Workgroup: IPPM Working Group
Internet-Draft:
draft-ietf-ippm-ioam-conf-state-05
Published: 21 September 2022
Intended Status: Standards Track
Expires: 25 March 2023
Authors: X. Min G. Mirsky L. Bo
ZTE Corp. Ericsson China Telecom

Echo Request/Reply for Enabled In-situ OAM Capabilities

Abstract

This document describes an extension to the echo request/reply mechanisms used in IPv6 (including Segment Routing with IPv6 data plane (SRv6)), MPLS (including Segment Routing with MPLS data plane (SR-MPLS)), Service Function Chain (SFC) and Bit Index Explicit Replication (BIER) environments, which can be used within the In situ Operations, Administration, and Maintenance (IOAM) domain, allowing the IOAM encapsulating node to discover the enabled IOAM capabilities of each IOAM transit and IOAM decapsulating node.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

In situ Operations, Administration, and Maintenance (IOAM) ([RFC9197] [I-D.ietf-ippm-ioam-direct-export]) defines data fields that record OAM information within the packet while the packet traverses a particular network domain, called an IOAM domain. IOAM can complement or replace other OAM mechanisms, such as ICMP or other types of probe packets.

As specified in [RFC9197], within the IOAM domain, the IOAM data may be updated by network nodes that the packet traverses. The device which adds an IOAM header to the packet is called an "IOAM encapsulating node". In contrast, the device which removes an IOAM header is referred to as an "IOAM decapsulating node". Nodes within the domain that are aware of IOAM data and read and/or write and/or process IOAM data are called "IOAM transit nodes". IOAM encapsulating or decapsulating nodes can also serve as IOAM transit nodes at the same time. IOAM encapsulating or decapsulating nodes are also referred to as IOAM domain edge devices, which can be hosts or network devices.

As specified in [RFC9197], IOAM is focused on "limited domains" as defined in [RFC8799]. In a limited domain, a control entity that has control over every IOAM device may be deployed. If that's the case, the control entity can provision both the explicit transport path and the IOAM header applied to data packet at every IOAM encapsulating node.

In a case when a control entity that has control over every IOAM device is not deployed in the IOAM domain, the IOAM encapsulating node needs to discover the enabled IOAM capabilities at the IOAM transit and decapsulating nodes. For example, what types of IOAM tracing data can be added by the transit nodes along the transport path of the data packet IOAM is applied to. The IOAM encapsulating node can then add the correct IOAM header to the data packet according to the discovered IOAM capabilities. Specifically, the IOAM encapsulating node first identifies the types and lengths of IOAM options included in the IOAM data according to the discovered IOAM capabilities. Then the IOAM encapsulating node can add the IOAM header to the data packet based on the identified types and lengths of IOAM options included in the IOAM data. The IOAM encapsulating node may use NETCONF/YANG or IGP to discover these IOAM capabilities. However, NETCONF/YANG or IGP has some limitations:

* When NETCONF/YANG is used in this scenario, each IOAM encapsulating node (including the host when it takes the role of an IOAM encapsulating node) needs to implement a NETCONF Client, each IOAM transit and IOAM decapsulating node (including the host when it takes the role of an IOAM decapsulating node) needs to implement a NETCONF Server, the complexity can be an issue. Furthermore, each IOAM encapsulating node needs to establish NETCONF Connection with each IOAM transit and IOAM decapsulating node, the scalability can be an issue.

* When IGP is used in this scenario, the IGP and IOAM domains don't always have the same coverage. For example, when the IOAM encapsulating node or the IOAM decapsulating node is a host, the availability can be an issue. Furthermore, it might be too challenging to reflect enabled IOAM capabilities at the IOAM transit and IOAM decapsulating node if these are controlled by a local policy depending on the identity of the IOAM encapsulating node.

This document describes an extension to the echo request/reply mechanisms used in IPv6 (including SRv6), MPLS (including SR-MPLS), SFC and BIER environments, which can be used within the IOAM domain, allowing the IOAM encapsulating node to discover the enabled IOAM capabilities of each IOAM transit and IOAM decapsulating node.

