Network Working Group F. Brockners
Internet-Draft S. Bhandari
Intended status: Experimental C. Pignataro
Expires: January 9, 2017 Cisco
H. Gredler
RtBrick Inc.
July 8, 2016

Data Formats for In-band OAM
draft-brockners-inband-oam-data-00

Abstract

In-band operation, administration and maintenance (OAM) records operational and telemetry information in the packet while the packet traverses a path between two points in the network. This document discusses the data types and data formats for in-band OAM data records. In-band OAM data records can be embedded into a variety of transports such as NSH, Segment Routing, VXLAN-GPE, native IPv6 (via extension header), or IPv4. In-band OAM is to complement current out-of-band OAM mechanisms based on ICMP or other types of probe packets.

Table of Contents

1. Introduction

This document defines data record types for "in-band" operation, administration, and maintenance (OAM). In-band OAM records OAM information within the packet while the packet traverses a particular network domain. The term "in-band" 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. A discussion of the motivation and requirements for in-band OAM can be found in [draft-brockners-inband-oam-requirements]. In-band OAM is to complement "out-of-band" or "active" mechanisms such as ping or traceroute, or more recent active probing mechanisms as described in [I-D.lapukhov-dataplane-probe]. In-band OAM mechanisms can be leveraged where current out-of-band mechanisms do not apply or do not offer the desired results, such as proving that a certain set of traffic 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 where probe traffic is potentially handled differently from regular data traffic by the network devices.

This document defines the data types and data formats for in-band OAM data records. The in-band OAM data records can be transported by a variety of transport protocols, including NSH, Segment Routing, VXLAN-GPE, IPv6, IPv4. Encapsulation details for these different transport protocols are outside the scope of this document.

2. Conventions

Abbreviations used in this document:

MTU: Maximum Transmit Unit

OAM: Operations, Administration, and Maintenance

SR: Segment Routing

SID: Segment Identifier

NSH: Network Service Header

SFC: Service Function Chain

TLV: Type-Length-Value

VXLAN-GPE: Virtual eXtensible Local Area Network, Generic Protocol Extension

3. In-band OAM Data Types and Data Format

This section defines in-band OAM data types and data formats of the data records required for in-band OAM. The different uses of in-band OAM require the definition of different types of data. The in-band OAM data format for the data being carried corresponds to the three main categories of in-band OAM data defined in [draft-brockners-inband-oam-requirements], which are edge-to-edge, per node, and for selected nodes only.

Transport options for in-band OAM data are found in [draft-brockners-inband-oam-transport]. In-band OAM data is defined as options in Type-Length-Value (TLV) format. The TLV format for each of the three different types of in-band OAM data is defined in this document.

In-band OAM is expected to be deployed in a specific domain rather than on the overall Internet. The part of the network which employs in-band OAM is referred to as "in-band OAM-domain". In-band OAM data is added to a packet on entering the in-band OAM-domain and is removed from the packet when exiting the domain. Within the in-band

OAM-domain, the in-band OAM data may be updated by network nodes that the packet traverses. The device which adds in-band OAM data to the packet is called the "in-band OAM encapsulating node", whereas the device which removed the in-band OAM data is referred to as the "in-band OAM decapsulating node". Nodes within the domain which are aware of in-band OAM data and read and/or write or process the in-band OAM data are called "in-band OAM transit nodes". Note that not every node in an in-band OAM domain needs to be an in-band OAM transit node. For example, a Segment Routing deployment might require the segment routing path to be verified. In that case, only the SR nodes would also be in-band OAM transit nodes rather than all nodes.

