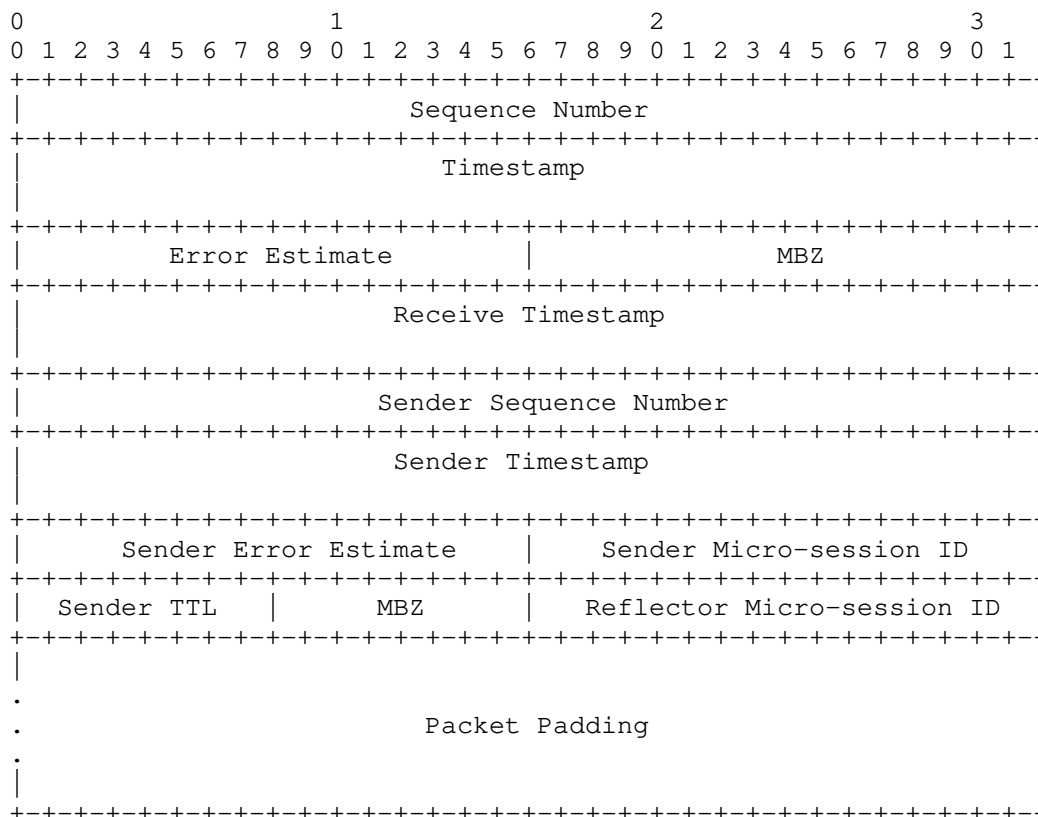
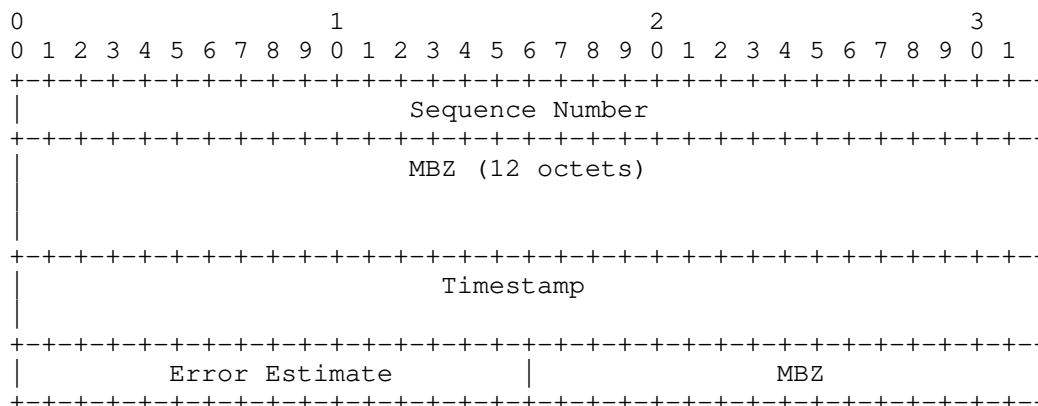


Figure 4: Micro Session-Reflector Packet Format in Unauthenticated Mode

For authenticated mode, the format is as below:



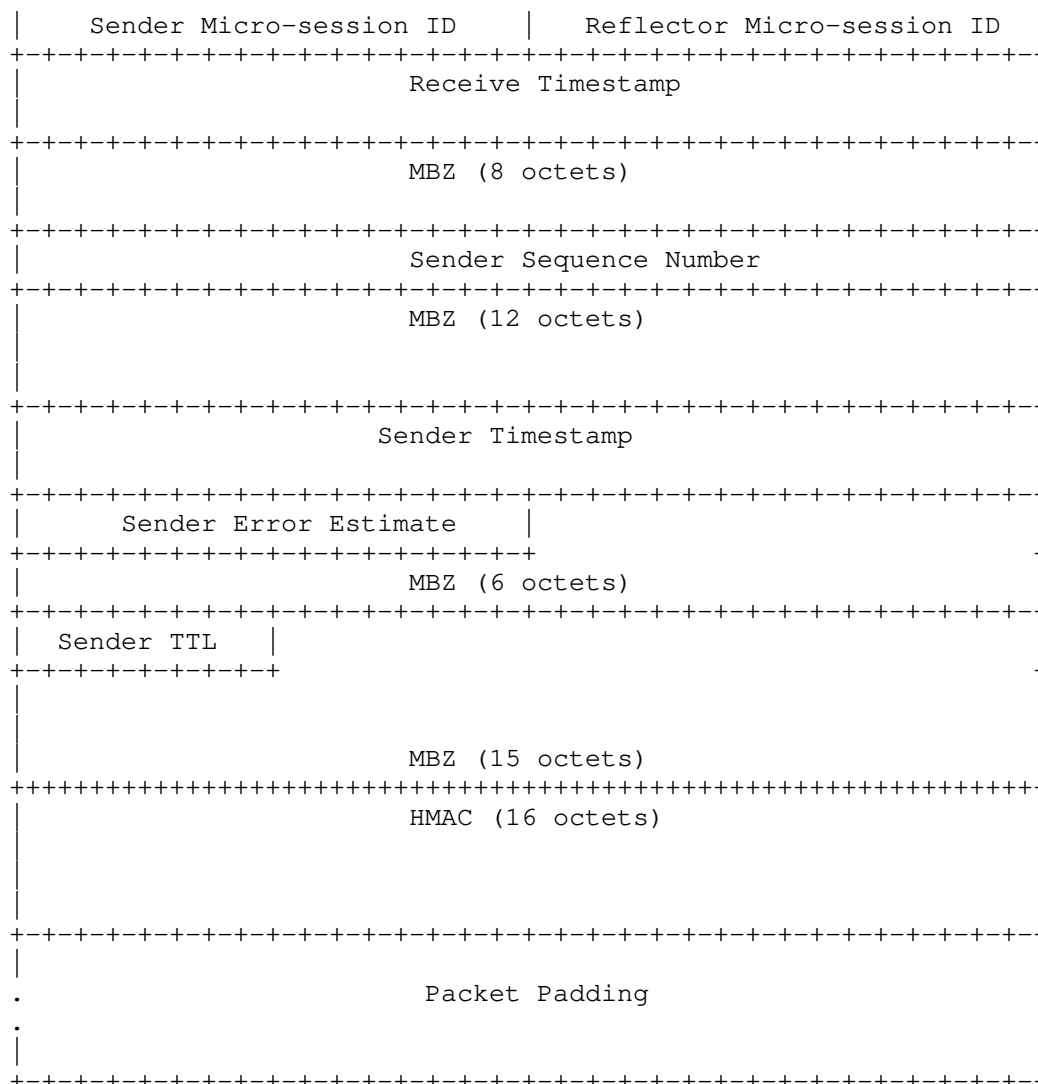


Figure 5: Micro Session-Reflector Packet Format in Authenticated Mode

Except for the Sender/Reflector Micro-session 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 Micro-session ID (2-octets in length): it is defined to carry the Micro-session identifier of the Sender side. The value of the Sender Micro-session ID MUST be unique at the Session-Sender.
- * Reflector Micro-session ID (2-octets in length): it is defined to carry the Micro-session identifier of the Reflector side. The value of the Reflector Micro-session ID MUST be unique at the Session-Reflector.

4.2.4. 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 receiving 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 the Reflector Micro-session ID is not zero, the Reflector MUST use the Reflector Micro-session ID to validate whether it associates with the receiving member link. If the validation fails, the Test packet MUST be discarded.

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

5. IANA Considerations

5.1. 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-Sessions	This document, Section 3.1

5.2. 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
TBD2	Request-TW-Micro-Sessions	This document, Section 4.1

6. Security Considerations

This document does not introduce additional security requirements and mechanisms other than those described in [RFC4656], and [RFC5357].

7. Acknowledgements

The authors would like to thank Fang Xin, Henrik Nydell, Mach Chen, Min Xiao, Jeff Tantsura for the valuable comments to this work.

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>>.
- [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>>.
- [RFC7799] Morton, A., "Active and Passive Metrics and Methods (with Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799, May 2016, <<https://www.rfc-editor.org/info/rfc7799>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8668] Ginsberg, L., Ed., Bashandy, A., Filsfils, C., Nanduri, M., and E. Aries, "Advertising Layer 2 Bundle Member Link Attributes in IS-IS", RFC 8668, DOI 10.17487/RFC8668, December 2019, <<https://www.rfc-editor.org/info/rfc8668>>.

8.2. Informative References

[I-D.ietf-spring-segment-routing-policy]

Filsfils, C., Talaulikar, K., Voyer, D., Bogdanov, A., and
P. Mattes, "Segment Routing Policy Architecture", Work in
Progress, Internet-Draft, draft-ietf-spring-segment-
routing-policy-18, 17 February 2022,
<[https://www.ietf.org/archive/id/draft-ietf-spring-
segment-routing-policy-18.txt](https://www.ietf.org/archive/id/draft-ietf-spring-segment-routing-policy-18.txt)>.

[IEEE802.1AX]

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

Authors' Addresses

Zhenqiang Li
China Mobile
China
Email: li_zhenqiang@hotmail.com

Tianran Zhou
Huawei
China
Email: zhoutianran@huawei.com

Jun Guo
ZTE Corp.
China
Email: guo.jun2@zte.com.cn

Greg Mirsky
Ericsson
United States of America
Email: gregimirsky@gmail.com

Rakesh Gandhi
Cisco
Canada
Email: rgandhi@cisco.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: 14 August 2022

Y. Li
T. Sun
H. Yang
D. Chen
China Mobile
Y. Wang
Huawei
10 February 2022

One-way Delay Measurement Based on Reference Delay
draft-li-ippm-ref-delay-measurement-02

Abstract

The end-to-end network one-way delay is an important performance metric in the 5G network. For realizing the accurate one-way delay measurement, existing methods requires the end-to-end deployment of accurate clock synchronization mechanism, such as PTP or GPS, which results in relatively high deployment cost. Another method can derive the one-way delay from the round-trip delay. In this case, since the delay of the downlink and uplink may be asymmetric, the measurement accuracy is relatively low. Hence, this document introduces a method to measure the end-to-end network one-way delay based on a reference delay guaranteed by deterministic networking without clock synchronization. The advantage of this solution is that it has high measurement accuracy and can test any flow type.

Status of This Memo

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

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

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

This Internet-Draft will expire on 14 August 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction	2
2. Conventions Used in This Document	4
2.1. Terminology	4
2.2. Requirements Language	4
3. The method of One-way Delay Measurement Based on Reference Delay	4
3.1. One-way Delay Measurement Method	5
3.2. Packet and Measurement Header Format	7
4. Acquisition of Reference Delay	8
5. Security Considerations	8
6. IANA Considerations	8
7. Normative References	8
Authors' Addresses	9

1. Introduction

With the gradual promotion of new-generation network technologies (such as 5G networks) and their application in various industries, SLA guarantees for network quality become more and more important. For example, different 5G services have different requirements for network performance indicators such as delay, jitter, packet loss, and bandwidth. Among them, the 5G network delay is defined as end-to-end one-way delay of the network. Real-time and accurate measurement of the end-to-end one-way delay is very important for the SLA guarantee of network services, and has become an urgent and important requirement.

As shown in figure 1, 5G network HD video surveillance service is a common scenario having requirement of end-to-end one-way delay measurement. In this case, one end of the network is a high-definition surveillance camera in the wireless access side, and the other end of the network is a video server. The end-to-end one-way

delay from the surveillance camera to the video server is the sum of T1, T2, T3 and T4, which is composed of delay in wireless access network, optical transmission network, 5G core network, and IP data network.

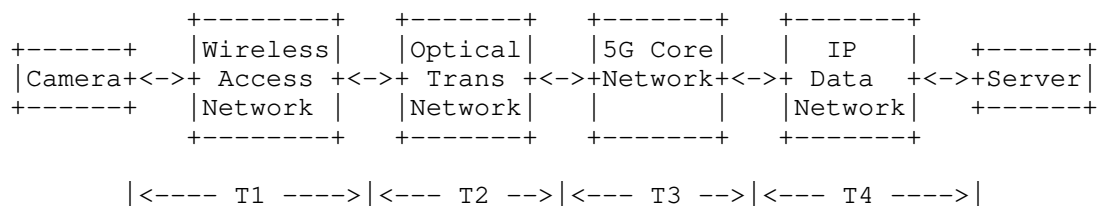


Figure 1: Figure 1:A Scenario for End-to-end One-way Delay

The existing one-way delay measurement solutions are divided into two types. One type of mechanism to calculate one-way delay is based on the measurement of round-trip delay. However, for example, because upstream traffic and downstream traffic do not share the same path in 5G network, the accuracy of the end-to-end one-way delay calculated from the round-trip delay is low. Another type of mechanism is in-band OAM with accurate network time synchronization mechanism, such as NTP[RFC5905] or PTP[IEEE.1588.2008].

The one-way delay measurement solution based on precise network time synchronization requires the deployment of an end-to-end time synchronization mechanism. The current time synchronization accuracy based on the NTP protocol can only reach millisecond level, which cannot fully meet the measurement accuracy requirements. The time synchronization accuracy based on the GPS module or the PTP protocol can meet the requirements. However, because many data centers are actually located underground or in rooms without GPS signals, so GPS clock information cannot be continuously obtained for time synchronization. For time synchronization solutions based on the PTP protocol, each device in the wireless access network, 5G transport network, and 5G core network must support the PTP protocol, which is unrealistic at the moment. So the one-way delay measurement solution based on precise end-to-end time synchronization is expensive and difficult to be deployed.

This document introduces a one-way delay measurement mechanism for Deterministic Networking (DetNet) [RFC8655]. The one-way delay measurement is based on a stable one-way delay of a reference DetNet packet, named as reference delay, which is known in advance and has extremely low jitter. We can use the reference delay provided by the reference DetNet packet to derive the one-way delay of other common service packets.

2. Conventions Used in This Document

2.1. Terminology

NTP Network Time Protocol

PTP Precision Time Protocol

SLA Service Level Agreement

2.2. Requirements Language

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

3. The method of One-way Delay Measurement Based on Reference Delay

The end-to-end one-way delay of a reference packet with a stable delay in the network can be used as a reference delay, denoted as Dref, which is known in advance and has extremely low jitter. This section will describe in detail the end-to-end one-way delay measurement method based on reference delay of the reference packet. Assume that the end-to-end one-way delay from the sender to the receiver is measured, as shown in figure 2. The intermediate network devices other than the sender and receiver are hidden in the figure.

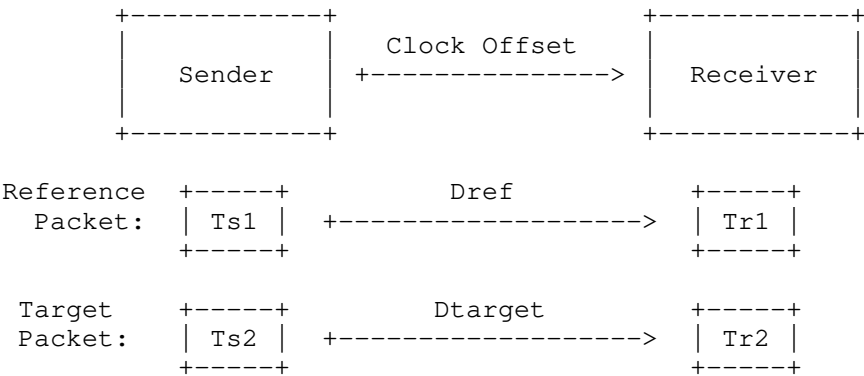


Figure 2: Figure 2:Topology of One-way Delay Measurement

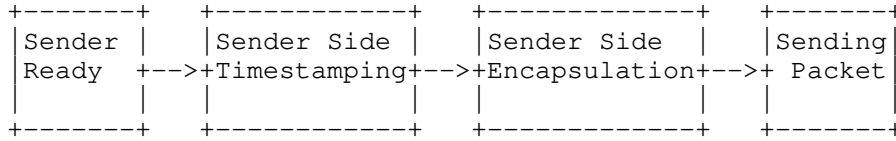
3.1. One-way Delay Measurement Method

The measurement steps are shown in figure 3, which describe the measurement steps at the sender side and receiver side respectively. For the sender side, a reference packet is sent. In the first step, the sender gets ready to send a reference packet; in the second step, the sender marks an egress timestamp $Ts1$ for the reference packet; in the third step, the sender encapsulates the egress timestamp of the reference packet in the measurement header of the reference packet; in the fourth step, the sender sends the reference packet. For the target packet, the sender side procedures are the same, we omit it for simplicity. The sending time of the target packet is according to the traffic model of real applications. On the other hand, the sender can send the reference packet according to a fixed frequency or adjust the sending frequency according to the link usage rate, so that the target packet can always find a nearby reference packet to make sure that the sending time interval between the reference packet and the target packet is small.

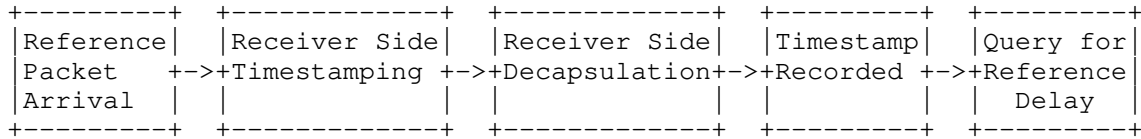
For the reference packet, the processing steps at the receiver are shown in figure 3. In the first step, the reference packet arrives at the receiver, and the receiver receives the reference packet; in the second step, the receiver timestamps the reference packet at the entrance, which is denoted as $Tr1$; in the third step, the receiver decapsulates the measurement header of the reference packet to obtain the sender side timestamp $Ts1$; in the fourth step, the receiver records the timestamp information of $Ts1$ and $Tr1$; in the fifth step, the receiver uses the source/destination pair obtained by decapsulation in the third step as the search key, queries the reference delay table and records the reference delay search result, denoted as $Dref$.

For the target packet, the processing steps at the receiver are also shown in figure 3. In the first step, the target packet arrives at the receiver, and the receiver receives the target packet; in the second step, the receiver timestamps the target packet at the entrance, which is denoted as $Tr2$; in the third step, the receiver decapsulates the measurement header of the target packet to obtain the sender side timestamp $Ts2$; in the fourth step, the receiver records the timestamp information of $Ts2$ and $Tr2$; in the fifth step, the receiver calculates the target one-way delay, which we want to measure, according to the recorded timestamp information $Ts1$, $Ts2$, $Tr1$, $Tr2$ and reference delay information $Dref$. The target one-way delay of the target packet is recorded as $Dtarget$.

Sender Side Procedures for both Reference and Target Packet:



Receiver Side Procedures for Reference Packet:



Receiver Side Procedures for Target Packet:

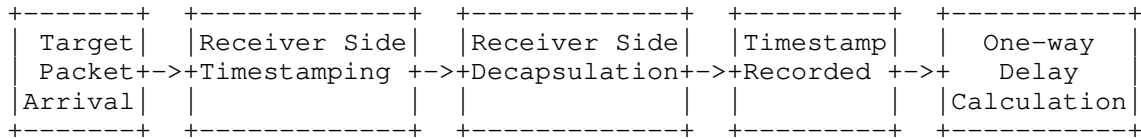


Figure 3: Figure 3: Measurement steps for Sender and Receiver
Respectively

Now we describe the fifth step of the receiver procedures for the target packet in figure 3, that is, calculating the one-way delay D_{target} of the target packet based on the recorded timestamp information $Ts1$, $Ts2$, $Tr1$, $Tr2$ and the reference delay information D_{ref} . The calculation method is the core of this solution. For the reference packet, leveraging the receiver timestamp minus the sender timestamp, we can get Equation 1.

$$\text{Equation 1: } Tr1 - Ts1 = D_{ref} + \text{Offset1}$$

where Offset1 is the time offset between the sender and the receiver when the reference packet transmission occurs. Similarly, for the target packet, we can get Equation 2 using the same method.

$$\text{Equation 2: } Tr2 - Ts2 = D_{target} + \text{Offset2}$$

where Offset2 is the time offset between the sender and the receiver when the target packet transmission occurs. Assuming that the sending time interval between the reference packet and the target packet is very small, we can get that $\text{Offset1} = \text{Offset2}$. By (Equation 2 - Equation 1), we can get Equation 3.

Equation 3: $D_{target} = (Tr2 + Ts1) - (Tr1 + Ts2) + D_{ref}$

So the one-way delay of the target packet can be calculated by Equation 3.

3.2. Packet and Measurement Header Format

The sender encapsulates the timestamp information and sender-receiver pair information in the measurement header of the sent packet, as shown in figure 4. The position of measurement header is in the option field of the TCP protocol header. The delay measurement option format is defined in figure 5. The Length value is 8 octets, which is in accordance with TCP option. The sender ID is one octet, and the receiver ID is also one octet. The sender side timestamp is 4 octets, which can store accurate timestamp information.