The following documents contain references to the echo request/reply mechanisms used in IPv6 (including SRv6), MPLS (including SR-MPLS), SFC and BIER environments:

* [RFC4443] ("Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification"), [RFC4620] ("IPv6 Node Information Queries"), [RFC4884] ("Extended ICMP to Support Multi-Part Messages") and [RFC8335] ("PROBE: A Utility for Probing Interfaces")

* [RFC8029] ("Detecting Multiprotocol Label Switched (MPLS) Data-Plane Failures")

* [I-D.ietf-sfc-multi-layer-oam] ("Active OAM for Service Function Chaining (SFC)")

* [I-D.ietf-bier-ping] ("BIER Ping and Trace")

Note that specification details for these different echo request/reply protocols are outside the scope of this document. It is expected that each such protocol extension would be specified by an RFC and jointly designed by the working group that develops or maintains the echo request/reply protocol and the IETF IP Performance Measurement (IPPM) Working Group.

Fate sharing is a common requirement for all kinds of active OAM packets, echo request is among them, in this document that means echo request is required to traverse a path of IOAM data packet. This requirement can be achieved by, e.g., applying same explicit path or ECMP processing to both echo request and IOAM data packet.

2. Conventions

2.1. Requirements Language

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

2.2. Abbreviations

BIER: Bit Index Explicit Replication

BGP: Border Gateway Protocol

ECMP: Equal-Cost Multipath

E2E: Edge to Edge

ICMP: Internet Control Message Protocol

IGP: Interior Gateway Protocol

IOAM: In situ Operations, Administration, and Maintenance

LSP: Label Switched Path

MPLS: Multi-Protocol Label Switching

MTU: Maximum Transmission Unit

NTP: Network Time Protocol

OAM: Operations, Administration, and Maintenance

PCEP: Path Computation Element (PCE) Communication Protocol

POSIX: Portable Operating System Interface

POT: Proof of Transit

PTP: Precision Time Protocol

SR-MPLS: Segment Routing with MPLS data plane

SRv6: Segment Routing with IPv6 data plane

SFC: Service Function Chain

TTL: Time to Live

3. IOAM Capabilities Formats

3.1. IOAM Capabilities Query Container

For echo request, IOAM Capabilities Query uses a container which has the following format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.            IOAM Capabilities Query Container Header           .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.                   List of IOAM Namespace-IDs                  .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: IOAM Capabilities Query Container of Echo Request

When this container is present in the echo request sent by an IOAM encapsulating node, that means the IOAM encapsulating node requests the receiving node to reply with its enabled IOAM capabilities. If there is no IOAM capability to be reported by the receiving node, then this container SHOULD be ignored by the receiving node, which means the receiving node SHOULD send an echo reply without IOAM capabilities or no echo reply, in the light of whether the echo request includes other containers than the IOAM Capabilities Query Container. A list of IOAM Namespace-IDs (one or more Namespace-IDs) MUST be included in this container in the echo request, and if present, the Default-Namespace-ID 0x0000 MUST be placed at the begining of the list of IOAM Namespace-IDs. The IOAM encapsulating node requests only the enabled IOAM capabilities that match one of the Namespace-IDs. The Namespace-ID has the same definition as what's specified in Section 4.3 of [RFC9197].

The IOAM Capabilities Query Container has a container header that is used to identify the type and optionally length of the container payload, and the container payload (List of IOAM Namespace-IDs) is zero-padded to align to a 4-octet boundary.

The length, structure, and definition of the IOAM Capabilities Query Container Header depends on the specific environment it is applied at.

3.2. IOAM Capabilities Response Container

For echo reply, IOAM Capabilities Response uses a container which has the following format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.          IOAM Capabilities Response Container Header          .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.               List of IOAM Capabilities Objects               .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: IOAM Capabilities Response Container of Echo Reply

When this container is present in the echo reply sent by an IOAM transit node or IOAM decapsulating node, that means the IOAM function is enabled at this node, and this container contains the enabled IOAM capabilities of the sender. A list of IOAM capabilities objects (one or more objects) which contains the enabled IOAM capabilities SHOULD be included in this container of echo reply.

