Workgroup: ippm
Internet-Draft:
draft-ietf-ippm-ioam-deployment-05
Published: 4 January 2023
Intended Status: Informational
Expires: 8 July 2023
Authors: F. Brockners, Ed.
Cisco
S. Bhandari, Ed.
Thoughtspot
D. Bernier
Bell Canada
T. Mizrahi, Ed.
Huawei

In-situ OAM Deployment

Abstract

In-situ Operations, Administration, and Maintenance (IOAM) collects operational and telemetry information in the packet while the packet traverses a path between two points in the network. This document provides a framework for IOAM deployment and provides IOAM deployment considerations and guidance.

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 July 2023.

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

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

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

2. Conventions

Abbreviations used in this document:

BIER: Bit Index Explicit Replication [RFC8279]

Geneve: Generic Network Virtualization Encapsulation [RFC8926]

GRE: Generic Routing Encapsulation [RFC2784]

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

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

3. IOAM Deployment: Domains And Nodes

IOAM is focused on "limited domains" as defined in [RFC8799]. IOAM is not targeted for a deployment on the global Internet. The part of the network which employs IOAM is referred to as the "IOAM-Domain". For example, an IOAM-domain can include an enterprise campus using physical connections between devices or an overlay network using virtual connections / tunnels for connectivity between said devices. An IOAM-domain is defined by its perimeter or edge. The operator of an IOAM-domain is expected to put provisions in place to ensure that packets which contain IOAM data fields do not leak beyond the edge of an IOAM domain, e.g., using for example packet filtering methods. The operator should consider the potential operational impact of IOAM to mechanisms such as ECMP load-balancing schemes (e.g., load-balancing schemes based on packet length could be impacted by the increased packet size due to IOAM), path MTU (i.e., ensure that the

MTU of all links within a domain is sufficiently large to support the increased packet size due to IOAM) and ICMP message handling.

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

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

An "IOAM transit node" updates one or more of the IOAM-Data-Fields. If both the Pre-allocated and the Incremental Trace Option-Types are present in the packet, each IOAM transit node will update at most one of these Option-Types. Note that in case both Trace Option-Types are present in a packet, it is up to the IOAM data processing systems (see Section 6) to integrate the data from the two Trace Option-Types to form a view of the entire journey of the packet. 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. Please refer to Section 7.1 for a discussion of IOAM-Namespaces.

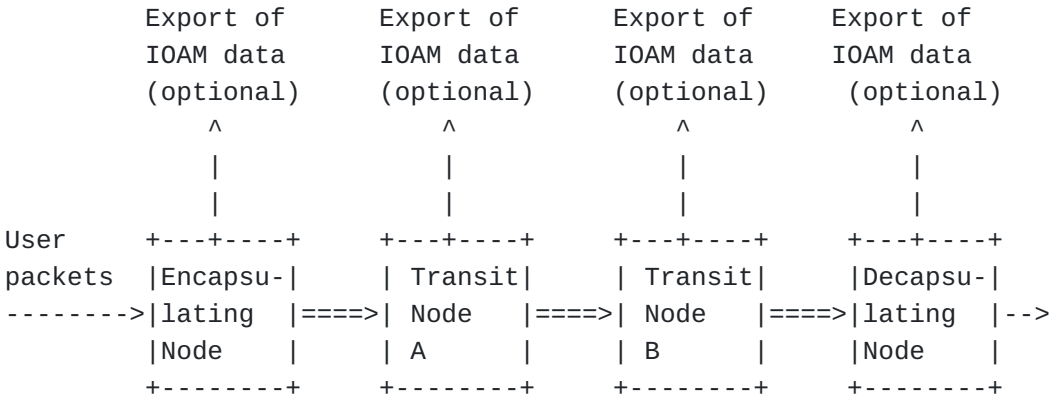

Figure 1: Roles of IOAM nodes

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.

