Stream: Internet Engineering Task Force (IETF)
RFC: 9359
Category: Standards Track
Published: March 2023
ISSN: 2070-1721
Authors: X. Min, *ZTE Corp.*; G. Mirsky, *Ericsson*; L. Bo, *China Telecom*

RFC 9359
Echo Request/Reply for Enabled In Situ OAM (IOAM) Capabilities

Abstract

This document describes a generic format for use in echo request/reply mechanisms, which can be used within an IOAM-Domain, allowing the IOAM encapsulating node to discover the enabled IOAM capabilities of each IOAM transit and IOAM decapsulating node. The generic format is intended to be used with a variety of data planes such as IPv6, MPLS, Service Function Chain (SFC), and Bit Index Explicit Replication (BIER).

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9359.

Copyright Notice

Table of Contents

1. Introduction

In situ Operations, Administration, and Maintenance (IOAM) ([RFC9197] [RFC9326]) 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 that adds an IOAM header to the packet is called an "IOAM encapsulating node". In contrast, the device that removes an IOAM header is referred to as an "IOAM decapsulating node". Nodes within the domain that are aware of IOAM data and that read, 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. [RFC9197] defines four IOAM option types, and [RFC9326] introduces a new IOAM option type called the "Direct Export (DEX) Option-Type", which is different from the other four IOAM option types defined in [RFC9197] regarding how to collect the operational and telemetry information defined in [RFC9197].

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 the 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 or exported 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 fields 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 fields. 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, and 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, so complexity can be an issue. Furthermore, each IOAM encapsulating node needs to establish a NETCONF Connection with each IOAM transit and IOAM decapsulating node, so 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 specifies formats and objects that can be used in the extension of echo request/reply mechanisms used in IPv6 (including Segment Routing over IPv6 (SRv6) data plane), MPLS (including Segment Routing over MPLS (SR-MPLS) data plane), Service Function Chain (SFC), and Bit Index Explicit Replication (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:

- "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification" [RFC4443]
- "IPv6 Node Information Queries" [RFC4620]
- "Extended ICMP to Support Multi-Part Messages" [RFC4884]
- "PROBE: A Utility for Probing Interfaces" [RFC8335]
- "Detecting Multiprotocol Label Switched (MPLS) Data-Plane Failures" [RFC8029]
- "Active OAM for Service Function Chaining (SFC)" [OAM-for-SFC]
- "BIER Ping and Trace" [BIER-PING]

It is expected that the specification of the instantiation of each of these extensions will be done in the form of an RFC jointly designed by the working group that develops or maintains the echo request/reply protocol and the IETF IP Performance Measurement (IPPM) Working Group.

In this document, note that the echo request/reply mechanism used in IPv6 does not mean ICMPv6 Echo Request/Reply [RFC4443] but does mean IPv6 Node Information Query/Reply [RFC4620].

Fate sharing is a common requirement for all kinds of active OAM packets, including echo requests. In this document, that means an echo request is required to traverse the path of an IOAM data packet. This requirement can be achieved by, e.g., applying the same explicit path or ECMP processing to both echo request and IOAM data packets. Specifically, the same ECMP processing can be applied to both echo request and IOAM data packets, by populating the same value or values in any ECMP affecting fields of the packets.

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

DEX: Direct Export

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: Multiprotocol Label Switching

MTU: Maximum Transmission Unit

NETCONF: Network Configuration Protocol

NTP: Network Time Protocol

OAM: Operations, Administration, and Maintenance

PCEP: Path Computation Element Communication Protocol

POSIX: Portable Operating System Interface

POT: Proof of Transit

PTP: Precision Time Protocol

SoP: Size of POT

SR-MPLS: Segment Routing over MPLS

SRv6: Segment Routing over IPv6

SFC: Service Function Chain

TTL: Time to Live (this is also the Hop Limit field in the IPv6 header)

TSF: TimeStamp Format

3. IOAM Capabilities Formats

3.1. IOAM Capabilities Query Container

For echo requests, the IOAM Capabilities Query uses a container that has the following format:

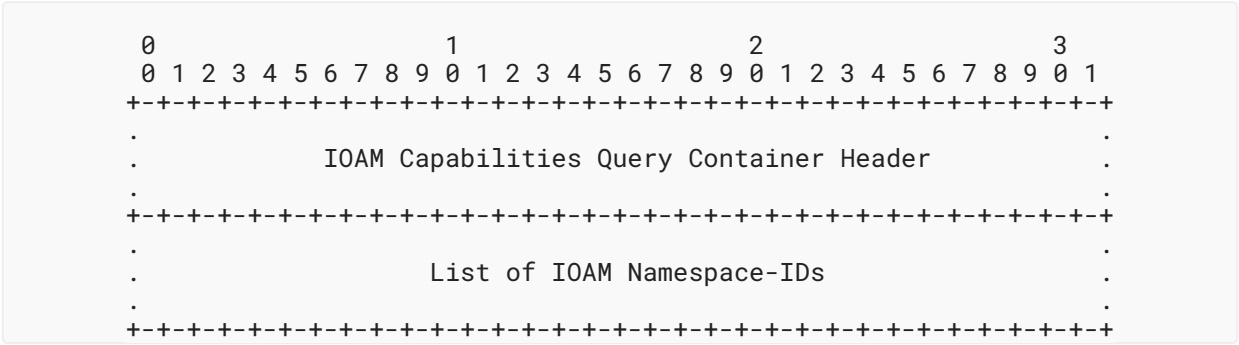

Figure 1: IOAM Capabilities Query Container of an Echo Request

When this container is present in the echo request sent by an IOAM encapsulating node, the IOAM encapsulating node requests that the receiving node reply with its enabled IOAM capabilities. If there is no IOAM capability to be reported by the receiving node, then this container **MUST** be ignored by the receiving node. This means the receiving node **MUST** send an echo reply without IOAM capabilities or no echo reply, in the light of whether the echo request includes containers other 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; if present, the Default-Namespace-ID 0x0000 **MUST** be placed at the beginning of the list of IOAM Namespace-IDs. The IOAM encapsulating node requests only the enabled IOAM capabilities that match one of the Namespace-IDs. Inclusion of the Default-Namespace-ID 0x0000 elicits replies only for capabilities that are configured with the Default-Namespace-ID 0x0000. 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, the length of the container payload. The container payload (List of IOAM Namespace-IDs) is zero-padded to align with a 4-octet boundary. Since the Default-Namespace-ID 0x0000 is mandated to appear first in the list, any other occurrences of 0x0000 **MUST** be disregarded.

The length, structure, and definition of the IOAM Capabilities Query Container Header depend on the specific deployment environment.

3.2. IOAM Capabilities Response Container

For echo replies, the IOAM Capabilities Response uses a container that has the following format:

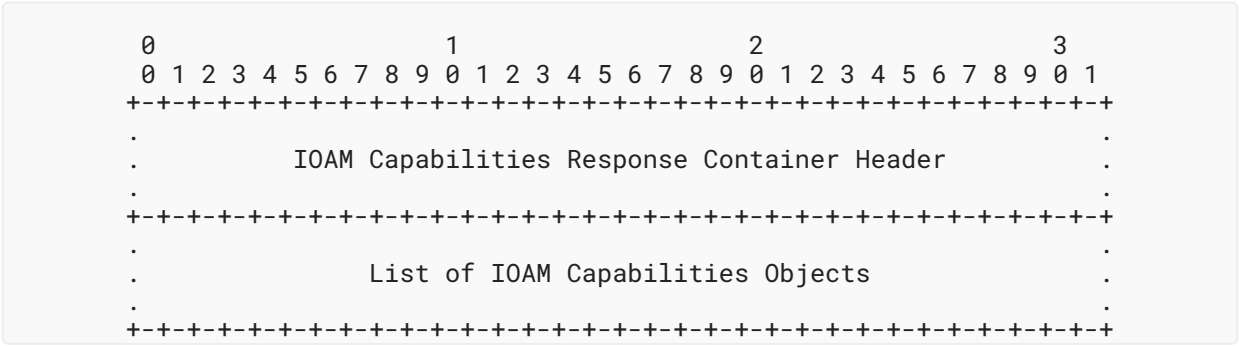

Figure 2: IOAM Capabilities Response Container for an Echo Reply

When this container is present in the echo reply sent by an IOAM transit node or IOAM decapsulating node, 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) that contains the enabled IOAM capabilities **MUST** be included in this container of the echo reply unless the sender encounters an error (e.g., no matched Namespace-ID).