The IOAM Capabilities Response Container has a container header that is used to identify the type and optionally length of the container payload, and the container payload (List of IOAM Capabilities Objects) is zero-padded to align to a 4-octet boundary.

The length, structure, and definition of the IOAM Capabilities Response Container Header depends on the specific environment it is applied at.

Based on the IOAM data fields defined in [RFC9197] and [I-D.ietf-ippm-ioam-direct-export], six types of objects are defined in this document. The same type of object MAY be present in the IOAM Capabilities Response Container more than once, only if with a different Namespace-ID.

Similar to the container, each object has an object header that is used to identify the type and length of the object payload, and the object payload is zero-padded to align to a 4-octet boundary.

The length, structure, and definition of Object Header depends on the specific environment it is applied at.

3.2.1. IOAM Pre-allocated Tracing Capabilities Object

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.     IOAM Pre-allocated Tracing Capabilities Object Header     .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               IOAM-Trace-Type                 |  Reserved   |W|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Namespace-ID          |         Ingress_MTU           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Ingress_if_id (short or wide format)         ......          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: IOAM Pre-allocated Tracing Capabilities Object

When this Object is present in the IOAM Capabilities Response Container, that means the sending node is an IOAM transit node and the IOAM pre-allocated tracing function is enabled at this IOAM transit node.

IOAM-Trace-Type field has the same definition as what's specified in Section 4.4 of [RFC9197].

Reserved field is reserved for future use and MUST be set to zero.

W flag indicates whether Ingress_if_id is in short or wide format. The W-bit is set if the Ingress_if_id is in wide format. The W-bit is clear if the Ingress_if_id is in short format.

Namespace-ID field has the same definition as what's specified in Section 4.3 of [RFC9197], it should be one of the Namespace-IDs listed in the IOAM Capabilities Query Object of the echo request.

Ingress_MTU field has 16 bits and specifies the MTU (in octets) of the ingress interface from which the sending node received echo request.

Ingress_if_id field has 16 bits (in short format) or 32 bits (in wide format) and specifies the identifier of the ingress interface from which the sending node received echo request. If the W-bit is cleared that indicates Ingress_if_id field has 16 bits, then the 16 bits following the Ingress_if_id field are reserved for future use and MUST be set to zero.