*Per-hop tracing: OAM information about each IOAM node a packet traverses is collected and stored within the user data packet as the packet progresses through the IOAM domain. Potential uses of IOAM per-hop tracing include:

-Understand the different paths different packets traverse between an IOAM encapsulating and an IOAM decapsulating node in a network that uses load balancing such as Equal Cost Multi-Path (ECMP). This information could be used to tune the algorithm for ECMP for optimized network resource usage.

-Operations/Troubleshooting: Understand which path a particular packet or set of packets take as well as what amount of jitter and delay different IOAM nodes in the path contribute to the overall delay and jitter between the IOAM encapsulating node and the IOAM decapsulating node.

*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.

*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.

*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 for example collect the following types of information:

- 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.

- Identification of the interface that a packet was received on, i.e. ingress interface.

- Identification of the interface that a packet was sent out on, i.e. egress interface.

- 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.

- 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 geolocation 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.

- Information to detect whether IOAM trace data was added at every hop or whether certain hops in the domain weren't IOAM transit nodes.

- 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 [RFC9197].

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 could use methods like nested hashing or nested encryption of the IOAM data.

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 [RFC9197].

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 [RFC9197].

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 [RFC9326].

5. IOAM Encapsulations

IOAM data fields and associated data types for in-situ OAM are defined in [RFC9197]. The in-situ OAM data field can be transported by a variety of transport protocols, including NSH, Segment Routing, Geneve, BIER, IPv6, etc.

5.1. IPv6

IOAM encapsulation for IPv6 is defined in [I-D.ietf-ippm-ioam-ipv6-options], which also discussed IOAM deployment considerations for IPv6 networks

5.2. NSH

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

5.3. BIER

IOAM encapsulation for BIER is defined in [I-D.xzlnp-bier-ioam].

5.4. GRE

IOAM encapsulation for GRE is outlined as part of the "EtherType Protocol Identification of In-situ OAM Data" in [I-D.weis-ippm-ioam-eth].

5.5. Geneve

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

5.6. Segment Routing

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

5.7. Segment Routing for IPv6

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

5.8. VXLAN-GPE

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

6. IOAM Data Export

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

Raw data export of IOAM data using IPFIX is discussed in [I-D.spiegel-ippm-ioam-rawexport]. "Raw export of IOAM data" refers to a mode of operation where a node exports the IOAM data as it is received in the packet. The exporting node neither interprets, aggregates nor reformats the IOAM data before it is exported. Raw export of IOAM data is to support an operational model where the processing and interpretation of IOAM data is decoupled from the operation of encapsulating/updating/decapsulating IOAM data, which is also referred to as IOAM data-plane operation. The figure below shows the separation of concerns for IOAM export: Exporting IOAM data is performed by the "IOAM node" which performs IOAM data-plane operation, whereas the interpretation of IOAM data is performed by one or several IOAM data processing systems. The separation of concerns is to off-load interpretation, aggregation and formatting of IOAM data from the node which performs data-plane operations. In other words, a node which is focused on data-plane operations, i.e. forwarding of packets and handling IOAM data will not be tasked to also interpret the IOAM data, but can leave this task to another system or a set of systems. For scalability reasons, a single IOAM node could choose to export IOAM data to several IOAM data processing systems. Similarly, there several monitoring systems or analytics systems can be used to further process the data received from the IOAM preprocessing systems. Figure 2 shows an overview of IOAM export, including IOAM data processing systems and monitoring/ analytics systems.

