SPRING Working Group
Internet Draft
Intended status: Standards Track
Expires: December 2016

Dave Allan
Ericsson
Jeff Tantsura

June 2016

A Framework for Computed Multicast applied to MPLS based Segment Routing
draft-allan-spring-mpls-multicast-framework-01

Abstract

This document describes a multicast solution for Segment Routing with MPLS data plane. It is consistent with the Segment Routing architecture in that an IGP is augmented to distribute information in addition to the link state. In this solution it is multicast group membership information sufficient to synchronize state in a given network domain. Computation is employed to determine the topology of any loosely specified multicast distribution tree.

Status of this Memo

Copyright and License Notice

Table of Contents

1. Introduction...3
1.1. Authors..3
1.2. Requirements Language......................................3
2. Conventions used in this document............................3
2.1. Terminology..3
3. Solution Overview..4
3.1. Mapping source specific trees onto the segment routing architecture...5
3.2. Role of the Routing System.................................5
3.3. MDT Construction Requirements..............................6
3.4. Pruning - theory of operation..............................6
4. Elements of Procedure..7
4.1. Triggers for Computation...................................7
4.2. FIB Determination..7
4.2.1. Information in the IGP...................................7
4.2.2. Computation of individual segments.......................8
4.3. FIB Generation..10
4.4. FIB installation..11
5. Related work..11
5.1. IGP Extensions..11
5.2. BGP Extensions..11
6. Observations..12
7. Acknowledgements..12
8. Security Considerations.....................................12
9. IANA Considerations...12
10. References...12
10.1. Normative References.....................................12
10.2. Informative References...................................12
11. Authors' Addresses...13

1. Introduction

This memo describes a solution for multicast for Segment Routing with MPLS data plane in which source specific multicast distribution trees (MDTs) are computed from information distributed via an IGP. Computation can use information in the IGP to determine if a given node in the network has a role as a root, leaf or replication point in a given MDT. Unicast tunnels are employed to interconnect the nodes determined to have a role. Therefore state only need be installed in nodes that have one of these three roles to fully instantiate an MDT.
Although this approach is computationally intensive, a significant amount of computation can be avoided when the computing agent determines that the node it is computing for has no role in a given MDT. This permits a computed approach to multicast convergence to be computationally tractable.

1.1. Authors

Dave Allan, Jeff Tantsura

1.2. Requirements Language

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

2. Conventions used in this document

2.1. Terminology

Candidate replication point - is a node that potentially needs to install state to replicate multicast traffic as determined at an intermediate step in multicast segment computation. It will either resolve to having no role or a role as a replication point once multicast has converged.

Candidate role - refers to any potential combination of roles on a given multicast segment as determined at some intermediate step in MDT computation. For example, a node with a candidate role may be a leaf and may be a candidate replication point.

Downstream - refers to the direction along the shortest path to one or more leaves for a given multicast distribution tree

Multicast convergence - is when all computation and state installation to ensure the FIB reflects the multicast information in the IGP is complete.

MDT - multicast distribution tree. Is a tree composed of one or more multicast segments.

Multicast segment - is a portion of the multicast tree where only the root and the leaves have been specified, and computation based upon the current state of the IGP database is employed to determine and install the required state to implement the segment. For MPLS a multicast segment is implemented as a p2mp LSP. A multicast segment is identified by a multicast SID.

Multicast SID - Is the data plane identifier that is used to implement a multicast segment. As per a unicast MPLS segment, the rightmost 20 bits of a multicast SID is encoded as a label. It is drawn from an SRGB that is global to the SR domain.

Pinned path - Is a unique shortest path extending from a leaf upstream towards the root for a given multicast segment. Therefore is a component of the multicast segment that it has been determined must be there. It will not necessarily extend from the leaf all the way to the root during intermediate computation steps. A pinned path can result from pruning operations.

Role - refers specifically to a node that is either a root, a leaf, a replication node, or a pinned waypoint for a given MDT.

Unicast convergence - is when all computation and state installation to ensure the FIB reflects the unicast information in the IGP is complete.

Upstream - refers to the direction along the shortest path to the root of a given MDT.

3. Solution Overview

This memo describes a multicast architecture in which multicast state is only installed in those nodes that have roles as a root, leaves, and replication points for a given multicast segment. The a-priori established segment routing unicast tunnels are used as interconnect between the nodes that have a role in a given multicast SID.

A loosely specified MDT is composed of a single multicast segment and the routing of the MDT is delegated entirely to computation driven by information in the IGP database.

Explicitly routed MDTs are expressed as a tree of concatenated multicast segments where both the leaves of each segment and the waypoints coupling a given segment to the upstream and/or downstream segment(s) is specified in information flooded in the IGP by the overall root of the MDT. The segments themselves will be computed as per a loosely specified MDT.

A PE acting as an overall root for a given tree is expected to be configured by the operator as to where to source multicast traffic from, be it an attachment circuit, interworking function for client technology or other. Similarly a leaf for a given tree is expected to be configured by the operator as to the disposition of received multicast traffic.

A computed segment is guaranteed to be loop free in a stable system. A concatenation of segments to construct an MDT will similarly be loop free as any collision of segments can be disambiguated in the data plane via the SIDs.

This architecture significantly reduces the amount of state that needs to be installed in the data plane to support multicast. This also means that the impact of many failures in the network on multicast traffic distribution will be recovered by unicast local repair or unicast convergence with subsequent multicast convergence acting in the role of network re-optimization (as opposed to restoration).

3.1. Mapping source specific trees onto the segment routing architecture

A computed source specific tree for a given multicast group corresponds to one or more multicast segments in the SR architecture. Each multicast segment is assigned a SID, typically by management configuration of the node that will be the overall root for the source specific tree. The root node then uses the IGP to advertise this information to all nodes in the IGP area/domain.

A multicast group is implemented as the set of source specific trees from all nodes that have registered transmit interest to all nodes that have registered receive interest in a multicast group.

3.2. Role of the Routing System

The role of the IGP is to communicate topology information, multicast capability and associated algorithm, multicast registrations, unicast to SID bindings, multicast to SID bindings and waypoints in multi-segment MDTs. No changes to topology or unicast to SID binding advertisements are proposed by this memo.

The multicast registrations/bindings will be in the form of source, group, transmit/receive interest and the SID to use for the source specific multicast tree. Registrations are originated by any node that has send or receive interest in a given multicast group. Nodes will use the combination of topology and multicast registrations to determine the nodes that have a role in each source specific tree and the SID information to then derive the required FIB state.

3.3. MDT Construction Requirements

A multicast segment in an MDT is constructed such that between any pair of nodes that have a role in the segment and are connected by a unicast tunnel, there is not another node on the shortest path between the two with a role in that segment. This ensures that copies of a packet forwarded by an multicast segment will traverse a link only once in a stable system.

Note that this can be satisfied by a minimum cost shortest path tree, but is not an absolute requirement. The pruning rules specified in this memo will meet this requirement without necessarily producing absolutely minimum cost multicast segment (or incurring the associated computational cost).

3.4. Pruning - theory of operation

The role of nodes in a given multicast segment is determined by first producing an inclusive shortest path tree with all possible paths between the root and leaves, and then applying a set of pruning rules repeatedly until an acyclic tree is produced or no further prunes are possible.

For the majority of multicast segments these rules will authoritatively produce a minimum cost tree. For those segments that have not yet been authoritatively resolved, there is a set of pruning operations applied that are not guaranteed to produce a tree that meets the requirements of 3.3, therefore these trees require auditing and potential correction according to a further set of agreed rules. This avoids the necessity of an exhaustive search of the solution space.

A node during computation of a segment may conclude that it will absolutely not have a role at any of numerous points in the computation process and abandon computation of that segment.

4. Elements of Procedure

4.1. Triggers for Computation

MDT computation is triggered by changes to the IGP database. These are in the form of either changes in registered multicast group interest, addition or removal of a multi-segment MDT descriptor, or topology changes.

A change in registered interest for a group will require re-computation of all MDTs that implement the multicast group.

A topology change will require the computation of some number of multicast segments, the actual number will depend on the implementation of tree computation but at a minimum will be all trees for which there is not an optimal shortest path solution as a result of the topology change.

4.2. FIB Determination

4.2.1. Information in the IGP

Group membership information for a multicast segment is obtained from the IGP. This is true for single segment MDTs as well as multi-segment MDTs. Included in the multi-segment MDT specification is the waypoint nodes in MDT and the upstream and downstream SIDs. The specified node is expected to cross connect the SIDs to join the segments together acting in the role of leaf for the upstream segment and root for the downstream segment.

When a waypoint in an MDT descriptor does not exist in the IGP, the assumption is that the node identified by the waypoint SID has failed. The response of the other nodes in the system in FIB determination is to add the leaves of the downstream segment to the upstream segment.

An example of this would be consider a node "x", and another node "y". At some point in time, "x" advertises a tree that identifies "y" as a waypoint that cross connects upstream SID "a" to downstream SID "b". At some later point node "y" fails. The other nodes in the network will compute segment "a" as if it included all leaves and waypoints in segment "b". All apriori state installed for segment "b" would be removed as the failure of "y" has required "b" to be subsumed by "a".

4.2.2. Computation of individual segments

FIB generation for a multicast segment is the result of computation, ultimately as applied to all source specific trees in the network. All computing nodes implement a common algorithm for tree generation, as all MUST agree on the solution.

One algorithm is as follows:

All possible shortest paths to the set of leaves for the MDT is determined. Then pruning rules are repeatedly applied until no further prunes are possible.

The philosophy of the application of these rules could be expressed as "simplify as much as possible, and prune that which cannot be". The rules are:

1) Eliminate any links and nodes not on a potential shortest path from the root to the leaves for the MDT under consideration.

2) Simplify via the replacement of any nodes that do not have a potential role in the MDT with links.

   This will be nodes that are not a leaf, a root or a candidate replication point. For example:

```
Root---------A----------B
```

   B is a leaf. A is not but is in a potential shortest path from root to B. However A will have no role in the MDT that serves B as it provides simple transit therefore is replaced with a direct connection between the root and B.

```
Root--------------------B
```

   Note that such pruning also needs to avoid the creation of duplicate parallel links. For example:

```
   /----------A----------\

Root                       B

   \----------C----------/
```

   Where A and C have no role and the cost root-A-B = cost root-C-B, they can be replaced with a single link from Root to B.

3) Simplify via the elimination of fewer hop paths

When for a given set of leaves, a node has multiple downstream links that converge on a common downstream point, and that set of leaves is only a subset of the leaves reachable on one or more of the links, any link that only serves that subset of leaves can be pruned.

For example:

```
--A---------------------------B

   \                         /

     -----------C-----------

                 \

                   ----D
```

Link AB is cost 2, link AC and CB are cost 1 (cost of link CD does not affect the example).

B and D are leaves of a root upstream of A. From A, link AB can reach leaf B. Path AC can reach leaf B and D. In this case path A-B can be pruned from consideration. The set of leaves reachable via link A-B is a subset of that reachable by A-C, and the paths from A that serves that subset converges at B.

4) Prune via the elimination of upstream links where the nearest reachable leaf is further than the closest leaf or pinned path, and that path does not have a candidate replication point closer than the closet leaf or pinned path, as the resulting tree will require the shortest path to transit the closest upstream leaf or pinned path.

For each upstream link for each leaf in a segment the nearest leaf or pinned path is determined. Those links for which the nearest leaf is further upstream than the closest leaf are pruned.

If, at the end of pruning and simplification, all leaves in a multicast segment have a unique shortest path to the root, the tree is considered resolved, and the computation can progress directly to the FIB generation step.

If not all leaves have a unique shortest path, additional pruning steps are applied. These steps are NOT guaranteed to produce a lowest cost tree, and therefore require an additional audit and possible modification to ensure when forwarding a maximum of one copy of a packet will traverse an interface.