The IOAM Capabilities Response Container has a container header that is used to identify the type and, optionally, the length of the container payload. The container header **MUST** be defined such that it falls on a 4-octet boundary.

The length, structure, and definition of the IOAM Capabilities Response Container Header depends on the specific deployment environment.

Based on the IOAM data fields defined in [RFC9197] and [RFC9326], 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 listed 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. The object payload **MUST** be defined such that it falls on a 4-octet boundary.

The length, structure, and definition of the object header depends on the specific deployment environment.

3.2.1. IOAM Pre-allocated Tracing Capabilities Object

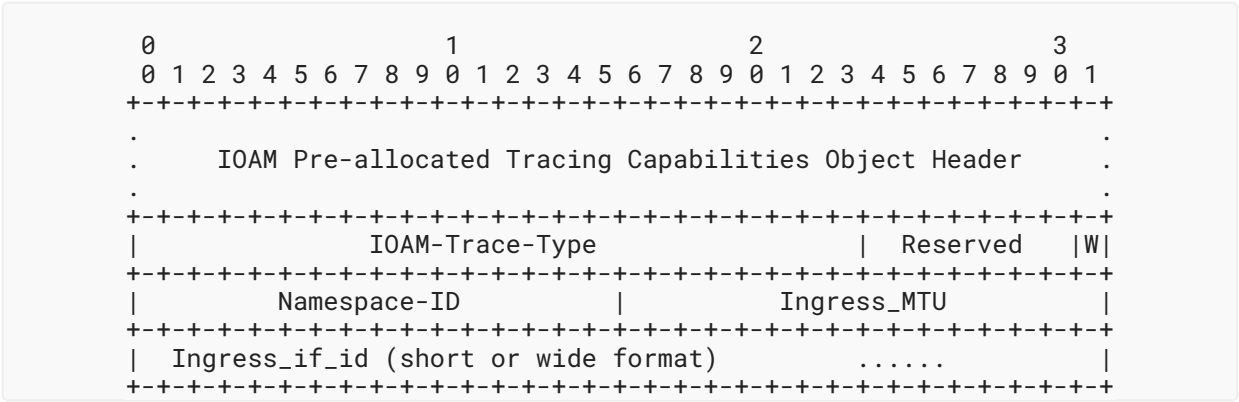

Figure 3: IOAM Pre-allocated Tracing Capabilities Object

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

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

The Reserved field **MUST** be zeroed on transmission and ignored on receipt.

The 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.

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

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

The 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 the echo request. If the W-bit is cleared, the Ingress_if_id field has 16 bits; then the 16 bits following the Ingress_if_id field are reserved for future use, **MUST** be set to zero, and **MUST** be ignored when non-zero.