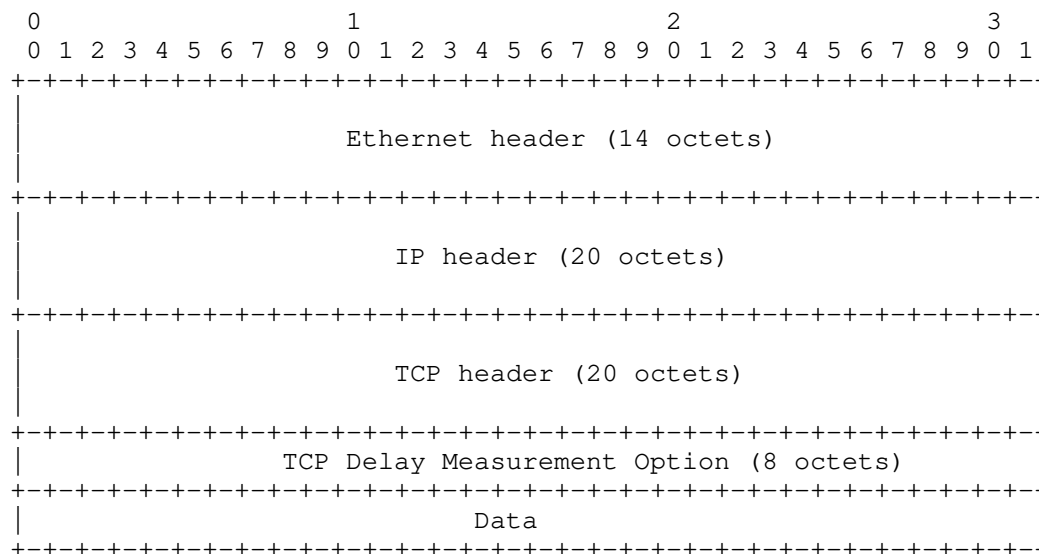


Figure 4: Figure 4: Format of Reference or Target Packet

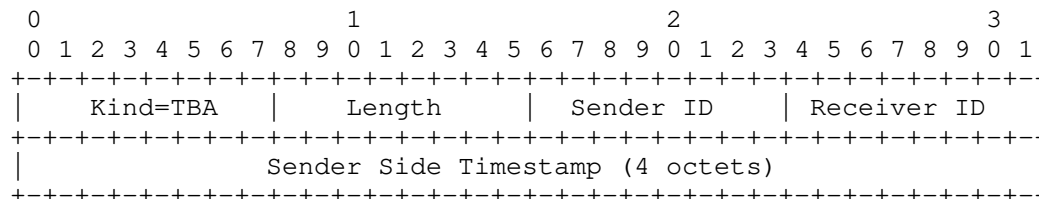


Figure 5: Figure 5: TCP Delay Measurement Option Format

4. Acquisition of Reference Delay

The end-to-end one-way delay includes three parts, namely the transmission delay, the internal processing delay of the network devices, and the internal queueing delay of the network devices. Among them, fixed parts of the delay include transmission delay and internal processing delay. The transmission delay is related to transmission distance and transmission media. For example, in optical fiber, it is about 5ns per meter. With transmission path and media determined, it is basically a fixed value. The internal processing delay of a network device includes processing delay of the device's internal pipeline or processor and serial-to-parallel conversion delay of the interface, which is related to in/out port rate of the device, message length and forwarding behavior. The magnitude of the internal processing delay is at microsecond level, and it is basically a fixed value related to the chip design specifications of a particular network device. Variable part of the delay is the internal queueing delay. The queueing delay of the device internal buffer is related to the queue depth, queue scheduling algorithm, message priority and message length. For each device along the end-to-end path, the queueing delay can reach microsecond or even millisecond level, depending on values of the above parameters and network congestion state.

With the continuous development of networking technologies and application requirements, a series of new network technologies have emerged which can guarantee bounded end-to-end delay and ultra small jitter. For example, deterministic network[RFC8655], by leveraging novel scheduling algorithms and packet priority settings, can stabilize queuing delay of network device on the end-to-end path. As a result, the end-to-end one-way delay is extremely low and bounded. So packets transmitted by a deterministic network with delay guarantee can be used as reference packets, and their end-to-end one-way delay can be used as reference delays. The acquisition method of reference delay is not limited to the above method based on deterministic network technology.

5. Security Considerations

TBD.

6. IANA Considerations

This document requests IANA to assign a Kind Number in TCP Option to indicate TCP Delay Measurement option.

7. Normative References

- [IEEE.1588.2008]
IEEE, "IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems", July 2008.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <<https://www.rfc-editor.org/info/rfc5905>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8655] Finn, N., Thubert, P., Varga, B., and J. Farkas, "Deterministic Networking Architecture", RFC 8655, DOI 10.17487/RFC8655, October 2019, <<https://www.rfc-editor.org/info/rfc8655>>.

Authors' Addresses

Yang Li
China Mobile
Beijing
100053
China

Email: liyangzn@chinamobile.com

Tao Sun
China Mobile
Beijing
100053
China

Email: suntao@chinamobile.com

Hongwei Yang
China Mobile
Beijing
100053
China

Email: yanghongwei@chinamobile.com

Danyang Chen
China Mobile
Beijing
100053
China

Email: chendanyang@chinamobile.com

Yali Wang
Huawei
156 Beiqing Rd., Haidian District
Beijing
China

Email: wangyalil1@huawei.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: 8 September 2022

Z. Li
China Mobile
T. Zhou
Huawei
J. Guo
ZTE Corp.
G. Mirsky
Ericsson
R. Gandhi
Cisco
7 March 2022

Simple Two-Way Active Measurement Protocol Extensions for Performance
Measurement on LAG
draft-li-ippm-stamp-on-lag-02

Abstract

This document extends Simple Two-Way Active Measurement Protocol (STAMP) to implement performance measurement on every member link of a Link Aggregation Group (LAG). Knowing the measured metrics of each member link of a LAG enables operators to enforce a performance based traffic steering policy across 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

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

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

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

This Internet-Draft will expire on 8 September 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction	2
2. Micro Session on LAG	3
3. Member Link Validation	4
3.1. Micro-session ID TLV	4
3.2. Micro STAMP-Test Procedures	5
4. IANA Considerations	6
5. Security Considerations	6
6. Acknowledgements	7
7. References	7
7.1. Normative References	7
7.2. Informative References	7
Authors' Addresses	8

1. Introduction

Link Aggregation Group (LAG), as defined in [IEEE802.1AX], provides mechanisms to combine multiple physical links into a single logical link. This logical link offers 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.

Usually, when forwarding traffic over LAG, the hash-based mechanism is used to load balance the traffic across the LAG member links. Link delay of each member link varies because of different transport paths. To provide low latency service for time sensitive traffic, we need to explicitly steer the traffic across the LAG member links based on the link delay, loss and so on. That requires a solution to measure the performance metrics of each member link of a LAG. Hence the measured performance metrics can work together with layer 2 bundle member link attributes advertisement [RFC8668] for traffic steering.

Simple Two-Way Active Measurement Protocol (STAMP) [RFC8762] is an active measurement method according to the classification given in [RFC7799], which can complement passive and hybrid methods. It provides a mechanism to measure both one-way and round-trip performance metrics, like delay, delay variation, and packet loss. Running a single STAMP test session over the aggregation without the knowledge of each member link would make it impossible to measure the performance of a given physical member link. The measured metrics can only reflect the performance of one member link or an average of some/all member links of the LAG.

This document extends STAMP to implement performance measurement on every member link of a LAG. The proposed method could also potentially apply to layer 3 ECMP (Equal Cost Multi-Path), e.g., with SR-Policy [I-D.ietf-spring-segment-routing-policy].

2. Micro Session on LAG

This document intends to address the scenario (e.g., Figure 1) where a LAG (e.g., the LAG includes four member links) directly connects two nodes (A and B). The goal is to measure the performance of each link of the LAG.

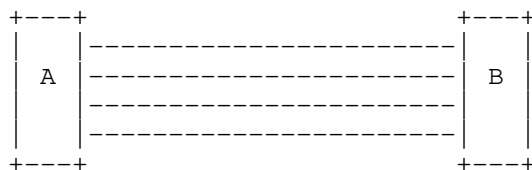


Figure 1: PM for LAG

To measure the performance metrics of every member link of a LAG, multiple sessions (one session for each member link) need to be established between the two end points 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 IP Address and Receiver IP Address. As for the UDP Port, the micro sessions may share the same Sender Port and Receiver Port pair, or each micro session is configured with a different Sender Port and Receiver Port pair. But from the operational point of view, the former is simpler and is recommended.

At the Sender side, each micro STAMP session MUST be assigned with a unique SSID [RFC8972]. Both the micro STAMP Session Sender and Reflector MUST use SSID to correlate the Test packet to a micro session. If there is no such a session, or the SSID is not correct, the Test packet MUST be discarded.

Test packets MAY carry the member link information for validation check. For example, when a micro STAMP Session-Sender receives a reflected Test packet, it may need to check whether the Test packet is from the expected member link. The detailed description about the member link validation is in section 3.

A micro STAMP Session-Sender MAY include the Follow-Up Telemetry TLV [RFC8972] to request information from the micro Session-Reflector. This timestamp might be important for the micro Session-Sender, as it improves the accuracy of network delay measurement by minimizing the impact of egress queuing delays on the measurement.

3. Member Link Validation

Test packets MAY carry the member link information for validation check. The micro Session Sender can verify whether the test packet is received from the expected member link. It can also verify whether the packet is sent from the expected member link at the Reflector side. The micro Session Reflector can verify whether the test packet is received from the expected member link.

3.1. Micro-session ID TLV

STAMP TLV [RFC8972] mechanism extends STAMP Test packets with one or more optional TLVs. This document defines the TLV Type (value TBA1) for the Micro-session ID TLV that carries the micro STAMP Session-Sender member link identifier and Session-Reflector member link identifier. The format of the Micro-session ID TLV is shown as follows:

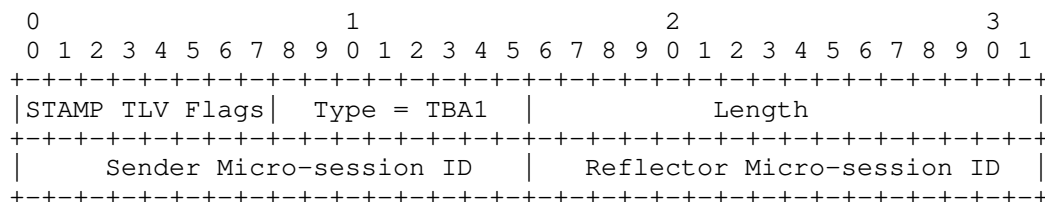


Figure 2: Micro-session ID TLV

- * Type: A one-octet field. Value TBA1 is allocated by IANA (Section 5).
- * Length: A two-octet field equal to the length of the Value field in octets. The Length field value MUST be 4 octets.
- * Sender Micro-session ID (2-octets in length): it is defined to carry the Micro-session identifier of the Sender side. The value of the Sender Member Link ID MUST be unique at the Session-Sender.
- * Reflector Micro-session ID (2-octets in length): it is defined to carry the Micro-session identifier of the Reflector side. The value of the Reflector Member ID MUST be unique at the Session-Reflector.

3.2. 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 Micro-session ID field with the member link identifier associated with the micro STAMP session. If the Session-Sender knows the Reflector member link identifier, the Reflector Micro-session ID field MUST be set. Otherwise, the Reflector Micro-session ID field MUST be zero. The Reflector member link identifier can be obtained from pre-configuration or learned from data plane (e.g., the reflected Test packet). How to obtain/learn the Reflector member link identifier is outside of this document's scope.

When the micro STAMP Session-Reflector receives a Test packet, if the Reflector Micro-session ID is not zero, the micro STAMP Session-Reflector MUST use the Reflector member link identifier to check whether it is associated with the micro STAMP session. If the validation fails, the Test packet MUST be discarded. If all validations 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 Micro-session ID and Reflector Micro-session ID fields respectively. The Sender member link identifier is copied from the received Test packet.

When receiving a reflected Test packet, the micro Session-Sender MUST use the Sender Micro-session ID to validate whether the reflected Test packet is correctly transmitted over the expected member link. If the validation fails, the Test packet MUST be discarded. The micro Session-Sender MUST use the Reflector Micro-session ID to validate the Reflector's behavior. If the validation fails, the Test packet MUST be discarded.

4. IANA Considerations

In the "STAMP TLV Types" registry created for [RFC8972], a new STAMP TLV Type for Micro-session ID TLV is requested from IANA as follows:

STAMP TLV Type Value	Description	Semantics Definition	Reference
TBA1	Micro-session ID TLV	Section 3	This Document

Figure 3: New STAMP TLV Type

5. Security Considerations

The STAMP extension defined in this document is intended for deployment in LAG scenario where Session-Sender and Session-Reflector are directly connected. As such, it's assumed that a node involved in STAMP protocol operation has previously verified the integrity of the LAG connection and the identity of its one-hop-away peer node.

This document does not introduce any additional security issues and the security mechanisms defined in [RFC8762] and [RFC8972] apply in this document.

6. Acknowledgements

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

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>>.
- [RFC7799] Morton, A., "Active and Passive Metrics and Methods (with Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799, May 2016, <<https://www.rfc-editor.org/info/rfc7799>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8668] Ginsberg, L., Ed., Bashandy, A., Filsfils, C., Nanduri, M., and E. Aries, "Advertising Layer 2 Bundle Member Link Attributes in IS-IS", RFC 8668, DOI 10.17487/RFC8668, December 2019, <<https://www.rfc-editor.org/info/rfc8668>>.
- [RFC8762] Mirsky, G., Jun, G., Nydell, H., and R. Foote, "Simple Two-Way Active Measurement Protocol", RFC 8762, DOI 10.17487/RFC8762, March 2020, <<https://www.rfc-editor.org/info/rfc8762>>.
- [RFC8972] Mirsky, G., Min, X., Nydell, H., Foote, R., Masputra, A., and E. Ruffini, "Simple Two-Way Active Measurement Protocol Optional Extensions", RFC 8972, DOI 10.17487/RFC8972, January 2021, <<https://www.rfc-editor.org/info/rfc8972>>.

7.2. Informative References

- [I-D.ietf-spring-segment-routing-policy] Filsfils, C., Talaulikar, K., Voyer, D., Bogdanov, A., and P. Mattes, "Segment Routing Policy Architecture", Work in Progress, Internet-Draft, draft-ietf-spring-segment-routing-policy-18, 17 February 2022, <<https://www.ietf.org/archive/id/draft-ietf-spring-segment-routing-policy-18.txt>>.

[IEEE802.1AX]

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

Authors' Addresses

Zhenqiang Li
China Mobile
Email: li_zhenqiang@hotmail.com

Tianran Zhou
Huawei
China
Email: zhoutianran@huawei.com

Jun Guo
ZTE Corp.
China
Email: guo.jun2@zte.com.cn

Greg Mirsky
Ericsson
United States of America
Email: gregimirsky@gmail.com

Rakesh Gandhi
Cisco
Canada
Email: rgandhi@cisco.com

IPPM
Internet-Draft
Intended status: Informational
Expires: January 13, 2022

M. Cociglio
Telecom Italia - TIM
A. Ferrieux
Orange Labs
G. Fioccola
Huawei Technologies
I. Lubashev
Akamai Technologies
F. Bulgarella
Telecom Italia - TIM
I. Hamchaoui
Orange Labs
M. Nilo
Telecom Italia - TIM
R. Sisto
Politecnico di Torino
D. Tikhonov
LiteSpeed Technologies
July 12, 2021

Explicit Flow Measurements Techniques
draft-mdt-ippm-explicit-flow-measurements-02

Abstract

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

Status of This Memo

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

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

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

This Internet-Draft will expire on January 13, 2022.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction	3
2. Notational Conventions	4
3. Latency Bits	5
3.1. Spin Bit	5
3.2. Delay Bit	6
3.2.1. Generation Phase	8
3.2.2. Reflection Phase	8
3.2.3. T_Max Selection	9
3.2.4. Delay Measurement using Delay Bit	10
3.2.5. Observer's Algorithm	12
3.2.6. Two Bits Delay Measurement: Spin Bit + Delay Bit . .	13
3.2.7. Hidden Delay Bit - Delay Bit with Privacy Protection	13
4. Loss Bits	13
4.1. T Bit - Round Trip Loss Bit	14
4.1.1. Round Trip Packet Loss Measurement	15
4.1.2. Setting the Round Trip Loss Bit on Outgoing Packets .	17
4.1.3. Observer's Logic for Round Trip Loss Signal	18
4.1.4. Loss Coverage and Signal Timing	19
4.2. Q Bit - Square Bit	19
4.2.1. Q Block Length Selection	19
4.2.2. Upstream Loss	20
4.2.3. Identifying Q Block Boundaries	21
4.3. L Bit - Loss Event Bit	21
4.3.1. End-To-End Loss	22

4.3.2. Loss Profile Characterization	22
4.4. L+Q Bits - Upstream, Downstream, and End-to-End Loss Measurements	22
4.4.1. Correlating End-to-End and Upstream Loss	23
4.5. R Bit - Reflection Square Bit	24
4.5.1. R+Q Bits - Using R and Q Bits for Passive Loss Measurement	25
4.5.2. Enhancement of R Block Length Computation	29
4.5.3. Improved Resilience to Packet Reordering	29
4.6. Improved Q and R Bits Resilience to Burst Losses	29
5. Summary of Delay and Loss Marking Methods	30
6. ECN-Echo Event Bit	32
6.1. Setting the ECN-Echo Event Bit on Outgoing Packets	32
6.2. Using E Bit for Passive ECN-Reported Congestion Measurement	32
7. Protocol Ossification Considerations	33
8. Examples of Application	33
8.1. QUIC	33
8.2. TCP	34
9. Security Considerations	34
9.1. Optimistic ACK Attack	35
10. Privacy Considerations	35
11. IANA Considerations	36
12. Change Log	36
13. Contributors	36
14. Acknowledgements	36
15. References	36
15.1. Normative References	36
15.2. Informative References	37
Authors' Addresses	39

1. Introduction

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

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

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

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

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

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

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

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

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

2. Notational Conventions

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

3. Latency Bits

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

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

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

3.1. Spin Bit

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

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

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

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

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

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

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

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

Note that packet reordering can cause spurious edges that require heuristics to correct. The spin bit performance deteriorates as soon as network impairments arise as explained in Section 3.2.

3.2. Delay Bit

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

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

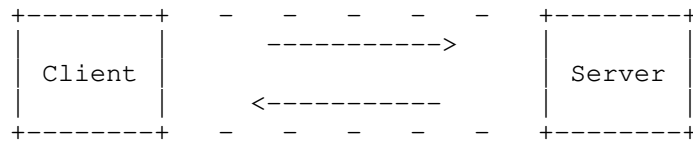
Unlike the spin bit, which is set in every packet transmitted on the network, the delay bit is set only once per round trip.

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

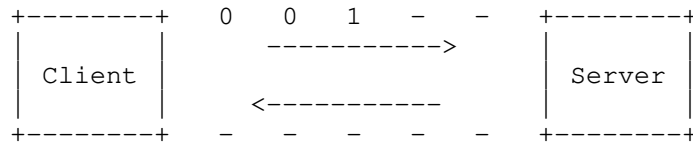
An observer placed at an intermediate point, observing a single direction of traffic, tracking the delay sample and the relative timestamp, can measure the round trip delay of the connection.

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

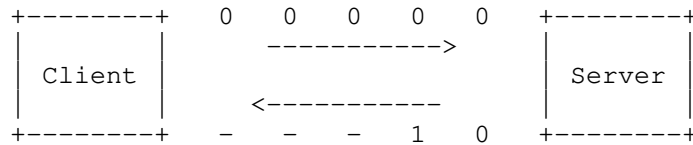
The next figure describes the elementary Delay bit mechanism.



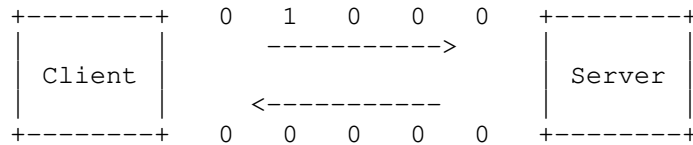
(a) No traffic at beginning.



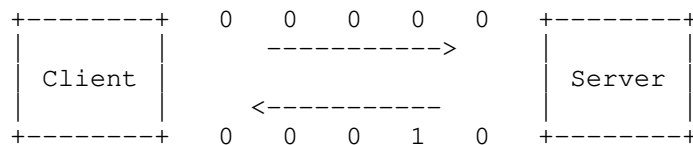
(b) The Client starts sending data and sets the first packet as Delay Sample.



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



(d) The Client reflects the Delay Sample.



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

Delay bit mechanism

3.2.1. Generation Phase

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

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

The server initializes the delay bit to 0 at the beginning of the connection, and its only task during the connection is described in Section 3.2.2.

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

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

To deal with these problems, the client generates a new delay sample if more than a predetermined time ("T_Max") has elapsed since the last delay sample transmission (including reflections). Note that "T_Max" should be greater than the max measurable RTT on the network. See Section 3.2.3 for details.

3.2.2. Reflection Phase

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

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

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

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

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

3.2.3. T_Max Selection

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

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

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

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

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

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

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

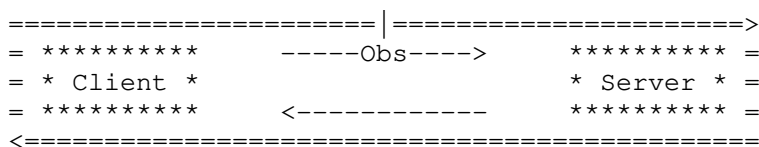
3.2.4. Delay Measurement using Delay Bit

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

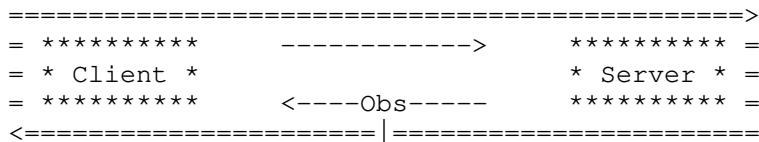
3.2.4.1. RTT Measurement

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

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



(a) client-server RTT



(b) server-client RTT

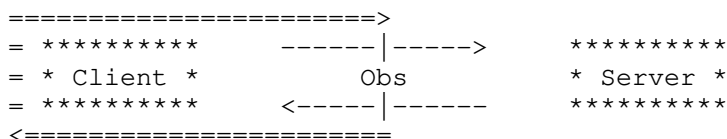
Round-trip time (both direction)

3.2.4.2. Half-RTT Measurement

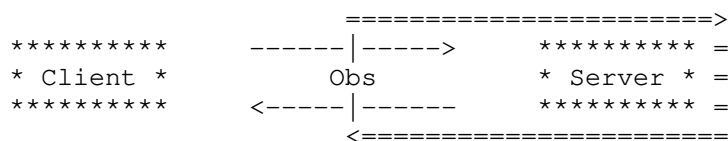
An observer that is able to observe both forward and return traffic directions can use the delay samples to measure "upstream" and "downstream" RTT components, also known as the half-RTT measurements. It does this by measuring the time between a delay sample observed in one direction and the delay sample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

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

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



(a) client-observer half-RTT

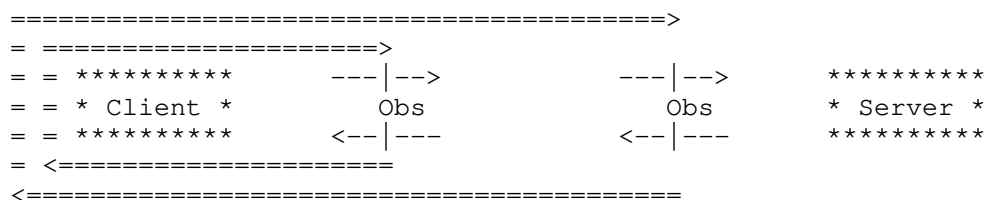


(b) observer-server half-RTT

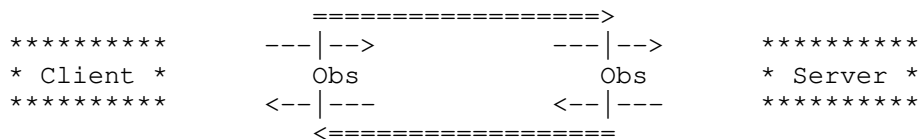
Half Round-trip time (both direction)

3.2.4.3. Intra-Domain RTT Measurement

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



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



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

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

3.2.5. Observer's Algorithm

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

A unidirectional observer, upon detecting a delay sample:

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

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

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

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

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

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

3.2.7. Hidden Delay Bit - Delay Bit with Privacy Protection

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

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

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

4. Loss Bits

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

- T: the "round Trip loss" bit is used in combination with the Spin bit to measure round-trip loss. See Section 4.1.
- Q: the "Square signal" bit is used to measure upstream loss. See Section 4.2.
- L: the "Loss event" bit is used to measure end-to-end loss. See Section 4.3.
- R: the "Reflection square signal" bit is used in combination with Q bit to measure end-to-end loss. See Section 4.1.