For segments not authoritatively resolved by the above rules, a prune that will not authoritatively result in a minimum cost tree is applied. For the purpose of interoperability, the following rule is proposed: A computing node will select the closest node to the root with a candidate role that does not have a unique shortest path to the root. Where more than one such node exists, the one with the lowest unicast SID is selected. For that node, the best upstream link is selected and all other upstream links pruned. The best upstream link is defined as the link with the closest node with a candidate role that potentially serves the highest number of leaves. Where there is a tie, once again the node with the lowest SID is selected.

Once the links have been pruned, rules 2 through 4 are repeatedly applied until either the tree is fully resolved, or again no further prunes are possible, in which case the next closest remaining unresolved node has the same prune applied.

For all segments not resolved by the initial prune rules, they are audited to ensure all nodes that have a role in the tree do not have a node with a role between them and their upstream node on the tree. If they do, the old upstream adjacency is removed, and the superior one added.

4.3. FIB Generation

The topology components that remain at the end of the pruning operation will reflect all nodes that have a role in a given multicast segment plus the necessary tunnels (as all intervening multi-path scenarios will have been simplified away). From this the FIB can be generated:

All nodes that have a role in a given multicast segment and have nodes upstream in the segment will need to accept the SID for the MDT from at minimum, all upstream interfaces.

All nodes that have a role in a given segment and have nodes immediately downstream in the segment will need to replicate packets simply labelled with the multicast SID onto those interfaces.

All nodes that have a role in a given segment and have nodes reachable via a tunnel downstream set the FIB to push the tunnel unicast SID for the downstream node onto any replicated copies of a received packet, and identify the set of interfaces on the shortest path for the tunnel SID.

4.4. FIB installation

FIB installation needs to acknowledge two aspects of the hybrid tunnel and role model of multicast tree construction. The first is that because of the sparse state model simple tree adds, moves, and changes may require the installation of state where it did not previously exist, and such changes may impact existing services. The second is that it is possible to retain the knowledge to prioritize computation of those trees impacted the failure of a node with a role.

To address this, there are three stages of state installation for multicast convergence:

1) Immediate:

   a. Installation of state for multicast segments impacted by the failure of a node in the network, and installation of state for segments in nodes that have not previously had a role in the given segment.

   b. Installation of state for waypoints in multi-segment MDTs.

2) After T1: Update state for nodes that both had and have a role in a given multicast segment.

3) After T2: Removal of state for nodes that transition from having a role to not having a role for a given multicast segment.

T1 and T2 are network wide configurable values.

5. Related work

5.1. IGP Extensions

The required IGP changes are documented in [MCAST-ISIS] and [MCAST-OPSF].

5.2. BGP Extensions

This memo will require the specification of a new PMSI Tunnel Attribute (SPRING P2MP tunnel, tentatively 0x09) to order to integrate into the multicast framework documented in RFC 6514

6. Observations

This technique is not confined to segment routing, and with the provision of a global label space (to be employed as per a multicast SID), an MPLS-LDP network would also provide the requisite mesh of unicast tunnels and be capable of implementing this approach to multicast.

This memo focuses on an implementation based upon nodes that are IGP speakers and converge independently so is written in a form that assumes a node, computing node and IGP speaker are one in the same. It should be observed that the relative frugality of data plane state would suggest that separation of computation from nodes in the data plane combined with management or "software defined networking" based population of the multicast FIB entries may also be useful modes of network operation.

7. Acknowledgements

Thanks to Uma Chunduri for his detailed review and suggestions.

8. Security Considerations

For a future version of this document.

9. IANA Considerations

This document requires the allocation of a PMSI tunnel type to identify a SPRING P2MP tunnel type from the P-Multicast Service Interface Tunnel (PMSI Tunnel) Tunnel Types registry.

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

10.2. Informative References

[MCAST-ISIS] Allan et.al., "IS-IS extensions for Computed Multicast applied to MPLS based Segment Routing", IETF work in progress, draft-allan-isis-spring-multicast-00, July 2016

[MCAST-OSPF] Allan et.al., "OSPF extensions for Computed Multicast applied to MPLS based Segment Routing", IETF work in progress, draft-allan-ospf-spring-multicast-00, July 2016

[RFC6514] Aggarwal et.al., "BGP Encodings and Procedures for Multicast in MPLS/BGP IP VPNs", IETF RFC 6514, February 2012

[RFC7385] Andersson & Swallow "IANA Registry for P-Multicast Service Interface (PMSI) Tunnel Type Code Points", IETF RFC 7385, October 2014

11. Authors' Addresses

Dave Allan (editor)
Ericsson
300 Holger Way
San Jose, CA 95134
USA
Email: david.i.allan@ericsson.com

Jeff Tantsura
Email: jefftant.ietf@gmail.com

SPRING Working Group   Madhukar Anand
Internet-Draft   Sanjoy Bardhan
Intended Status: Informational   Ramesh Subrahmaniam
                                                     Infinera Corporation

                                                           Jeff Tantsura
                                                              Individual
Expires: January 7, 2017   July 6, 2016

Packet-Optical Integration in Segment Routing
draft-anand-spring-poi-sr-01

Abstract

This document illustrates a way to integrate a new class of nodes and links in segment routing to represent transport networks in an opaque way into the segment routing domain. An instance of this class would be optical networks that are typically transport centric. In the IP centric network, this will help in defining a common control protocol for packet optical integration that will include optical paths as 'transport segments' or sub-paths as an augmentation to the defined extensions of segment routing. The transport segment option also defines a general mechanism to allow for future extensibility of segment routing into non-packet domains.

Requirements Language

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

The list of current Internet-Drafts can be accessed at
http://www.ietf.org/1id-abstracts.html

The list of Internet-Draft Shadow Directories can be accessed at
http://www.ietf.org/shadow.html

Copyright and License Notice

Table of Contents

1 Introduction 3
2. Reference Taxonomy 3
3. Use case - Packet Optical Integration 3
4. Mechanism overview 5
5. PCEP-LS extensions for supporting the transport segment . . . 6
6. OSPF extensions for supporting the transport segment 7
7. OSPFv3 extensions for supporting the transport segment 9
8. IS-IS extensions for supporting the transport segment 10
9. BGP-LS extensions for supporting the transport segment 12
10. Summary 14
11. Security Considerations 14
12 IANA Considerations 15
12.1 PCEP 15
12.2 OSPF 15
12.3 OSPFv3 15
12.4 IS-IS 15
12.5 BGP-LS 16
13 References 16
13.1 Normative References 16
13.2 Informative References 17
Authors' Addresses 17

1 Introduction

Packet and optical transport networks have evolved independently with different control plane mechanisms that have to be provisioned and maintained separately. Consequently, coordinating packet and optical networks for delivering services such as end-to-end traffic engineering or failure response has proved challenging. To address this challenge, a unified control and management paradigm that provides an incremental path to complete packet-optical integration while leveraging existing signaling and routing protocols in either domains is needed. This document introduces such a paradigm based on Segment Routing (SR) [I-D.ietf-spring-segment-routing].

This document introduces a new type of segment, Transport segment. Transport segment can be used to model abstracted paths through the optical transport domain and integrate it with the packet network for delivering end-to-end services. In addition, this also introduces a notion of a Packet optical gateway (POG). These are nodes in the network that map packet services to the optical domain that originate and terminate these transport segments. Given a transport segment, a POG will expand it to a path in the optical transport network.

2. Reference Taxonomy

POG - Packet optical gateway Device

SR Edge Router - The Edge Router which is the ingress device

CE - Customer Edge Device that is outside of the SR domain

PCE - Path Computation Engine

Controller - A network controller

3. Use case - Packet Optical Integration

Many operators build and operate their networks that are both multi-layer and multi-domain. Services are built around these layers and domains to provide end-to-end services. Due to the nature of the different domains, such as packet and optical, the management and service creation has always been problematic and time consuming. With segment routing, enabling a head-end node to select a path and embed the information in the packet is a powerful construct that would be used in the Packet Optical Gateways (POG). The path is usually constructed for each domain that may be manually derived or through a stateful PCE which is run specifically in that domain.

```
P1---------O1---------P2---------O2---------P3---------O3---------P4
```

Figure 1: Representation of a packet-optical path

In Figure 1 above, the nodes represent a packet optical network. P1, P2, P3 and P4 are packet optical devices that are connected via optical paths O1, O2 and O3. Nodes P1 and P4 are edge devices that have customer facing devices (denoted as Border POGs) and P2 and P3 are core nodes (denoted as Transit POGs) in the network. A packet service is established by specifying a path between P1 and P4. Note that in defining this path, we will need to specify both the nodes and the links that make up this service. POGs advertise themselves along with their adjacencies and the domains they belong to. To leverage segment routing to define the above service, the ingress node P1 would append all outgoing packets in a SR header consisting of the SIDs that constitute the path. In the packet domain this would mean P1 would send its packets towards P4 using the segment list {P2, P4}. The operator would need to use a different mechanism in the optical domain to set up the optical paths denoted by O1, O2 and O3. Each POG would announce the active optical path as a transport segment - for example, in the case of P1, the optical path O1 would represent an optical path that includes the optical nodes Om and On as shown on Figure 2. This path is not known to the packet SR domain and is only relevant to the optical domain D between P1 and P2. A PCE that is run in Domain D would be responsible for calculating path O1.

```
          |-----Om--------On-----|

    P1----|         (D)          |------P2

          |-----Ox---------Oy----|
```

Figure 2: POG with multiple optical paths through an optical domain

Similarly, the transit POGs P2 and P3 in Figure 1 would announce transport segments O2 and O3. The border POG would include the optical paths O1, O2 and O3 to the segment list for P1 to P4. The expanded segment list would read as {O1, P2, O2, P3, O3, P4}.

There are potentially two locations for Borders POGs - one that has last-mile access nodes and the other being Data Center Interconnect nodes. The POGs that are in the core of the network which connect with long haul optical networks are usually Transit POGs.

```
                       +------------------------+
                       |                        |
   +--------------+----'    PCE or Controller   |----+---------------+
   |              |    |                        |    |               |
   |              |    +------------------------+    |               |
   |              |                                  |               |
   |              |            .-----.               |               |
   |              |           (       )              |               |
+-------+     +-------+   .--(         )--.     +-------+      +-------+
|  SR   |     |Packet |  (                 )    |Packet |      |  SR   |
| Edge  |     |Optical|-( Optical Transport )_  |Optical|      | Edge  |
|Router | ... |Gateway| (       Domain      )   |Gateway| ...  |Router |
+---+.--+     +-------+  (                 )    +-------+      +---+.--+
    |                      '--(         )--'                       |
  ,--+.                        (       )                         ,-+-.
 ( CE  )                        '-----'                         ( CE  )
  '---'                                                          '---'
```

Figure 3. Reference Topology for Transport Segment

4. Mechanism overview

The current proposal assumes that the SR domains run standard IGP protocols to discover the topology and distribute labels without any modification. There are also no modifications to the control plane mechanisms in the Optical transport domains. The mechanism for supporting the transport segment is as follows.

1. Firstly, the Packet Optical Gateway (POG) devices announce themselves in the SR domain. This is indicated by advertising a new SR node capability flag. The exact extensions to support this capability are described in the subsequent sections of this document.

2. Then, the POG devices announce paths to other POGs through the optical transport domain as a transport segment (transport segment binding SID) in the SR domain. The paths are announced with an appropriate optical transport domain ID, and a label (Packet-Optical Label) to be used to bind to the transport segment. The appropriate

IGP segment routing extensions to carry this information is described in the subsequent sections of this document.

3. The transport segment can also optionally be announced with a set of attributes that characterizes the path in the optical transport domain between the two POG devices. For instance, those attributes could define the OTN mapping used (e.g., ODU4, ODU3,ODU3e1....ODU1), timeslots (1-8 or 4,6,7 or 1-2,5), or optical path protection schemes.