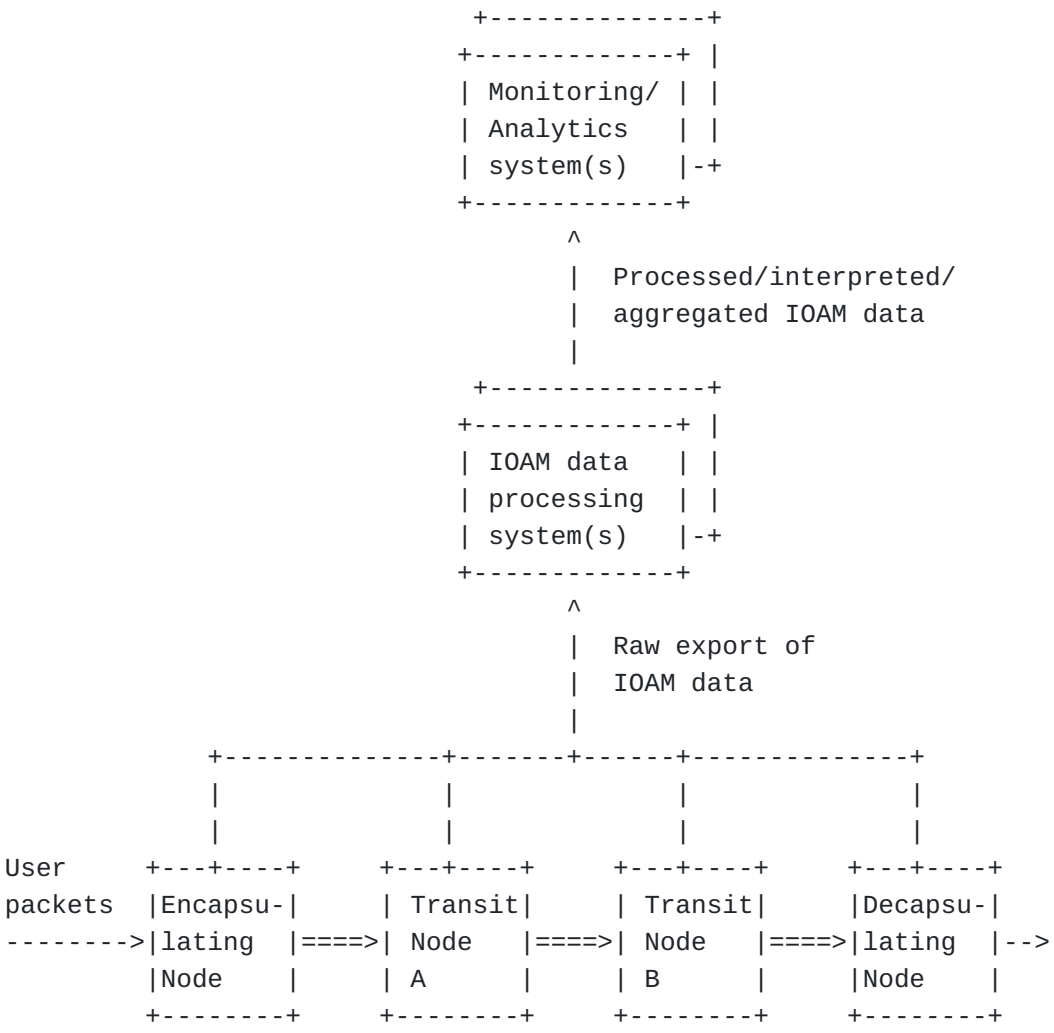

Figure 2: IOAM framework with data export

7. IOAM Deployment Considerations

This section describes several concepts of IOAM, and provides considerations that need to be taken to account when implementing IOAM in a network domain. This includes concepts like IOAM Namespaces, IOAM Layering, traffic-sets that IOAM is applied to and IOAM loopback mode. For a definition of IOAM Namespaces and IOAM layering, please refer to [RFC9197]. IOAM loopback mode is defined in [RFC9322].

7.1. IOAM Namespaces

IOAM-Namespaces add further context to IOAM-Option-Types and associated IOAM-Data-Fields. IOAM-Namespaces are defined in Section 4.3 of [RFC9197]. The Namespace-ID is part of the IOAM Option-Type definition, see e.g., Section 4.4 of [RFC9197] for IOAM Trace

Option-Types or Section 4.6 of [RFC9197] for the IOAM Edge-to-Edge Option-Type. IOAM-Namespaces support several uses:

* 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.

* 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. The Namespace-ID could also be used to distinguish between different types of interfaces: 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). Namespace-IDs could be used to distinguish between the different types of interfaces.

* IOAM-Namespaces can be used to identify different sets of devices (e.g., different types of devices) in a deployment: If an operator desires to insert different IOAM-Data-Fields based on the device, the devices could be grouped into multiple IOAM-Namespaces. This could be due to the fact that the IOAM feature set differs between different sets of devices, or it could be for reasons of optimized space usage in the packet header. It could also stem from hardware or operational limitations on the size of the trace data that can be added and processed, preventing collection of a full trace for a flow.