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

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

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

4.1. T Bit - Round Trip Loss Bit

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

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

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

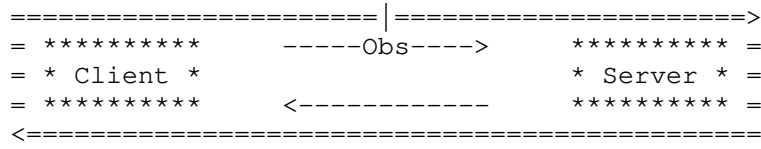
Packets belonging to the first round trip (first and second train) represent the Generation Phase, while those belonging to the second round trip (third and fourth train) represent the Reflection Phase.

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

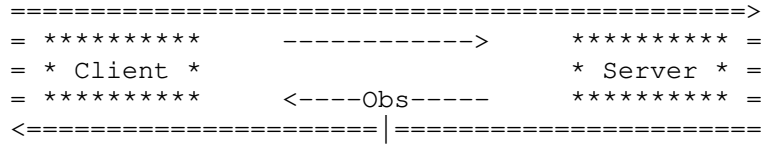
Since packet rates in two directions may be different, the number of marked packets in the train is determined by the direction with the lowest packet rate. See Section 4.1.2 for details on packet generation and for a mechanism to allow an observer to distinguish between trains belonging to different phases (Generation and Reflection).

4.1.1. Round Trip Packet Loss Measurement

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



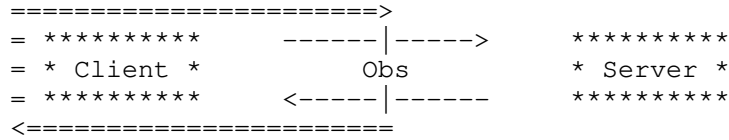
(a) client-server RTPL



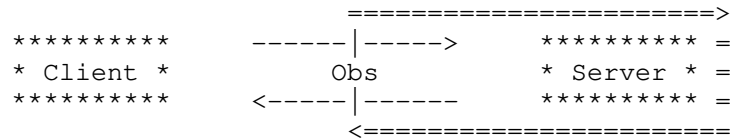
(b) server-client RTPL

Round-trip packet loss (both direction)

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

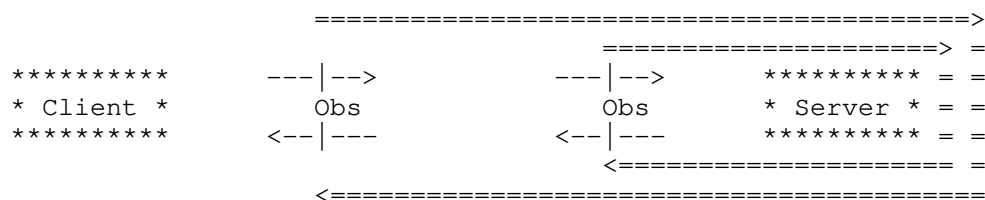


(a) client-observer half-RTPL

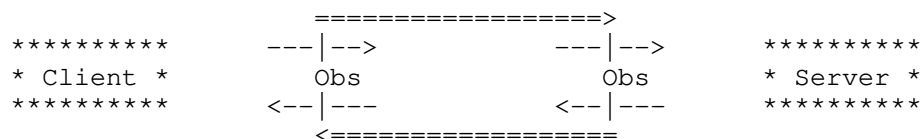


(b) observer-server half-RTPL

Half Round-trip packet loss (both direction)



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



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

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

4.1.2. Setting the Round Trip Loss Bit on Outgoing Packets

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

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

1. Generation Phase. The client starts generating marked packets for two consecutive spin-bit periods; it maintains a "generation token" count that is reset to zero at the beginning of the algorithm phase and is incremented every time a packet arrives. When the client transmits a packet and a "generation token" is available, the client marks the packet and retires a "generation token". If no token is available, the outgoing packet is transmitted unmarked. At the end of the first spin-bit period spent in generation, the reflection counter is unlocked to start counting incoming marked packets that will be reflected later;
2. Pause Phase. When the generation is completed, the client pauses till it has observed one entire spin bit period with no marked packets. That spin bit period is used by the observer as a separator between generated and reflected packets. During this marking pause, all the outgoing packets are transmitted with T

bit set to 0. The reflection counter is still incremented every time a marked packet arrives;

3. Reflection Phase. The client starts transmitting marked packets, decrementing the reflection counter for each transmitted marked packet until the reflection counter reached zero. The "generation token" method from the generation phase is used during this phase as well. At the end of the first spin-period spent in reflection, the reflection counter is locked to avoid incoming reflected packets incrementing it;
4. Pause Phase 2. The pause phase is repeated after the reflection phase and serves as a separator between the reflected packet train and a new packet train.

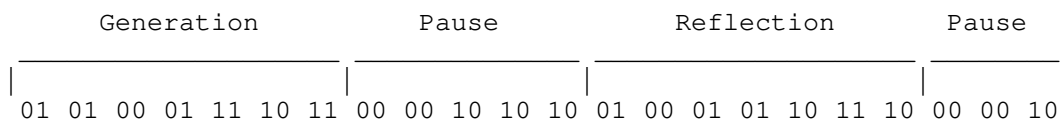
The generation token counter should be capped to limit the effects of a subsequent sudden reduction in the other endpoint's packet rate that could prevent that endpoint from reflecting collected packets. The most conservative cap value is "1".

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

4.1.3. Observer's Logic for Round Trip Loss Signal

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

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



Round Trip Loss signal example

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

4.1.4. Loss Coverage and Signal Timing

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

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

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

4.2. Q Bit - Square Bit

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

Observation points can estimate upstream losses by watching a single direction of the traffic flow and counting the number of packets in each observed Q Block, as described in Section 4.2.2.

4.2.1. Q Block Length Selection

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

In this latter scenario, the sender is expected to choose N (Q Block length) based on the expected amount of loss and reordering on the path. The choice of N strikes a compromise - the observation could become too unreliable in case of packet reordering and/or severe loss if N is too small, while short flows may not yield a useful upstream loss measurement if N is too large (see Section 4.2.2).

The value of N should be at least 64 and be a power of 2. This requirement allows an Observer to infer the Q Block length by observing one period of the square signal. It also allows the Observer to identify flows that set the loss bits to arbitrary values (see Section 7).

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

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

4.2.2. Upstream Loss

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

The observer needs to be able to tolerate packet reordering that can blur the edges of the square signal, as explained in Section 4.2.3.

```

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

```

(a) in client-server channel (uloss_up)

```

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

```

(b) in server-client channel (uloss_down)

Upstream loss

4.2.3. Identifying Q Block Boundaries

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

The choice of X represents a trade-off between resiliency to reordering and resiliency to loss. A very large Marking Block Threshold will be able to reconstruct Q Blocks despite a significant amount of 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 sSquare and Reflection square bits are toggled at most every N packets (except for the first transition of the R-bit as explained before), an on-path observer can count the number of packets of each marking block and, knowing the value of N, can estimate the amount of loss experienced by the connection. An observer can calculate different measurements depending on whether it is able to observe a single direction of the traffic or both directions.

Single directional observer:

- upstream loss in the observed direction: the loss between the sender and the observation point (see Section 4.2.2)

- "three-quarters" connection loss: the loss between the receiver and the sender in the unobserved direction plus the loss between the sender and the observation point in the observed direction
- end-to-end loss in the unobserved direction: the loss between the receiver and the sender in the opposite direction

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

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

4.5.1.1. Three-Quarters Connection Loss

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

```

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

```

(a) in client-server channel (tqloss_up)

```

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

```

(b) in server-client channel (tqloss_down)

Three-quarters connection loss

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

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

End-to-end loss in the unobserved direction ("eloss_unobserved") relates to the "three-quarters" connection loss ("tqloss") and upstream loss in the observed direction ("uloss") as $(1 - \text{eloss_unobserved})(1 - \text{uloss}) = 1 - \text{tqloss}$. Hence, $\text{eloss_unobserved} = (\text{tqloss} - \text{uloss}) / (1 - \text{uloss})$.

```

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

```

(a) in client-server channel (eloss_down)

```

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

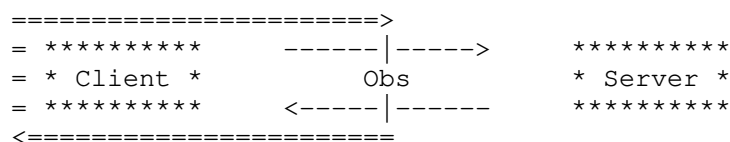
```

(b) in server-client channel (eloss_up)

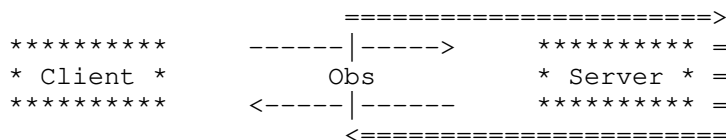
End-To-End loss in the opposite direction

4.5.1.3. Half Round-Trip Loss

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



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



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

Half Round-trip loss (both direction)

4.5.1.4. Downstream Loss

If the observer is able to observe both directions of traffic, it is able to calculate two downstream loss measurements using either end-to-end loss and upstream loss, similar to the calculation in Section 4.4.1.1 or using "half round-trip" loss and upstream loss in the opposite direction.

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

```

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

```

(a) in client-server channel (dloss_up)

```

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

```

(b) in server-client channel (dloss_down)

Downstream loss

4.5.2. Enhancement of R Block Length Computation

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

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

4.5.3. Improved Resilience to Packet Reordering

When a protocol implementing the marking mechanism is able to detect when packets are received out of order, it can improve resilience to packet reordering beyond what is possible using methods described in Section 4.2.3.

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

4.6. Improved Q and R Bits Resilience to Burst Losses

Burst losses can affect Q and R measurements accuracy. Generally, burst losses can be absorbed and correctly measured if smaller than the established Q Block length. On the other hand, entire periods might be wiped out if the burst sizes become too large thus making the observer completely unaware of their loss.

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

5. Summary of Delay and Loss Marking Methods

This section summarizes the marking methods described in this draft.

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

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

x2 Same metric for both directions

* Both algorithms work independtly; an observer could use approximate spin bit measures when delay bit ones aren't available

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

Figure 1: Delay Comparison

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

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

* All protocols

Protocols employing loss detection (w/ or w/o pure ACK loss detection)

\$ Require a working spin bit

! Metric relative to the opposite channel

x2 Same metric for both directions

ppa Packets-Per-Ack

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

Figure 2: Loss Comparison

6. ECN-Echo Event Bit

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

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

- E: The "ECN-Echo Event" bit is set to 0 or 1 according to the Unreported ECN Echo counter, as explained below in Section 6.1.

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

The Unreported ECN-Echo counter operates identically to Unreported Loss counter (Section 4.3), except it counts packets delivered by the network with CE markings, according to the ECN-Echo feedback from the receiver.

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

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

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

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

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

7. Protocol Ossification Considerations

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

The delay and loss bits are amenable to "greasing" described in [RFC8701], if the protocol designers are not ready to dedicate (and ossify) bits used for loss reporting to this function. The greasing could be accomplished similarly to the Latency Spin bit greasing in [QUIC-TRANSPORT]. Namely, implementations could decide that a fraction of flows should not encode loss and delay information and, instead, the bits would be set to arbitrary values. The observers would need to be ready to ignore flows with delay and loss information more resembling noise than the expected signal.

8. Examples of Application

8.1. QUIC

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

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

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

```

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

Scheme 1

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

```

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

```

Scheme 2A

```

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

```

Scheme 2B

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