4. The POG device is also responsible for programming its forwarding table to map every transport segment label entry into an appropriate forwarding action relevant in the optical domain, such as mapping it to a label-switched path.

5. The transport segment is communicated to the PCE or Controller using extensions to BGP-LS or PCEP-LS as described in subsequent sections of this document.

6. Finally, the PCE or Controller then uses the transport segment label to influence the path leaving the SR domain into the optical domain, thereby defining the end-to-end path for a given service.

5. PCEP-LS extensions for supporting the transport segment

To communicate the Packet-Optical Gateway capability of the device, we introduce a new PCEP capabilities TLV is defined as follows(extensions to [I-D.draft-sivabalan-pce-segment-routing]):

Value	Meaning	Reference
27	TRANSPORT-SR-PCE-CAPABILITY	This document

A new type of TLV to accommodate a transport segment is defined by extending Binding SIDs [I-D.draft-sivabalan-pce-binding-label-sid-01]

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             Type              |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Binding Type (BT)      |             Domain ID         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Binding Value                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~       Transport Segment Sub TLVs (variable length)            ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

Type: TBD, suggested value 32

Length: variable.

Binding Type: 0 or 1 as defined in [I-D.draft-sivabalan-pce-binding-label-sid-01]

Domain ID: An identifier for the transport domain

Binding Value: is the transport segment label

Transport Segment Sub TLVs: TBD

IANA will be requested to allocate a new TLV type (recommended value is 32) for TRANSPORT-SEGMENT-BINDING-TLV as specified in this document:

1   Transport Segment Label (This document)

6. OSPF extensions for supporting the transport segment

To communicate the Packet-Optical Gateway capability of the device, we introduce an new optical informational capability bit in the Router Information capabilities TLV (as defined in [RFC4970]).

Bit-24 - Optical - If set, then the router is capable of performing Packet Optical Gateway function.

Further, a new OSPF sub-TLV (similar to the ERO SubTLV) of SID/Label Binding Sub-TLV (TRANSPORT-SEGMENT-BINDING-SUBTLV) to carry the transport segment label is defined as follows.

```
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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Type               |            Length           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Domain ID                |   Flags       |  Reserved   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Packet-Optical Label                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~       Transport Segment Sub TLVs (variable length)           ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

Type : TBD, Suggested Value 9

Length: variable.

Domain ID: An identifier for the transport domain

Flags: 1 octet field of following flags:
V - Value flag. If set, then the optical label carries a value. By default the flag is SET.
L - Local. Local Flag. If set, then the value/index carried by the Adj-SID has local significance. By default the flag is SET.

```
0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|V|L|
+-+-+-+-+-+-+-+-+
```

Packet-Optical Label : according to the V and L flags, it contains either:

* A 3 octet local label where the 20 rightmost bits are used for encoding the label value. In this case the V and L flags MUST be set.

* A 4 octet index defining the offset in the label space advertised by this router. In this case V and L flags MUST be unset.

Transport Segment Sub TLVs: TBD

Multiple TRANSPORT-SEGMENT-BINDING-SUBTLV MAY be associated with a pair of POG devices to represent multiple paths within the optical domain

7. OSPFv3 extensions for supporting the transport segment

To communicate the Packet-Optical Gateway capability of the device, we introduce an new optical informational capability bit in the Router Information capabilities TLV (as defined in [RFC4970]).

Bit-24 - Optical - If set, then the router is capable of performing Packet Optical Gateway function.

Further, a new OSPFv3 sub-TLV similar to the ERO SubTLV) of SID/Label Binding Sub-TLV (TRANSPORT-SEGMENT-BINDING-SUBTLV) to carry the transport segment label is defined as follows.

```
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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Type             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Domain ID              |   Flags       |  Reserved     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Packet-Optical Label                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~       Transport Segment Sub TLVs (variable length)          ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

Type : TBD,Suggested Value 12

Length: variable.

Domain ID: An identifier for the transport domain

Flags: 1 octet field of following flags:
  V - Value flag. If set, then the optical label carries a value.
      By default the flag is SET.
  L - Local. Local Flag. If set, then the value/index carried by
      the Adj-SID has local significance. By default the flag is SET.

```
0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|V|L|
+-+-+-+-+-+-+-+-+
```

Packet-Optical Label : according to the V and L flags, it contains either:

* A 3 octet local label where the 20 rightmost bits are used for encoding the label value. In this case the V and L flags MUST be set.

* A 4 octet index defining the offset in the label space advertised by this router. In this case V and L flags MUST be unset.

Transport Segment Sub TLVs: TBD

Multiple TRANSPORT-SEGMENT-BINDING-SUBTLV MAY be associated with a pair of POG devices to represent multiple paths within the optical domain

8. IS-IS extensions for supporting the transport segment

To communicate the Packet-Optical Gateway capability of the device, we introduce a new flag O in the SR Node Capabilities sub-TLV:

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|I|V|H|O|       |
+-+-+-+-+-+-+-+-+
```

I, V, H flags are defined in [I-D.ietf-isis-segment-routing-extensions]

O-Flag: If set, then the router is capable of performing Packet Optical Gateway function.

Further, a new IS-IS sub-TLV (similar to the ERO SubTLV) of SID/Label Binding Sub-TLV (TRANSPORT-SEGMENT-BINDING-SUBTLV) to carry the transport segment label is defined as follows.

First, we define the O flag in the SID/Label Binding TLV

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|F|M|S|D|A|O|   |
+-+-+-+-+-+-+-+-+
```

F, M, S, D, and A flags: are defined in [I-D.ietf-isis-segment-routing -extensions]

O-Flag: If set, then the F flag, Range, Prefix Length FEC Prefix, must be ignored in the SID/Label Binding TLV

Secondly, we define the SubTLV of the SID/Label Binding Sub-TLV:

```
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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Type             |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Domain ID              |   Flags       |  Reserved     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Packet-Optical Label                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~       Transport Segment Sub TLVs (variable length)          ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

Type : TBD, Suggested Value 151

Length: variable.

Domain ID: An identifier for the transport domain

Flags: 1 octet field of following flags:
V - Value flag. If set, then the optical label carries a value.
    By default the flag is SET.
L - Local. Local Flag. If set, then the value/index carried by
    the Adj-SID has local significance. By default the flag is SET.

```
0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|V|L|
+-+-+-+-+-+-+-+-+
```

Packet-Optical Label : according to the V and L flags, it contains either:

* A 3 octet local label where the 20 rightmost bits are used for encoding the label value. In this case the V and L flags MUST be set.

* A 4 octet index defining the offset in the label space advertised by this router. In this case V and L flags MUST be unset.

Transport Segment Sub TLVs: TBD

Multiple TRANSPORT-SEGMENT-BINDING-SUBTLV MAY be associated with a pair of POG devices to represent multiple paths within the optical domain with perhaps different characteristics.

9. BGP-LS extensions for supporting the transport segment

9.1 Node Attribuites TLV

To communicate the Packet-Optical Gateway capability of the device, we introduce an new optical informational capability the following new Node Attribute TLV is defined:

TLV Code Point	Description	Length	Section
1172	SR-Optical-Node-Capability TLV	variable	

Table 1: Node Attribute TLVs

These TLVs can ONLY be added to the Node Attribute associated with the node NLRI that originates the corresponding SR TLV.

9.2 SR-Optical-Node-Capability TLV

The SR Capabilities sub-TLV has following format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Type            |               Length          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Flags    |   RESERVED    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

Type : TBD, Suggested Value 1157

Length: variable.

Flags: The Flags field currently has only one bit defined. If the bit is set it has the capability of an Packet Optical Gateway.

9.3 Prefix Attribute TLVs

The following Prefix Attribute Binding SID Sub-TLVs have been added:

TLV Code Point	Description	Length	Section
1173	TRANSPORT-SEGMENT-SID	12	

Table 4: Prefix Attribute - Binding SID Sub-TLVs

The Transport segment TLV allows a node to advertise an transport segment within a single IGP domain. The transport segment SID TLV TRANSPORT-SEGMENT-TLV has the following format:

9.3.1 Transport Segment SID Sub-TLV

Further, a new sub-TLV (similar to the IPV4 ERO SubTLV) of Binding SID Sub-TLV (TRANSPORT-SEGMENT-BINDING-SUBTLV) to carry the transport segment label is defined as follows.

```
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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Type             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Domain ID              |    Flags      |  Reserved     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Packet-Optical Label                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~       Transport Segment Sub TLVs (variable length)          ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

Type : TBD

Length: variable.

Domain ID: An identifier for the transport domain

Flags: 1 octet field of following flags:
  V - Value flag. If set, then the optical label carries a value.
      By default the flag is SET.
  L - Local. Local Flag. If set, then the value/index carried by
      the Adj-SID has local significance. By default the flag is SET.