3.1. In-band OAM Tracing Option

"In-band OAM tracing data" is expected to be collected at every hop that a packet traverses, i.e., in a typical deployment all nodes in an in-band OAM-domain would participate in in-band OAM and thus be in-band OAM transit nodes, in-band OAM encapsulating or in-band OAM decapsulating nodes. The network diameter of the in-band OAM domain is assumed to be known. For in-band OAM tracing, the in-band OAM encapsulating node allocates an array which is to store operational data retrieved from every node while the packet traverses the domain. Every entry is to hold information for a particular in-band OAM transit node that is traversed by a packet. In-band OAM transit nodes update the content of the array. A pointer which is part of the in-band OAM trace data points to the next empty slot in the array, which is where the next in-band OAM transit node fills in its data. The in-band OAM decapsulating node removes the in-band OAM data and process and/or export the metadata. In-band OAM data uses its own name-space for information such as node identifier or interface identifier. This allows for a domain-specific definition and interpretation. For example: In one case an interface-id could 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).

The following in-band OAM data is defined for in-band OAM tracing:

o Identification of the in-band OAM node. An in-band OAM node identifier can match to a device identifier or a particular control point or subsystem within a device.

o Identification of the interface that a packet was received on.

o Identification of the interface that a packet was sent out on.

- Time of day when the packet was processed by the node. Different definitions of processing time are feasible and expected, though it is important that all devices of an in-band 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 deployment, all in-band OAM nodes should interpret the generic data the same way. Examples for generic in-band OAM data include geo-location information (location of the node at the time the packet was processed), buffer queue fill level or cache fill level at the time the packet was processed, or even a battery charge level.

- A mechanism to detect whether in-band OAM trace data was added at every hop or whether certain hops in the domain weren't in-band OAM transit nodes.

The "Node data List" array in the packet is populated iteratively as the packet traverses the network, starting with the last entry of the array, i.e., "Node data List [n]" is the first entry to be populated, "Node data List [n-1]" is the second one, etc.

In-band OAM Tracing Option:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Option Type  |  Opt Data Len | OAM-trace-type| Elements-left |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|                                                               |  |
|                        Node data List [0]                     |  |
|                                                               |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  D
|                                                               |  a
|                        Node data List [1]                     |  t
|                                                               |  a
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                               .                               .  S
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  p
|                                                               |  a
|                        Node data List [n-1]                   |  c
|                                                               |  e
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
|                                                               |  |
|                        Node data List [n]                     |  |
|                                                               |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
```

Option Type: 8-bit identifier of the type of option. Option number is defined based on the encapsulation protocol.

Opt Data Len: 8-bit unsigned integer. Length of the Option Data field of this option, in octets.

OAM-trace-type: 8-bit identifier of a particular trace element variant.

The trace type value can be interpreted as a bit field. The following bit fields are defined in this document, with details on each field described in the next section. The order of packing the trace data in each Node-data element follows the bit order for setting each trace data element. Only a valid combination of these fields defined in this document are valid in-band OAM-trace-types.

Bit 0   When set indicates presence of node_id in the Node data.

Bit 1 When set indicates presence of ingress_if_id in the Node data.

Bit 2 When set indicates presence of egress_if_id in the Node data.

Bit 3 When set indicates presence of timestamp in the Node data.

Bit 4 When set indicates presence of app_data in the Node data.

Bit 5-7 Undefined in this document.

Section 3.1.1 describes the format of a number of trace types. Specifically, it exemplifies OAM-trace-types 0x00011111, 0x00000111, 0x00001001, 0x00010001, and 0x00011001.

Elements-left: 8-bit unsigned integer. A pointer that indicates the next data recording point in the data space of the packet in octets. It is the index into the "Node data List" array shown above.

Node data List [n]: Variable-length field. The format of which is determined by the OAM Type representing the n-th Node data in the Node data List. The Node data List is encoded starting from the last Node data of the path. The first element of the node data list (Node data List [0]) contains the last node of the path while the last node data of the Node data List (Node data List[n]) contains the first Node data of the path traced. The index contained in "Elements-left" identifies the current active Node data to be populated.

3.1.1. In-band OAM Trace Type and Node Data Element

An entry in the "Node data List" array can have different formats, following the needs of the a deployment. Some deployments might only be interested in recording the node identifiers, whereas others might be interested in recording node identifier and timestamp. The section defines different formats that an entry in "Node data List" can take.

Node data has the following format:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Hop_Lim     |    <trace-data elements packed as indicated   ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~             by in-band OAM-trace-type bits> .....             ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

0x00011111: In-band OAM-trace-type is 0x00011111 then the format of node data is:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Hop_Lim     |              node_id                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     ingress_if_id             |         egress_if_id          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           timestamp                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            app_data                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

0x00000111: In-band OAM-trace-type is 0x00000111 then the format is:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Hop_Lim     |              node_id                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     ingress_if_id             |         egress_if_id          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

0x00001001: In-band OAM-trace-type is 0x00001001 then the format is:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Hop_Lim     |              node_id                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           timestamp                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

0x00010001: In-band OAM-trace-type is 0x00010001 then the format is:

```
        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
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |   Hop_Lim     |              node_id                          |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                            app_data                           |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

0x00011001: In-band OAM-trace-type is 0x00011001 then the format is:

```
        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
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |   Hop_Lim     |              node_id                          |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                           timestamp                           |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                            app_data                           |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Trace data elements in Node data are defined as follows:

Hop_Lim: 1 octet Hop limit that is set to the TTL value in the packet at the node that records this data.

node_id: Node identifier node_id is a 3 octet field to uniquely identify a node within in-band OAM domain. The procedure to allocate, manage and map the node_ids is beyond the scope of this document.

ingress_if_id: 2 octet interface identifier to record the ingress interface the packet was received on.

egress_if_id: 2 octet interface identifier to record the egress interface the packet is forwarded out of.

timestamp: 4 octet timestamp when packet has been processed by the node.

app_data: 4 octet placeholder which can be used by the node to add application specific data.

Hop Limit information is used to identify the location of the node in the communication path.

3.2. In-band OAM Proof of Transit Option

In-band OAM Proof of Transit data is to support the path or service function chain [RFC7665] verification use cases. Proof-of-transit uses methods like nested hashing or nested encryption of the in-band OAM data or mechanisms such as Shamir's Secret Sharing Schema (SSSS). While details on how the in-band OAM data for the proof of transit option is processed at in-band OAM encapsulating, decapsulating and transit nodes are outside the scope of the document, all of these approaches share the need to uniquely identify a packet as well as iteratively operate on a set of information that is handed from node to node. Correspondingly, two pieces of information are added as in-band OAM data to the packet:

o Random: Unique identifier for the packet (e.g., 64-bits allow for the unique identification of 2^64 packets).

o Cumulative: Information which is handed from node to node and updated by every node according to a verification algorithm.

In-band OAM Proof of Transit option:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Option Type  |  Opt Data Len |   POT type = 0 |F|  reserved   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|                          Random                               |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  P
|                       Random(contd)                           |  O
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  T
|                         Cumulative                            |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
|                         Cumulative (contd)                    |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
```

Option Type: 8-bit identifier of the type of option.

Opt Data Len: 8-bit unsigned integer. Length of the Option Data field of this option, in octets.

POT Type: 8-bit identifier of a particular POT variant that dictates the POT data that is included.

* 16 Octet field as described below

Flag (F): 1-bit. Indicates which POT-profile is active. 0 means the even POT-profile is active, 1 means the odd POT-profile is active.

Reserved: 7-bit. (Reserved Octet) Reserved octet for future use.

Random: 64-bit Per packet Random number.

Cumulative: 64-bit Cumulative that is updated at specific nodes by processing per packet Random number field and configured parameters.

Note: Larger or smaller sizes of "Random" and "Cumulative" data are feasible and could be required for certain deployments (e.g. in case of space constraints in the transport protocol used). Future versions of this document will address different sizes of data for "proof of transit".