```

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

```

Scheme 3A

```

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

```

Scheme 3B

8.2. TCP

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

9. Security Considerations

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

In the absence of packet loss, Q and R bits signals do not provide any information that cannot be observed by simply counting packets

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

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

9.1. Optimistic ACK Attack

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

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

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

10. Privacy Considerations

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

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

incrementing Unreported Loss counter for loss of packets sent to a different destination or with a different Connection ID.

11. IANA Considerations

This document makes no request of IANA.

12. Change Log

TBD

13. Contributors

The following people provided valuable contributions to this document:

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

14. Acknowledgements

TBD

15. References

15.1. Normative References

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

- [IP] Postel, J., "Internet Protocol", STD 5, RFC 791, DOI 10.17487/RFC0791, September 1981, <<https://www.rfc-editor.org/info/rfc791>>.
- [IPM-Methods] Morton, A., "Active and Passive Metrics and Methods (with Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799, May 2016, <<https://www.rfc-editor.org/info/rfc7799>>.
- [IPv6] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <<https://www.rfc-editor.org/info/rfc8200>>.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC8558] Hardie, T., Ed., "Transport Protocol Path Signals", RFC 8558, DOI 10.17487/RFC8558, April 2019, <<https://www.rfc-editor.org/info/rfc8558>>.
- [TCP] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <<https://www.rfc-editor.org/info/rfc793>>.

15.2. Informative References

- [ACCURATE] Briscoe, B., Kuehlewind, M., and R. Scheffenegger, "More Accurate ECN Feedback in TCP", draft-ietf-tcpm-accurate-ecn-14 (work in progress), February 2021.
- [AltMark] Fioccola, G., Ed., Capello, A., Cociglio, M., Castaldelli, L., Chen, M., Zheng, L., Mirsky, G., and T. Mizrahi, "Alternate-Marking Method for Passive and Hybrid Performance Monitoring", RFC 8321, DOI 10.17487/RFC8321, January 2018, <<https://www.rfc-editor.org/info/rfc8321>>.
- [ANRW19-PM-QUIC] Bulgarella, F., Cociglio, M., Fioccola, G., Marchetto, G., and R. Sisto, "Performance measurements of QUIC communications", Proceedings of the Applied Networking Research Workshop, DOI 10.1145/3340301.3341127, July 2019.

- [I-D.trammell-ippm-spin]
Trammell, B., "An Explicit Transport-Layer Signal for Hybrid RTT Measurement", draft-trammell-ippm-spin-00 (work in progress), January 2019.
- [I-D.trammell-tsvwg-spin]
Trammell, B., "A Transport-Independent Explicit Signal for Hybrid RTT Measurement", draft-trammell-tsvwg-spin-00 (work in progress), July 2018.
- [IPv6AltMark]
Fioccola, G., Zhou, T., Cociglio, M., Qin, F., and R. Pang, "IPv6 Application of the Alternate Marking Method", draft-ietf-6man-ipv6-alt-mark-04 (work in progress), March 2021.
- [QUIC-TRANSPORT]
Iyengar, J. and M. Thomson, "QUIC: A UDP-Based Multiplexed and Secure Transport", draft-ietf-quic-transport-34 (work in progress), January 2021.
- [RFC8517] Dolson, D., Ed., Snellman, J., Boucadair, M., Ed., and C. Jacquenet, "An Inventory of Transport-Centric Functions Provided by Middleboxes: An Operator Perspective", RFC 8517, DOI 10.17487/RFC8517, February 2019, <<https://www.rfc-editor.org/info/rfc8517>>.
- [RFC8701] Benjamin, D., "Applying Generate Random Extensions And Sustain Extensibility (GREASE) to TLS Extensibility", RFC 8701, DOI 10.17487/RFC8701, January 2020, <<https://www.rfc-editor.org/info/rfc8701>>.
- [SPIN-BIT]
Trammell, B., Vaere, P. D., Even, R., Fioccola, G., Fossati, T., Ihlar, M., Morton, A., and E. Stephan, "Adding Explicit Passive Measurability of Two-Way Latency to the QUIC Transport Protocol", draft-trammell-quic-spin-03 (work in progress), May 2018.
- [TRANSPORT-ENCRYPT]
Fairhurst, G. and C. Perkins, "Considerations around Transport Header Confidentiality, Network Operations, and the Evolution of Internet Transport Protocols", draft-ietf-tsvwg-transport-encrypt-21 (work in progress), April 2021.

[UDP-OPTIONS]

Touch, J., "Transport Options for UDP", draft-ietf-tsvwg-udp-options-12 (work in progress), May 2021.

[UDP-SURPLUS]

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

Authors' Addresses

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

EMail: mauro.cociglio@telecomitalia.it

Alexandre Ferrieux
Orange Labs

EMail: alexandre.ferrieux@orange.com

Giuseppe Fioccola
Huawei Technologies
Riesstrasse, 25
Munich 80992
Germany

EMail: giuseppe.fioccola@huawei.com

Igor Lubashev
Akamai Technologies

EMail: ilubashe@akamai.com

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

EMail: fabio.bulgarella@guest.telecomitalia.it

Isabelle Hamchaoui
Orange Labs

EMail: isabelle.hamchaoui@orange.com

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

EMail: massimo.nilo@telecomitalia.it

Riccardo Sisto
Politecnico di Torino

EMail: riccardo.sisto@polito.it

Dmitri Tikhonov
LiteSpeed Technologies

EMail: dtikhonov@litespeedtech.com

IPPM Working Group
Internet-Draft
Intended status: Standards Track
Expires: 27 October 2022

G. Mirsky
Ericsson
W. Lingqiang
G. Zhui
ZTE Corporation
H. Song
Futurewei Technologies
P. Thubert
Cisco Systems, Inc
25 April 2022

Hybrid Two-Step Performance Measurement Method
draft-mirsky-ippm-hybrid-two-step-13

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.

Status of This Memo

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

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

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

This Internet-Draft will expire on 27 October 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction	2
2. Conventions used in this document	3
2.1. Acronyms	3
2.2. Requirements Language	4
3. Problem Overview	4
4. Theory of Operation	5
4.1. Operation of the HTS Ingress Node	7
4.2. Operation of the HTS Intermediate Node	9
4.3. Operation of the HTS Egress Node	10
4.4. Considerations for HTS Timers	11
4.5. Deploying HTS in a Multicast Network	11
5. Authentication in HTS	12
6. IANA Considerations	13
6.1. IOAM Option-Type for HTS	13
6.2. HTS TLV Registry	13
6.3. HTS Sub-TLV Type Sub-registry	14
6.4. HMAC Type Sub-registry	15
7. Security Considerations	16
8. Acknowledgments	16
9. References	16
9.1. Normative References	16
9.2. Informative References	17
Authors' Addresses	19

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 improves 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. Acronyms

RTM Residence Time Measurement

ECMP Equal Cost Multipath

MTU Maximum Transmission Unit

HTS Hybrid Two-Step

HMAC Hashed Message Authentication Code

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 when 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 recorded can differ from the ideal case by any fixed amount. 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 on the path, 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 the client layer or risk missing network state data for the part, most probably the latter part, of the path.

In some networks, for example, wireless that are in the scope of [I-D.ietf-raw-use-cases], it is beneficial to collect the telemetry, including the calculated performance metrics, that reflects conditions experienced by the monitored flow at a node, other than the egress. For example, a head-end can optimize path selection based on the compounded information that reflects network conditions, resource utilization. This mode is referred to as the upstream collection and the other - downstream collection to differentiate between two modes of telemetry collection.

4. Theory of Operation

The HTS method consists of two phases:

- * performing a measurement and/or obtaining network state information on a node;
- * collecting and transporting the measurement and/or the telemetry information.

HTS may use an HTS Trigger carried in a data packet or a specially constructed test packet. For example, an HTS Trigger could be a packet that has IOAM Option-Type set to the "IOAM Hybrid Two-Step Option-Type" value (TBA1) allocated by IANA (see Section 6.1). The HTS Trigger also includes IOAM Namespace-ID and IOAM-Trace-Type information s defined in Section 5.3 and Section 5.4.1 [I-D.ietf-ippm-ioam-data] respectively (shown in Figure 1). A packet in the flow to which the Alternate-Marking method, defined in [RFC8321] and [RFC8889], is applied can be used as an HTS Trigger.

The 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.

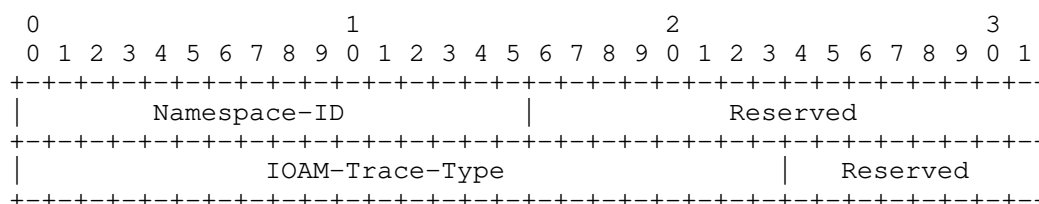


Figure 1: Hybrid Two-Step Trace IOAM Header

The HTS method uses the HTS Follow-up packet, 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. In some use cases, e.g., when HTS is used to collect the telemetry, including performance metrics, calculated based on a series of measurements, an HTS follow-up packet can be originated without using the HTS Trigger. The follow-up packet contains characteristic information sufficient for participating HTS nodes to associate it with the monitored data flow. The characteristic information can be obtained using the information of the trigger packet or constructed by a node that originates the follow-up packet. As the follow-up packet is expected to traverse the same sequence of nodes, one element of the characteristic information is the information that determines the path in the data plane. For example, in a segment routing domain [RFC8402], a list of segment identifiers of the trigger packet is applied to the follow-up packet. And in the case of the service function chain based on the Network Service Header [RFC8300], the Base Header and Service Path Header of the trigger packet will be applied to the follow-up packet. Also, when HTS is used to collect the telemetry information in an IOAM domain, the IOAM trace option header [I-D.ietf-ippm-ioam-data] of the trigger packet is applied in the follow-up packet. The follow-up packet also uses the same network information used to load-balance flows in equal-cost multipath (ECMP) as the trigger packet, e.g., IPv6 Flow Label [RFC6437] or an entropy label [RFC6790]. The exact composition of the characteristic information is specific for each transport network, and its definition is outside the scope of this document.

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.

The following sections describe the operation of HTS nodes in the downstream mode of collecting the telemetry information. In the upstream mode, the behavior of HTS nodes, in general, identical with the exception that the HTS Trigger packet does not precede the HTS Follow-up packet.

4.1. Operation of the HTS Ingress Node

A node that originates the HTS Trigger is referred to as the 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 2 displays an example of the follow-up packet format.

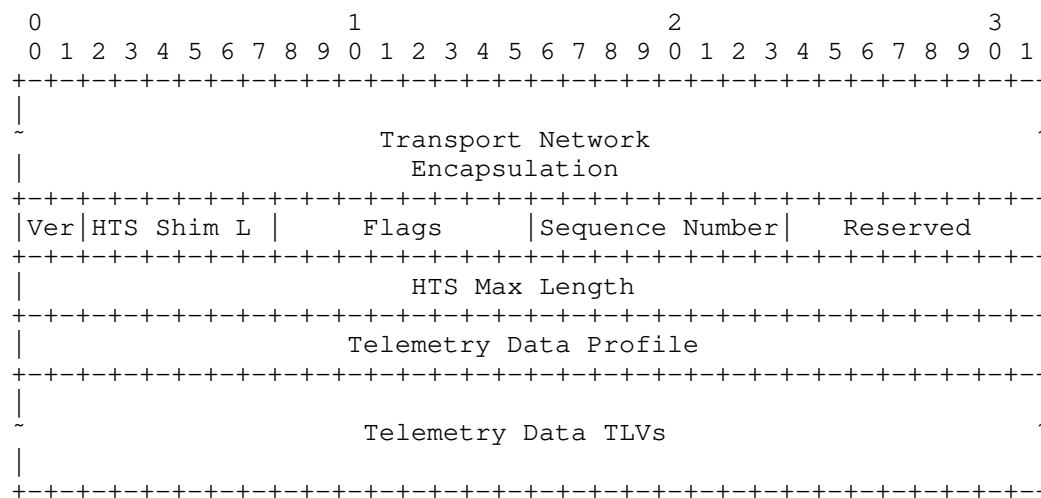


Figure 2: Follow-up Packet Format

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 octets. The minimal value of the field is eight octets.

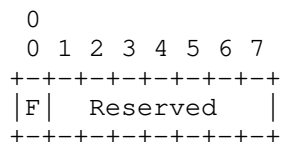


Figure 3: Flags Field Format

Flags is eight-bits long. The format of the Flags field displayed in Figure 3.

- 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 one octet-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 that originated in response to the same HTS trigger. The ingress node MUST set the value of the field to zero.

Reserved is one octet-long field. It MUST be zeroed on transmission and ignored on receipt.

HTS Max Length is four octet-long field. The value of the HTS Max Length field indicates the maximum length of the HTS Follow-up packet in octets. An operator MUST be able to configure the HTS Max Length field's value. The value SHOULD be set equal to the path MTU.

Telemetry Data Profile is the optional variable-length field of bit-size flags. Each flag indicates the requested type of telemetry data to be collected at 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.

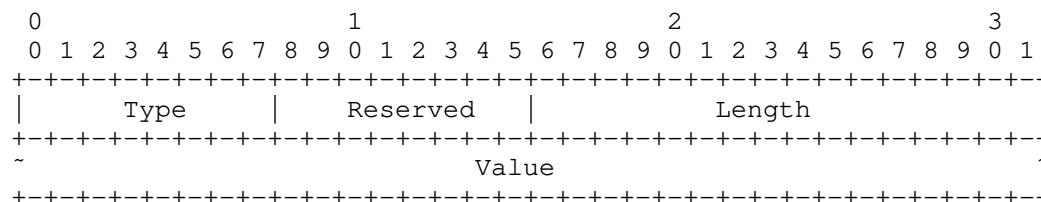


Figure 4: Telemetry Data TLV Format

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 4 presents the format of a Telemetry Data TLV, where fields are defined as the following:

- Type - a one-octet-long field that characterizes the interpretation of the Value field.
- Reserved - one-octet-long field.
- Length - two-octet-long field equal to the length of the Value field in octets.
- Value - a variable-length field. The value of the Type field determines its interpretation and encoding. 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;
- * transmit the trigger packet.

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

1. verify that the matching transport information exists and the Full flag is cleared, then stop the associated HTS Follow-up Timer;

2. otherwise, transmit the received packet. Proceed to Step 8;
3. collect telemetry data requested in the Telemetry Data Profile field or defined by the local HTS policy;
4. if adding the collected telemetry would not exceed HTS Max Length field's value, then append data as a new Telemetry Data TLV and transmit the follow-up packet. Proceed to Step 8;
5. otherwise, set the value of the Full flag to one, copy the transport information from the received follow-up packet and transmit it accordingly. Proceed to Step 8;
6. originate the new follow-up packet using the transport information copied from the received follow-up packet. 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;
7. copy collected telemetry data into the first Telemetry Data TLV's Value field and then transmit the packet;
8. processing completed.

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

- * originate the follow-up packet using transport information associated with the expired timer;
- * initialize the HTS shim by setting the Version field's value 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.
- * copy telemetry information into Telemetry Data TLV's Value 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:

- * copy the transport information;
- * 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 for each of Telemetry Data TLVs MUST:

- * if HTS is used in the authenticated mode, verify the authentication of the Telemetry Data TLV using the Authentication sub-TLV (see Section 5);
- * copy telemetry information from the Value field;
- * 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. For the particular flow, there MUST be no 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 invaluable 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 the HTS packet as defined below:

- * the first transmitted multicast packet MUST be followed by the received corresponding HTS packet as described in Section 4.2;
- * 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 an 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 reconstruct and analyze the state of the particular multicast distribution tree based on HTS packets collected from egress nodes.

5. Authentication in HTS

Telemetry information may be used to drive network operation, closing the control loop for self-driving, self-healing networks. Thus it is critical to provide a mechanism to protect the telemetry information collected using the HTS method. This document defines an optional authentication of a Telemetry Data TLV that protects the collected information's integrity.

The format of the Authentication sub-TLV is displayed in Figure 5.

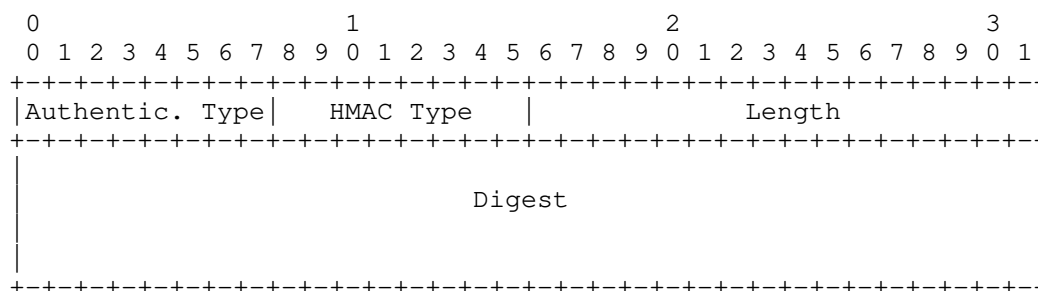


Figure 5: HMAC sub-TLV

where fields are defined as follows:

- * Authentication Type - is a one-octet-long field, value TBA2 allocated by IANA Section 6.2.
- * Length - two-octet-long field, set equal to the length of the Digest field in octets.
- * HMAC Type - is a one-octet-long field that identifies the type of the HMAC and the length of the digest and the length of the digest according to the HTS HMAC Type sub-registry (see Section 6.4).
- * Digest - is a variable-length field that carries HMAC digest of the text that includes the encompassing TLV.

This specification defines the use of HMAC-SHA-256 truncated to 128 bits ([RFC4868]) in HTS. Future specifications may define the use in HTS of more advanced cryptographic algorithms or the use of digest of a different length. HMAC is calculated as defined in [RFC2104] over

text as the concatenation of the Sequence Number field of the follow-up packet (see Figure 2) and the preceding data collected in the Telemetry Data TLV. The digest then MUST be truncated to 128 bits and written into the Digest field. Distribution and management of shared keys are outside the scope of this document. In the HTS authenticated mode, the Authentication sub-TLV MUST be present in each Telemetry Data TLV. HMAC MUST be verified before using any data in the included Telemetry Data TLV. If HMAC verification fails, the system MUST stop processing corresponding Telemetry Data TLV and notify an operator. Specification of the notification mechanism is outside the scope of this document.

6. IANA Considerations

6.1. IOAM Option-Type for HTS

The IOAM Option-Type registry is requested in [I-D.ietf-ippm-ioam-data]. IANA is requested to allocate a new code point as listed in Table 1.

Value	Description	Reference
TBA1	IOAM Hybrid Two-Step Option-Type	This document

Table 1: IOAM Option-Type for HTS

6.2. HTS TLV Registry

IANA is requested to create the HTS TLV Type registry. All code points in the range 1 through 175 in this registry shall be allocated according to the "IETF Review" procedure 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 specified in [RFC8126]. The remaining code points are allocated according to Table 2:

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 2: HTS TLV Type Registry

6.3. HTS Sub-TLV Type Sub-registry

IANA is requested to create the HTS sub-TLV Type sub-registry as part of the HTS TLV Type registry. All code points in the range 1 through 175 in this registry shall be allocated according to the "IETF Review" procedure 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 specified in [RFC8126]. The remaining code points are allocated according to Table 3:

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 3: HTS Sub-TLV Type Sub-registry

This document defines the following new values in the IETF Review range of the HTS sub-TLV Type sub-registry:

Value	Description	TLV Used	Reference
TBA2	HMAC	Any	This document

Table 4: HTS sub-TLV Types

6.4. HMAC Type Sub-registry

IANA is requested to create the HMAC Type sub-registry as part of the HTS TLV Type registry. All code points in the range 1 through 127 in this registry shall be allocated according to the "IETF Review" procedure specified in [RFC8126]. Code points in the range 128 through 239 in this registry shall be allocated according to the "First Come First Served" procedure specified in [RFC8126]. The remaining code points are allocated according to Table 5:

Value	Description	Reference
0	Reserved	This document
1- 127	Unassigned	This document
128 - 239	Unassigned	This document
240 - 249	Experimental	This document
250 - 254	Private Use	This document
255	Reserved	This document

Table 5: HMAC Type Sub-registry

This document defines the following new values in the HMAC Type sub-registry:

Value	Description	Reference
1	HMAC-SHA-256 16 octets long	This document

Table 6: HMAC Types

7. Security Considerations

Nodes that practice the 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.

This document defines the use of authentication (Section 5) to protect the integrity of the telemetry information collected using the HTS method. Privacy protection can be achieved by, for example, sharing the IPsec tunnel with a data flow that generates information that is collected using HTS.

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.

8. 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.

9. References

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

- [RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

9.2. Informative References

- [I-D.ietf-ippm-ioam-data]
Brockners, F., Bhandari, S., and T. Mizrahi, "Data Fields for In-situ OAM", Work in Progress, Internet-Draft, draft-ietf-ippm-ioam-data-17, 13 December 2021, <<https://datatracker.ietf.org/doc/html/draft-ietf-ippm-ioam-data-17>>.
- [I-D.ietf-ippm-ioam-direct-export]
Song, H., Gafni, B., Zhou, T., Li, Z., Brockners, F., Bhandari, S., Sivakolundu, R., and T. Mizrahi, "In-situ OAM Direct Exporting", Work in Progress, Internet-Draft, draft-ietf-ippm-ioam-direct-export-07, 13 October 2021, <<https://datatracker.ietf.org/doc/html/draft-ietf-ippm-ioam-direct-export-07>>.
- [I-D.ietf-raw-use-cases]
Bernardos, C. J., Papadopoulos, G. Z., Thubert, P., and F. Theoleyre, "RAW use-cases", Work in Progress, Internet-Draft, draft-ietf-raw-use-cases-05, 23 February 2022, <<https://datatracker.ietf.org/doc/html/draft-ietf-raw-use-cases-05>>.
- [I-D.song-ippm-postcard-based-telemetry]
Song, H., Mirsky, G., Filsfils, C., Abdelsalam, A., Zhou, T., Li, Z., Shin, J., and K. Lee, "In-Situ OAM Marking-based Direct Export", Work in Progress, Internet-Draft, draft-song-ippm-postcard-based-telemetry-11, 15 November 2021, <<https://datatracker.ietf.org/doc/html/draft-song-ippm-postcard-based-telemetry-11>>.
- [P4.INT] "In-band Network Telemetry (INT)", P4.org Specification, October 2017.
- [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>>.

- [RFC6437] Amante, S., Carpenter, B., Jiang, S., and J. Rajahalme, "IPv6 Flow Label Specification", RFC 6437, DOI 10.17487/RFC6437, November 2011, <<https://www.rfc-editor.org/info/rfc6437>>.
- [RFC6790] Kompella, K., Drake, J., Amante, S., Henderickx, W., and L. Yong, "The Use of Entropy Labels in MPLS Forwarding", RFC 6790, DOI 10.17487/RFC6790, November 2012, <<https://www.rfc-editor.org/info/rfc6790>>.
- [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>>.
- [RFC8300] Quinn, P., Ed., Elzur, U., Ed., and C. Pignataro, Ed., "Network Service Header (NSH)", RFC 8300, DOI 10.17487/RFC8300, January 2018, <<https://www.rfc-editor.org/info/rfc8300>>.
- [RFC8321] Fioccola, G., Ed., Capello, A., Cociglio, M., Castaldelli, L., Chen, M., Zheng, L., Mirsky, G., and T. Mizrahi, "Alternate-Marking Method for Passive and Hybrid Performance Monitoring", RFC 8321, DOI 10.17487/RFC8321, January 2018, <<https://www.rfc-editor.org/info/rfc8321>>.
- [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>>.
- [RFC8889] Fioccola, G., Ed., Cociglio, M., Sapio, A., and R. Sisto, "Multipoint Alternate-Marking Method for Passive and Hybrid Performance Monitoring", RFC 8889, DOI 10.17487/RFC8889, August 2020, <<https://www.rfc-editor.org/info/rfc8889>>.

Authors' Addresses

Greg Mirsky
Ericsson
Email: gregimirsky@gmail.com

Wang Lingqiang
ZTE Corporation
No 19 ,East Huayuan Road
Beijing
Phone: +86 10 82963945
Email: wang.lingqiang@zte.com.cn

Guo Zhui
ZTE Corporation
No 19 ,East Huayuan Road
Beijing
Phone: +86 10 82963945
Email: guo.zhui@zte.com.cn

Haoyu Song
Futurewei Technologies
2330 Central Expressway
Santa Clara,
United States of America
Email: hsong@futurewei.com

Pascal Thubert
Cisco Systems, Inc
Building D
45 Allée des Ormes - BP1200
06254 MOUGINS - Sophia Antipolis
France
Phone: +33 497 23 26 34
Email: pthubert@cisco.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 28, 2022

L. Ciavattone
A. Morton
AT&T Labs
October 25, 2021

Test Protocol for One-way IP Capacity Measurement
draft-morton-ippm-capacity-metric-protocol-02

Abstract

This memo addresses the problem of protocol support for measuring Network Capacity metrics in RFC xxxx: draft-ietf-ippm-capacity-metric-method, where the method deploys a feedback channel from the receiver to control the sender's transmission rate in near-real-time. It supplies a simple protocol to perform the measurements.

See Section 10: The authors seek feedback to determine what additional features will be necessary for an IETF Standards Track Protocol, beyond what is present in the running code available now.

Status of This Memo

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

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

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

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

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

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

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
1.1. Requirements Language	3
2. Scope, Goals, and Applicability	3
3. Protocol Overview	4
4. General Parameters and Definitions	5
5. Setup Request and Response Exchange	7
5.1. Setup Response Processing at the Client	11
6. Test Activation Request and Response	11
6.1. Test Activation Request at the client	11
6.2. Test Activation Request - server response	13
6.3. Test Activation Response action at the client	15
7. Test Stream Transmission and Measurement Feedback Messages .	15
7.1. Test Packet PDU and Roles	15
7.2. Status PDU	18
8. Stopping the Test	23
9. Method of Measurement	24
9.1. Running Code	24
10. Security Considerations	24
11. IANA Considerations	26
12. Acknowledgments	27
13. References	27
13.1. Normative References	27
13.2. Informative References	28
Authors' Addresses	30

1. Introduction

The IETF's efforts to define Network and Bulk Transport Capacity have been chartered and finally progressed after over twenty years.

Over that time, the performance community has seen development of Informative definitions in [RFC3148] for Framework for Bulk Transport Capacity (BTC), RFC 5136 for Network Capacity and Maximum IP-layer Capacity, and the Experimental metric definitions and methods in [RFC8337], Model-Based Metrics for BTC.

This memo looks at the problem of measuring Network Capacity metrics defined in [I-D.ietf-ippm-capacity-metric-method] where the method deploys a feedback channel from the receiver to control the sender's transmission rate in near-real-time.

Although there are several test protocol already available for support and manage active measurements, this protocol is a major departure from their operation:

1. UDP transport is used for all setup, test activation, and control messages, and for results feedback (not TCP), simplifying operations.
2. TWAMP [RFC5357] and STAMP [RFC8762] use the philosophy that one host is a Session-Reflector, sending test packets every time they receive a test packet. This protocol supports a one-way test with periodic status messages returned to the sender. These messages are also a basis for on-path Round-trip delay measurements, which are a key input to the load adjustment search algorithm.
3. OWAMP [RFC4656] supports one-way testing with results Fetch at the end of the test session. This protocol supports a one-way test and requires periodic status messages returned to the sender to support the load adjustment search algorithm.
4. The security features of OWAMP [RFC4656] and TWAMP [RFC5357] have been described as "unusual", to the point that IESG approved their use while also asking that these methods not be used again. Further, the common OWAMP [RFC4656] and TWAMP [RFC5357] approach to security is over 15 years old at this time.

Note: the -02 update of this draft will be the last that describes version 8 of the protocol in the running code. Future updates of the draft will correspond to protocol version 9 and higher versions.

1.1. Requirements Language

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

2. Scope, Goals, and Applicability

The scope of this memo is to define a protocol to measure the Maximum IP-Layer Capacity metric and according to the standardized method.

The continued goal is to harmonize the specified metric and method across the industry, and this protocol supports the specifications of IETF and other Standards Development Organizations.

All active testing protocols currently defined by the IPPM WG are UDP-based, but this protocol specifies both control and test protocols using UDP transport. Also, the control protocol continues operating during testing to convey results and dynamic configurations.

The primary application of the protocol described here is the same as in Section 2 of [RFC7497] where:

- o The access portion of the network is the focus of this problem statement. The user typically subscribes to a service with bidirectional access partly described by rates in bits per second.

3. Protocol Overview

This section gives an informative overview of the communication protocol between two test end-points (without expressing requirements: later sections provide details and requirements).

One end-point takes the role of server, awaiting connection requests on a well-known port from the other end-point, the client.

The client requires configuration of a test direction parameter (upstream or downstream test) as well as the hostname or IP address of the server in order to begin the setup and configuration exchanges with the server.

The protocol uses UDP transport and has four phases:

1. Setup Request and Response Exchange: The client requests to begin a test by communicating its protocol version, intended security mode, and jumbo datagram support. The server either confirms matching configuration or rejects the connection. The server also communicates the ephemeral port for further communication when accepting the client's request.
2. Test Activation Request and Response: the client composes a request conveying parameters such as the testing direction, the duration of the test interval and test sub-intervals, and various thresholds. The server then chooses to accept, ignore or modify any of the test parameters, and communicates the set that will be used unless the client rejects the modifications. Note that the client assumes that the Test Activation exchange has opened any co-located firewalls and network address/port translators for the test connection (in response to the Request packet on the ephemeral port) and the traffic that follows. If the Test Activation is rejected or fails, the client assumes that the

firewall will close the address/port combination after the firewall's configured idle traffic time-out.

3. Test Stream Transmission and Measurement Feedback Messages:
Testing proceeds with one end-point sending load PDUs and the other end-point receiving the load PDUs and sending frequent status messages to communicate status and transmission conditions there. The feedback messages are input to a load-control algorithm at the server, which controls future sending rates at either end-point as needed. The choice to locate the load-control algorithm at the server, regardless of transmission direction, means that the algorithm can be updated more easily at a host within the network, and at a fewer number of hosts than the number of clients.
4. Stopping the Test: The server initiates the phase to stop the test by setting the STOP1 indication in load PDUs or sstatus feedback messages. The client acknowledges by setting the STOP2 in further load PDUs or messages, and a graceful connection termination at each end-point follows. (Since the load PDUs and feedback messages are used, this phase is kind of a sub-phase of 3.) If the Test traffic stops or the communication path fails, the client assumes that the firewall will close the address/port combination after the firewall's configured idle traffic time-out.

4. General Parameters and Definitions

This section lists the REQUIRED input factors to specify a Sender or Receiver metric.

- o Src, the address of a host (such as the globally routable IP address).
- o Dst, the address of a host (such as the globally routable IP address).
- o MaxHops, the limit on the number of Hops a specific packet may visit as it traverses from the host at Src to the host at Dst (implemented in the TTL or Hop Limit).
- o T0, the time at the start of measurement interval, when packets are first transmitted from the Source.
- o I, the nominal duration of a measurement interval at the destination (default 10 sec)

- o dt, the nominal duration of m equal sub-intervals in I at the destination (default 1 sec)
- o dtn, the beginning boundary of a specific sub-interval, n, one of m sub-intervals in I
- o FT, the feedback time interval between status feedback messages communicating measurement results, sent from the receiver to control the sender. The results are evaluated to determine how to adjust the current offered load rate at the sender (default 50ms)
- o Tmax, a maximum waiting time for test packets to arrive at the destination, set sufficiently long to disambiguate packets with long delays from packets that are discarded (lost), such that the distribution of one-way delay is not truncated.
- o F, the number of different flows synthesized by the method (default 1 flow)
- o flow, the stream of packets with the same n-tuple of designated header fields that (when held constant) result in identical treatment in a multi-path decision (such as the decision taken in load balancing). Note: The IPv6 flow label MAY be included in the flow definition when routers have complied with [RFC6438] guidelines.
- o Type-P, the complete description of the test packets for which this assessment applies (including the flow-defining fields). Note that the UDP transport layer is one requirement for test packets specified below. Type-P is a parallel concept to "population of interest" defined in clause 6.1.1 of[Y.1540].
- o PM, a list of fundamental metrics, such as loss, delay, and reordering, and corresponding target performance threshold. At least one fundamental metric and target performance threshold MUST be supplied (such as One-way IP Packet Loss [RFC7680] equal to zero).

A non-Parameter which is required for several metrics is defined below:

- o T, the host time of the *first* test packet's *arrival* as measured at the destination Measurement Point, or MP(Dst). There may be other packets sent between source and destination hosts that are excluded, so this is the time of arrival of the first packet used for measurement of the metric.

Note that time stamp format and resolution, sequence numbers, etc. will be established by this memo.

5. Setup Request and Response Exchange

All messages defined in this section SHALL use UDP transport. The hosts SHALL calculate and include the UDP checksum, or check the UDP checksum as necessary.

The client SHALL begin the Control protocol connection by sending a Setup Request message to the server's control port.

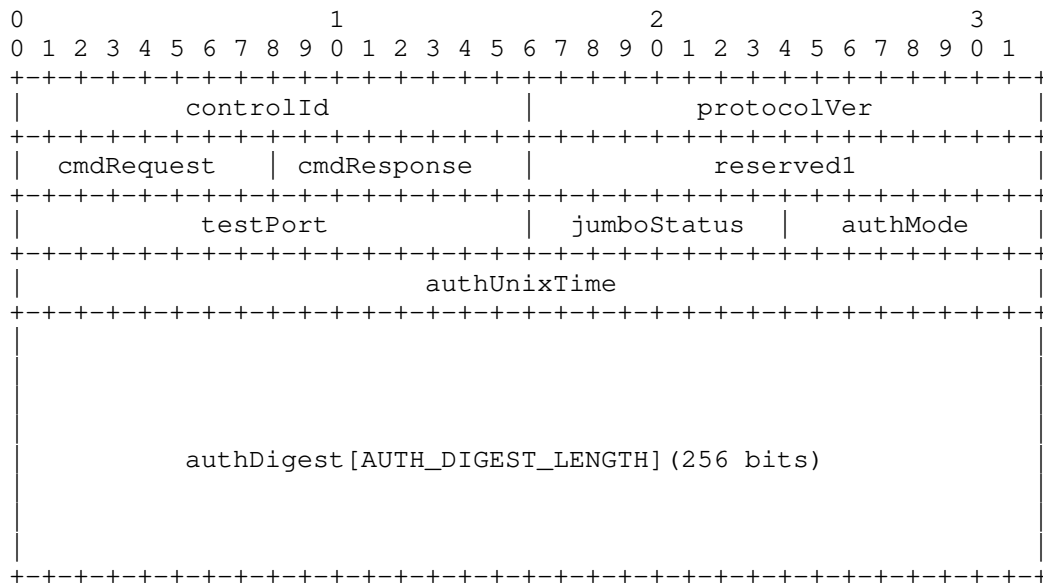
The client SHALL simultaneously start a test initiation timer so that if the control protocol fails to complete all exchanges in the allocated time, the client software SHALL exit (close the UDP socket and indicate an error message to the user).

(Note: in version 8, the watchdog time-out is configured, in udpst.h, as `#define WARNING_NOTRAFFIC 1` // Receive traffic stopped warning threshold (sec) `#define TIMEOUT_NOTRAFFIC (WARNING_NOTRAFFIC + 4)` or 5 seconds)

The Setup Request message PDU SHALL be organized as follows:

```
uint16_t controlId;    // Control ID = 0xACE1
uint16_t protocolVer;  // Protocol version = 0x08
uint8_t cmdRequest;    // Command request = 1 (request)
uint8_t cmdResponse;   // Command response = 0
uint16_t reserved1;    // Reserved (alignment)
uint16_t testPort;     // Test port on server (=0 for Request)
uint8_t jumboStatus;   // Jumbo datagram support status (BOOL)
uint8_t authMode;      // Authentication mode
uint32_t authUnixTime; // Authentication time stamp
unsigned char authDigest[AUTH_DIGEST_LENGTH] // SHA256_DIGEST_LENGTH = 32
oct
```

The UDP PDU format layout SHALL be as follows (big-endian AB):



When the server receives the Setup Request it SHALL validate the request by checking the protocol version, the jumbo datagram support indicator, and the authentication data if utilized. If the client has selected options for:

- o Jumbo datagram support status (BOOL),
- o Authentication mode, and
- o Authentication time stamp

that do not match the server configuration, the server MUST reject the Setup Request.

(Note: in version 8, the watchdog time is configured, in udpst.h, as `#define WARNING_NOTRAFFIC 1 // Receive traffic stopped warning threshold (sec) #define TIMEOUT_NOTRAFFIC (WARNING_NOTRAFFIC + 4) or 5 seconds`)

If the Setup Request must be rejected (due to any of the reasons in the Command response codes listed below), a Setup Response SHALL be sent back to the client with a corresponding command response value indicating the reason for the rejection.