3.2.2. IOAM Incremental Tracing Capabilities Object

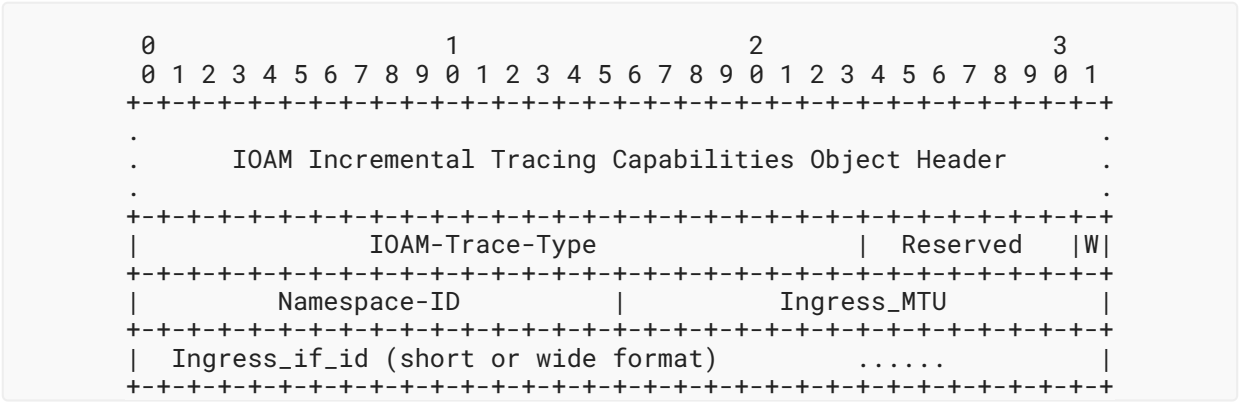

Figure 4: IOAM Incremental Tracing Capabilities Object

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

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

The Reserved field **MUST** be zeroed on transmission and ignored on receipt.

The 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.

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

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

The 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 the echo request. If the W-bit is cleared, the Ingress_if_id field has 16 bits; then the 16 bits following the Ingress_if_id field are reserved for future use, **MUST** be set to zero, and **MUST** be ignored when non-zero.

3.2.3. IOAM Proof of Transit Capabilities Object

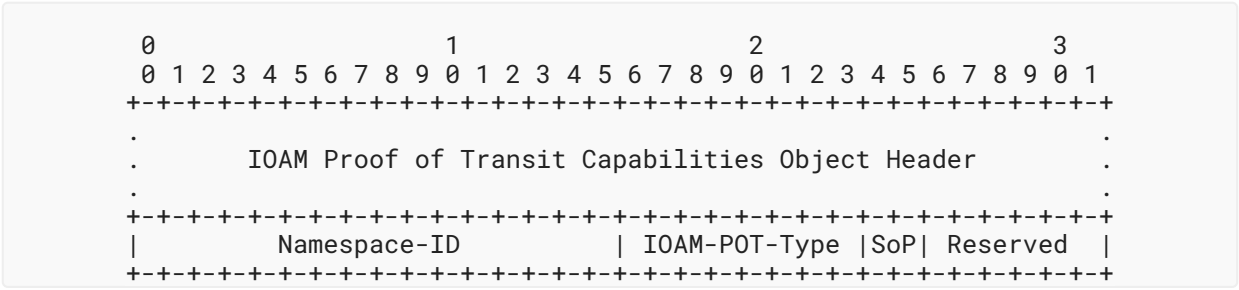

Figure 5: IOAM Proof of Transit Capabilities Object

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

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

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

The SoP (Size of POT) field has two bits that indicate the size of "PktID" and "Cumulative" data, which are specified in Section 4.5 of [RFC9197]. This document defines SoP as follows:

0b00: 64-bit "PktID" and 64-bit "Cumulative" data

0b01~0b11: reserved for future standardization

The Reserved field **MUST** be zeroed on transmission and ignored on receipt.

3.2.4. IOAM Edge-to-Edge Capabilities Object

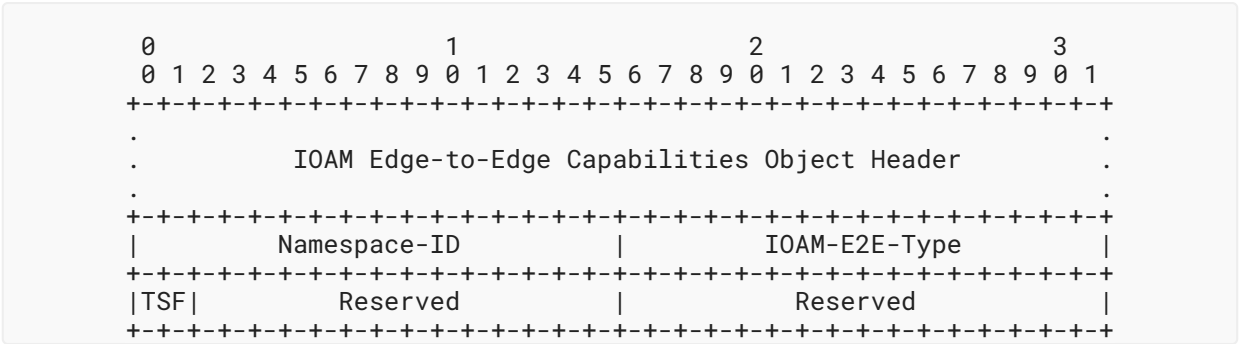

Figure 6: IOAM Edge-to-Edge Capabilities Object

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

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

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

The 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

The Reserved field **MUST** be zeroed on transmission and ignored on receipt.