3.2.2. IOAM Incremental Tracing Capabilities Object

```
   0                   1                   2                   3
   0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  .                                                               .
  .      IOAM Incremental Tracing Capabilities Object Header      .
  .                                                               .
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |               IOAM-Trace-Type                 |  Reserved   |W|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |         Namespace-ID          |         Ingress_MTU           |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |  Ingress_if_id (short or wide format)         ......          |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4: IOAM Incremental Tracing Capabilities Object

When this Object is present in the IOAM Capabilities Response Container, that means the sending node is an IOAM transit node and the IOAM incremental tracing function is enabled at this IOAM transit node.

IOAM-Trace-Type field has the same definition as what's specified in Section 4.4 of [RFC9197].

Reserved field is reserved for future use and MUST be set to zero.

W flag indicates whether Ingress_if_id is in short or wide format. The W-bit is set if the Ingress_if_id is in wide format. The W-bit is clear if the Ingress_if_id is in short format.

Namespace-ID field has the same definition as what's specified in Section 4.3 of [RFC9197], it should be one of the Namespace-IDs listed in the IOAM Capabilities Query Object of the echo request.

Ingress_MTU field has 16 bits and specifies the MTU (in octets) of the ingress interface from which the sending node received echo request.

Ingress_if_id field has 16 bits (in short format) or 32 bits (in wide format) and specifies the identifier of the ingress interface from which the sending node received echo request. If the W-bit is cleared that indicates Ingress_if_id field has 16 bits, then the 16 bits following the Ingress_if_id field are reserved for future use and MUST be set to zero.

3.2.3. IOAM Proof-of-Transit Capabilities Object

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.       IOAM Proof-of-Transit Capabilities Object Header        .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Namespace-ID          | IOAM-POT-Type |SoP| Reserved  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 5: IOAM Proof-of-Transit Capabilities Object

When this Object is present in the IOAM Capabilities Response Container, that means the sending node is an IOAM transit node and the IOAM Proof of Transit function is enabled at this IOAM transit node.

Namespace-ID field has the same definition as what's specified in Section 4.3 of [RFC9197], it should be one of the Namespace-IDs listed in the IOAM Capabilities Query Object of the echo request.

IOAM-POT-Type field has the same definition as what's specified in Section 4.5 of [RFC9197].

SoP field has two bits, which means the size of "PktID" and "Cumulative" data that are specified in Section 4.5 of [RFC9197]. This document defines SoP as follow:

   0b00 means 64-bit "PktID" and 64-bit "Cumulative" data.

   0b01~0b11: Reserved for future standardization

Reserved field is reserved for future use and MUST be set to zero.

3.2.4. IOAM Edge-to-Edge Capabilities Object

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.          IOAM Edge-to-Edge Capabilities Object Header         .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Namespace-ID          |         IOAM-E2E-Type         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|TSF|         Reserved          |            Reserved           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 6: IOAM Edge-to-Edge Capabilities Object

When this Object is present in the IOAM Capabilities Response Container, that means the sending node is an IOAM decapsulating node and IOAM edge-to-edge function is enabled at this IOAM decapsulating node.

Namespace-ID field has the same definition as what's specified in Section 4.3 of [RFC9197], it should be one of the Namespace-IDs listed in the IOAM Capabilities Query Object of the echo request.

IOAM-E2E-Type field has the same definition as what's specified in Section 4.6 of [RFC9197].

TSF field specifies the timestamp format used by the sending node. Aligned with three possible timestamp formats specified in Section 5 of [RFC9197], this document defines TSF as follows:

0b00: PTP truncated timestamp format

0b01: NTP 64-bit timestamp format

0b10: POSIX-based timestamp format

0b11: Reserved for future standardization

Reserved field is reserved for future use and MUST be set to zero.

3.2.5. IOAM DEX Capabilities Object

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.              IOAM DEX Capabilities Object Header              .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               IOAM-Trace-Type                 |    Reserved   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Namespace-ID          |           Reserved            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 7: IOAM DEX Capabilities Object