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|V|L|
+-+-+-+-+-+-+-+-+
```

Packet-Optical Label : according to the V and L flags, it contains either:

* A 3 octet local label where the 20 rightmost bits are used for encoding the label value. In this case the V and L flags MUST be set.

* A 4 octet index defining the offset in the label space advertised by this router. In this case V and L flags MUST be unset.

Transport Segment Sub TLVs: TBD

Multiple TRANSPORT-SEGMENT-TLV MAY be associated with a pair of POG devices to represent multiple paths within the optical domain

10. Summary

The motivation for introducing a new type of segment - transport segment - is to integrate transport networks with the segment routing domain and expose characteristics of the transport domain into the packet domain. An end-to-end path across packet and transport domains can then be specified by attaching appropriate SIDs to the packet. An instance of transport segments has been defined here for optical networks, where paths between packet-optical gateway devices has been abstracted using binding SIDs. Extensions to various protocols to announce the transport segment have been proposed in this document.

11. Security Considerations

This document does not introduce any new security considerations.

12 IANA Considerations

This documents request allocation for the following TLVs and subTLVs.

12.1 PCEP
Packet-Optical Gateway capability of the device

Value	Meaning	Reference
27	TRANSPORT-SR-PCE-CAPABILITY	This document

A new type of TLV to accommodate a transport segment is defined by extending Binding SIDs [I-D.draft-sivabalan-pce-binding-label-sid-01]

Value	Description	Reference
32	TRANSPORT-SR-PCEP-TLV	This document

This document requests that a registry is created to manage the value of the Binding Type field in the TRANSPORT-SR-PCEP TLV.

Value	Description	Reference
1	Transport Segment Label	This document

12.2 OSPF
Transport-Segment SubTLV of OSPF Extended Prefix LSA

Value	Description	Reference
9	TRANSPORT-SR-OSPF-SUBTLV	This document

12.3 OSPFv3
Transport-Segment SubTLV of OSPFv3 Extend-LSA Sub-TLV registry

Value	Description	Reference
12	TRANSPORT-SR-OSPFv3-SUBTLV	This document

12.4 IS-IS
Transport-Segment SubTLV of Segment Identifier / Label Binding TLV

Value	Description	Reference
151	TRANSPORT-SR-ISIS-SUBTLV	This document

12.5 BGP-LS
Node Attributes TLV:

Value	Description	Reference
1172	TRANSPORT-SR-BGPLS-CAPABILITY	This document

Prefix Attribute Binding SID SubTLV:

Value	Description	Reference
1173	TRANSPORT-SR-BGPLS-TLV	This document

13 References

13.1 Normative References

[I-D.ietf-spring-segment-routing]
Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and r. rjs@rob.sh, "Segment Routing Architecture", draft-ietf-spring-segment-routing-04 (work in progress), July 2015.

[I-D.ietf-isis-segment-routing-extensions]
Previdi, S., Filsfils, C., Bashandy, A., Gredler, H., Litkowski, S., Decraene, B., and J. Tantsura, "IS-IS Extensions for Segment Routing", draft-ietf-isis-segment-routing-extensions-05 (work in progress), June 2015.

[I-D.ietf-ospf-segment-routing-extensions]
Psenak, P., Previdi, S., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPF Extensions for Segment Routing", draft-ietf-ospf-segment-routing-extensions-05 (work in progress), June 2015.

[RFC4915] L. Nguyen, P. Psenak, S. Mirtorabi, P. Pillay-Esnault, and A. Roy, "Multi-Topology (MT) Routing in OSPF.", RFC4915, <http://tools.ietf.org/html/rfc4915>.

[I-D.ietf-ospf-ospfv3-segment-routing-extensions]

Psenak, P., Previdi, S., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPFv3 Extensions for Segment Routing", draft-ietf-ospf-ospfv3-segment-routing-extensions-03 (work in progress), June 2015.

[I-D.ietf-idr-ls-distribution]
Gredler, H., Medved, J., Previdi, S., Farrel, A., and S. Ray, "North-Bound Distribution of Link-State and TE Information using BGP", draft-ietf-idr-ls-distribution-13 (work in progress), October 2015.

[RFC4970] Lindem, A., Ed., Shen, N., Vasseur, JP., Aggarwal, R., and S. Shaffer, "Extensions to OSPF for Advertising Optional Router Capabilities", RFC 4970, DOI 10.17487/RFC4970, July 2007, <http://www.rfc-editor.org/info/rfc4970>.

[I-D.sivabalan-pce-binding-label-sid]
Sivabalan, S., Filsfils, C., Previdi, S., Tantsura, J., Hardwick, J., and M. Nanduri, "Carrying Binding Label/ Segment-ID in PCE-based Networks.", draft-sivabalan-pce-binding-label-sid-01 (work in progress), March 2016.

[I-D.ietf-pce-segment-routing]
Sivabalan, S., Medved, J., Filsfils, C., Crabbe, E., Lopez, V., Tantsura, J., Henderickx, W., and J. Hardwick, "PCEP Extensions for Segment Routing", draft-ietf-pce-segment-routing-07 (work in progress), March 2016.

13.2 Informative References

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

Authors' Addresses

Madhukar Anand
Infinera Corporation
169 W Java Dr, Sunnyvale, CA 94089

Email: manand@infinera.com

Sanjoy Bardhan
Infinera Corporation
169 W Java Dr, Sunnyvale, CA 94089

Email: sbardhan@infinera.com

Ramesh Subrahmaniam
Infinera Corporation
169 W Java Dr, Sunnyvale, CA 94089

Email: RSubrahmaniam@@infinera.com

Jeff Tantsura

Email: jefftant.ietf@gmail.com

SPRING Working Group   Madhukar Anand
Internet-Draft   Ciena Corporation
Intended Status: Standard Track

Sanjoy Bardhan
Infinera Corporation

Ramesh Subrahmaniam
Individual

Jeff Tantsura
Apstra

Utpal Mukhopadhyaya
Equinix Inc

Clarence Filsfils
Cisco Systems, Inc.

Expires: January 30, 2020   July 29, 2019

Packet-Optical Integration in Segment Routing
draft-anand-spring-poi-sr-08

Abstract

This document illustrates a way to integrate a new class of nodes and links in segment routing to represent transport/optical networks in an opaque way into the segment routing domain. An instance of this class would be optical networks that are typically transport centric by having very few devices with the capability to process packets. In the IP centric network, this will help in defining a common control protocol for packet optical integration that will include optical paths as 'transport segments' or sub-paths as an augmentation to packet paths. The transport segment option also defines a general mechanism to allow for future extensibility of segment routing into non-packet domains.

Requirements Language

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

Status of this Memo

Copyright and License Notice

Table of Contents

1 Introduction 4
2. Reference Taxonomy 4
3. Use case - Packet Optical Integration 5
4. Mechanism overview 8
5. Transport Segments as SR Policy 9
6. PCEP extensions for supporting the transport segment 10
7. BGP-LS extensions for supporting the transport segment 11
  7.1 Node Attributes TLV 11
  7.2 SR-Optical-Node-Capability Sub-TLV 11
  7.3 Prefix Attribute TLVs 12
    7.3.1 Transport Segment SID Sub-TLV 12

8. Note about Transport Segments and Scalability 13
9. Summary 14
10. Security Considerations 14
11 IANA Considerations 14
11.1 PCEP 14
11.2 BGP-LS 15
12 Acknowledgements 15
13 References 15
13.1 Normative References 15
13.2 Informative References 16
Authors' Addresses 16

1 Introduction

Packet and optical transport networks have evolved independently with different control plane mechanisms that have to be provisioned and maintained separately. Consequently, coordinating packet and optical networks for delivering services such as end-to-end traffic engineering or failure response has proved challenging. To address this challenge, a unified control and management paradigm that provides an incremental path to complete packet-optical integration while leveraging existing signaling and routing protocols in either domains is needed. This document introduces such a paradigm based on Segment Routing (SR) [RFC8402].

This document introduces a new type of segment, Transport segment, as a special case of SR traffic engineering (SR-TE) policy (Type 1, Sec 5. [I-D.draft-ietf-spring-segment-routing-policy]). Specifically, the structure of SR-TE policy and constraints associated in the transport/optical network are different from those outlined for the packet networks. Transport segment can be used to model abstracted paths through the transport/optical domain and integrate it with the packet network for delivering end-to-end services. In addition, this also introduces a notion of a Packet optical gateway (POG). These are nodes in the network that map packet services to the optical domain that originate and terminate these transport segments. Given a transport segment, a POG will expand it to a path in the optical transport network. A POG can be viewed as SR traffic engineering policy headend.

The concept of POG introduced here allows for multiple instantiations of the concept. In one case, the packet device is distinct from the transport/optical device, and the POG is a logical entity that spans these two devices. In this case, the POG functionality is achieved with the help of external coordination between the packet and optical devices. In another case, the packet and optical components are integrated into one physical device, and the co-ordination required for functioning of the POG is performed by this integrated device. It must be noted that in either case, it is the packet/optical data plane that is either disaggregated or integrated. Control of the devices can be logically centralized or distributed in either scenario. The focus of this document is to define the logical functions of a POG without going into the exact instantiations of the concept.

2. Reference Taxonomy

POG - Packet optical gateway Device

SR Edge Router - The Edge Router which is the ingress device

CE - Customer Edge Device that is outside of the SR domain

PCE - Path Computation Engine

Controller - A network controller

3. Use case - Packet Optical Integration

Many operators build and operate their networks that are both multi-layer and multi-domain. Services are built around these layers and domains to provide end-to-end services. Due to the nature of the different domains, such as packet and optical, the management and service creation has always been problematic and time consuming. With segment routing, enabling a head-end node to select a path and embed the information in the packet is a powerful construct that would be used in the Packet Optical Gateways (POG). The path is usually constructed for each domain that may be manually derived or through a stateful PCE which is run specifically in that domain.

```
                                P5
     P1 _                      .-'-._                    ,'P4
         `._                .-'      `-.               ,'
            `.          _._'            `-._         ,'
              `-.    .-'                    `-.    ,'
               P2`.-'--------------------------`-.- P3
                 |\                            /|
                 | \                          / |   Packet
            ''''''''''''''''''''''''''''''''''''''''''''''''''
                 |   \                      /   |
                 |    \                    /    |   Optical
                 |     \                  /     |
                 |       ................/      |
                 |    ,'O2              O3`.    |
                 |  ,'                      `.  |
                 |,'                          `.|
                O1\                             | O4
                   \                           ,'
                    \                        ,'
                     ......................._
                    O6                     O5
```

Figure 1: Representation of a packet-optical path

In Figure 1 above, the nodes represent a packet optical network. P1,...,P5 are packet devices. Nodes P2 and P3 are connected via optical network (e.g., DWDM) comprising of nodes O1,...,O6. Nodes P2 and P3 are POGs that communicate with other packet devices and also with the devices in the transport/optical domain. POGs P2 and P3 are connected to optical nodes O2/O3 and O3/O4 respectively via multiple links that are visible to the packet network. In defining a path between nodes P2 and P3, we will need to specify the nodes and the links in both the packet and transport/optical domains.

To leverage segment routing to define a service between P1 and P4, the ingress node P1 would append all outgoing packets in a SR header consisting of the SIDs that constitute the path. In the packet domain this would mean P1 would send its packets towards P4 using a segment list {P2, P3, P4} or {P2, P5, P3, P4} as the case may be. The operator would need to use a different mechanism in the optical domain to set up the paths between the two POGs P2 and P3. For instance, if the packet is forwarded on the link from P2 towards O1 with the expectation that it would come out on the link O4-P3, it could be routed in the optical network using either path {O1, O2, O3, O4} or {O1, O6, O5, O4}. Currently, this decision is made in the optical domain, and there are no mechanisms in the packet network to influence that. The transport segment mechanism proposed in this draft has been designed with an explicit goal of providing better control of optical path selection to the packet network and applications running on them.

Under the proposed scheme, each POG would announce active optical paths to the other POG as a transport segment - for example, the optical path from P2 to P3 comprising {O1, O2, O3, O4} could be represented as a transport segment label Om and the optical path from P2 to P3 comprising devices {O1, O5, O6, O4} could be represented as a transport segment label On. Both Om and On will be advertised by POG P2 as two optical paths between P2 and P3 with specific properties. The specifics of the optical paths, including specific intermediate devices, need not be exposed to the packet SR domain and are only relevant to the optical domain between P2 and P3. A PCE that is run in the optical domain would be responsible for calculating paths corresponding to label Om and On. The expanded segment list would read as {P2, Om, P3, P4} or {P2, On, P3, P4}. Multiple optical paths between P2 and P3 corresponding to different properties can be exposed as transport segments in the packet domain. For example, some optical paths can be low operational cost paths, some could be low-latency, and some others can be high-bandwidth paths. Transport segments for all these candidate viable alternative paths may be generated statically or dynamically.They may be pre-computed or may be generated on the fly when a customer at node P1 requests a service towards node P4. A discussion on transport segments and scalability can be found in Section 8.

Use-case examples of transport segments.

1. Consider the scenario where there are multiple fibers between two packet end points. The network operator may choose to route packet traffic on the first fiber, and reserve the second fiber only for maintenance or low priority traffic.

2. As a second use-case, consider the case where the packet end points are connected by transport/optical network provided by two different service providers. The packet operator wants to preferentially route traffic over one of the providers and use the second provider as a backup.

3. Finally, let the packet end points be connected by optical paths that may span multiple optical domains i.e. different administrative control. For instance, one transport/optical path may lie completely in one country while the other transport/optical path transits another country. Weather, tariffs, security considerations and other factors may determine how the packet operator wants to route different types of traffic on this network.

All of the above use-cases can be supported by first mapping distinct transport/optical paths to different transport segments and then, depending on the need, affixing appropriate transport segment identifier to the specific packet to route it appropriately through the transport domain.