  - Assigning different IOAM Namespace-IDs to different sets of nodes or network partitions and using the Namespace-ID as a selector at the IOAM encapsulating node, a full trace for a flow could be collected and constructed via partial traces in different packets of the same flow. Example: An operator could choose to group the devices of a domain into two IOAM-Namespaces, in a way that at average, only every second hop

would be recorded by any device. To retrieve a full view of the deployment, the captured IOAM-Data-Fields of the two IOAM-Namespaces need to be correlated.

-Assigning different IOAM Namespace-IDs to different sets of nodes or network partitions and using a separate instance of an IOAM-Option-Type for each Namespace-ID, a full trace for a flow could be collected and constructed via partial traces from each IOAM-Option-Type in each of the packets in the flow. Example: An operator could choose to group the devices of a domain into two IOAM-Namespaces, in a way that each IOAM-Namespace is represented by one of two IOAM-Option-Types in the packet. Each node would record data only for the IOAM-Namespace that it belongs to, ignoring the other IOAM-Option-Type with a IOAM-Namespace to which it doesn't belong. To retrieve a full view of the deployment, the captured IOAM-Data-Fields of the two IOAM-Namespaces 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 of 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 of 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, but do not have to be associated to cross 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 3 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 the 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 the IOAM encapsulating node (into Geneve) and node D is the 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.

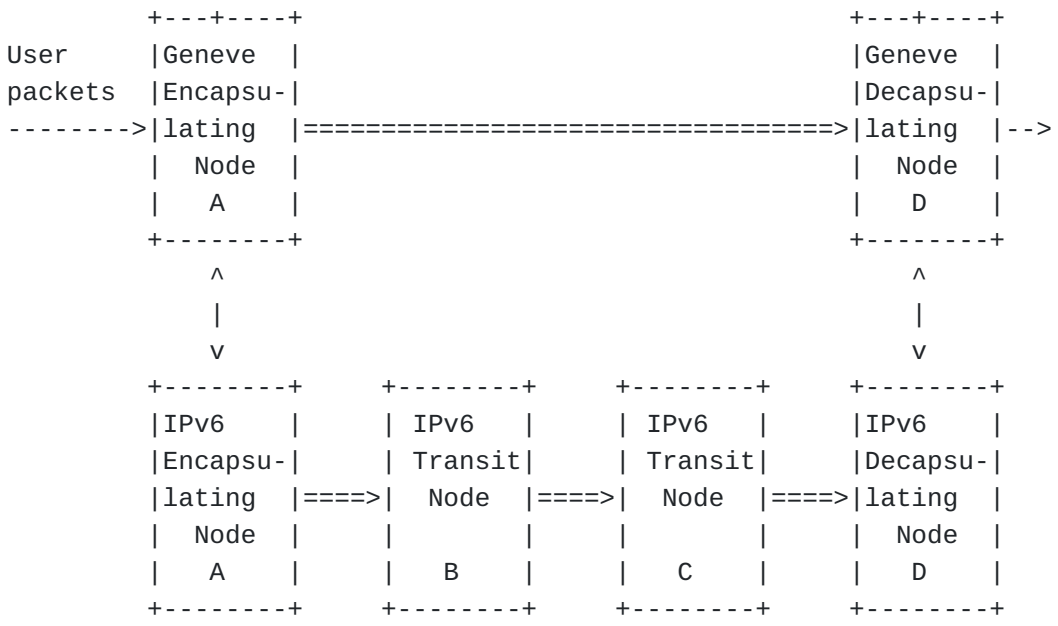

Figure 3: IOAM layering example

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.