3.2.5. IOAM DEX Capabilities Object

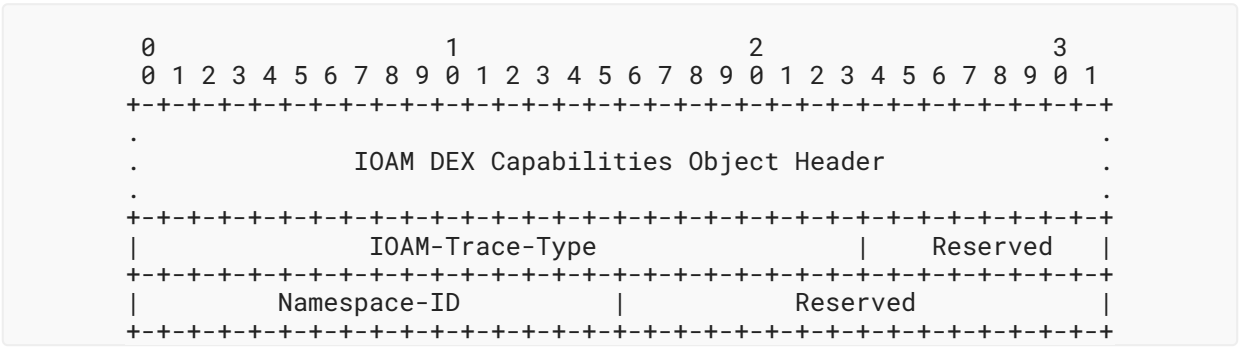

Figure 7: IOAM DEX Capabilities Object

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

The IOAM-Trace-Type field has the same definition as what's specified in Section 3.2 of [RFC9326].

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

The Reserved field **MUST** be zeroed on transmission and ignored on receipt.

3.2.6. IOAM End-of-Domain Object

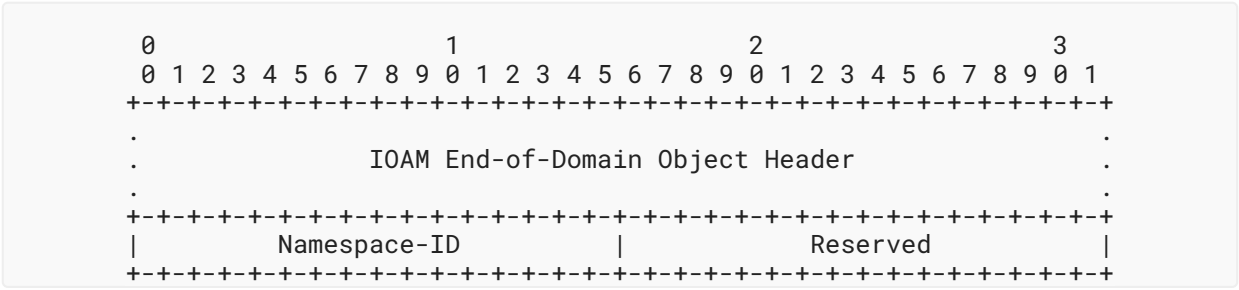

Figure 8: IOAM End-of-Domain Object

When the IOAM End-of-Domain Object is present in the IOAM Capabilities Response Container, 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 IOAM 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, including only the IOAM Edge-to-Edge Capabilities Object, not the IOAM End-of-Domain Object, is **RECOMMENDED**.

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

Reserved field **MUST** be zeroed on transmission and ignored on receipt.

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 as follows:

- First, with TTL equal to 1 to reach the closest node (which may or may not be an IOAM transit node).
- Then, with TTL equal to 2 to reach the second-nearest node (which also may or may not be an IOAM transit node).
- Then, 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 contains the IOAM Capabilities Response Container including the IOAM Edge-to-Edge Capabilities Object or the IOAM End-of-Domain Object).

As a result, the echo requests sent by the IOAM encapsulating node will reach all nodes one by one along the transport path of IOAM data packet.