```
                       +------------------------+
                       |                        |
   +--------------+----'   PCE or Controller    |----+---------------+
   |              |    |                        |    |               |
   |              |    +------------------------+    |               |
   |              |                                  |               |
   |              |            .-----.               |               |
   |              |           (       )              |               |
+-------+     +-------+   .--(         )--.     +-------+      +-------+
|  SR   |     |Packet |  (                 )    |Packet |      |  SR   |
| Edge  |     |Optical|-( Transport/optical )_  |Optical|      | Edge  |
|Router | ... |Gateway| (      Domain        )  |Gateway| ...  |Router |
+---+.--+     +-------+  (                 )    +-------+      +---+.--+
    |                      '--(         )--'                       |
  ,--+.                        (       )                         ,-+-.
 ( CE  )                        '-----'                         ( CE  )
  `---'                                                          `---'
```

Figure 3. Reference Topology for Transport Segment Mechanism

4. Mechanism overview

The current proposal assumes that the SR domains run standard protocols without any modification to discover the topology and distribute labels. There are also no modifications necessary in the control plane mechanisms in the transport/optical domains. The only requirement of a transport segment is that the optical path be setup before they are announced to the packet network. For example, the optical paths may be setup using a domain-specific controller or a PCE based on requirements from the packet domain (such as bandwidth, QoS, latency and cost) taking into consideration the constraints in the optical network.

The mechanism for supporting the transport segment is as follows.

1. Firstly, the Packet Optical Gateway (POG) devices are announced in the packet domain. This is indicated by advertising a new SR node capability flag. The exact extensions to support this capability are described in the subsequent sections of this document.

2. Then, the POG devices announce candidate transport/optical paths between that POG (Source POG) and other POGs (Destination POG) via appropriate mechanisms in the packet domain. The paths are announced with an appropriate transport/optical domain ID and a Binding SID representing the transport segment from a source POG to a destination POG. The appropriate protocol-specific extensions to carry path characteristics and Binding SID corresponding to a optical path are described in the subsequent sections of this document.

3.The transport SR policy can also optionally be announced with a set of attributes that characterizes the path in the transport/optical domain between the two POG devices. For instance, those could define the path attributes such as path identifier, latency, bandwidth, quality, directionality, or optical path protection schemes. These attributes can be used to determine the "color" of the SR-TE policy in the tuple <Source POG, Destination POG, color> used to prioritize different candidate paths between the POGs.

4. The POG device is also responsible for programming its forwarding table to map every transport segment Binding SID entry

into an appropriate forwarding action relevant in the optical domain, such as mapping it to a optical label-switched path.

5. The transport SR policy is communicated to the PCE or Controller using extensions to BGP-LS or PCEP as described in subsequent sections of this document.

6. Finally, the PCE or Controller in the packet domain then uses the transport segment binding SID in the overall SR policy to influence the path traversed by the packet in the optical domain, thereby defining the end-to-end path for a given service.

In the next few sections, we outline a few representative protocol specific extensions to carry the transport segment.

5. Transport Segments as SR Policy

The Segment Routing Traffic Engineering (SRTE) [ietf-spring-segment-routing-policy] process installs the transport segment SR policy in the forwarding plane of the POG. The Transport SR policy is identified by using a transport segment Binding SID. Corresponding to each transport segment Binding SID, the SRTE process MAY learn about multiple candidate paths. The SRTE-DB includes information about the candidate paths including optical domain, topology and path characteristics. All of the information can be learned from different sources including but not limited to: Netconf/Restconf, PCEP and BGP-LS.

The information model for Transport SR policy is as follows:

```
Transport SR Policy FO1
       Candidate-paths
         path preference 200 (selected)
             BSID1
         path preference 100
             BSID2
         path preference 100
             BSID3
         path preference 50
             BSID4
```

A transport SR policy is identified through the tuple <Source POG, Destination POG, color>. Each TSR policy is associated with one or more candidate paths, each of them associated with a (locally) unique Binding SID and a path preference. For each transport SR policy, the candidate path with the highest path preference (at most one) is selected and used for forwarding traffic that is being steered onto that policy. When candidate paths change (or a new candidate path is set up), the path selection process can be re-executed. The validity of each path is to be verified by the POG before announcement in the packet domain. If there are no valid paths, then the transport SR policy is deemed invalid.

The allocation of BSID to a path can include dynamic, explicit or generic allocation strategies as discussed in [ietf-spring-segment-routing-policy]. We discuss PCEP and BGP-LS specific extensions in the subsequent section.

6. PCEP extensions for supporting the transport segment

To communicate the Packet-Optical Gateway capability of the device, we introduce a new PCEP capabilities TLV is defined as follows(extensions to [I-D.ietf-pce-segment-routing]):

Value	Meaning	Reference
TBD1	TRANSPORT-SR-PCE-CAPABILITY	This document

A new type of TLV to accommodate a transport segment is defined by extending Binding SIDs [I-D.sivabalan-pce-binding-label-sid]

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             Type              |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Binding Type (BT)      |             Domain ID         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Binding Value                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~       Transport Segment Sub TLVs (variable length)            ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

Type: TBD

Length: variable.

Binding Type: 0 or 1 as defined in [I-D.sivabalan-pce-binding-label-sid]

Domain ID: An identifier for the transport domain

Binding Value: is the transport segment label

Transport Segment Sub TLVs: TBD

IANA will be requested to allocate a new TLV type for TRANSPORT-SEGMENT-BINDING-TLV as specified in this document:

TBD   Transport Segment Label (This document)

7. BGP-LS extensions for supporting the transport segment

7.1 Node Attributes TLV

To communicate the Packet-Optical Gateway capability of the device, we introduce an new optical informational capability the following new Node Attribute TLV is defined:

TLV Code Point	Description	Length	Section
TBD	SR-Optical-Node-Capability TLV	variable	

Table 1: Node Attribute TLVs

These TLVs can ONLY be added to the Node Attribute associated with the node NLRI that originates the corresponding SR TLV.

7.2 SR-Optical-Node-Capability Sub-TLV

The SR Capabilities sub-TLV has following format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Type             |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Flags    |   RESERVED    |
```

```
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

Type : TBD, Suggested Value 1157

Length: variable.

Flags: The Flags field currently has only one bit defined. If the bit is set it has the capability of an Packet Optical Gateway.

7.3 Prefix Attribute TLVs

The following Prefix Attribute Binding SID Sub-TLVs have been added:

TLV Code Point	Description	Length	Section
TBD	TRANSPORT-SEGMENT-SID	12	

Table 4: Prefix Attribute - Binding SID Sub-TLVs

The Transport segment TLV allows a node to advertise an transport segment within a single IGP domain. The transport segment SID TLV TRANSPORT-SEGMENT-TLV has the following format:

7.3.1 Transport Segment SID Sub-TLV

Further, a new sub-TLV (similar to the IPV4 ERO SubTLV) of Binding SID Sub-TLV (TRANSPORT-SEGMENT-BINDING-SUBTLV) to carry the transport segment label is defined as follows.

```
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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Type               |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Domain ID                |   Flags       |   Reserved    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Packet-Optical Label                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~       Transport Segment Sub TLVs (variable length)            ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
where:

Type : TBD

Length: variable.

Domain ID: An identifier for the transport domain

Flags: 1 octet field of following flags:
  V - Value flag. If set, then the optical label carries a value.
      By default the flag is SET.
  L - Local. Local Flag. If set, then the value/index carried by
      the Adj-SID has local significance. By default the flag is SET.