```

uint16_t controlId;    // Control ID = 0xACE1
uint16_t protocolVer;  // Protocol version = 0x08
uint8_t cmdRequest;    // Command request = 2 (reply)
uint8_t cmdResponse;   // Command response = <see table below>
uint16_t reserved1;    // Reserved (alignment)
uint16_t testPort;     // Test port on server (available port in Response
)

uint8_t jumboStatus;   // Jumbo datagram support status (BOOL)
uint8_t authMode;      // Authentication mode
uint32_t authUnixTime; // Authentication time stamp
unsigned char authDigest[AUTH_DIGEST_LENGTH] // 32 octets, MBZ

```

Command Response Codes

Control Header Setup Request Code CHSR_CRSP_NONE	0 = None
Control Header Setup Request Code CHSR_CRSP_ACKOK	1 = Acknowledgement
Control Header Setup Request Code CHSR_CRSP_BADVER	2 = Bad Protocol Version
Control Header Setup Request Code CHSR_CRSP_BADJS	3 = Invalid Jumbo datagram option
Control Header Setup Request Code CHSR_CRSP_AUTHNC	4 = Unexpected Authentication in Setup Request
Control Header Setup Request Code CHSR_CRSP_AUTHREQ	5 = Authentication missing in Setup Request
Control Header Setup Request Code CHSR_CRSP_AUTHINV	6 = Invalid authentication method
Control Header Setup Request Code CHSR_CRSP_AUTHFAIL	7 = Authentication failure
Control Header Setup Request Code CHSR_CRSP_AUTHTIME	8 = Authentication time is invalid in Setup Request

@@@ To Do: How do we communicate multiple errors when the server sends the Setup Response? Is an error hierarchy possible, where Bad Protocol Version means that none of the other aspects (higher error numbers) were checked? New text to address this issue appears below:

There is a hierarchy of Command Response codes, beginning with: "2 = Bad Protocol Version", which SHALL be checked first because it is a fixed field length and the most reliable check. The server SHOULD communicate the first error condition detected in the order listed below:

```

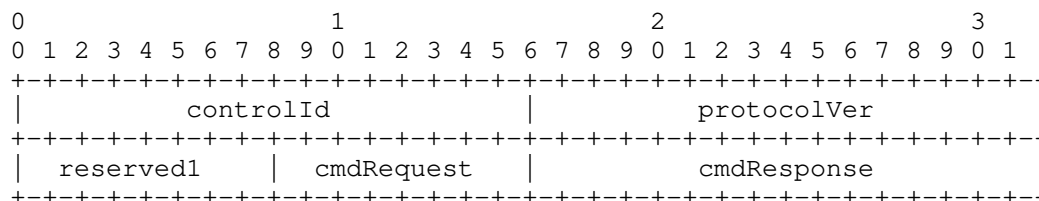
3 = Invalid Jumbo datagram option
5 = Authentication missing in Setup Request
4 = Unexpected Authentication in Setup Request
6 = Invalid authentication method (SHA-256 not used)
7 = Authentication failure (both shared secret and time)
8 = Authentication time is invalid in Setup Request (replay attack)

```

The only circumstance when a server would not communicate the appropriate Command Response Code for an error condition above is when an attack has been detected, in which case the server will allow setup attempts with errors to terminate silently.

@@@ Or, if multiple error codes are wanted, would a flag system work better? For an expanded field multiple error codes, we could use decimal values that set 1 bit in the cmdResponse (0,1,2,4,8,...) for

each code, and expand the cmdResponse field to 16 bits by using the nearby reserved field as shown below (and moving the fields within the 32-bit word):



@@@ - end text for discussion -

If the server finds that the Setup Request matches its configuration and is otherwise acceptable, the server SHALL initiate a new connection for the client, using a new UDP socket allocated from the UDP ephemeral port range. Then, the server SHALL start a watchdog timer (to terminate the connection in case the client goes silent), and sends the Setup Response back to the client (see below for composition).

If the Setup Request is accepted by the server, a Setup Response SHALL be sent back to the client with a corresponding command response value indicating 1 = Acknowledgement.

```

uint16_t controlId;    // Control ID = 0xACE1
uint16_t protocolVer;  // Protocol version = 0x08
uint8_t cmdRequest;    // Command request = 2 (reply)
uint8_t cmdResponse;   // Command response = 1 (Acknowledgement)
uint16_t reserved1;    // Reserved (alignment)
uint16_t testPort;     // Test port on server (available port in Response
)
uint8_t jumboStatus;   // Jumbo datagram support status (BOOL)
uint8_t authMode;      // Authentication mode
uint32_t authUnixTime; // Authentication time stamp
unsigned char authDigest[AUTH_DIGEST_LENGTH] // 32 octets, MBZ
...

```

The new connection is associated with a new UDP socket allocated from the UDP ephemeral port range at the server. The server SHALL set a timer for the new connection as a watchdog (in case the client goes quiet) and send the Setup response back to the client.

(Note: in version 8, the watchdog time-out is configured at 5 seconds)

The Setup Response SHALL include the port number at the server for the new socket, and this UDP port-pair SHALL be used for all

subsequent communication. The server SHALL also include the values of:

- o Jumbo datagram support status (BOOL),
- o Authentication mode, and
- o Authentication time stamp

for the client's use on the new connection in its Setup Response, and the remaining 32 octets MUST Be Zero (MBZ).

Finally, the new UDP connection associated with the new socket and port number is opened, and the server awaits communication there.

If a Test Activation request is not subsequently received from the client on this new port number before the watchdog timer expires, the server SHALL close the socket and deallocate the port.

5.1. Setup Response Processing at the Client

When the client receives the Setup response from the server it first checks the cmdResponse value. If this value indicates an error the client SHALL display/report a relevant message to the user or management process and exit. If the client receives a Command Response code (CRSP) that is not equal to one of the codes defined above, then the client MUST terminate the connection and terminate operation of the current Setup Request. If the Command Response code (CRSP) value indicates success the client SHALL compose a Test Activation Request with all the test parameters it desires such as the test direction, the test duration, etc.

6. Test Activation Request and Response

This section is divided according to the sending and processing of the client, server, and again at the client.

All messages defined in this section SHALL use UDP transport. The hosts SHALL calculate and include the UDP checksum, or check the UDP checksum as necessary.

6.1. Test Activation Request at the client

Upon a successful setup, the client SHALL then send the Test Activation Request to the UDP port number the server communicated in the Setup Response.

@@@ To Do: Add Options for UDP payload content (beyond the Test PDU), such as all zeroes, all ones, alternating one and zero, and pseudo-random.

The client SHALL compose Test Activation Request as follows:

```

uint16_t controlId;    // Control ID
uint16_t protocolVer; // Protocol version
uint8_t cmdRequest;    // Command request, 1 = upstream, 2 = downstream
uint8_t cmdResponse;   // Command response (set to 0)
uint16_t lowThresh;    // Low delay variation threshold
uint16_t upperThresh;  // Upper delay variation threshold
uint16_t trialInt;     // Status feedback/trial interval (ms)
uint16_t testIntTime;  // Test interval time (sec)
uint8_t subIntPeriod;  // Sub-interval period (sec)
uint8_t ipTosByte;     // IP ToS byte for testing
uint16_t srIndexConf;  // Configured sending rate index (see Note below)
uint8_t useOwDelVar;   // Use one-way delay instead of RTT
uint8_t highSpeedDelta; // High-speed row adjustment delta
uint16_t slowAdjThresh; // Slow rate adjustment threshold
uint16_t seqErrThresh; // Sequence error threshold
uint8_t ignoreOooDup;  // Ignore Out-of-Order/Duplicate datagrams
uint8_t reserved1;     // (Alignment)
uint16_t reserved2;    // (Alignment)

```

Control Header Test Activation Command Request Values:

```

CHTA_CREQ_NONE      0 = No Request
CHTA_CREQ_TESTACTUS 1 = Request test in Upstream direction (client to server, client takes the role of sending test packets)
CHTA_CREQ_TESTACTDS 2 = Request test in Downstream direction (server to client, client takes the role of receiving test packets)

```

Control Header Test Activation Command Response Values:

```

CHTA_CRSP_NONE      0 = Used by client when making a Request
CHTA_CRSP_ACKOK      1 = Used by Server in affirmative Response
CHTA_CRSP_BADPARAM 2 = Used by Server to indicate an error; bad parameter; reject;

```

Note: uint16_t srIndexConf is the table index of the configured *fixed* sending rate index to use. The client can request the specified rate, or the server can use this field to coerce a maximum rate in its response. If the server sets to 0 in its response, client SHALL not use fixed rate.

The UDP PDU format of the Test Activation Request is as follows (big-endian AB):

0										1										2										3																																							
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1																																						
-----										controlId										-----										protocolVer										-----																													
-----										cmdRequest										-----										cmdResponse										-----										lowThresh										-----									
-----										upperThresh										-----										trialInt										-----																													
-----										testIntTime										-----										subIntPeriod										-----										ipTosByte										-----									
-----										srIndexConf										-----										useOwDelVar										-----										highSpeedDelta										-----									
-----										slowAdjThresh										-----										seqErrThresh										-----																													
-----										ignoreOooDup										-----										reserved1										-----										reserved2										-----									
-----																				-----																				-----																													

Note: This is only 28 octets of the 56 octet PDU sent, the rest are MBZ for a Test Activation Request.

The client SHALL use the configuration for

- o Jumbo datagram support status (BOOL),
- o Authentication mode, and
- o Authentication time stamp

requested and confirmed by the server.

6.2. Test Activation Request - server response

After the server receives the Test Activation request on the new connection, it MUST choose to accept, ignore or modify any of the test parameters.

When the server sends the Test Activation response back, it SHALL set the cmd Response field to:

```
uint8_t cmdResponse; // Command response (set to 1, ACK, or 2 error)
```

The server SHALL include all the test parameters again to make the client aware of any changes.

If the client has requested an upstream test, the server SHALL include the transmission parameters from the first row of the sending rate table.

The remaining 28 octets of the Test Activation Request (normally read from the first row of the sending rate table) are called the Sending Rate Structure, and SHALL be organized as follows:

```
uint32_t txIntervall; // Transmit interval (us)
uint32_t udpPayload1; // UDP payload (bytes)
uint32_t burstSize1; // UDP burst size per interval
uint32_t txInterval2; // Transmit interval (us)
uint32_t udpPayload2; // UDP payload (bytes)
uint32_t burstSize2; // UDP burst size per interval
uint32_t udpAddon2; // UDP add-on (bytes)
```

with

```
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     txIntervall1                             |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     udpPayload1                             |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     burstSize1                             |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     txInterval2                             |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     udpPayload2                             |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     burstSize2                             |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     udpAddon2                             |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
```

Note that the server additionally has the option of completely rejecting the request and sending back an appropriate command response value:

```
uint8_t cmdResponse; // Command response (set to 2, error)
```

If activation continues, the new connection is prepared for an upstream OR downstream test.

In the case of a downstream test, the server prepares to send with either a single timer to send status PDUs at the specified interval OR dual timers to send load PDUs based on the first row of the sending rate table.

The server SHALL then send a Test Activation response back to the client, update the watchdog timer with a new time-out value, and set a test duration timer to eventually stop the test.

The new connection is now ready for testing.

6.3. Test Activation Response action at the client

When the client receives the Test Activation response, it first checks the command response value.

If the client receives a Test Activation command response value that indicates an error, the client SHALL display/report a relevant message to the user or management process and exit.

If the client receives a Test Activation command response value that is not equal to one of the codes defined above, then the client MUST terminate the connection and terminate operation of the current Setup Request.

If the client receives a Test Activation command response value that indicates success (CHTA_CRSP_ACKOK) the client SHALL update its configuration to use any test parameters modified by the server.

Next, the client SHALL prepare its connection for either an upstream test with dual timers set to send load PDUs (based on the starting transmission parameters sent by the server), OR a downstream test with a single timer to send status PDUs at the specified interval.

Then, the client SHALL stop the test initiation timer, set a new time-out value for the watchdog timer, and start the timer (in case the server goes quiet).

The connection is now ready for testing.

7. Test Stream Transmission and Measurement Feedback Messages

This section describes the testing phase of the protocol. The roles of sender and receiver vary depending whether the direction of testing is from server to client, or the reverse.

All messages defined in this section SHALL use UDP transport. The hosts SHALL calculate and include the UDP checksum, or check the received UDP checksum before further processing, as necessary.

7.1. Test Packet PDU and Roles

Testing proceeds with one end point sending load PDUs, based on transmission parameters from the sending rate table, and the other end point receiving the load PDUs and sending status messages to communicate the traffic conditions at the receiver.

The watchdog timer at the receiver SHALL be reset each time a test PDU is received. See non-graceful test stop in Section 8 for handling the watchdog/NOTRAFFIC time-out expiration at each end-point.

When the server is sending Load PDUs in the role of sender, it SHALL use the transmission parameters directly from the sending rate table via the index that is currently selected (which was based on the feedback in its received status messages).

However, when the client is sending load PDUs in the role of sender, it SHALL use the discreet transmission parameters that were communicated by the server in its periodic status messages (and not referencing a sending rate table). This approach allows the server to control the individual sending rates as well as the algorithm used to decide when and how to adjust the rate.

The server uses a load adjustment algorithm which evaluates measurements, either it's own or the contents of received feedback messages. This algorithm is unique to udpst; it provides the ability to search for the Maximum IP Capacity that is absent from other testing tools. Although the algorithm depends on the protocol, it is not part of the protocol per se.

The current algorithm has three paths to its decision on the next sending rate:

1. When there are no impairments present (no sequence errors, low delay variation), resulting in sending rate increase.
2. When there are low impairments present (no sequence errors but higher levels of delay variation), so the same sending rate is retained.
3. When the impairment levels are above the thresholds set for this purpose and "congestion" is inferred, resulting in sending rate decrease.

The algorithm also has two modes for increasing/decreasing the sending rate:

- o A high-speed mode to achieve high sending rates quickly, but also back-off quickly when "congestion" is inferred from the measurements. Any two consecutive feedback intervals that have a sequence number anomaly and/or contain an upper delay variation threshold exception in both of the two consecutive intervals, count as the two consecutive feedback measurements required to declare "congestion" within a test.

- o A single-step mode where all rate adjustments use the minimum increase or decrease of one step in the sending rate table. The single step mode continues after the first inference of "congestion" from measured impairments.

On the other hand, the test configuration MAY use a fixed sending rate requested by the client, using the field below:

```
uint16_t srIndexConf; // Configured sending rate index
```

The client MAY communicate the desired fixed rate in it's activation request.

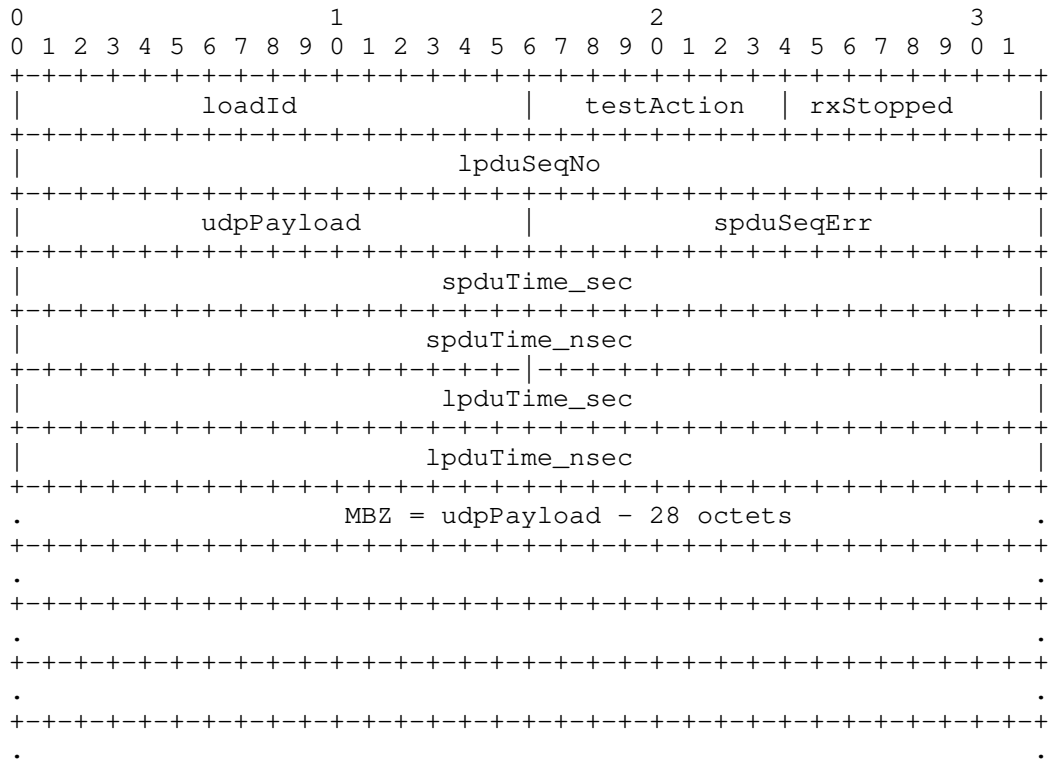
The Load PDU SHALL have the following format and field definitions:

```
uint16_t loadId; // Load ID (=0xBEEF for the LLoad PDU)
uint8_t testAction; // Test action (= 0x00 normally, until test stop)
uint8_t rxStopped; // Receive traffic stopped indicator (BOOL)
uint32_t lpduSeqNo; // Load PDU sequence number (starts at 1)
uint16_t udpPayload; // UDP payload LENGTH(bytes)
uint16_t spduSeqErr; // Status PDU sequence error count
//
uint32_t spduTime_sec; // Send time in last received status PDU
uint32_t spduTime_nsec; // Send time in last received status PDU
uint32_t lpduTime_sec; // Send time of this load PDU
uint32_t lpduTime_nsec; // Send time of this load PDU
```

Test Action Codes

```
TEST_ACT_TEST 0 // normal
TEST_ACT_STOP1 1 // normal stop at end of test: server sends in STATUS or Test P
DU
TEST_ACT_STOP2 2 // ACK of STOP1: sent by client in STATUS or Test PDU
```

The Test Load UDP PDU format is as follows (big-endian AB):



7.2. Status PDU

The receiver SHALL send a Status PDU to the sender during a test at the configured feedback interval.

The watchdog timer at the test PDU sender SHALL be reset each time a Status PDU is received. See non-graceful test stop in Section 8 for handling the watchdog/NOTRAFFIC time-out expiration at each end-point.

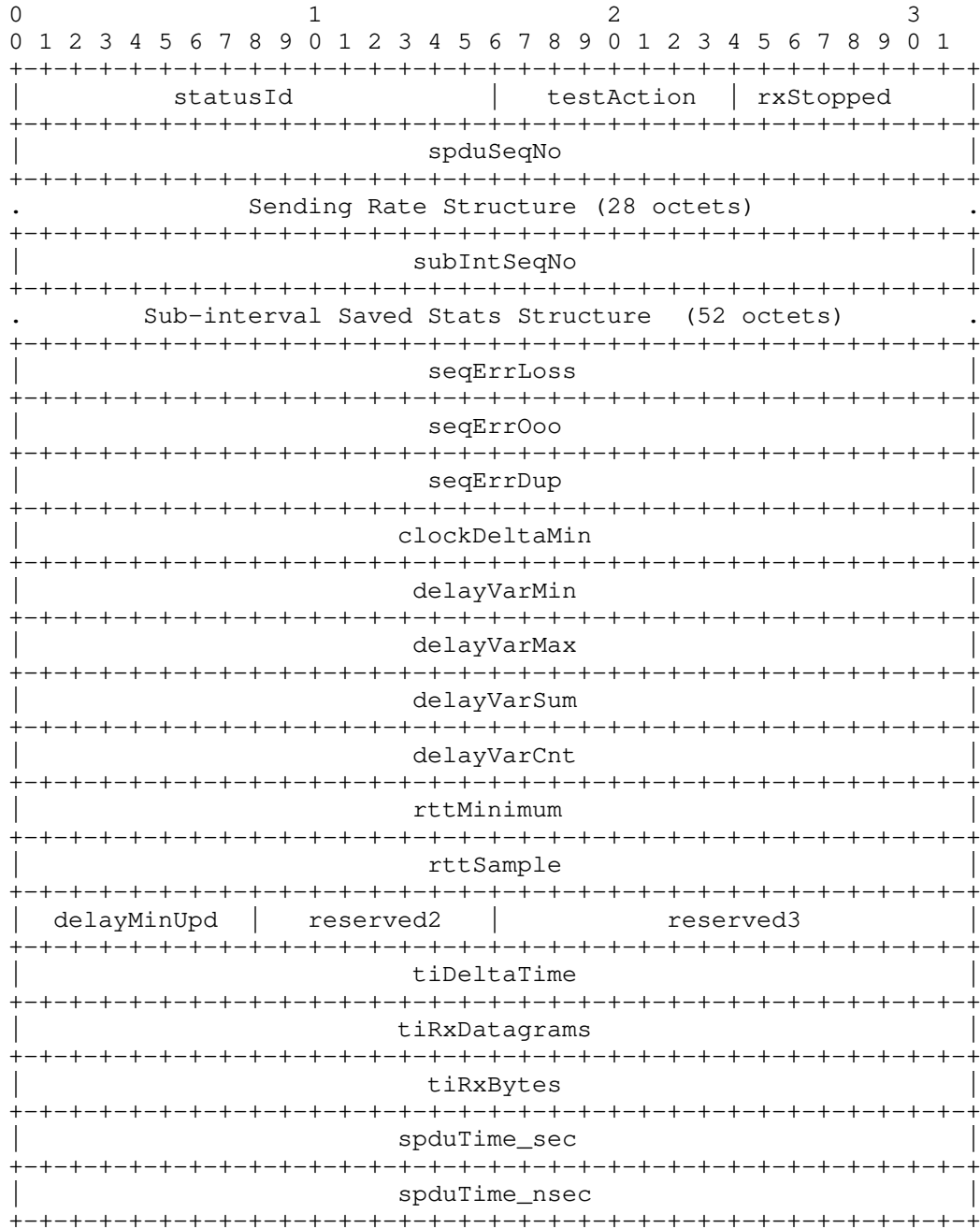
@@@ To Do: What protections from bit errors (checksum) or on-path attacks (something stronger) are warrented for teh Status PDUs? These PDUs are a key part of the server-client control loop. Added a requirement to calculate and include/check the UDP checksum.

The Status Header PDU SHALL have the following format and field definitions:

```
// Status feedback header for UDP payload of status PDUs
//
```

```
    uint16_t statusId; // Status ID = 0xFEED
    uint8_t testAction; // Test action
    uint8_t rxStopped; // Receive traffic stopped indicator (BOOL)
    uint32_t spduSeqNo; // Status PDU sequence number (starts at 1)
    //
    struct sendingRate srStruct; // Sending Rate Structure (28 octets)
    //
    uint32_t subIntSeqNo; // Sub-interval sequence number
    struct subIntStats sisSav; // Sub-interval Saved Stats Structure (52 oct
ets)
    //
    uint32_t seqErrLoss; // Loss sum
    uint32_t seqErrOoo; // Out-of-Order sum
    uint32_t seqErrDup; // Duplicate sum
    //
    uint32_t clockDeltaMin; // Clock delta minimum (either RTT or 1-way delay
)
    uint32_t delayVarMin; // Delay variation minimum
    uint32_t delayVarMax; // Delay variation maximum
    uint32_t delayVarSum; // Delay variation sum
    uint32_t delayVarCnt; // Delay variation count
    uint32_t rttMinimum; // Minimum round-trip time sampled
    uint32_t rttSample; // Last round-trip time sample
    uint8_t delayMinUpd; // Delay minimum(s) updated observed, communicate
d in both directions.
    uint8_t reserved2; // (alignment)
    uint16_t reserved3; // (alignment)
    //
    uint32_t tiDeltaTime; // Trial interval delta time
    uint32_t tiRxDatagrams; // Trial interval receive datagrams
    uint32_t tiRxBytes; // Trial interval receive bytes
    //
    uint32_t spduTime_sec; // Send time of this status PDU
    uint32_t spduTime_nsec; // Send time of this status PDU
```

The Status feedback UDP payload PDUs format is as follows (big-endian AB):



Note that the Sending Rate Structure (28 octets) is defined in the Test Activation section.

Also note that the Sub-interval Saved Stats Structure (52 octets) SHALL be included (and populated as required when the server is in the receiver role) as defined below.

The Sub-interval saved statistics structure for received traffic measurements SHALL be organized and formatted as follows:

```

uint32_t rxDatagrams; // Received datagrams
uint32_t rxBytes;     // Received bytes
uint32_t deltaTime;   // Time delta
uint32_t seqErrLoss;   // Loss sum
uint32_t seqErrOoo;   // Out-of-Order sum
uint32_t seqErrDup;    // Duplicate sum
uint32_t delayVarMin;  // Delay variation minimum
uint32_t delayVarMax;  // Delay variation maximum
uint32_t delayVarSum;  // Delay variation sum
uint32_t delayVarCnt;  // Delay variation count
uint32_t rttMinimum;   // Minimum round-trip time
uint32_t rttMaximum;   // Maximum round-trip time
uint32_t accumTime;    // Accumulated time

```

0										1										2										3									
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9
rxDatagrams																																							
rxBytes																																							
deltaTime																																							
seqErrLoss																																							
seqErrOoo																																							
seqErrDup																																							
delayVarMin																																							
delayVarMax																																							
delayVarSum																																							
delayVarCnt																																							
rttMinimum																																							
rttMaximum																																							
accumTime																																							

Note that the 52 octet saved statistics structure above has slight differences from the 40 octets that follow in the status feedback PDU, particularly the time-related fields.

Upon receiving the Status Feedback PDU or expiration of the feedback interval, the server SHALL perform calculations required by the Load adjustment algorithm and adjust its sending rate, or signal that the client do so in its role as as sender.

@@@ To Do: Additional measurements, like interface byte counters from a client at a residential gateway, would change the Status Feedback PDU (and the protocol version number as a result). Interface byte counters seem useful for specific circumstances, such as when the client application has acces to an interface that sees all traffic to/from a service subscriber's location.

8. Stopping the Test

When the test duration timer on the server expires, it SHALL set the connection test action to STOP and also starts marking all outgoing load or status PDUs with a test action of STOP1.

```
uint8_t testAction; // Test action (server sets STOP1)
```

This is simply a non-reversible state for all future messages sent from the server.

When the client receives a load or status PDU with the STOP1 indication, it SHALL finalize testing, display the test results, and also mark its connection with a test action of STOP (so that any PDUs received subsequent to the STOP1 are ignored).

With the test action of the client's connection set to STOP, the very next expiry of a send timer for either a load or status PDU SHALL cause the client to schedule an immediate end time to exit.

The client SHALL then send all subsequent load or status PDUs with a test action of STOP2