When this Object is present in the IOAM Capabilities Response Container, that means the sending node is an IOAM transit node and the IOAM direct exporting function is enabled at this IOAM transit node.

IOAM-Trace-Type field has the same definition as what's specified in Section 3.2 of [I-D.ietf-ippm-ioam-direct-export].

Namespace-ID field has the same definition as what's specified in Section 4.3 of [RFC9197], it should be one of the Namespace-IDs listed in the IOAM Capabilities Query Object of the echo request.

Reserved field is reserved for future use and MUST be set to zero.

3.2.6. IOAM End-of-Domain Object

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.               IOAM End-of-Domain Object Header                .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Namespace-ID          |         Must Be Zero          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 8: IOAM End-of-Domain Object

When this Object is present in the IOAM Capabilities Response Container, that means the sending node is an IOAM decapsulating node. Unless the IOAM Edge-to-Edge Capabilities Object is present, which also indicates that the sending node is an IOAM decapsulating node, the End-of-Domain Object MUST be present in the IOAM Capabilities Response Container sent by an IOAM decapsulating node. When the IOAM edge-to-edge function is enabled at the IOAM decapsulating node, it's RECOMMENDED to include only the IOAM Edge-to-Edge Capabilities Object but not the IOAM End-of-Domain Object.

Namespace-ID field has the same definition as what's specified in Section 4.3 of [RFC9197], it SHOULD be one of the Namespace-IDs listed in the IOAM Capabilities Query Container.

4. Operational Guide

Once the IOAM encapsulating node is triggered to discover the enabled IOAM capabilities of each IOAM transit and IOAM decapsulating node, the IOAM encapsulating node will send echo requests that include the IOAM Capabilities Query Container. First, with TTL equal to 1 to reach the closest node, which may be an IOAM transit node or not. Then with TTL equal to 2 to reach the second nearest node, which also may be an IOAM transit node or not. And further, increasing by 1 the TTL every time the IOAM encapsulating node sends a new echo request, until the IOAM encapsulating node receives an echo reply sent by the IOAM decapsulating node, which should contain the IOAM Capabilities Response Container including the IOAM Edge-to-Edge Capabilities Object or the IOAM End-of-Domain Object. Alternatively, if the IOAM encapsulating node knows precisely all the IOAM transit and IOAM decapsulating nodes beforehand, once the IOAM encapsulating node is triggered to discover the enabled IOAM capabilities, it can send an echo request to each IOAM transit and IOAM decapsulating node directly, without TTL expiration.

The IOAM encapsulating node may be triggered by the device administrator, the network management system, the network controller, or data traffic. The specific triggering mechanisms are outside the scope of this document.

Each IOAM transit and IOAM decapsulating node that receives an echo request containing the IOAM Capabilities Query Container will send an echo reply to the IOAM encapsulating node. For the echo reply, there should be an IOAM Capabilities Response Container containing one or more Objects. The IOAM Capabilities Query Container of the echo request would be ignored by the receiving node unaware of IOAM.

5. IANA Considerations

This document requests the following IANA Actions.

IANA is requested to create a registry group named "In-Situ OAM (IOAM) Capabilities Parameters".

This group will include the following registries:

* IOAM SoP Capability

* IOAM TSF Capability

New registries in this group can be created via RFC Required process as per [RFC8126].

The subsequent sub-sections detail the registries herein contained.

Considering the Containers/Objects defined in this document would be carried in different types of Echo Request/Reply messages, such as ICMPv6 or LSP Ping, it is intended that the registries for Container/Object Type would be requested in subsequent documents.

5.1. IOAM SoP Capability Registry

This registry defines 4 code points for the IOAM SoP Capability field for identifying the size of "PktID" and "Cumulative" data as explained in Section 4.5 of [RFC9197]. The following code points are defined in this document:

SoP	Description
0b00	64-bit "PktID" and 64-bit "Cumulative" data

0b01 - 0b11 are available for assignment via RFC Required process as per [RFC8126].

5.2. IOAM TSF Capability Registry

This registry defines 4 code points for the IOAM TSF Capability field of identifying the timestamp format as explained in Section 5 of [RFC9197]. The following code points are defined in this document:

TSF	Description
0b00	PTP Truncated Timestamp Format
0b01	NTP 64-bit Timestamp Format
0b10	POSIX-based Timestamp Format
0b11	Reserved for future standardization

0b11 is available for assignment via RFC Required process as per [RFC8126].

6. Security Considerations

Overall, the security needs for IOAM capabilities query mechanisms used in different environments are similar.

To avoid potential Denial-of-Service (DoS) attacks, it is RECOMMENDED that implementations apply rate-limiting to incoming echo requests and replies.

To protect against unauthorized sources using echo request messages to obtain IOAM Capabilities information, it is RECOMMENDED that implementations provide a means of checking the source addresses of echo request messages against an access list before accepting the message.

When echo request/reply is used within an administrative domain, a deployment can increase security by using border filtering of incoming and outgoing echo requests/replies.

The integrity protection on IOAM Capabilities information carried in echo reply messages can be achieved by the underlaying transport. For example, if the environment is an IPv6 network, the IP Authentication Header [RFC4302] or IP Encapsulating Security Payload Header [RFC4303] can be used.

The collected IOAM Capabilities information by queries may be considered confidential. An implementation can use secure underlaying transport of echo request/reply to provide privacy protection. For example, if the environment is an IPv6 network, confidentiality can be achieved by using the IP Encapsulating Security Payload Header [RFC4303]. Besides, implementations SHOULD provide a means of filtering the addresses to which echo reply messages carrying IOAM Capabilities information may be sent.

An implementation can also directly secure the data carried in echo requests and replies if needed, the specific mechanism on how to secure the data is beyond the scope of this document.

7. Acknowledgements

The authors would like to acknowledge Tianran Zhou, Dhruv Dhody, Frank Brockners, Cheng Li, Gyan Mishra, Marcus Ihlar and Martin Duke 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-direct-export]
Song, H., Gafni, B., Brockners, F., Bhandari, S., and T. Mizrahi, "In-situ OAM Direct Exporting", Work in Progress, Internet-Draft, draft-ietf-ippm-ioam-direct-export-10, 18 August 2022, <https://www.ietf.org/archive/id/draft-ietf-ippm-ioam-direct-export-10.txt>.

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

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

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

[RFC9197]
Brockners, F., Ed., Bhandari, S., Ed., and T. Mizrahi, Ed., "Data Fields for In Situ Operations, Administration, and Maintenance (IOAM)", RFC 9197, DOI 10.17487/RFC9197, May 2022, <https://www.rfc-editor.org/info/rfc9197>.

8.2. Informative References

[I-D.ietf-bier-ping] Kumar, N., Pignataro, C., Akiya, N., Zheng, L., Chen, M., and G. Mirsky, "BIER Ping and Trace", Work in Progress, Internet-Draft, draft-ietf-bier-ping-07, 11 May 2020, <https://www.ietf.org/archive/id/draft-ietf-bier-ping-07.txt>.

[I-D.ietf-sfc-multi-layer-oam] Mirsky, G., Meng, W., Ao, T., Khasnabish, B., Leung, K., and G. Mishra, "Active OAM for Service Function Chaining (SFC)", Work in Progress, Internet-Draft, draft-ietf-sfc-multi-layer-oam-22, 25 July 2022, <https://www.ietf.org/archive/id/draft-ietf-sfc-multi-layer-oam-22.txt>.

[RFC4302] Kent, S., "IP Authentication Header", RFC 4302, DOI 10.17487/RFC4302, December 2005, <https://www.rfc-editor.org/info/rfc4302>.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, DOI 10.17487/RFC4303, December 2005, <https://www.rfc-editor.org/info/rfc4303>.

[RFC4443] Conta, A., Deering, S., and M. Gupta, Ed., "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", STD 89, RFC 4443, DOI 10.17487/RFC4443, March 2006, <https://www.rfc-editor.org/info/rfc4443>.

[RFC4620] Crawford, M. and B. Haberman, Ed., "IPv6 Node Information Queries", RFC 4620, DOI 10.17487/RFC4620, August 2006, <https://www.rfc-editor.org/info/rfc4620>.

[RFC4884] Bonica, R., Gan, D., Tappan, D., and C. Pignataro, "Extended ICMP to Support Multi-Part Messages", RFC 4884, DOI 10.17487/RFC4884, April 2007, <https://www.rfc-editor.org/info/rfc4884>.

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

[RFC8335]
Bonica, R., Thomas, R., Linkova, J., Lenart, C., and M. Boucadair, "PROBE: A Utility for Probing Interfaces", RFC 8335, DOI 10.17487/RFC8335, February 2018, <https://www.rfc-editor.org/info/rfc8335>.

[RFC8799] Carpenter, B. and B. Liu, "Limited Domains and Internet Protocols", RFC 8799, DOI 10.17487/RFC8799, July 2020, <https://www.rfc-editor.org/info/rfc8799>.

Authors' Addresses

Xiao Min
ZTE Corp.
Nanjing
China

Phone: +86 25 88013062
Email: xiao.min2@zte.com.cn

Greg Mirsky
Ericsson
United States of America

Email: gregimirsky@gmail.com

Lei Bo
China Telecom
Beijing
China

Phone: +86 10 50902903
Email: leibo@chinatelecom.cn