Which IOAM Trace-Option-Types can be supported is not only a function of operator-defined configuration, but may also be limited by protocol constraints unique to a given encapsulating protocol. For encapsulating protocols which support both IOAM Trace-Option-Types, the operator decides by means of configuration which Trace-Option-Type(s) will be used for a particular domain. In this case, deployments can mix devices which include either the Incremental

Trace-Option-Type or the Pre-allocated Trace-Option-Type. If for example different types of packet forwarders and associated different types of IOAM implementations exist in a deployment and the encapsulating protocol supports both IOAM Trace-Option-Types, a deployment can mix devices which include either the Incremental Trace-Option-Type or the Pre-allocated Trace-Option-Type. 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 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 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. Loopack 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 [RFC9322].

7.6. IOAM Active Mode

The IOAM active mode flag indicates that a packet is an active OAM packet as opposed to regular user data traffic. Active mode is expected to be used for active measurement using IOAM. For details on IOAM active mode, please refer to [RFC9322].

Example use-cases for IOAM Active Mode include:

* Endpoint detailed active measurement: Synthetic probe packets are sent between the source and destination. These probe packets include a Trace Option-Type (i.e., either incremental or pre-allocated). Since the probe packets are sent between the endpoints, these packets are treated as data packets by the IOAM domain, and do not require special treatment at the IOAM layer. The source, which is also the IOAM encapsulating node can choose to set the Active flag, providing an explicit indication that these probe packets are meant for telemetry collection.

* IOAM active measurement using probe packets: Probe packets are generated and transmitted by an IOAM encapsulating node towards a destination which is also the IOAM decapsulating node. Probe packets include a Trace Option-Type (i.e., either incremental or pre-allocated) which has its Active flag set.

* IOAM active measurement using replicated data packets: Probe packets are created by an IOAM encapsulating node by selecting some or all of the en route data packets and replicating them. A

selected data packet that is replicated, and its (possibly truncated) copy is forwarded with one or more IOAM option, while the original packet is forwarded normally, without IOAM options. To the extent possible, the original data packet and its replica are forwarded through the same path. The replica includes a Trace Option-Type that has its Active flag set, indicating that the IOAM decapsulating node should terminate it. In this case the IOAM Active flag ensures that the replicated traffic is not forwarded beyond the IOAM domain.

7.7. Brown Field Deployments: IOAM Unaware Nodes

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

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

8. IOAM Manageability

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

A deployment can leverage IOAM profiles 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].

For deployments where the IOAM capabilities of a node are unknown, [I-D.ietf-ippm-ioam-conf-state] could be used to discover the enabled IOAM capabilities of nodes.

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 [RFC9322].

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 interconnected sites (for example, using an overlay network), the inter-site links are expected to be secured (e.g., by IPsec) in order to avoid external eavesdropping and introduction of malicious or false data. Another possible mitigation approach is to use the "direct exporting" mode [RFC9326]. 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 limiting the rate and/or the percentage of packets that an IOAM encapsulating node adds IOAM information to, as well as limiting rate and/or percentage of packets that an IOAM transit or an IOAM decapsulating node creates to export IOAM information extracted from the data packets that carry IOAM information.

Even though IOAM focused on limited domains [RFC8799], there might be deployments for which it is important for IOAM transit nodes and IOAM decapsulating nodes to know that the data received hadn't been tampered with. In those cases, the IOAM data should be integrity protected. Integrity protection of IOAM data fields is described in [I-D.ietf-ippm-ioam-data-integrity]. In addition, 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 limited network domains ([RFC8799]), thus confining the potential attack vectors to within the limited 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 the IOAM domain, and prevent an attacker from introducing malicious or false IOAM data to be processed and used within the IOAM domain. IOAM data leakage could lead to privacy issues. Consider an IOAM encapsulating node that is a home gateway in an operator's network. A home gateway is often identified with an individual, and revealing IOAM data such as "IOAM node identifier" or geolocation information outside of the limited domain could be harmful for that user. Note that the Direct Export mode [RFC9326] 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. Informative References

[I-D.ali-spring-ioam-srv6]
Ali, Z., Gandhi, R., Filsfils, C., Brockners, F., Nainar, N. K., Pignataro, C., Li, C., Chen, M., and G. Dawra, "Segment Routing Header encapsulation for In-situ OAM Data", Work in Progress, Internet-Draft, draft-ali-spring-ioam-srv6-06, 10 July 2022, <https://www.ietf.org/archive/id/draft-ali-spring-ioam-srv6-06.txt>.