```
uint8_t testAction; // Test action (client sets STOP2)
```

as confirmation to the server, and a graceful termination of the test can begin.

When the server receives the STOP2 confirmation in the load or status PDU, the server SHALL schedule an immediate end time for the connection which closes the socket and deallocates it.

In a non-graceful test stop, the watchdog/NOTRAFFIC time-outs at each end-point will expire (sometimes at one end-point first), notifications in logs, STDOUT, and/or formateed output SHALL be made,

and the test action of each end-point's connection SHALL be set to STOP.

9. Method of Measurement

The architecture of the method REQUIRES two cooperating hosts operating in the roles of Src (test packet sender) and Dst (receiver), with a measured path and return path between them.

The duration of a test duration, parameter I, MUST be constrained in a production network, since this is an active test method and it will likely cause congestion on the Src to Dst host path during a test.

9.1. Running Code

This section is for the benefit of the Document Shepherd's form, and will be deleted prior to final review.

Much of the development of the method and comparisons with existing methods conducted at IETF Hackathons and elsewhere have been based on the example udpst Linux measurement tool (which is a working reference for further development) [udpst]. The current project:

- o is a utility that can function as a client or server daemon
- o requires a successful client-initiated setup handshake between cooperating hosts and allows firewalls to control inbound unsolicited UDP which either go to a control port [expected and w/ authentication] or to ephemeral ports that are only created as needed. Firewalls protecting each host can both continue to do their job normally. This aspect is similar to many other test utilities available.
- o is written in C, and built with gcc (release 9.3) and its standard run-time libraries
- o allows configuration of most of the parameters described in Sections 4 and 7.
- o supports IPv4 and IPv6 address families.
- o supports IP-layer packet marking.

10. Security Considerations

Active metrics and measurements have a long history of security considerations. The security considerations that apply to any active

measurement of live paths are relevant here. See [RFC4656] and [RFC5357].

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

There are some new considerations for Capacity measurement as described in this memo.

1. Cooperating source and destination hosts and agreements to test the path between the hosts are REQUIRED. Hosts perform in either the Src or Dst roles.
2. It is REQUIRED to have a user client-initiated setup handshake between cooperating hosts that allows firewalls to control inbound unsolicited UDP traffic which either goes to a control port [expected and w/authentication] or to ephemeral ports that are only created as needed. Firewalls protecting each host can both continue to do their job normally.
3. Client-server authentication and integrity protection for feedback messages conveying measurements is RECOMMENDED. To accomodate different host limitations and testing circumstances, different modes of operation are recommended:

A. Unauthenticated mode (for all phases)

AND

B. OPTIONAL Authenticated set-up only
SHA-256 HMAC time-window verification (5 min time stamp verification)
(could add silent failure option)

C. Encrypted setup and test-activation
(currently using OpenSSL Library, so KISS, but may be too slow for
test packets)

----- Old/lowpower host performance impacts -----

D. Encrypted feedback messages (maybe split into Integrity and encrypt?)

E. Integrity protection for test packets SHA-256 HMAC

F. Encrypted test packets (maybe also valuable to defeat compression on links)

4. Hosts MUST limit the number of simultaneous tests to avoid resource exhaustion and inaccurate results.
5. Senders MUST be rate-limited. This can be accomplished using a pre-built table defining all the offered load rates that will be supported (Section 8.1). The recommended load-control search algorithm results in "ramp up" from the lowest rate in the table.
6. Service subscribers with limited data volumes who conduct extensive capacity testing might experience the effects of Service Provider controls on their service. Testing with the Service Provider's measurement hosts SHOULD be limited in frequency and/or overall volume of test traffic (for example, the range of I duration values SHOULD be limited).

The exact specification of these features was hopefully accomplished during this protocol development.

11. IANA Considerations

This memo requests IANA to assign a UDP port.

12. Acknowledgments

Thanks to Ruediger Geib, Lincoln Lavoie, Can Desem, and Greg Mirsky for reviewing this draft and providing helpful suggestions and areas for further development.

13. References

13.1. Normative References

- [I-D.ietf-ippm-capacity-metric-method]
Morton, A., Geib, R., and L. Ciavattone, "Metrics and Methods for One-way IP Capacity", draft-ietf-ippm-capacity-metric-method-12 (work in progress), June 2021.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC2330] Paxson, V., Almes, G., Mahdavi, J., and M. Mathis, "Framework for IP Performance Metrics", RFC 2330, DOI 10.17487/RFC2330, May 1998, <<https://www.rfc-editor.org/info/rfc2330>>.
- [RFC2681] Almes, G., Kalidindi, S., and M. Zekauskas, "A Round-trip Delay Metric for IPPM", RFC 2681, DOI 10.17487/RFC2681, September 1999, <<https://www.rfc-editor.org/info/rfc2681>>.
- [RFC6438] Carpenter, B. and S. Amante, "Using the IPv6 Flow Label for Equal Cost Multipath Routing and Link Aggregation in Tunnels", RFC 6438, DOI 10.17487/RFC6438, November 2011, <<https://www.rfc-editor.org/info/rfc6438>>.
- [RFC7497] Morton, A., "Rate Measurement Test Protocol Problem Statement and Requirements", RFC 7497, DOI 10.17487/RFC7497, April 2015, <<https://www.rfc-editor.org/info/rfc7497>>.
- [RFC7680] Almes, G., Kalidindi, S., Zekauskas, M., and A. Morton, Ed., "A One-Way Loss Metric for IP Performance Metrics (IPPM)", STD 82, RFC 7680, DOI 10.17487/RFC7680, January 2016, <<https://www.rfc-editor.org/info/rfc7680>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

- [RFC8468] Morton, A., Fabini, J., Elkins, N., Ackermann, M., and V. Hegde, "IPv4, IPv6, and IPv4-IPv6 Coexistence: Updates for the IP Performance Metrics (IPPM) Framework", RFC 8468, DOI 10.17487/RFC8468, November 2018, <<https://www.rfc-editor.org/info/rfc8468>>.

13.2. Informative References

- [copycat] Edleine, K., Kuhlewind, K., Trammell, B., and B. Donnet, "copycat: Testing Differential Treatment of New Transport Protocols in the Wild (ANRW '17)", July 2017, <<https://irtf.org/anrw/2017/anrw17-final5.pdf>>.
- [LS-SG12-A] 12, I. S., "LS - Harmonization of IP Capacity and Latency Parameters: Revision of Draft Rec. Y.1540 on IP packet transfer performance parameters and New Annex A with Lab Evaluation Plan", May 2019, <<https://datatracker.ietf.org/liaison/1632/>>.
- [LS-SG12-B] 12, I. S., "LS on harmonization of IP Capacity and Latency Parameters: Consent of Draft Rec. Y.1540 on IP packet transfer performance parameters and New Annex A with Lab & Field Evaluation Plans", March 2019, <<https://datatracker.ietf.org/liaison/1645/>>.
- [RFC2544] Bradner, S. and J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices", RFC 2544, DOI 10.17487/RFC2544, March 1999, <<https://www.rfc-editor.org/info/rfc2544>>.
- [RFC3148] Mathis, M. and M. Allman, "A Framework for Defining Empirical Bulk Transfer Capacity Metrics", RFC 3148, DOI 10.17487/RFC3148, July 2001, <<https://www.rfc-editor.org/info/rfc3148>>.
- [RFC4656] Shalunov, S., Teitelbaum, B., Karp, A., Boote, J., and M. Zekauskas, "A One-way Active Measurement Protocol (OWAMP)", RFC 4656, DOI 10.17487/RFC4656, September 2006, <<https://www.rfc-editor.org/info/rfc4656>>.
- [RFC5136] Chimento, P. and J. Ishac, "Defining Network Capacity", RFC 5136, DOI 10.17487/RFC5136, February 2008, <<https://www.rfc-editor.org/info/rfc5136>>.

- [RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, DOI 10.17487/RFC5357, October 2008, <<https://www.rfc-editor.org/info/rfc5357>>.
- [RFC6815] Bradner, S., Dubray, K., McQuaid, J., and A. Morton, "Applicability Statement for RFC 2544: Use on Production Networks Considered Harmful", RFC 6815, DOI 10.17487/RFC6815, November 2012, <<https://www.rfc-editor.org/info/rfc6815>>.
- [RFC7312] Fabini, J. and A. Morton, "Advanced Stream and Sampling Framework for IP Performance Metrics (IPPM)", RFC 7312, DOI 10.17487/RFC7312, August 2014, <<https://www.rfc-editor.org/info/rfc7312>>.
- [RFC7594] Eardley, P., Morton, A., Bagnulo, M., Burbridge, T., Aitken, P., and A. Akhter, "A Framework for Large-Scale Measurement of Broadband Performance (LMAP)", RFC 7594, DOI 10.17487/RFC7594, September 2015, <<https://www.rfc-editor.org/info/rfc7594>>.
- [RFC7799] Morton, A., "Active and Passive Metrics and Methods (with Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799, May 2016, <<https://www.rfc-editor.org/info/rfc7799>>.
- [RFC8337] Mathis, M. and A. Morton, "Model-Based Metrics for Bulk Transport Capacity", RFC 8337, DOI 10.17487/RFC8337, March 2018, <<https://www.rfc-editor.org/info/rfc8337>>.
- [RFC8762] Mirsky, G., Jun, G., Nydell, H., and R. Foote, "Simple Two-Way Active Measurement Protocol", RFC 8762, DOI 10.17487/RFC8762, March 2020, <<https://www.rfc-editor.org/info/rfc8762>>.
- [RFC8972] Mirsky, G., Min, X., Nydell, H., Foote, R., Masputra, A., and E. Ruffini, "Simple Two-Way Active Measurement Protocol Optional Extensions", RFC 8972, DOI 10.17487/RFC8972, January 2021, <<https://www.rfc-editor.org/info/rfc8972>>.
- [TR-471] Morton, A., "Broadband Forum TR-471: IP Layer Capacity Metrics and Measurement", July 2020, <<https://www.broadband-forum.org/technical/download/TR-471.pdf>>.

- [udpst] udpst Project Collaborators, "UDP Speed Test Open Broadband project", December 2020, <<https://github.com/BroadbandForum/obudpst>>.
- [Y.1540] Y.1540, I. R., "Internet protocol data communication service - IP packet transfer and availability performance parameters", December 2019, <<https://www.itu.int/rec/T-REC-Y.1540-201912-I/en>>.
- [Y.Sup60] Morton, A., "Recommendation Y.Sup60, (09/20) Interpreting ITU-T Y.1540 maximum IP-layer capacity measurements", September 2020, <<https://www.itu.int/rec/T-REC-Y.Sup60/en>>.

Authors' Addresses

Len Ciavattone
AT&T Labs
200 Laurel Avenue South
Middletown,, NJ 07748
USA

Email: lencia@att.com

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

Phone: +1 732 420 1571
Fax: +1 732 368 1192
Email: acm@research.att.com

IPPM
Internet-Draft
Intended status: Informational
Expires: 13 November 2022

H. Song
Futurewei Technologies
G. Mirsky
Ericsson
C. Filsfils
A. Abdelsalam
Cisco Systems, Inc.
T. Zhou
Z. Li
Huawei
G. Mishra
Verizon Inc.
J. Shin
SK Telecom
K. Lee
LG U+
12 May 2022

In-Situ OAM Marking-based Direct Export
draft-song-ippm-postcard-based-telemetry-12

Abstract

The document describes a packet-marking variation of the IOAM DEX option, referred to as IOAM Marking. Similar to IOAM DEX, IOAM Marking does not carry the telemetry data in user packets but send the telemetry data through a dedicated packet. Unlike IOAM DEX, IOAM Marking does not require an extra instruction header. IOAM Marking raises some unique issues that need to be considered. This document formally describes the high level scheme and cover the common requirements and issues when applying IOAM Marking in different networks. IOAM Marking is complementary to the other on-path telemetry schemes such as IOAM trace and E2E options.

Status of This Memo

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

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

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

This Internet-Draft will expire on 13 November 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Motivation	3
2. IOAM Marking: Marking-based IOAM Direct Export	3
3. New Challenges	5
4. IOAM Marking Design Considerations	6
4.1. Packet Marking	6
4.2. Flow Path Discovery	7
4.3. Packet Identity for Export Data Correlation	7
4.4. Control the Load	8
5. Implementation Recommendation	8
5.1. Configuration	8
5.2. Postcard Format	8
5.3. Data Correlation	9
6. Use Cases	9
7. Security Considerations	10
8. IANA Considerations	10
9. Contributors	10
10. Acknowledgments	10
11. Informative References	10
Authors' Addresses	12

1. Motivation

To gain detailed data plane visibility to support effective network OAM, it is essential to be able to examine the trace of user packets along their forwarding paths. Such on-path flow data reflect the state and status of each user packet's real-time experience and provide valuable information for network monitoring, measurement, and diagnosis.

The telemetry data include but not limited to the detailed forwarding path, the timestamp/latency at each network node, and, in case of packet drop, the drop location, and the reason. The emerging programmable data plane devices allow user-defined data collection or conditional data collection based on trigger events. Such on-path flow data are from and about the live user traffic, which complements the data acquired through other passive and active OAM mechanisms such as IPFIX [RFC7011] and ICMP [RFC2925].

On-path telemetry was developed to cater to the need of collecting on-path flow data. There are two basic modes for on-path telemetry: the passport mode and the postcard mode. In the passport mode which is represented by IOAM trace option [I-D.ietf-ippm-ioam-data], each node on the path adds the telemetry data to the user packets (i.e., stamp the passport). The accumulated data-trace carried by user packets are exported at a configured end node. In the postcard mode which is represented by IOAM direct export option (DEX) [I-D.ietf-ippm-ioam-direct-export], each node directly exports the telemetry data using an independent packet (i.e., send a postcard) to avoid carrying the data with user packets. The postcard mode is complementary to the passport mode.

IOAM DEX uses an instruction header to explicitly instruct the telemetry data to be collected. This document describes another variation of the postcard mode on-path telemetry, IOAM Marking. Unlike IOAM DEX, IOAM Marking does not require a telemetry instruction header. However, IOAM Marking has unique issues that need to be considered. This document discusses the challenges and their solutions which are common to the high-level scheme of IOAM Marking.

2. IOAM Marking: Marking-based IOAM Direct Export

As the name suggests, IOAM Marking only needs a marking-bit in the existing headers of user packets to trigger the telemetry data collection and export. The sketch of IOAM Marking is as follows. If on-path data need to be collected, the user packet is marked at the path head node. At each IOAM Marking-aware node, if the mark is detected, a postcard (i.e., the dedicated OAM packet triggered by a

marked user packet) is generated and sent to a collector. The postcard contains the data requested by the management plane. The requested data are configured by the management plane. Once the collector receives all the postcards for a single user packet, it can infer the packet's forwarding path and analyze the data set. The path end node is configured to unmark the packets to its original format if necessary.

The overall architecture of IOAM Marking is depicted in Figure 1.

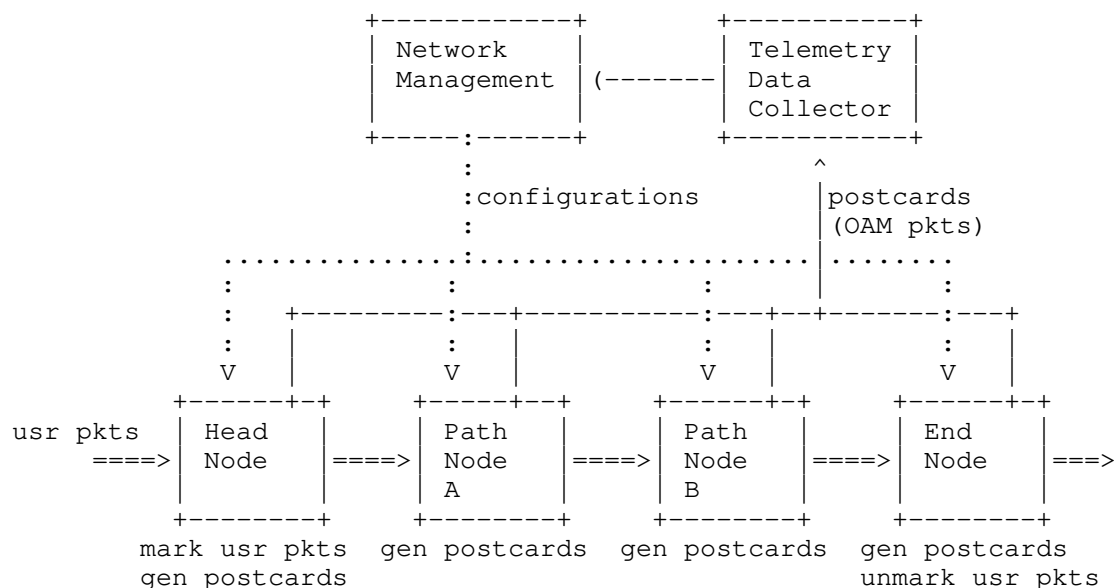


Figure 1: Architecture of IOAM Marking

The advantages of IOAM Marking are summarized as follows.

- * 1: IOAM Marking avoids augmenting user packets with new headers and the signaling for telemetry data collection remains in the data plane.
- * 2: IOAM Marking is extensible for collecting arbitrary new data to support possible future use cases. The data set to be collected can be configured through the management plane or control plane.

- * 3: IOAM Marking can avoid interfering with the normal forwarding. The collected data are free to be transported independently through in-band or out-of-band channels. The data collecting, processing, assembly, encapsulation, and transport are, therefore, decoupled from the forwarding of the corresponding user packets and can be performed in data-plane slow-path if necessary.
- * 4: For IOAM Marking, the types of data collected from each node can vary depending on application requirements and node capability.
- * 5: IOAM Marking makes it easy to secure the collected data without exposing it to unnecessary entities. For example, both the configuration and the telemetry data can be encrypted and/or authenticated before being transported, so passive eavesdropping and a man-in-the-middle attack can both be deterred.
- * 6: Even if a user packet under inspection is dropped at some node in the network, the postcards collected from the preceding nodes are still valid and can be used to diagnose the packet drop location and reason.

3. New Challenges

Although IOAM Marking has some unique features compared to the passport mode telemetry and the instruction-based IOAM DEX, it introduces a few new challenges.

- * Challenge 1 (Packet Marking): A user packet needs to be marked to trigger the path-associated data collection. Since IOAM Marking does not augment user packets with any new header fields, it needs to reserve or reuse bits from the existing header fields. This raises a similar issue as in the Alternate Marking Scheme [RFC8321]
- * Challenge 2 (Configuration): Since the packet header will not carry IOAM instructions anymore, the data plane devices need to be configured to know what data to collect. However, in general, the forwarding path of a flow packet (due to ECMP or dynamic routing) is unknown beforehand (note that there are some notable exceptions, such as segment routing). If the per-flow customized data collection is required, configuring the data set for each flow at all data plane devices might be expensive in terms of configuration load and data plane resources.
- * Challenge 3 (Data Correlation): Due to the variable transport latency, the dedicated postcard packets for a single packet may arrive at the collector out of order or be dropped in networks for

some reason. In order to infer the packet forwarding path, the collector needs some information from the postcard packets to identify the user packet affiliation and the order of path node traversal.

- * Challenge 4 (Load Overhead): Since each postcard packet has its header, the overall network bandwidth overhead of IOAM Marking can be high. A large number of postcards could add processing pressure on data collecting servers. That can be used as an attack vector for DoS.

4. IOAM Marking Design Considerations

To address the above challenges, we propose several design details of IOAM Marking.

4.1. Packet Marking

To trigger the path-associated data collection, usually, a single bit from some header field is sufficient. While no such bit is available, other packet-marking techniques are needed. We discuss several possible application scenarios.

- * IPv4. Alternate Marking (AM) [RFC8321] is an IP flow performance measurement framework that also requires a single bit for packet coloring. The difference is that AM does in-network measurement while IOAM Marking only collects and exports data at network nodes (i.e., the data analysis is done at the collector rather than in the network nodes). AM suggests to use some reserved bit of the Flag field or some unused bit of the TOS field. Actually, AM can be considered a sub-case of IOAM Marking, so that the same bit can be used for IOAM Marking. The management plane is responsible for configuring the actual operation mode.
- * SFC NSH. The OAM bit in the NSH header can be used to trigger the on-path data collection [RFC8300]. IOAM Marking does not add any other metadata to NSH.
- * MPLS. Instead of choosing a header bit, we take advantage of the synonymous flow label [I-D.bryant-mpls-synonymous-flow-labels] approach to mark the packets. A synonymous flow label indicates the on-path data should be collected and forwarded through a postcard.
- * SRv6: A flag bit in SRH can be reserved to trigger the on-path data collection [I-D.song-6man-srv6-pbt]. SRv6 OAM [I-D.ietf-6man-spring-srv6-oam] has adopted the O-bit in SRH flags as the marking bit to trigger the telemetry.

4.2. Flow Path Discovery

In case the path that a flow traverses is unknown in advance, all IOAM Marking-aware nodes should be configured to react to the marked packets by exporting some basic data, such as node ID and TTL before a data set template for that flow is configured. This way, the management plane can learn the flow path dynamically.

If the management plane wants to collect the on-path data for some flow, it configures the head node(s) with a probability or time interval for the flow packet marking. When the first marked packet is forwarded in the network, the IOAM Marking-aware nodes will export the basic data set to the collector. Hence, the flow path is identified. If other data types need to be collected, the management plane can further configure the data set's template to the target nodes on the flow's path. The IOAM Marking-aware nodes collect and export data accordingly if the packet is marked and a data set template is present.

If the flow path is changed for any reason, the new path can be quickly learned by the collector. Consequently, the management plane controller can be directed to configure the nodes on the new path. The outdated configuration can be automatically timed out or explicitly revoked by the management plane controller.

4.3. Packet Identity for Export Data Correlation

The collector needs to correlate all the postcard packets for a single user packet. Once this is done, the TTL (or the timestamp, if the network time is synchronized) can be used to infer the flow forwarding path. The key issue here is to correlate all the postcards for the same user packet.

The first possible approach includes the flow ID plus the user packet ID in the OAM packets. For example, the flow ID can be the 5-tuple IP header of the user traffic, and the user packet ID can be some unique information pertaining to a user packet (e.g., the sequence number of a TCP packet).

If the packet marking interval is large enough, the flow ID is enough to identify a user packet. As a result, it can be assumed that all the exported postcard packets for the same flow during a short time interval belong to the same user packet.

Alternatively, if the network is synchronized, then the flow ID plus the timestamp at each node can also infer the postcard affiliation. However, some errors may occur under some circumstances. For example, two consecutive user packets from the same flows are marked,

but one exported postcard from a node is lost. It is difficult for the collector to decide to which user packet the remaining postcard is related. In many cases, such a rare error has no catastrophic consequence. Therefore it is tolerable.

4.4. Control the Load

IOAM Marking should not be applied to all the packets all the time. It is better to be used in an interactive environment where the network telemetry applications dynamically decide which subset of traffic is under scrutiny. The network devices can limit the packet marking rate through sampling and metering. The postcard packets can be distributed to different servers to balance the processing load.

It is important to understand that the total amount of data exported by IOAM Marking is identical to that of IOAM trace option. The only extra overhead is the packet header of the postcards. In the case of IOAM trace option, it carries the data from each node throughout the path to the end node before exporting the aggregated data. On the other hand, IOAM Marking directly exports local data. The overall network bandwidth impact depends on the network topology and scale, and in some cases IOAM Marking could be more bandwidth efficient.

5. Implementation Recommendation

5.1. Configuration

The head node's ACL should be configured to filter out the target flows for telemetry data collection. Optionally, a flow packet sampling rate or probability could be configured to monitor a subset of the flow packets.

The telemetry data set that should be exported by postcards at each path node could be configured using the data set templates specified, for example, in IPFIX [RFC7011]. In future revisions, we will provide more details.

The IOAM Marking-aware path nodes could be configured to respond or ignore the marked packets.

5.2. Postcard Format

The postcard should use the same data export format as that used by IOAM. [I-D.spiegel-ippm-ioam-rawexport] proposes a raw format that can be interpreted by IPFIX. In future revisions, we will provide more details.

5.3. Data Correlation

Enough information should be included to help the collector to correlate and order the postcards for a single user packet. Section 4.3 provides several possible means. The application scenario and network protocol are important factors to determine the means to use. In future revisions, we will provide details for representative applications.

6. Use Cases

The MPLS Design Team has been investigating extensibility options for the MPLS data plane.

The challenge has been to continue to support existing MPLS architecture, backwards compatibility as well as not excessively increase the depth of the MPLS label stack with a variety of functional SPL labels and NAI indicators similar in concept to the MPLS Entropy label ELI, EL added to the label stack, as well as the MPLS extension headers being in Stack or post stack.

Reference Augmented Forwarding (RAF) [I-D.raszuk-mpls-raf-fwk] utilizes In Stack Data (ISD) with parity to Entropy Label stack {TL,RFI,RFV,AL} and control plane extension to distribute special network actions and forwarding behaviors.

Reference Augmented Forwarding (RAF) keeps the ISD and PSD stack depth in check by using an alternative means of carrying the IOAM data using IGP control plane extension TLV to carry the data to provide In-Situ IOAM on path telemetry using the postcard based telemetry.

The MPLS Design Team may come up with other alternatives to carry IOAM data such as the IGP extension mentioned and maybe other solutions, which will heavily rely on the the postcard based solution.

With Segment Routing SR-MPLS and SRv6 as Maximum SID Depth(MSD) as well as PMTU in SR Policy are critical issues for SR path instantiation by a controller, postcard based telemetry will become a critical solution to ensure that IOAM telemetry can be viable for operators by eliminating IOAM data from being carried in-situ in the SR-TE policy path.

This draft provides a critical optimization that fills the gaps with IOAM DEX related to packet marking triggers using existing mechanisms as well as flow path discovery mechanisms to avoid configuration of on path data plane node complexity and helps mitigate SR MSD and PMTU issues.

7. Security Considerations

Several security issues need to be considered.

- * Eavesdrop and tamper: the postcards can be encrypted and authenticated to avoid such security threats.
- * DoS attack: IOAM Marking can be limited to a single administrative domain. The mark must be removed at the egress domain edge. The node can rate-limit the extra traffic incurred by postcards.

8. IANA Considerations

No requirement for IANA is identified.

9. Contributors

We thank Alfred Morton who provided valuable suggestions and comments helping improve this draft.

10. Acknowledgments

TBD.

11. Informative References

[I-D.bryant-mpls-synonymous-flow-labels]

Bryant, S., Swallow, G., Sivabalan, S., Mirsky, G., Chen, M., and Z. Li, "RFC6374 Synonymous Flow Labels", Work in Progress, Internet-Draft, draft-bryant-mpls-synonymous-flow-labels-01, 4 July 2015, <<https://www.ietf.org/archive/id/draft-bryant-mpls-synonymous-flow-labels-01.txt>>.

[I-D.ietf-6man-spring-srv6-oam]

Ali, Z., Filsfils, C., Matsushima, S., Voyer, D., and M. Chen, "Operations, Administration, and Maintenance (OAM) in Segment Routing Networks with IPv6 Data plane (SRv6)", Work in Progress, Internet-Draft, draft-ietf-6man-spring-srv6-oam-13, 23 January 2022, <<https://www.ietf.org/archive/id/draft-ietf-6man-spring-srv6-oam-13.txt>>.

- [I-D.ietf-ippm-ioam-data]
Brockners, F., Bhandari, S., and T. Mizrahi, "Data Fields for In-situ OAM", Work in Progress, Internet-Draft, draft-ietf-ippm-ioam-data-17, 13 December 2021, <<https://www.ietf.org/archive/id/draft-ietf-ippm-ioam-data-17.txt>>.
- [I-D.ietf-ippm-ioam-direct-export]
Song, H., Gafni, B., Zhou, T., Li, Z., Brockners, F., Bhandari, S., Sivakolundu, R., and T. Mizrahi, "In-situ OAM Direct Exporting", Work in Progress, Internet-Draft, draft-ietf-ippm-ioam-direct-export-07, 13 October 2021, <<https://www.ietf.org/archive/id/draft-ietf-ippm-ioam-direct-export-07.txt>>.
- [I-D.raszuk-mpls-raf-fwk]
Raszuk, R., "Framework of MPLS Reference Augmented Forwarding", Work in Progress, Internet-Draft, draft-raszuk-mpls-raf-fwk-00, 25 April 2022, <<https://www.ietf.org/archive/id/draft-raszuk-mpls-raf-fwk-00.txt>>.
- [I-D.song-6man-srv6-pbt]
Song, H., "Support Postcard-Based Telemetry for SRv6 OAM", Work in Progress, Internet-Draft, draft-song-6man-srv6-pbt-01, 14 October 2019, <<https://www.ietf.org/archive/id/draft-song-6man-srv6-pbt-01.txt>>.