3.3. In-band OAM Edge-to-Edge Option

The in-band OAM Edge-to-Edge Option is to carry data which is to be interpreted only by the in-band OAM encapsulating and in-band OAM decapsulating node, but not by in-band OAM transit nodes.

Currently only sequence numbers use the in-band OAM Edge-to-Edge option. In order to detect packet loss, packet reordering, or packet duplication in an in-band OAM-domain, sequence numbers can be added to packets of a particular tube (see [I-D.hildebrand-spud-prototype]). Each tube leverages a dedicated namespace for its sequence numbers.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Option Type  |  Opt Data Len | OAM-E2E-Type  |    reserved   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      E2E Option data format determined by iOAM-E2E-Type       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Option Type: 8-bit identifier of the type of option.

Opt Data Len: 8-bit unsigned integer. Length of the Option Data field of this option, in octets.

iOAM-E2E-Type: 8-bit identifier of a particular in-band OAM E2E variant.

0: E2E option data is a 64-bit sequence number added to a specific tube which is used to identify packet loss and reordering for that tube.

Reserved: 8-bit. (Reserved Octet) Reserved octet for future use.

4. In-band OAM Data Export

In-band OAM nodes collect information for packets traversing a domain that supports in-band OAM. The device at the domain edge (which could also be an end-host) which receives a packet with in-band OAM information chooses how to process the in-band OAM data collected within the packet. This decapsulating node can simply discard the information collected, can process the information further, or export the information using e.g., IPFIX.

The discussion of in-band OAM data processing and export is left for a future version of this document.

5. IANA Considerations

IANA considerations will be added in a future version of this document.

6. Manageability Considerations

Manageability considerations will be addressed in a later version of this document..

7. Security Considerations

Security considerations will be addressed in a later version of this document. For a discussion of security requirements of in-band OAM, please refer to [draft-brockners-inband-oam-requirements].

8. Acknowledgements

The authors would like to thank Steve Youell, Eric Vyncke, Nalini Elkins, Srihari Raghavan, Ranganathan T S, Karthik Babu Harichandra Babu, Akshaya Nadahalli, and Andrew Yourtchenko for the comments and advice. This document leverages and builds on top of several concepts described in [draft-kitamura-ipv6-record-route]. The authors would like to acknowledge the work done by the author Hiroshi Kitamura and people involved in writing it.

9. References

9.1. Normative References

[draft-brockners-inband-oam-requirements]
Brockners, F., Bhandari, S., and S. Dara, "Requirements for in-band OAM", July 2016.

9.2. Informative References

[draft-brockners-inband-oam-transport]
Brockners, F., Bhandari, S., Pignataro, C., and H. Gredler, "Encapsulations for in-band OAM", July 2016.

[draft-brockners-proof-of-transit]
Brockners, F., Bhandari, S., and S. Dara, "Proof of transit", July 2016.

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

[FD.io] "Fast Data Project: FD.io", <https://fd.io/>.

[I-D.hildebrand-spud-prototype]
Hildebrand, J. and B. Trammell, "Substrate Protocol for User Datagrams (SPUD) Prototype", draft-hildebrand-spud-prototype-03 (work in progress), March 2015.

[I-D.lapukhov-dataplane-probe]
Lapukhov, P. and r. remy@barefootnetworks.com, "Data-plane probe for in-band telemetry collection", draft-lapukhov-dataplane-probe-01 (work in progress), June 2016.

[P4] Kim, , "P4: In-band Network Telemetry (INT)", September 2015.

[RFC7665] Halpern, J., Ed. and C. Pignataro, Ed., "Service Function Chaining (SFC) Architecture", RFC 7665, DOI 10.17487/RFC7665, October 2015, <http://www.rfc-editor.org/info/rfc7665>.

Authors' Addresses

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

Email: fbrockne@cisco.com

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

Email: shwethab@cisco.com

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

Email: cpignata@cisco.com

Hannes Gredler
RtBrick Inc.

Email: hannes@rtbrick.com