[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", Work in Progress, Internet-Draft, draft-brockners-ippm-ioam-geneve-05, 19 November 2020, <https://www.ietf.org/archive/id/draft-brockners-ippm-ioam-geneve-05.txt>.

[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", Work in Progress, Internet-Draft, draft-brockners-ippm-ioam-vxlan-gpe-03, 4 November 2019, <https://www.ietf.org/archive/id/draft-brockners-ippm-ioam-vxlan-gpe-03.txt>.

[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", Work in Progress, Internet-Draft, draft-gandhi-spring-ioam-sr-mpls-02, 22 August 2019, <https://www.ietf.org/archive/id/draft-gandhi-spring-ioam-sr-mpls-02.txt>.

[I-D.ietf-ippm-ioam-conf-state] Min, X., Mirsky, G., and L. Bo, "Echo Request/Reply for Enabled In-situ OAM Capabilities", Work in Progress, Internet-Draft, draft-ietf-ippm-ioam-conf-state-10, 21 November 2022, <https://www.ietf.org/archive/id/draft-ietf-ippm-ioam-conf-state-10.txt>.

[I-D.ietf-ippm-ioam-data-integrity] Brockners, F., Bhandari, S., Mizrahi, T., and J. Iurman, "Integrity of In-situ OAM Data Fields", Work in Progress, Internet-Draft, draft-

ietf-ippm-ioam-data-integrity-03, 24 November 2022, <https://www.ietf.org/archive/id/draft-ietf-ippm-ioam-data-integrity-03.txt>.

[I-D.ietf-ippm-ioam-ipv6-options] Bhandari, S. and F. Brockners, "In-situ OAM IPv6 Options", Work in Progress, Internet-Draft, draft-ietf-ippm-ioam-ipv6-options-09, 11 October 2022, <https://www.ietf.org/archive/id/draft-ietf-ippm-ioam-ipv6-options-09.txt>.

[I-D.ietf-nvo3-vxlan-gpe] Maino, F., Kreeger, L., and U. Elzur, "Generic Protocol Extension for VXLAN (VXLAN-GPE)", Work in Progress, Internet-Draft, draft-ietf-nvo3-vxlan-gpe-12, 22 September 2021, <https://www.ietf.org/archive/id/draft-ietf-nvo3-vxlan-gpe-12.txt>.

[I-D.ietf-sfc-ioam-nsh] Brockners, F. and S. Bhandari, "Network Service Header (NSH) Encapsulation for In-situ OAM (IOAM) Data", Work in Progress, Internet-Draft, draft-ietf-sfc-ioam-nsh-11, 30 September 2022, <https://www.ietf.org/archive/id/draft-ietf-sfc-ioam-nsh-11.txt>.

[I-D.ietf-sfc-proof-of-transit] Brockners, F., Bhandari, S., Mizrahi, T., Dara, S., and S. Youell, "Proof of Transit", Work in Progress, Internet-Draft, draft-ietf-sfc-proof-of-transit-08, 1 November 2020, <https://www.ietf.org/archive/id/draft-ietf-sfc-proof-of-transit-08.txt>.

[I-D.mizrahi-ippm-ioam-profile]
Mizrahi, T., Brockners, F., Bhandari, S., Sivakolundu, R., Pignataro, C., Kfir, A., Gafni, B., Spiegel, M., and T. Zhou, "In Situ OAM Profiles", Work in Progress, Internet-Draft, draft-mizrahi-ippm-ioam-profile-06, 17 February 2022, <https://www.ietf.org/archive/id/draft-mizrahi-ippm-ioam-profile-06.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>.

[I-D.weis-ippm-ioam-eth]
Weis, B., Brockners, F., Hill, C., Bhandari, S., Govindan, V. P., Pignataro, C., Nainar, N. K., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Kfir, A., Gafni, B., Lapukhov, P., and M. Spiegel, "EtherType Protocol Identification of In-situ OAM Data", Work in Progress,

Internet-Draft, draft-weis-ippm-ioam-eth-05, 21 February 2022, <https://www.ietf.org/archive/id/draft-weis-ippm-ioam-eth-05.txt>.