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|V|L|
+-+-+-+-+-+-+-+-+
```

Packet-Optical Label : according to the V and L flags, it contains
       either:

* A 3 octet local label where the 20 rightmost bits are used for encoding the label value. In this case the V and L flags MUST be set.

* A 4 octet index defining the offset in the label space advertised by this router. In this case V and L flags MUST be unset.

Transport Segment Sub TLVs: TBD

Multiple TRANSPORT-SEGMENT-TLV MAY be associated with a pair of POG devices to represent multiple paths within the optical domain

8. Note about Transport Segments and Scalability

In most operational scenarios, there would be multiple, distinct paths between the POGs. There is no requirement that every distinct path in the optical domain be advertised as a separate transport segment. Transport segments are designed to be consumed in the packet domain, and the correspondence between transport segments and exact paths in the optical domain are determined by their utility to the packet world. Therefore, the number of transport segments is to be determined by the individual packet-optical use-case. The number of actual paths in the optical domain between the POG is expected to be large (counting the number of active and passive devices in the optical network), it is likely that multiple actual paths are to be advertised as one transport segment. Of course, in the degenerate case, it is possible that there is a one-to-one correspondence between an optical path and a transport segment. Given this view of network operation, the POG is not expected to handle a large number of transport segments (and identifiers). This framework does leave open the possibility of handling a large number of transport segments in future. For instance, a hierarchical partitioning of the optical domain along with stacking of multiple transport segment identifiers could be explored towards reducing the overall number of transport segment identifiers.

9. Summary

The motivation for introducing a new type of segment - transport segment - is to integrate transport/optical networks with the segment routing domain and expose characteristics of the transport/optical domain into the packet domain. An end-to-end path across packet and transport/optical domains can then be specified by attaching appropriate SIDs to the packet. An instance of transport segments has been defined here for optical networks, where paths between packet-optical gateway devices have been abstracted using binding SIDs. Extensions to various protocols to announce the transport segment have been proposed in this document.

10. Security Considerations

This document does not introduce any new security considerations.

11 IANA Considerations

This documents request allocation for the following TLVs and subTLVs.

11.1 PCEP

Packet-Optical Gateway capability of the device

Value	Meaning	Reference
TBD1	TRANSPORT-SR-PCE-CAPABILITY	This document

A new type of TLV to accommodate a transport segment is defined by extending Binding SIDs [I-D.sivabalan-pce-binding-label-sid]

Value	Description	Reference

TBD2	TRANSPORT-SR-PCEP-TLV	This document

This document requests that a registry is created to manage the value of the Binding Type field in the TRANSPORT-SR-PCEP TLV.

Value	Description	Reference
TBD3	Transport Segment Label	This document

11.2 BGP-LS
Node Attributes TLV:

Value	Description	Reference
TBD4	TRANSPORT-SR-BGPLS-CAPABILITY	This document

Prefix Attribute Binding SID SubTLV:

Value	Description	Reference
TBD5	TRANSPORT-SR-BGPLS-TLV	This document

12 Acknowledgements
We would like to thank Peter Psenak, Bruno Decraene, Ketan Talaulikar and Radhakrishna Valiveti for their comments and review of this document.

13 References

13.1 Normative References

[RFC8402]
Filsfils, C., Previdi, S., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, July 2018.

[I-D.sivabalan-pce-binding-label-sid]
Sivabalan, S., Tantsura, J., Filsfils, C., Previdi, S., Hardwick, J., and Dhody, D., "Carrying Binding Label/ Segment-ID in PCE-based Networks.", draft-sivabalan-pce-binding-label-sid-07 (work in progress), July 2019.

[I-D.ietf-pce-segment-routing]
Sivabalan, S., Filsfils, C., Tantsura, J., Henderickx, W.,

and J. Hardwick, "PCEP Extensions for Segment Routing", draft-ietf-pce-segment-routing-16 (work in progress), Mar 2019.

[I-D.draft-ietf-spring-segment-routing-policy]
Filsfils, C., Sivabalan, S., Voyer, D., Bogdanov, A., and Mattes, P., "Segment Routing Policy Architecture", draft-ietf-spring-segment-routing-policy-03.txt (work in progress), May 2019.

13.2 Informative References

[RFC5513] Farrel, A., "IANA Considerations for Three Letter Acronyms", RFC 5513, DOI 10.17487/RFC5513, April 1 2009, <http://www.rfc-editor.org/info/rfc5513>.

[RFC5514] Vyncke, E., "IPv6 over Social Networks", RFC 5514, DOI 10.17487/RFC5514, April 1 2009, <http://www.rfc-editor.org/info/rfc5514>.

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

Authors' Addresses

Madhukar Anand
Ciena Corporation
3939, N 1st Street, San Jose, CA, 95134
Email: madanand@ciena.com

Sanjoy Bardhan
Infinera Corporation
169 W Java Dr, Sunnyvale, CA 94089
Email: sbardhan@infinera.com

Ramesh Subrahmaniam

Email: svr_fremont@yahoo.com

Jeff Tantsura
Apstra
333 Middlefield Road Suite 200
Menlo Park, CA 94025
Email: jefftant.ietf@gmail.com

Utpal Mukhopadhyaya
Equinix Inc
1188 E. Arques, Sunnyvale, CA 94085
Email: umukhopadhyaya@equinix.com

Clarence Filsfils
Cisco Systems, Inc.
Brussels
BE
Email: cfilsfil@cisco.com

SPRING Working Group
Internet-Draft
Intended Status: Informational
Sanjoy Bardhan
Madhukar Anand
Ramesh Subrahmaniam
Infinera Corporation
Jeff Tantsura
Individual
Expires: January 7, 2017
July 7, 2016

OAM for Packet-Optical Integration in Segment Routing
draft-bardhan-spring-poi-sr-oam-00

Abstract

This document describes a list of functional requirements for transport segment OAM in Segment Routing (SR) based networks.

Status of this Memo

Copyright and License Notice

Table of Contents

1 Introduction 3
  1.1 Terminology 3
2. Detailed Requirement List 3
3 Security Considerations 6
4 IANA Considerations 6
5 References 6
  5.1 Normative References 6
  5.2 Informative References 6
Authors’ Addresses 6
Acknowledgments 7

1 Introduction

[I-D.filsfils-rtgwg-segment-routing] introduces and explains Segment Routing architecture that leverages source routing and tunneling standards which can be applied directly to MPLS dataplane with no changes on forwarding plane and on IPv6 dataplane with new Routing Extension Header. In addition [I-D. draft-anand-spring-poi-sr] introduces the concept of a Transport Segment at the edge of the packet and optical network that represents the optical path taken for a given flow. This document is a place holder to identify and list the OAM requirements for Segment Routing based network which can further be extended to produce OAM tools for path liveliness and service validation across the optical domain using Transport Segments.

1.1 Terminology

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

SR: Segment Routing

Initiator: Centralized OAM initiator

POG: Packet Optical Gateway that interworks between a packet and optical network

2. Detailed Requirement List

This section list the OAM requirement for Transport Segments in a Segment Routing based network. The below listed requirements MUST be supported within an optical dataplane.

REQ#1: Transport Segment OAM SHOULD support Continuity Check (liveliness of a path - BFD), Connectivity Verification (BFD, Ping), Fault Verification - exercised on demand to validate the reported fault (Ping).

REQ#2:   Transport Segment OAM MUST support both On-demand and Continuous OAM functionality.

REQ#3:   Transport Segment OAM packet MUST follow exactly the same path as the dataplane traffic.

REQ#4:   The Transport Segment OAM packet MUST have the ability to exercise any available paths as defined by the transport segment label.

REQ#5:   Transport Segment OAM SHOULD have the ability to allow the Initiator to add the Remote Transport Label and control the return path from egress responder. draft-ietf-mpls-bfd-directed has provided the semantics of a return path which would suit this need.

REQ#6:   Transport Segment OAM MUST have the ability to be initialized from an ingress POG node to perform connectivity verification and continuity check to any remote POG within the same optical domain ID based on the declared Transport Segment Label.

REQ#7:   In case of any failure with continuity check, Transport Segment OAM Layer SHOULD support rapid Connectivity Fault notification to the Packet Control plane of the POG to withdraw the Transport Segment Label associated with the affected path and/or take a local protection action.

REQ#8: Transport Segment OAM SHOULD also have the ability to be initialized from a centralized controller.

REQ#9: When Transport Segment OAM is initialized from centralized controller, the node on receiving the alert MAY take a local protection action and/or pop an informational message.

REQ#10: When Transport Segment OAM is initialized, it SHOULD support node redundancy based on network configuration. If primary Initiator fails, secondary one MUST take over the responsibility without having any impact on customer traffic.

REQ#11: Transport Segment OAM MUST have the ability to measure bidirectional packet loss, throughput measurement, delay variation, as well as unidirectional and dyadic measurements.

REQ#12: When a new path is instantiated, Transport Segment OAM SHOULD allow path verification without noticeable delay. It may be desired to check for liveliness of the optical path using Transport Segment OAM before announcing the Transport Segment.

REQ#13: The above listed requirements SHOULD be supported without any scalability limitation imposed and SHOULD be extensible to accommodate any new SR functionality.

REQ#14: Transport Segment OAM SHOULD maintain per Transport label state entry at the originating POG.

REQ#15: When traffic engineering is initiated by centralized controller device, and when Transport Segment OAM is performed by POGs, there MUST be a mechanism to communicate the failure to a centralized controller device.

REQ#16: When a local repair in the optical network takes place, the characteristics of the path between the POGS may have changed. If there is significant change in the path characteristics based on thresholds, the ingress POG SHALL trigger a re-advertisement of the transport segment label at the global level.

REQ#17: The format of the Transport Segment OAM Ping packet SHALL follow RFC 4379.

REQ#18: The format of the Transport Segment OAM BFD packet SHALL follow RFC 5884.

3 Security Considerations

This document does not introduce any new security considerations.

4 IANA Considerations

TBD.

5 References

5.1 Normative References

[I-D.ietf-spring-segment-routing]   Filsfils, C., Previdi, S., Decraene, B., Litkowski, S.,   and r. rjs@rob.sh, "Segment Routing Architecture", draft-ietf-spring-segment-routing-04 (work in progress), July 2015.

[I-D.ietf-mpls-bfd-directed]   Mirsky, G., Tantsura, J., Varlashkin, I., and M. Chen,   "Bidirectional Forwarding Detection (BFD) Directed Return   Path", draft-ietf-mpls-bfd-directed-02 (work in progress), March 2016.

[I-D.draft-anand-spring-poi-sr-01]   Madhukar Anand, Sanjoy Bardhan, Ramesh Subrahmaniam, Tantsura, J. "Packet-Optical Integration in Segment Routing", draft-anand-spring-poi-sr-01   (work in progress), July 2016.

5.2 Informative References

Authors’ Addresses

Sanjoy Bardhan
Infinera Corporation
169 W Java Dr, Sunnyvale, CA 94089

Email: sbardhan@infinera.com

Madhukar Anand
Infinera Corporation
169 W Java Dr, Sunnyvale, CA 94089

Email: manand@infinera.com

Ramesh Subrahmaniam
Infinera Corporation
169 W Java Dr, Sunnyvale, CA 94089

Email: RSubrahmaniam@infinera.com

Jeff Tantsura

Email: jefftant.ietf@gmail.com

Acknowledgments

The authors would like to thank Krish Verma for his comments and review of this document.

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 2
2. Conventions 3
3. In-band OAM Data Types and Data Format 3
3.1. In-band OAM Tracing Option 4
3.1.1. In-band OAM Trace Type and Node Data Element 7
3.2. In-band OAM Proof of Transit Option 10
3.3. In-band OAM Edge-to-Edge Option 11
4. In-band OAM Data Export 12
5. IANA Considerations 12
6. Manageability Considerations 12
7. Security Considerations 12
8. Acknowledgements 12
9. References 13
9.1. Normative References 13
9.2. Informative References 13
Authors' Addresses 13

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.

- o 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.

- o Generic data: Format-free information where syntax and semantic of the information is defined by the operator in a specific deployment. For a specific 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.

- o 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

ippm F. Brockners
Internet-Draft S. Bhandari
Intended status: Standards Track C. Pignataro
Expires: January 3, 2018 Cisco
H. Gredler
RtBrick Inc.
J. Leddy
Comcast
S. Youell
JPMC
T. Mizrahi
Marvell
D. Mozes
Mellanox Technologies Ltd.
P. Lapukhov
Facebook
R. Chang
Barefoot Networks
D. Bernier
Bell Canada
July 2, 2017

Data Fields for In-situ OAM
draft-brockners-inband-oam-data-07

Abstract

In-situ Operations, Administration, and Maintenance (IOAM) records operational and telemetry information in the packet while the packet traverses a path between two points in the network. This document discusses the data fields and associated data types for in-situ OAM. In-situ OAM data fields can be embedded into a variety of transports such as NSH, Segment Routing, Geneve, native IPv6 (via extension header), or IPv4. In-situ OAM can be used to complement OAM mechanisms based on e.g. ICMP or other types of probe packets.

Status of This Memo

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

This Internet-Draft will expire on January 3, 2018.

Copyright Notice

Table of Contents

1. Introduction 3
2. Conventions 3
3. Scope, Applicability, and Assumptions 4
4. IOAM Data Types and Formats 5
4.1. IOAM Tracing Options 6
4.1.1. Pre-allocated Trace Option 8
4.1.2. Incremental Trace Option 11
4.1.3. IOAM node data fields and associated formats 14
4.1.4. Examples of IOAM node data 19
4.2. IOAM Proof of Transit Option 21
4.3. IOAM Edge-to-Edge Option 23
5. IOAM Data Export 23
6. IANA Considerations 24
6.1. Creation of a New In-Situ OAM (IOAM) Protocol Parameters IANA registry 24
6.2. IOAM Trace Type Registry 24
6.3. IOAM Trace Flags Registry 24
6.4. IOAM POT Type Registry 25
6.5. IOAM E2E Type Registry 25
7. Manageability Considerations 25
8. Security Considerations 25
9. Acknowledgements 25
10. References 25
10.1. Normative References 25

10.2. Informative References 26
Authors' Addresses 27

1. Introduction

This document defines data fields for "in-situ" Operations, Administration, and Maintenance (IOAM). In-situ OAM records OAM information within the packet while the packet traverses a particular network domain. The term "in-situ" refers to the fact that the OAM data is added to the data packets rather than is being sent within packets specifically dedicated to OAM. A discussion of the motivation and requirements for in-situ OAM can be found in [I-D.brockners-inband-oam-requirements]. IOAM is to complement mechanisms such as Ping or Traceroute, or more recent active probing mechanisms as described in [I-D.lapukhov-dataplane-probe]. In terms of "active" or "passive" OAM, "in-situ" OAM can be considered a hybrid OAM type. While no extra packets are sent, IOAM adds information to the packets therefore cannot be considered passive. In terms of the classification given in [RFC7799] IOAM could be portrayed as Hybrid Type 1. "In-situ" mechanisms do not require extra packets to be sent and hence don't change the packet traffic mix within the network. IOAM mechanisms can be leveraged where mechanisms using e.g. ICMP do not apply or do not offer the desired results, such as proving that a certain traffic flow takes a pre-defined path, SLA verification for the live data traffic, detailed statistics on traffic distribution paths in networks that distribute traffic across multiple paths, or scenarios in which probe traffic is potentially handled differently from regular data traffic by the network devices.

2. Conventions

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

Abbreviations used in this document:

E2E Edge to Edge

Geneve: Generic Network Virtualization Encapsulation [I-D.ietf-nvo3-geneve]

IOAM: In-situ Operations, Administration, and Maintenance

MTU: Maximum Transmit Unit

NSH: Network Service Header [I-D.ietf-sfc-nsh]

OAM: Operations, Administration, and Maintenance

POT: Proof of Transit

SFC: Service Function Chain

SID: Segment Identifier

SR: Segment Routing

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

3. Scope, Applicability, and Assumptions

IOAM deployment assumes a set of constraints, requirements, and guiding principles which are described in this section.

Scope: This document defines the data fields and associated data types for in-situ OAM. The in-situ OAM data field can be transported by a variety of transport protocols, including NSH, Segment Routing, Geneve, IPv6, or IPv4. Specification details for these different transport protocols are outside the scope of this document.

Deployment domain (or scope) of in-situ OAM deployment: IOAM is a network domain focused feature, with "network domain" being a set of network devices or entities within a single administration. For example, a network domain can include an enterprise campus using physical connections between devices or an overlay network using virtual connections / tunnels for connectivity between said devices. A network domain is defined by its perimeter or edge. Designers of carrier protocols for IOAM must specify mechanisms to ensure that IOAM data stays within an IOAM domain. In addition, the operator of such a domain is expected to put provisions in place to ensure that IOAM data does not leak beyond the edge of an IOAM domain, e.g. using for example packet filtering methods. The operator should consider potential operational impact of IOAM to mechanisms such as ECMP processing (e.g. load-balancing schemes based on packet length could be impacted by the increased packet size due to IOAM), path MTU (i.e. ensure that the MTU of all links within a domain is sufficiently large to support the increased packet size due to IOAM) and ICMP message handling (i.e. in case of a native IPv6 transport, IOAM support for ICMPv6 Echo Request/Reply could desired which would translate into ICMPv6 extensions to enable IOAM data fields to be copied from an Echo Request message to an Echo Reply message).

IOAM control points: IOAM data fields are added to or removed from the live user traffic by the devices which form the edge of a domain.

Devices within an IOAM domain can update and/or add IOAM data-fields. Domain edge devices can be hosts or network devices.

Traffic-sets that IOAM is applied to: IOAM can be deployed on all or only on subsets of the live user traffic. It SHOULD be possible to enable IOAM on a selected set of traffic (e.g., per interface, based on an access control list or flow specification defining a specific set of traffic, etc.) The selected set of traffic can also be all traffic.

Encapsulation independence: Data formats for IOAM SHOULD be defined in a transport-independent manner. IOAM applies to a variety of encapsulating protocols. A definition of how IOAM data fields are carried by different transport protocols is outside the scope of this document.

Layering: If several encapsulation protocols (e.g., in case of tunneling) are stacked on top of each other, IOAM data-records could be present at every layer. The behavior follows the ships-in-the-night model.

Combination with active OAM mechanisms: IOAM should be usable for active network probing, enabling for example a customized version of traceroute. Decapsulating IOAM nodes may have an ability to send the IOAM information retrieved from the packet back to the source address of the packet or to the encapsulating node.

IOAM implementation: The IOAM data-field definitions take the specifics of devices with hardware data-plane and software data-plane into account.

4. IOAM Data Types and Formats

This section defines IOAM data types and data fields and associated data types required for IOAM. The different uses of IOAM require the definition of different types of data. The IOAM data fields for the data being carried corresponds to the three main categories of IOAM data defined in [I-D.brockners-inband-oam-requirements], which are: edge-to-edge, per node, and for selected nodes only.

Transport options for IOAM data are outside the scope of this memo, and are discussed in [I-D.brockners-inband-oam-transport]. IOAM data fields are fixed length data fields. A bit field determines the set of OAM data fields embedded in a packet. Depending on the type of the encapsulation, a counter field indicates how many data fields are included in a particular packet.

IOAM is expected to be deployed in a specific domain rather than on the overall Internet. The part of the network which employs IOAM is referred to as the "IOAM-domain". IOAM data is added to a packet upon entering the IOAM-domain and is removed from the packet when exiting the domain. Within the IOAM-domain, the IOAM data may be updated by network nodes that the packet traverses. The device which adds an IOAM data container to the packet to capture IOAM data is called the "IOAM encapsulating node", whereas the device which removes the IOAM data container is referred to as the "IOAM decapsulating node". Nodes within the domain which are aware of IOAM data and read and/or write or process the IOAM data are called "IOAM transit nodes". IOAM nodes which add or remove the IOAM data container can also update the IOAM data fields at the same time. Or in other words, IOAM encapsulation 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 Segment Routing deployment might require the segment routing path to be verified. In that case, only the SR nodes would also be IOAM transit nodes rather than all nodes.

4.1. IOAM Tracing Options

"IOAM tracing data" is expected to be collected at every 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 in-situ OAM-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 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.

To optimize hardware and software implementations tracing is defined as two separate options. Any deployment MAY choose to configure and support one or both of the following options. An implementation of the transport protocol that carries these in-situ OAM data MAY choose to support only one of the options. In the event that both options are utilized at the same time, the Incremental Trace Option MUST be placed before the Pre-allocated Trace Option. Given that the operator knows which equipment is deployed in a particular IOAM, the operator will decide by means of configuration which type(s) of trace options will be enabled for a particular domain.

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 software implementations where it is efficient to allocate the space once and index into the array to populate the data during transit. The IOAM encapsulating node allocates the option header and sets the fields in the option header. The in situ OAM encapsulating node allocates an array which is used to store operational data retrieved from every node while the packet traverses the domain. IOAM transit nodes update the content of the array. A pointer which is part of the IOAM trace data points to the next empty slot in the array, which is where the next IOAM transit node fills in its data.

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. The maximum length of the node data list is written into the option header. This type of trace recording is useful for some of the hardware implementations as this eliminates the need for the transit network elements to read the full array in the option and allows for arbitrarily long packets as the MTU allows. The in-situ OAM encapsulating node allocates the option header. The in-situ OAM encapsulating node based on operational state and configuration sets the fields in the header to control how large the node data list can grow. IOAM transit nodes push their node data to the node data list and increment the number of node data fields in the header.

Every node data entry is to hold information for a particular IOAM transit node that is traversed by a packet. The in-situ OAM decapsulating node removes the IOAM data and processes and/or exports the metadata. IOAM 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 IOAM data is defined for IOAM tracing:

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

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

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

- 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-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 deployment, all IOAM nodes should interpret the generic data the same way. Examples for generic IOAM data include geo-location information (location of the node at the time the packet was processed), buffer queue fill level or cache fill level at the time the packet was processed, or even a battery charge level.

- A mechanism to detect whether IOAM trace data was added at every hop or whether certain hops in the domain weren't in-situ 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.

4.1.1. Pre-allocated Trace Option

In-situ OAM pre-allocated trace option:

Pre-allocated trace option header:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         IOAM-Trace-Type       |NodeLen|  Flags  | Octets-left |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Pre-allocated Trace Option Data MUST be 4-octet aligned:

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|                                                               |  |
|                        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]                     |  |
|                                                               |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
```