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 is 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.

Note that the mechanism defined in this document applies to all kinds of IOAM option types, whether the four types of IOAM options defined in [RFC9197] or the DEX type of IOAM option defined in [RFC9326]. Specifically, when applied to the IOAM DEX option, the mechanism allows the IOAM encapsulating node to discover which nodes along the transport path support IOAM direct exporting and which trace data types are supported to be directly exported at these nodes.

5. IANA Considerations

IANA has created a registry named "In Situ OAM (IOAM) Capabilities".

This registry includes the following subregistries:

- IOAM SoP Capability
- IOAM TSF Capability

The subsequent subsections detail the registries herein contained.

Considering the Containers/Objects defined in this document that 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 four codepoints for the IOAM SoP Capability field for identifying the size of "PktID" and "Cumulative" data as explained in Section 4.5 of [RFC9197].

A new entry in this registry requires the following fields:

- SoP (Size of POT): a 2-bit binary field as defined in Section 3.2.3.
- Description: a terse description of the meaning of this SoP value.

The registry initially contains the following value:

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

Table 1: SoP and Description

0b01 - 0b11 are available for assignment via the IETF Review process as per [RFC8126].

5.2. IOAM TSF Capability Registry

This registry defines four code points for the IOAM TSF Capability field for identifying the timestamp format as explained in Section 5 of [RFC9197].

A new entry in this registry requires the following fields:

- TSF (TimeStamp Format): a 2-bit binary field as defined in Section 3.2.4.
- Description: a terse description of the meaning of this TSF value.

The registry initially contains the following values:

TSF	Description
0b00	PTP Truncated Timestamp Format
0b01	NTP 64-bit Timestamp Format
0b10	POSIX-based Timestamp Format

Table 2: TSF and Description

0b11 is available for assignment via the IETF Review 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, implementations **MUST** provide a means of checking the source addresses of echo request messages against an access list before accepting the message.

A deployment **MUST** ensure that border-filtering drops inbound echo requests with an IOAM Capabilities Container Header from outside of the domain and that drops outbound echo requests or replies with IOAM Capabilities Headers leaving the domain.

A deployment **MUST** support the configuration option to enable or disable the IOAM Capabilities Discovery feature defined in this document. By default, the IOAM Capabilities Discovery feature **MUST** be disabled.

The integrity protection on IOAM Capabilities information carried in echo reply messages can be achieved by the underlying 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 underlying transport of echo requests or replies 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].

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.

An implementation can also check whether the fields in received echo requests and replies strictly conform to the specifications, e.g., whether the list of IOAM Namespace-IDs includes duplicate entries and whether the received Namespace-ID is an operator-assigned or IANA-assigned one, once a check fails, an exception event indicating the checked field should be reported to the management.

Except for what's described above, the security issues discussed in [RFC9197] provide good guidance on implementation of this specification.

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>.

[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>.

[RFC9326] Song, H., Gafni, B., Brockners, F., Bhandari, S., and T. Mizrahi, "In Situ Operations, Administration, and Maintenance (IOAM) Direct Exporting", RFC 9326, DOI 10.17487/RFC9326, November 2022, <https://www.rfc-editor.org/info/rfc9326>.

7.2. Informative References

[BIER-PING] Nainar, N. K., Pignataro, C., Akiya, N., Zheng, L., Chen, M., and G. Mirsky, "BIER Ping and Trace", Work in Progress, Internet-Draft, draft-ietf-bier-ping-08, 6 March 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-bier-ping-08>.

[OAM-for-SFC] 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://datatracker.ietf.org/doc/html/draft-ietf-sfc-multi-layer-oam-22>.

[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>.

Acknowledgements

The authors would like to acknowledge Tianran Zhou, Dhruv Dhody, Frank Brockners, Cheng Li, Gyan Mishra, Marcus Ihlar, Martin Duke, Chris Lonvick, Éric Vyncke, Alvaro Retana, Paul Wouters, Roman Danyliw, Lars Eggert, Warren Kumari, John Scudder, Robert Wilton, Erik Kline, Zaheduzzaman Sarker, Murray Kucherawy, and Donald Eastlake 3rd 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.

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