[I-D.xzlnp-bier-ioam] Min, X., Zhang, Z., Liu, Y., Nainar, N. K., and C. Pignataro, "Bit Index Explicit Replication (BIER) Encapsulation for In-situ OAM (IOAM) Data", Work in Progress, Internet-Draft, draft-xzlnp-bier-ioam-04, 25 July 2022, <https://www.ietf.org/archive/id/draft-xzlnp-bier-ioam-04.txt>.

[I-D.zhou-ippm-ioam-yang] Zhou, T., Guichard, J., Brockners, F., and S. Raghavan, "A YANG Data Model for In-Situ OAM", Work in Progress, Internet-Draft, draft-zhou-ippm-ioam-yang-08, 30 July 2020, <https://www.ietf.org/archive/id/draft-zhou-ippm-ioam-yang-08.txt>.

[RFC2784] Farinacci, D., Li, T., Hanks, S., Meyer, D., Traina, P., and RFC Publisher, "Generic Routing Encapsulation (GRE)", RFC 2784, DOI 10.17487/RFC2784, March 2000, <https://www.rfc-editor.org/info/rfc2784>.

[RFC5905] Mills, D., Martin, J., Ed., Burbank, J., Kasch, W., and RFC Publisher, "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., Weingarten, Y., and RFC Publisher, "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. and RFC Publisher, "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., Pignataro, C., Ed., and RFC Publisher, "Service Function Chaining (SFC) Architecture", RFC 7665, DOI 10.17487/RFC7665, October 2015, <https://www.rfc-editor.org/info/rfc7665>.

[RFC7799] Morton, A. and RFC Publisher, "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., Mizrahi, T., and RFC Publisher, "Multipath Time Synchronization", RFC 8039, DOI 10.17487/RFC8039, December 2016, <https://www.rfc-editor.org/info/rfc8039>.

[RFC8279] Wijnands, IJ., Ed., Rosen, E., Ed., Dolganow, A., Przygienda, T., Aldrin, S., and RFC Publisher, "Multicast Using Bit Index Explicit Replication (BIER)", RFC 8279, DOI 10.17487/RFC8279, November 2017, <https://www.rfc-editor.org/info/rfc8279>.

[RFC8300] Quinn, P., Ed., Elzur, U., Ed., Pignataro, C., Ed., and RFC Publisher, "Network Service Header (NSH)", RFC 8300, DOI 10.17487/RFC8300, January 2018, <https://www.rfc-editor.org/info/rfc8300>.

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

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

[RFC8926] Gross, J., Ed., Ganga, I., Ed., Sridhar, T., Ed., and RFC Publisher, "Geneve: Generic Network Virtualization Encapsulation", RFC 8926, DOI 10.17487/RFC8926, November 2020, <https://www.rfc-editor.org/info/rfc8926>.

[RFC9197] Brockners, F., Ed., Bhandari, S., Ed., Mizrahi, T., Ed., and RFC Publisher, "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>.

[RFC9322] Mizrahi, T., Brockners, F., Bhandari, S., Gafni, B., Spiegel, M., and RFC Publisher, "In Situ Operations, Administration, and Maintenance (IOAM) Loopback and Active Flags", RFC 9322, DOI 10.17487/RFC9322, November 2022, <https://www.rfc-editor.org/info/rfc9322>.

[RFC9326] Song, H., Gafni, B., Brockners, F., Bhandari, S., Mizrahi, T., and RFC Publisher, "In Situ Operations, Administration, and Maintenance (IOAM) Direct Exporting",

RFC 9326, DOI 10.17487/RFC9326, November 2022, <https://www.rfc-editor.org/info/rfc9326>.

Authors' Addresses

Frank Brockners (editor)
Cisco Systems, Inc.
Hansaallee 249, 3rd Floor
40549 DUESSELDORF
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