IOAM-Trace-Type: A 16-bit identifier which specifies which data types are used in this node data list.

The IOAM-Trace-Type value is a bit field. The following bit fields are defined in this document, with details on each field described in the Section 4.1.3. The order of packing the data fields in each node data element follows the bit order of the IOAM-Trace-Type field, as follows:

Bit 0 (Most significant bit) When set indicates presence of Hop_Lim and node_id in the node data.

Bit 1 When set indicates presence of ingress_if_id and egress_if_id (short format) in the node data.

Bit 2 When set indicates presence of timestamp seconds in the node data

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

Bit 4 When set indicates presence of transit delay in the node data.

Bit 5 When set indicates presence of app_data (short format) in the node data.

Bit 6 When set indicates presence of queue depth in the node data.

Bit 7 When set indicates presence of variable length Opaque State Snapshot field.

Bit 8 When set indicates presence of Hop_Lim and node_id in wide format in the node data.

Bit 9 When set indicates presence of ingress_if_id and egress_if_id in wide format in the node data.

Bit 10 When set indicates presence of app_data wide in the node data.

Bit 11 When set indicates presence of the Checksum Complement node data.

Bit 12-15 Undefined in this draft.

Section 4.1.3 describes the IOAM data types and their formats. Within an in-situ OAM domain possible combinations of these bits making the IOAM-Trace-Type can be restricted by configuration knobs.

Node Data Length: 4-bit unsigned integer. This field specifies the length of data added by each node in multiples of 4-octets. For example, if 3 IOAM-Trace-Type bits are set and none of them is wide, then the Node Data Length would be 3. If 3 IOAM-Trace-Type bits are set and 2 of them are wide, then the Node Data Length would be 5.

Flags 5-bit field. Following flags are defined:

Bit 0 "Overflow" (O-bit) (most significant bit). This bit is set by the network element if there is not enough number of octets left to record node data, no field is added and the overflow "O-bit" must be set to "1" in the header. This is useful for transit nodes to ignore further processing of the option.

Bit 1 "Loopback" (L-bit). Loopback mode is used to send a copy of a packet back towards the source. Loopback mode assumes that a return path from transit nodes and destination nodes towards the source exists. The encapsulating node decides (e.g. using a filter) which packets loopback mode is enabled for by setting the loopback bit. The encapsulating node also needs to ensure that sufficient space is available in the IOAM header for loopback operation. The loopback bit when set indicates to the transit nodes processing this option to create a copy of the packet received and send this copy of the packet back to the source of the packet while it continues to forward the original packet towards the destination. The source address of the original packet is used as destination address in the copied packet. The address of the node performing the copy operation is used as the source address. The L-bit MUST be cleared in the copy of the packet a nodes sends it back towards the source. On its way back towards the source, the packet is processed like a regular packet with IOAM information. Once the return packet reaches the IOAM domain boundary IOAM decapsulation occurs as with any other packet containing IOAM information.

Bit 2-4 Reserved: Must be zero.

Octets-left: 7-bit unsigned integer. It is the data space in multiples of 4-octets remaining for recording the node data. This is used as an offset in data space to record the node data element.

Node data List [n]: Variable-length field. The type of which is determined by the IOAM-Trace-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 "Octets-left" identifies the offset for current active node data to be populated.

4.1.2. Incremental Trace Option

In-situ OAM incremental trace option:

In-situ OAM incremental trace option Header:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        IOAM-Trace-Type        |NodeLen|  Flags  | Max Length  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IOAM Incremental Trace Option Data MUST be 4-octet aligned:

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                        node data list [0]                     |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                        node data list [1]                     |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~                             ...                               ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                        node data list [n-1]                   |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                        node data list [n]                     |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IOAM-trace-type: A 16-bit identifier which specifies which data types are used in this node data list.

The IOAM-Trace-Type value is a bit field. The following bit fields are defined in this document, with details on each field described in the Section 4.1.3. The order of packing the data fields in each node data element follows the bit order of the IOAM-Trace-Type field, as follows:

Bit 0   (Most significant bit) When set indicates presence of Hop_Lim and node_id in the node data.

Bit 1   When set indicates presence of ingress_if_id and egress_if_id (short format) in the node data.

Bit 2 When set indicates presence of timestamp seconds in the node data.

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

Bit 4 When set indicates presence of transit delay in the node data.

Bit 5 When set indicates presence of app_data in the node data.

Bit 6 When set indicates presence of queue depth in the node data.

Bit 7 When set indicates presence of variable length Opaque State Snapshot field.

Bit 8 When set indicates presence of Hop_Lim and node_id wide in the node data.

Bit 9 When set indicates presence of ingress_if_id and egress_if_id in wide format in the node data.

Bit 10 When set indicates presence of app_data wide in the node data.

Bit 11 When set indicates presence of the Checksum Complement node data.

Bit 12-15 Undefined in this draft.

Section 4.1.3 describes the IOAM data types and their formats.

Node Data Length: 4-bit unsigned integer. This field specifies the length of data added by each node in multiples of 4-octets. For example, if 3 IOAM-Trace-Type bits are set and none of them is wide, then the Node Data Length would be 3. If 3 IOAM-Trace-Type bits are set and 2 of them are wide, then the Node Data Length would be 5.

Flags 5-bit field. Following flags are defined:

Bit 0 "Overflow" (O-bit) (least significant bit). This bit is set by the network element if there is not enough number of octets left to record node data, no field is added and the overflow "O-bit" must be set to "1" in the header. This is useful for transit nodes to ignore further processing of the option.

Bit 1 "Loopback" (L-bit). This bit when set indicates to the transit nodes processing this option to send a copy of the packet back to the source of the packet while it continues to forward the original packet towards the destination. The L-bit MUST be cleared in the copy of the packet before sending it.

Bit 2-4 Reserved. Must be zero.

Maximum Length: 7-bit unsigned integer. This field specifies the maximum length of the node data list in multiples of 4-octets. Given that the sender knows the minimum path MTU, the sender can set the maximum length according to the number of node data bytes allowed before exceeding the MTU. Thus, a simple comparison between "Opt data Len" and "Max Length" allows to decide whether or not data could be added.

Node data List [n]: Variable-length field. The type 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.

4.1.3. IOAM node data fields and associated formats

All the data fields MUST be 4-octet aligned. The IOAM encapsulating node MUST initialize data fields that it adds to the packet to zero. If a node which is supposed to update an IOAM data field is not capable of populating the value of a field set in the IOAM-Trace-Type, the field value MUST be left unaltered except when explicitly specified in the field description below. In the description of data below if zero is valid value then a non-zero value to mean not populated is specified.