- [I-D.spiegel-ippm-ioam-rawexport]
Spiegel, M., Brockners, F., Bhandari, S., and R. Sivakolundu, "In-situ OAM raw data export with IPFIX", Work in Progress, Internet-Draft, draft-spiegel-ippm-ioam-rawexport-06, 21 February 2022, <<https://www.ietf.org/archive/id/draft-spiegel-ippm-ioam-rawexport-06.txt>>.
- [RFC2925] White, K., "Definitions of Managed Objects for Remote Ping, Traceroute, and Lookup Operations", RFC 2925, DOI 10.17487/RFC2925, September 2000, <<https://www.rfc-editor.org/info/rfc2925>>.
- [RFC7011] Claise, B., Ed., Trammell, B., Ed., and P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information", STD 77, RFC 7011, DOI 10.17487/RFC7011, September 2013, <<https://www.rfc-editor.org/info/rfc7011>>.

- [RFC8300] Quinn, P., Ed., Elzur, U., Ed., and C. Pignataro, Ed.,
"Network Service Header (NSH)", RFC 8300,
DOI 10.17487/RFC8300, January 2018,
<<https://www.rfc-editor.org/info/rfc8300>>.
- [RFC8321] Fioccola, G., Ed., Capello, A., Cociglio, M., Castaldelli,
L., Chen, M., Zheng, L., Mirsky, G., and T. Mizrahi,
"Alternate-Marking Method for Passive and Hybrid
Performance Monitoring", RFC 8321, DOI 10.17487/RFC8321,
January 2018, <<https://www.rfc-editor.org/info/rfc8321>>.

Authors' Addresses

Haoyu Song
Futurewei Technologies
2330 Central Expressway
Santa Clara, 95050,
United States of America
Email: hsong@futurewei.com

Greg Mirsky
Ericsson
Email: gregimirsky@gmail.com

Clarence Filsfils
Cisco Systems, Inc.
Belgium
Email: cfilsfil@cisco.com

Ahmed Abdelsalam
Cisco Systems, Inc.
Italy
Email: ahabdels@cisco.com

Tianran Zhou
Huawei
156 Beiqing Road
Beijing, 100095
P.R. China
Email: zhoutianran@huawei.com

Zhenbin Li
Huawei
156 Beiqing Road
Beijing, 100095
P.R. China
Email: lizhenbin@huawei.com

Gyan Mishra
Verizon Inc.
Email: hayabusagsm@gmail.com

Jongyoon Shin
SK Telecom
South Korea
Email: jongyoon.shin@sk.com

Kyungtae Lee
LG U+
South Korea
Email: coolee@lguplus.co.kr

IPPM Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 9, 2021

X. Min
G. Mirsky
ZTE Corp.
L. Bo
China Telecom
June 7, 2021

Echo Request/Reply for Enabled In-situ OAM Capabilities
draft-xiao-ippm-ioam-conf-state-10

Abstract

This document describes an extension to the echo request/reply mechanisms used in IPv6, MPLS, SFC and BIER 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

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

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

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

This Internet-Draft will expire on December 9, 2021.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Conventions	4
2.1. Requirements Language	4
2.2. Abbreviations	4
3. IOAM Capabilities Formats	5
3.1. IOAM Capabilities Query TLV in the Echo Request	5
3.2. IOAM Capabilities Response TLV in the Echo Reply	6
3.2.1. IOAM Pre-allocated Tracing Capabilities sub-TLV	7
3.2.2. IOAM Incremental Tracing Capabilities sub-TLV	8
3.2.3. IOAM Proof of Transit Capabilities sub-TLV	9
3.2.4. IOAM Edge-to-Edge Capabilities sub-TLV	10
3.2.5. IOAM DEX Capabilities sub-TLV	11
3.2.6. IOAM End-of-Domain sub-TLV	12
4. Operational Guide	13
5. Security Considerations	13
6. IANA Considerations	14
6.1. IOAM SoR Capability Registry	14
6.2. IOAM TSF+TSL Capability Registry	15
7. Acknowledgements	15
8. References	16
8.1. Normative References	16
8.2. Informative References	16
Authors' Addresses	17

1. Introduction

The Data Fields for In-situ OAM (IOAM) [I-D.ietf-ippm-ioam-data] defines data fields that record 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". In contrast, the device which removes the IOAM data container is referred to as the "IOAM decapsulating node". Nodes within the domain that 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 the correct 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. A centralized controller could be used in some IOAM deployments. The IOAM encapsulating node can acquire these IOAM capabilities info from the centralized controller, through, e.g., NETCONF/YANG, PCEP, or BGP. In the IOAM deployment scenario where there is no centralized controller, NETCONF/YANG or IGP may be used for the IOAM encapsulating node to acquire these IOAM capabilities info, however, whether NETCONF/YANG or IGP has some limitations as follows.

- o When NETCONF/YANG is used in this scenario, each IOAM encapsulating node (including the host when it takes the role of an IOAM encapsulating node) needs to implement a NETCONF Client, each IOAM transit node and IOAM decapsulating node (including the host when it takes the role of an IOAM decapsulating node) needs to implement a NETCONF Server, the complexity can be an issue. Furthermore, each IOAM encapsulating node needs to establish NETCONF Connection with each IOAM transit node and IOAM decapsulating node, the scalability can be an issue.
- o When IGP is used in this scenario, the IGP domain and an IOAM domain don't always have the same coverage. For example, when the IOAM encapsulating node or the IOAM decapsulating node is a host, the availability can be an issue. Furthermore, it might be too challenging to reflect IOAM capabilities at the IOAM transit node and/or the IOAM decapsulating node if these are controlled by a local policy depending on the identity of the IOAM encapsulating node.

This document describes an extension to the echo request/reply mechanisms used in IPv6, MPLS, SFC and BIER environments, which can be used within an IOAM domain where no Centralized Controller exists, 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, SFC and BIER 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")
 - o [I-D.ietf-bier-ping] ("BIER Ping and Trace")

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

BIER: Bit Index Explicit Replication

BGP: Border Gateway Protocol

E2E: Edge to Edge

ICMP: Internet Control Message Protocol

IGP: Interior Gateway Protocol

IOAM: In-situ Operations, Administration, and Maintenance

LSP: Label Switched Path

MPLS: Multi-Protocol Label Switching

MBZ: Must Be Zero

MTU: Maximum Transmission Unit

NTP: Network Time Protocol

OAM: Operations, Administration, and Maintenance

PCEP: Path Computation Element (PCE) Communication Protocol

POSIX: Portable Operating System Interface

POT: Proof of Transit

PTP: Precision Time Protocol

SFC: Service Function Chain

TTL: Time to Live

3. IOAM Capabilities Formats

3.1. IOAM Capabilities Query TLV in the Echo Request

In echo request IOAM Capabilities Query uses TLV (Type-Length-Value tuple) which have the following format:

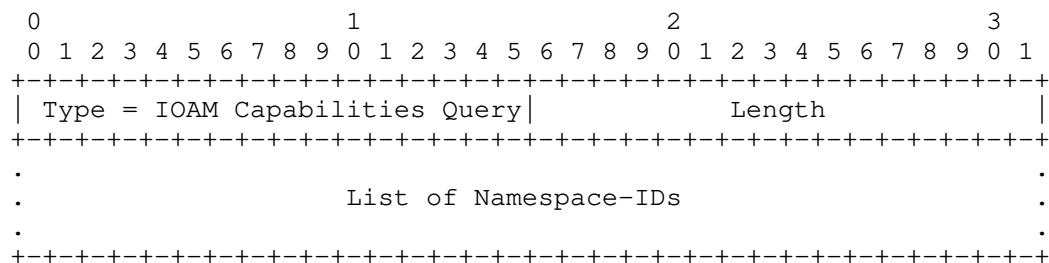


Figure 1: IOAM Capabilities Query TLV in the Echo Request

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 Query TLV. List of Namespace-IDs MAY be included in this TLV of the echo request. In that case, the IOAM encapsulating node requests only the IOAM capabilities that match 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 that identifies it as an IOAM Capabilities Query TLV.

Length is the length of the TLV's Value field in octets, including a List of Namespace-IDs.

Value field of this TLV is zero-padded to align to a 4-octet boundary.

3.2. IOAM Capabilities Response TLV in the Echo Reply

In echo reply IOAM Capabilities Response uses TLV which have the following format:

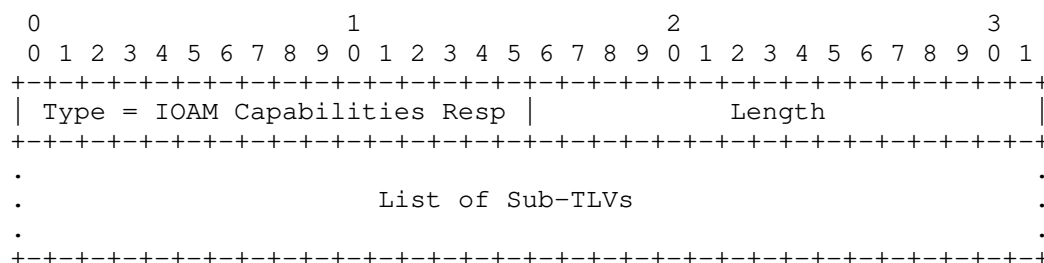


Figure 2: IOAM Capabilities Response TLV in the Echo Reply

When this TLV is present in the echo reply sent by an IOAM transit node and/or an IOAM decapsulating node, it means that the IOAM function is enabled at this node, and this TLV contains the enabled IOAM capabilities of the sender. A list of Sub-TLVs which contains 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 that identifies it as an IOAM Capabilities Response TLV.

Length is the length of the TLV's Value field in octets, including a List of Sub-TLVs.

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] and [I-D.ietf-ippm-ioam-direct-export], six kinds of Sub-TLVs are defined in this document. The same type of the sub-TLV MAY be in the IOAM Capabilities Response TLV more than once only if with a different Namespace-ID.

3.2.1. IOAM Pre-allocated Tracing Capabilities sub-TLV

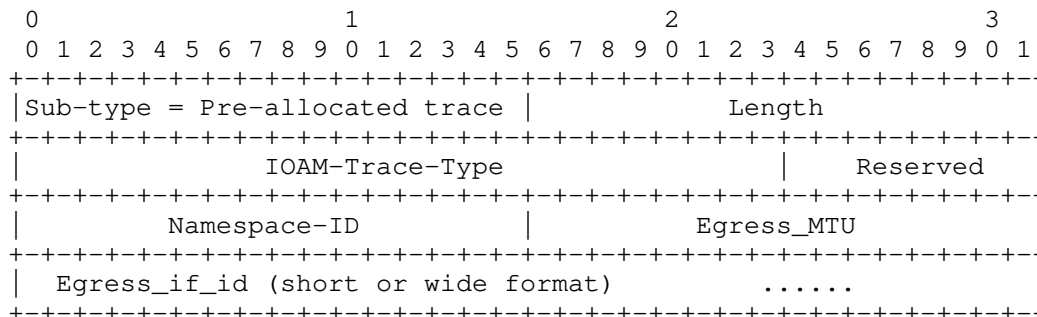


Figure 3: IOAM Pre-allocated Tracing Capabilities Sub-TLV

When this sub-TLV is present in the IOAM Capabilities Response TLV, it means that the sending node is an IOAM transit node and IOAM pre-allocated tracing function is enabled at this IOAM transit node.

Sub-type is set to the value that identifies it as 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. 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 5.4 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 5.3 of [I-D.ietf-ippm-ioam-data], it should be one of the Namespace-IDs listed in the IOAM Capabilities Query 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

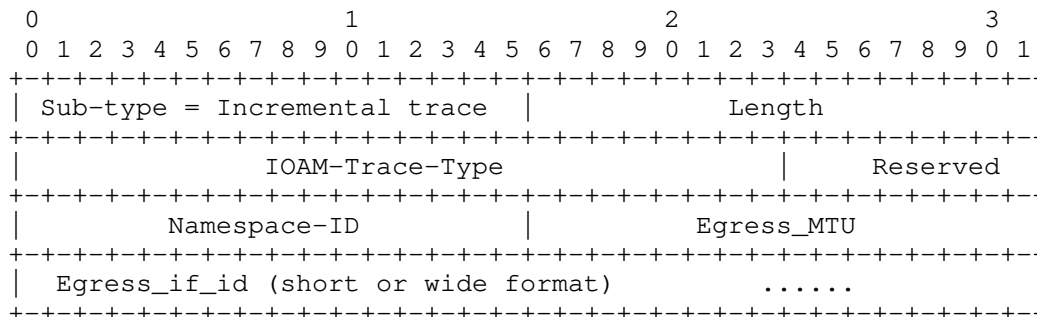


Figure 4: IOAM Incremental Tracing Capabilities Sub-TLV

When this sub-TLV is present in the IOAM Capabilities Response TLV, it means that the sending node is an IOAM transit node and IOAM incremental tracing function is enabled at this IOAM transit node.

Sub-type is set to the value that identifies it as 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. 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 5.4 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 5.3 of [I-D.ietf-ippm-ioam-data], it should be one of the Namespace-IDs listed in the IOAM Capabilities Query 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

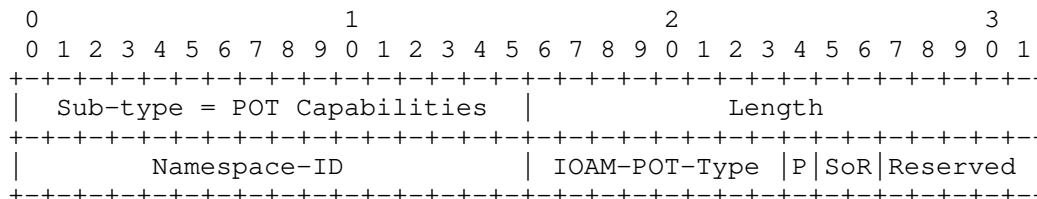


Figure 5: IOAM Proof of Transit Capabilities Sub-TLV

When this sub-TLV is present in the IOAM Capabilities Response 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 that identifies it as 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 5.3 of [I-D.ietf-ippm-ioam-data], it should be one of the Namespace-IDs listed in the IOAM Capabilities Query TLV of echo request.

IOAM-POT-Type field and P bit have the same definition as what's specified in section 5.5 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 that are specified in section 5.5 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

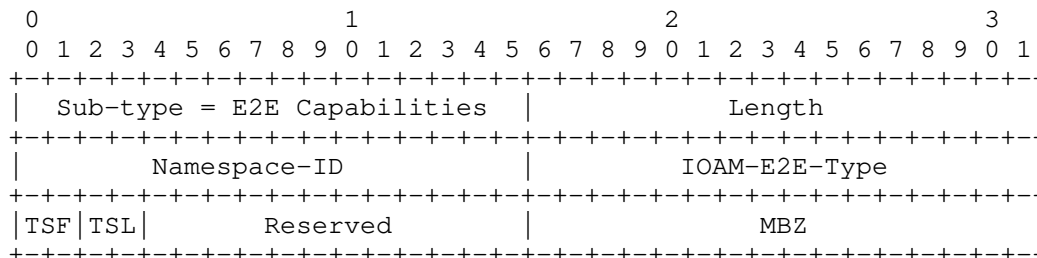


Figure 6: IOAM Edge-to-Edge Capabilities Sub-TLV

When this sub-TLV is present in the IOAM Capabilities Response 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 that identifies it as 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 5.3 of [I-D.ietf-ippm-ioam-data], it should be one of the Namespace-IDs listed in the IOAM Capabilities Query TLV of echo request.

IOAM-E2E-Type field has the same definition as what's specified in section 5.6 of [I-D.ietf-ippm-ioam-data].

TSF field specifies the timestamp format used by the sending node. 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 the TSF field is set to 0b00, which indicates the PTP timestamp format, the values of the TSL field are interpreted as follows:

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 the TSF field is set to 0b01, which indicates the NTP timestamp format, the values of the TSL field are interpreted as follows:

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 the 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

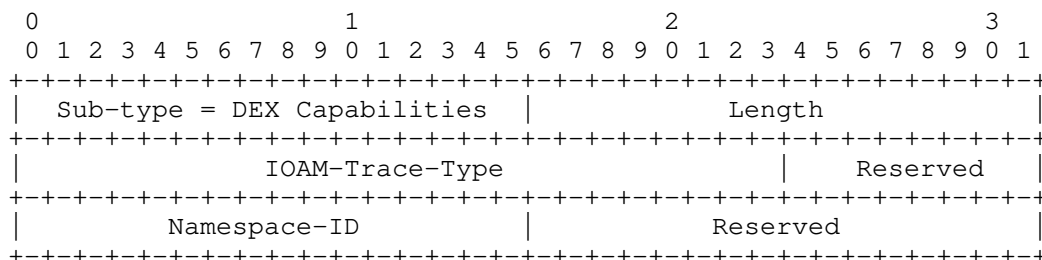


Figure 7: IOAM DEX Capabilities Sub-TLV

When this sub-TLV is present in the IOAM Capabilities Response TLV, it means that the sending node is an IOAM transit node and the IOAM DEX function is enabled at this IOAM transit node.

Sub-type is set to the value that identifies it as 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 Query 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

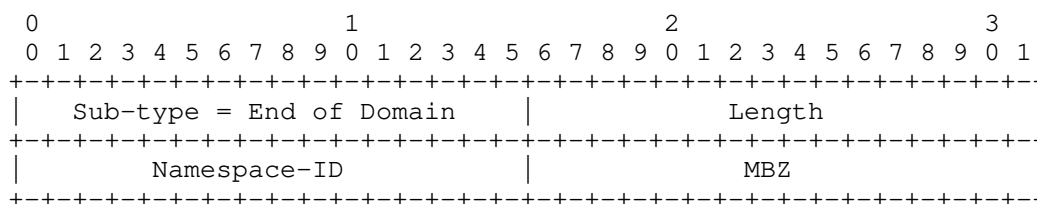


Figure 8: IOAM End of Domain Sub-TLV

When this sub-TLV is present in the IOAM Capabilities Response 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 Response 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 Response TLV, otherwise the End-of-Domain sub-TLV MUST be present in the IOAM Capabilities Response TLV sent by the IOAM decapsulating node. 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.

Sub-type is set to the value that identifies it as an IOAM End of Domain 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 5.3 of [I-D.ietf-ippm-ioam-data], it should be one of the Namespace-IDs listed in the IOAM Capabilities Query 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 echo requests that include the IOAM Capabilities Query 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. And further, increasing by 1 the TTL every time the IOAM encapsulating node sends a new echo request, until the IOAM encapsulating node receives an echo reply sent by the IOAM decapsulating node, which should contain the IOAM Capabilities Response 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 an 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. 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 Query TLV will send an echo reply to the IOAM encapsulating node, and within the echo reply, there should be an IOAM Capabilities Response TLV containing one or more sub-TLVs. The IOAM Capabilities Query TLV contained in the echo request would be ignored by the receiving node that is unaware of IOAM.

5. Security Considerations

Queries and responses about the state of an IOAM domain should be processed only from a trusted source. An unauthorized query MUST be discarded by an implementation that supports this specification. Similarly, unsolicited echo response with the IOAM Capabilities TLV MUST be discarded. Authentication of echo request/reply that

includes the IOAM Capabilities TLV is one of methods of the integrity protection. Implementations could also provide a means of filtering based on the source address of the received echo request/reply. The integrity protection for IOAM capabilities information collection can also be achieved using mechanisms in the underlay data plane. For example, if the underlay is an IPv6 network, IP Authentication Header [RFC4302] or IP Encapsulating Security Payload Header [RFC4303] can be used to provide integrity protection.

Information about the state of the IOAM domain collected in the IOAM Capabilities TLV is confidential. An implementation can use secure transport to provide privacy protection. For example, if the underlay is an IPv6 network, confidentiality can be achieved using the IP Encapsulating Security Payload Header [RFC4303].

6. IANA Considerations

This document requests the following IANA Actions.

IANA is requested to create a registry group named "In-Situ OAM (IOAM) Capabilities Parameters".

This group will include the following registries:

- o IOAM SoR Capability
- o IOAM TSF+TSL Capability

New registries in this group can be created via RFC Required process as per [RFC8126].

The subsequent sub-sections detail the registries herein contained.

Considering the TLVs/sub-TLVs defined in this document would be carried in different kinds of Echo Request/Reply message, such as ICMPv6 or LSP Ping, it is intended that the registries for Type and sub-Type would be requested in subsequent documents.

6.1. IOAM SoR Capability Registry

This registry defines 4 code points for the IOAM SoR Capability field for identifying the size of "Random" and "Cumulative" data as explained in section 5.5 of [I-D.ietf-ippm-ioam-data]. The following code points are defined in this draft:

SoR	Description
----	-----
0b00	64-bit "Random" and 64-bit "Cumulative" data

0b01 - 0b11 are available for assignment via RFC Required process as per [RFC8126].

6.2. IOAM TSF+TSL Capability Registry

This registry defines 3 code points for the IOAM TSF Capability field for identifying the timestamp format as explained in section 6 of [I-D.ietf-ippm-ioam-data].

- o When the code point for the IOAM TSF Capability field equals 0b00 which means PTP timestamp format, this registry defines 2 code points for the IOAM TSL Capability field for identifying the timestamp length.
- o When the code point for the IOAM TSF Capability field equals 0b01 which means NTP timestamp format, this registry defines 3 code points for the IOAM TSL Capability field for identifying the timestamp length.

The following code points are defined in this draft:

TSF ----	TSL ----	Description -----
0b00		PTP Timestamp Format
	0b00	64-bit PTPv1 timestamp
	0b01	80-bit PTPv2 timestamp
0b01		NTP Timestamp Format
	0b00	32-bit NTP timestamp
	0b01	64-bit NTP timestamp
	0b10	128-bit NTP timestamp
0b10		POSIX Timestamp Format

Unassigned code points of TSF+TSL are available for assignment via RFC Required process as per [RFC8126].

7. Acknowledgements

The authors would like to acknowledge Tianran Zhou, Dhruv Dhody, Frank Brockners and Cheng Li for their careful review and helpful comments.

The authors appreciate the f2f discussion with Frank Brockners on this document.

The authors would like to acknowledge Tommy Pauly and Ian Swett for their good suggestion and guidance.

8. References

8.1. Normative References

- [I-D.ietf-ippm-ioam-data]
Brockners, F., Bhandari, S., and T. Mizrahi, "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data-12 (work in progress), February 2021.
- [I-D.ietf-ippm-ioam-direct-export]
Song, H., Gafni, B., Zhou, T., Li, Z., Brockners, F., Bhandari, S., Sivakolundu, R., and T. Mizrahi, "In-situ OAM Direct Exporting", draft-ietf-ippm-ioam-direct-export-03 (work in progress), February 2021.
- [IEEE1588v2]
IEEE, "IEEE Std 1588-2008 - IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems", IEEE Std 1588-2008, 2008, <<http://standards.ieee.org/findstds/standard/1588-2008.html>>.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <<https://www.rfc-editor.org/info/rfc5905>>.
- [RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

8.2. Informative References

- [I-D.ietf-bier-ping]
Kumar, N., Pignataro, C., Akiya, N., Zheng, L., Chen, M., and G. Mirsky, "BIER Ping and Trace", draft-ietf-bier-ping-07 (work in progress), May 2020.

- [I-D.ietf-sfc-multi-layer-oam]
Mirsky, G., Meng, W., Khasnabish, B., and C. Wang, "Active OAM for Service Function Chaining", draft-ietf-sfc-multi-layer-oam-10 (work in progress), March 2021.
- [RFC4302] Kent, S., "IP Authentication Header", RFC 4302, DOI 10.17487/RFC4302, December 2005, <<https://www.rfc-editor.org/info/rfc4302>>.
- [RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, DOI 10.17487/RFC4303, December 2005, <<https://www.rfc-editor.org/info/rfc4303>>.
- [RFC4443] Conta, A., Deering, S., and M. Gupta, Ed., "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", STD 89, RFC 4443, DOI 10.17487/RFC4443, March 2006, <<https://www.rfc-editor.org/info/rfc4443>>.
- [RFC4884] Bonica, R., Gan, D., Tappan, D., and C. Pignataro, "Extended ICMP to Support Multi-Part Messages", RFC 4884, DOI 10.17487/RFC4884, April 2007, <<https://www.rfc-editor.org/info/rfc4884>>.
- [RFC8029] Kompella, K., Swallow, G., Pignataro, C., Ed., Kumar, N., Aldrin, S., and M. Chen, "Detecting Multiprotocol Label Switched (MPLS) Data-Plane Failures", RFC 8029, DOI 10.17487/RFC8029, March 2017, <<https://www.rfc-editor.org/info/rfc8029>>.
- [RFC8335] Bonica, R., Thomas, R., Linkova, J., Lenart, C., and M. Boucadair, "PROBE: A Utility for Probing Interfaces", RFC 8335, DOI 10.17487/RFC8335, February 2018, <<https://www.rfc-editor.org/info/rfc8335>>.

Authors' Addresses

Xiao Min
ZTE Corp.
Nanjing
China

Phone: +86 25 88013062
Email: xiao.min2@zte.com.cn

Greg Mirsky
ZTE Corp.
USA

Email: gregory.mirsky@ztetx.com

Lei Bo
China Telecom
Beijing
China

Phone: +86 10 50902903
Email: leibo@chinatelecom.cn

IPPM
Internet-Draft
Intended status: Standards Track
Expires: September 1, 2022

T. Zhou, Ed.
G. Fioccola
Huawei
Y. Liu
China Mobile
M. Cociglio
Telecom Italia
S. Lee
LG U+
W. Li
Huawei
February 28, 2022

Enhanced Alternate Marking Method
draft-zhou-ippm-enhanced-alternate-marking-09

Abstract

This document extends the IPv6 Alternate Marking Option to provide enhanced capabilities and allow advanced functionalities. With this extension, it can be possible to perform thicker packet loss measurements and more dense delay measurements with no limitation for the number of concurrent flows under monitoring.

Status of This Memo

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

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

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

This Internet-Draft will expire on September 1, 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction	2
1.1. Requirements Language	3
2. Data Fields Format	3
3. Security Considerations	6
4. IANA Considerations	6
5. References	7
5.1. Normative References	7
5.2. Informative References	7
Authors' Addresses	8

1. Introduction

The Alternate Marking [RFC8321] and Multipoint Alternate Marking [RFC8889] define the Alternate Marking technique that is a hybrid performance measurement method, per [RFC7799] classification of measurement methods. This method is based on marking consecutive batches of packets and it can be used to measure packet loss, latency, and jitter on live traffic.

The IPv6 AltMark Option [I-D.ietf-6man-ipv6-alt-mark] applies the Alternate Marking Method to IPv6, and defines an Extension Header Option to encode the Alternate Marking Method for both the Hop-by-Hop Options Header and the Destination Options Header. Similarly, SRv6 AltMark [I-D.fz-spring-srv6-alt-mark] defines how Alternate Marking data is carried as a TLV in the Segment Routing Header.

While the IPv6 AltMark Option implements the basic alternate marking methodology, this document defines extended data fields for the AltMark Option and provides enhanced capabilities to overcome some challenges and enable future proof applications.

It is worth mentioning that the enhanced capabilities are intended for further use and are optional.

Some possible enhanced applications MAY be:

1. thicker packet loss measurements: the single marking method of the base AltMark Option can be extended with additional marking bits in order to get shortest marking periods under the same timing conditions.
2. more dense delay measurements: than double marking method of the base AltMark Option can be extended with additional marking bits in order to identify down to each packet as delay sample.
3. increase the number of concurrent flows under monitoring: if the 20-bit FlowMonID is set independently and pseudo randomly, there is a 50% chance of collision for 1206 flows. The size of FlowMonID can be extended to raise the entropy and therefore to increase the number of concurrent flows that can be monitored.

1.1. Requirements Language

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

2. Data Fields Format

The Data Fields format is represented in Figure 1. A 4-bit NH(NextHeader) field is allocated from the Reserved field of IPv6 AltMark Option [I-D.ietf-6man-ipv6-alt-mark]. It is worth highlighting that remaining bits of the former Reserved field continue to be reserved.

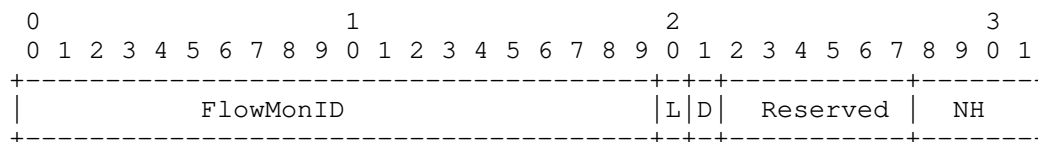


Figure 1: Data fields indicator for enhanced capabilities

The NH (NextHeader) field is used to indicate the extended data fields which are used for enhanced capabilities:

NextHeader value of 0x00 is reserved for backward compatibility. It means that there is no extended data field attached.

NextHeader values of 0x01-0x08 are reserved for private use or for experimentation.

NextHeader value of 0x09 indicates the extended data fields. The format is showed in Figure 2.

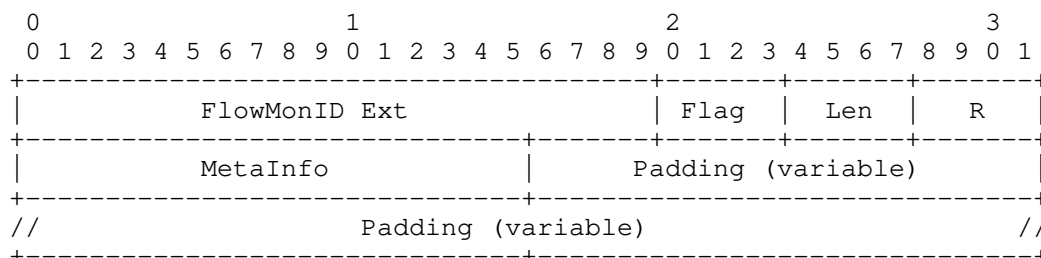


Figure 2: Data fields extension for enhanced alternate marking

where:

- o FlowMonID Ext - 20 bits unsigned integer. This is used to extend the FlowMonID in order to reduce the conflict when random allocation is applied. The disambiguation of the FlowMonID field is discussed in IPv6 AltMark Option [I-D.ietf-6man-ipv6-alt-mark].
- o Flag - A 4-bit flag to indicate the special purpose usage (see below).
- o Len - Length. It indicates the length of the enhanced alternate marking extension in bytes.
- o R - Reserved for further use. These bits MUST be set to zero on transmission and ignored on receipt.
- o MetaInfo - A 16-bit Bitmap to indicate more meta data attached for the enhanced function (see below).
- o Padding - These bits MUST be set to zero when not being used.

The Flag is defined in Figure 3 as:

- o bit 0 - Measurement mode, M bit. If M=0, it indicates that it is for hop-by-hop monitoring. If M=1, it indicates that it is for end-to-end monitoring.
- o bit 2 - Flow direction identification, F bit. This flag is used in the case backward direction flow monitoring is requested to be set up automatically. If F=1, it indicates that the flow direction is forward. If F=0, it indicates that the flow direction is backward.

- o others (shown as R) - Reserved. These bits MUST be set to zero and ignored on receipt.

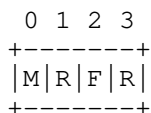


Figure 3: Flag data field

The MetaInfo is defined in the following Figure 4 as a bit map:

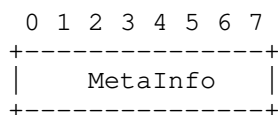


Figure 4: MetaInfo data field

- o bit 0: it indicates a 6 bytes Timestamp that is attached as Padding after the MetaInfo. Timestamp(s) stands for the number of seconds in the timestamp. It will overwrite the Padding after MetaInfo. Timestamp(ns) stands for the number of sub-seconds in the timestamp with the unit of nano second. This Timestamp is filled by the encapsulation node, and is taken all the way to the decapsulation node. So that all the intermediate nodes could compare it with its local time, and measure the one way delay.

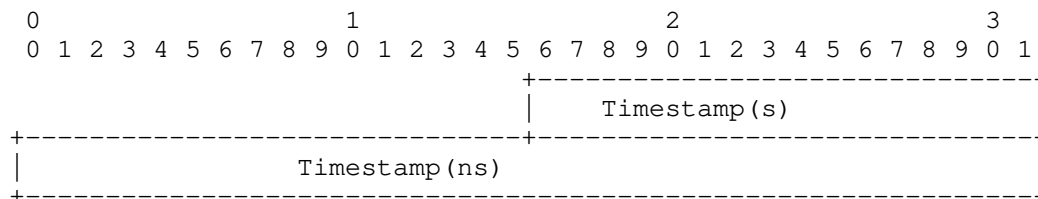


Figure 5: Timestamp data field

- o bit 1: it indicates the control information with the following data format that is attached as Padding after the MetaInfo:

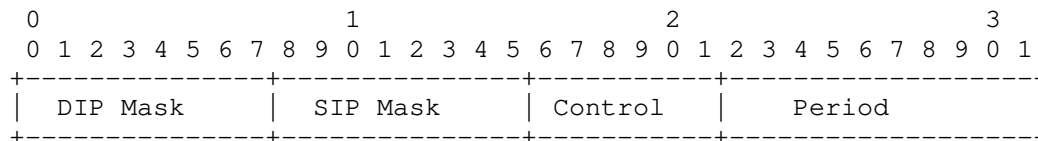


Figure 6: Control words for backward direction flow monitoring

This is used to set up the backward direction flow monitoring.
Where:

- * DIP Mask: it is the length of the destination IP prefix.
 - * SIP Mask: it is the length of the source IP prefix.
 - * Control: it indicates more match fields to set up the backward direction flow monitoring.
 - * Period: it indicates the alternate marking period with the unit of second.
- o bit 2: it indicates a 4 bytes Sequence number with the following data format that is attached as Padding after the MetaInfo. The unique Sequence could be used to detect the out-of-order packets, in addition to the normal loss measurement. More over, the Sequence can be used together with the latency measurement, so as to get the per packet timestamp.

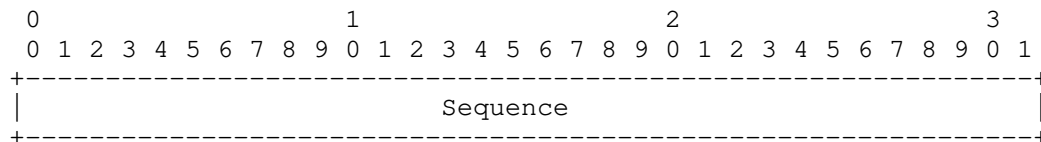


Figure 7: Sequence number data field

It is worth noting that the meta data information forming the Padding and specified above in Figure 5, Figure 6 and Figure 7 must be ordered according to the order of the MetaInfo bits.

3. Security Considerations

IPv6 AltMark Option [I-D.ietf-6man-ipv6-alt-mark] analyzes different security concerns and related solutions. These aspects are valid and applicable also to this document. In particular the fundamental security requirement is that Alternate Marking MUST only be applied in a specific limited domain, as also mentioned in [RFC8799].

4. IANA Considerations

This document has no request to IANA.

5. References

5.1. Normative References

- [I-D.fz-spring-srv6-alt-mark]
Fioccola, G., Zhou, T., and M. Cociglio, "Segment Routing Header encapsulation for Alternate Marking Method", draft-fz-spring-srv6-alt-mark-02 (work in progress), February 2022.
- [I-D.ietf-6man-ipv6-alt-mark]
Fioccola, G., Zhou, T., Cociglio, M., Qin, F., and R. Pang, "IPv6 Application of the Alternate Marking Method", draft-ietf-6man-ipv6-alt-mark-12 (work in progress), October 2021.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC7799] Morton, A., "Active and Passive Metrics and Methods (with Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799, May 2016, <<https://www.rfc-editor.org/info/rfc7799>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

5.2. Informative References

- [RFC8321] Fioccola, G., Ed., Capello, A., Cociglio, M., Castaldelli, L., Chen, M., Zheng, L., Mirsky, G., and T. Mizrahi, "Alternate-Marking Method for Passive and Hybrid Performance Monitoring", RFC 8321, DOI 10.17487/RFC8321, January 2018, <<https://www.rfc-editor.org/info/rfc8321>>.
- [RFC8799] Carpenter, B. and B. Liu, "Limited Domains and Internet Protocols", RFC 8799, DOI 10.17487/RFC8799, July 2020, <<https://www.rfc-editor.org/info/rfc8799>>.
- [RFC8889] Fioccola, G., Ed., Cociglio, M., Sapio, A., and R. Sisto, "Multipoint Alternate-Marking Method for Passive and Hybrid Performance Monitoring", RFC 8889, DOI 10.17487/RFC8889, August 2020, <<https://www.rfc-editor.org/info/rfc8889>>.

Authors' Addresses

Tianran Zhou
Huawei
156 Beiqing Rd.
Beijing 100095
China

Email: zhoutianran@huawei.com

Giuseppe Fioccola
Huawei
Riesstrasse, 25
Munich 80992
Germany

Email: giuseppe.fioccola@huawei.com

Yisong Liu
China Mobile
Beijing
China

Email: liuyisong@chinamobile.com

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

Email: mauro.cociglio@telecomitalia.it

Shinyoung Lee
LG U+
71, Magokjungang 8-ro, Gangseo-gu
Seoul
Republic of Korea

Email: leesy@lguplus.co.kr

Weidong Li
Huawei
156 Beiqing Rd.
Beijing 100095
China

Email: poly.li@huawei.com