Data field and associated data type for each of the data field is shown below:

Hop_Lim and node_id: 4-octet field defined as follows:

```
 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                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Hop_Lim: 1-octet unsigned integer. It is set to the Hop Limit value in the packet at the node that records this data. Hop Limit information is used to identify the location of the node in the communication path. This is copied from the lower layer, e.g., TTL value in IPv4 header or hop limit field from IPv6 header of the packet when the packet is ready for transmission. The semantics of the Hop_Lim field depend on the lower layer protocol that IOAM is encapsulated over, and therefore its specific semantics are outside the scope of this memo.

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

ingress_if_id and egress_if_id: 4-octet field defined as follows: When this field is part of the data field but a node populating the field is not able to fill it, the position in the field must be filled with value 0xFFFFFFFF to mean not populated.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     ingress_if_id             |         egress_if_id          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

ingress_if_id: 2-octet unsigned integer. Interface identifier to record the ingress interface the packet was received on.

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

timestamp seconds: 4-octet unsigned integer. Absolute timestamp in seconds that specifies the time at which the packet was received by the node. The structure of this field is identical to the most significant 32 bits of the 64 least significant bits of the [IEEE1588v2] timestamp. This truncated field consists of a 32-bit seconds field. As defined in [IEEE1588v2], the timestamp specifies the number of seconds elapsed since 1 January 1970 00:00:00 according to the International Atomic Time (TAI).

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        timestamp seconds                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

timestamp nanoseconds: 4-octet unsigned integer in the range 0 to 10^9-1. This timestamp specifies the fractional part of the wall clock time at which the packet was received by the node in units of nanoseconds. This field is identical to the 32 least significant bits of the [IEEE1588v2] timestamp. This fields allows for delay computation between any two nodes in the network when the nodes are time synchronized. When this field is part of the data field but a node populating the field is not able to fill it, the field position in the field must be filled with value 0xFFFFFFFF to mean not populated.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       timestamp nanoseconds                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

transit delay: 4-octet unsigned integer in the range 0 to 2^30-1. It is the time in nanoseconds the packet spent in the transit node. This can serve as an indication of the queuing delay at the node. If the transit delay exceeds 2^30-1 nanoseconds then the top bit 'O' is set to indicate overflow. When this field is part of the data field but a node populating the field is not able to fill it, the field position in the field must be filled with value 0xFFFFFFFF to mean not populated.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|O|                     transit delay                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

app_data: 4-octet placeholder which can be used by the node to add application specific data. App_data represents a "free-format" 4-octet bit field with its semantics defined by a specific deployment.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       app_data                                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

queue depth: 4-octet unsigned integer field. This field indicates the current length of the egress interface queue of the interface from where the packet is forwarded out. The queue depth is expressed as the current number of memory buffers used by the queue (a packet may consume one or more memory buffers, depending on its size). When this field is part of the data field but a node populating the field is not able to fill it, the field position in the field must be filled with value 0xFFFFFFFF to mean not populated.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       queue depth                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Opaque State Snapshot: Variable length field. It allows the network element to store an arbitrary state in the node data field , without a pre-defined schema. The schema needs to be made known to the analyzer by some out-of-band mechanism. The specification of this mechanism is beyond the scope of this document. The 24-bit "Schema Id" field in the field indicates which particular schema is used, and should be configured on the network element by the operator.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Length      |                     Schema ID                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                                                               |
|                        Opaque data                            |
~                                                               ~
.                                                               .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Length: 1-octet unsigned integer. It is the length in octets of the Opaque data field that follows Schema Id. It MUST always be a multiple of 4.

Schema ID: 3-octet unsigned integer identifying the schema of Opaque data.

Opaque data: Variable length field. This field is interpreted as specified by the schema identified by the Schema ID.

Hop_Lim and node_id wide: 8-octet field defined as follows:

```
 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                          ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~                         node_id (contd)                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Hop_Lim: 1-octet unsigned integer. It is set to the Hop Limit value in the packet at the node that records this data. Hop

Limit information is used to identify the location of the node in the communication path. This is copied from the lower layer for e.g. TTL value in IPv4 header or hop limit field from IPv6 header of the packet. The semantics of the Hop_Lim field depend on the lower layer protocol that IOAM is encapsulated over, and therefore its specific semantics are outside the scope of this memo.

node_id: 7-octet unsigned integer. Node identifier field to uniquely identify a node within in-situ OAM domain. The procedure to allocate, manage and map the node_ids is beyond the scope of this document.

ingress_if_id and egress_if_id wide: 8-octet field defined as follows: When this field is part of the data field but a node populating the field is not able to fill it, the field position in the field must be filled with value 0xFFFFFFFFFFFFFFFF to mean not populated.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        ingress_if_id                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        egress_if_id                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

ingress_if_id: 4-octet unsigned integer. Interface identifier to record the ingress interface the packet was received on.

egress_if_id: 4-octet unsigned integer. Interface identifier to record the egress interface the packet is forwarded out of.

app_data wide: 8-octet placeholder which can be used by the node to add application specific data. App data represents a "free-format" 8-octed bit field with its semantics defined by a specific deployment.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        app data                               ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~                        app data (contd)                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Checksum Complement: 4-octet node data which contains a two-octet Checksum Complement field, and a 2-octet reserved field. The Checksum Complement can be used when IOAM is transported over encapsulations that make use of a UDP transport, such as VXLAN-GPE or Geneve. In this case, incorporating the IOAM node data requires the UDP Checksum field to be updated. Rather than to recompute the Chekcsum field, a node can use the Checksum Complement to make a checksum-neutral update in the UDP payload; the Checksum Complement is assigned a value that complements the rest of the node data fields that were added by the current node, causing the existing UDP Checksum field to remain correct. Checksum Complement fields are used in a similar manner in [RFC7820] and [RFC7821].

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Checksum Complement      |           Reserved            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

4.1.4. Examples of IOAM node data

An entry in the "node data list" array can have different formats, following the needs of the 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 types that an entry in "node data list" can take.

0x002B: IOAM-Trace-Type is 0x2B 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 nanoseconds                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             app_data                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

0x0003: IOAM-Trace-Type is 0x0003 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          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

0x0009: IOAM-Trace-Type is 0x0009 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 nanoseconds                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

0x0021: IOAM-Trace-Type is 0x0021 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                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

0x0029: IOAM-Trace-Type is 0x0029 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 nanoseconds                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                app_data                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

0x018C: IOAM-Trace-Type is 0x104D 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      timestamp seconds                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    timestamp nanoseconds                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Length      |                     Schema Id                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                                                               |
|                        Opaque data                            |
~                                                               ~
.                                                               .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Hop_Lim     |              node_id                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         node_id(contd)                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

4.2. IOAM Proof of Transit Option

IOAM 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 IOAM data or mechanisms such as Shamir’s Secret Sharing Schema (SSSS). While details on how the IOAM data for the proof of transit option is processed at IOAM 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 IOAM 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.

IOAM proof of transit option:

IOAM proof of transit option header:

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|IOAM POT Type|P|
+-+-+-+-+-+-+-+-+
```

IOAM proof of transit option data MUST be 4-octet aligned:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|                           Random                              |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  P
|                         Random(contd)                         |  O
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  T
|                          Cumulative                           |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
|                          Cumulative (contd)                   |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
```

IOAM POT Type: 7-bit identifier of a particular POT variant that dictates the POT data that is included. This document defines POT Type 0:

0: POT data is a 16 Octet field as described below.

Profile to use (P): 1-bit. Indicates which POT-profile is used to generate the Cumulative. Any node participating in POT will have a maximum of 2 profiles configured that drive the computation of cumulative. The two profiles are numbered 0, 1. This bit conveys whether profile 0 or profile 1 is used to compute the Cumulative.

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".

4.3. IOAM Edge-to-Edge Option

The IOAM edge-to-edge option 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 without modifying it.

Currently only sequence numbers use the IOAM edge-to-edge option. In order to detect packet loss, packet reordering, or packet duplication in an in-situ 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.

IOAM edge-to-edge option:

IOAM edge-to-edge option header:

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
| IOAM-E2E-Type |
+-+-+-+-+-+-+-+-+
```

IOAM edge-to-edge option data MUST be 4-octet aligned:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       E2E Option data field determined by IOAM-E2E-Type       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IOAM-E2E-Type: 8-bit identifier of a particular in situ 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.

5. 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.

The discussion of IOAM data processing and export is left for a future version of this document.

6. IANA Considerations

This document requests the following IANA Actions.

6.1. Creation of a New In-Situ OAM (IOAM) Protocol Parameters IANA registry

IANA is requested to create a new protocol registry for "In-Situ OAM (IOAM) Protocol Parameters". This is the common registry that will include registrations for all IOAM namespaces. Each Registry, whose names are listed below:

IOAM Trace Type

IOAM Trace flags

IOAM POT Type

IOAM E2E Type

will contain the current set of possibilities defined in this document. New registries in this name space are created via RFC Required process as per [RFC8126].

The subsequent sub-sections detail the registries herein contained.

6.2. IOAM Trace Type Registry

This registry defines code point for each bit in the 16-bit IOAM-Trace-Type field for Pre-allocated trace option and Incremental trace option defined in Section 4.1. The meaning of Bit 0 - 11 for trace type are defined in this document in Paragraph 1 of (Section 4.1.1). The meaning for Bit 12 - 15 are available for assignment via RFC Required process as per [RFC8126].

6.3. IOAM Trace Flags Registry

This registry defines code point for each bit in the 5 bit flags for Pre-allocated trace option and Incremental trace option defined in Section 4.1. The meaning of Bit 0 - 1 for trace flags are defined in this document in Paragraph 5 of Section 4.1.1. The meaning for Bit 2 - 4 are available for assignment via RFC Required process as per [RFC8126].

6.4. IOAM POT Type Registry

This registry defines 128 code points to define IOAM POT Type for IOAM proof of transit option Section 4.2. The code point value 0 is defined in this document, 1 - 127 are available for assignment via RFC Required process as per [RFC8126].

6.5. IOAM E2E Type Registry

This registry defines 256 code points to define IOAM-E2E-Type for IOAM E2E option Section 4.3. The code point value 0 is defined in this document, 1 - 255 are available for assignments via RFC Required process as per [RFC8126].

7. Manageability Considerations

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

8. Security Considerations

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

9. Acknowledgements

The authors would like to thank Eric Vyncke, Nalini Elkins, Srihari Raghavan, Ranganathan T S, Karthik Babu Harichandra Babu, Akshaya Nadahalli, LJ Wobker, Erik Nordmark, Vengada Prasad Govindan, and Andrew Yourtchenko for the comments and advice.

This document leverages and builds on top of several concepts described in [I-D.kitamura-ipv6-record-route]. The authors would like to acknowledge the work done by the author Hiroshi Kitamura and people involved in writing it.

The authors would like to gracefully acknowledge useful review and insightful comments received from Joe Clarke, Al Morton, and Mickey Spiegel.

10. References

10.1. Normative References

[IEEE1588v2]
Institute of Electrical and Electronics Engineers, "1588-2008 - IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems", IEEE Std 1588-2008, 2008, <http://standards.ieee.org/findstds/standard/1588-2008.html>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://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, <http://www.rfc-editor.org/info/rfc8126>.

10.2. Informative References

[I-D.brockners-inband-oam-requirements]
Brockners, F., Bhandari, S., Dara, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mozes, D., Mizrahi, T., <>, P., and r. remy@barefootnetworks.com, "Requirements for In-situ OAM", draft-brockners-inband-oam-requirements-03 (work in progress), March 2017.

[I-D.brockners-inband-oam-transport]
Brockners, F., Bhandari, S., Govindan, V., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Mozes, D., Lapukhov, P., and R. <>, "Encapsulations for In-situ OAM Data", draft-brockners-inband-oam-transport-03 (work in progress), March 2017.

[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.ietf-nvo3-geneve]
Gross, J., Ganga, I., and T. Sridhar, "Geneve: Generic Network Virtualization Encapsulation", draft-ietf-nvo3-geneve-04 (work in progress), March 2017.

[I-D.ietf-nvo3-vxlan-gpe]
Maino, F., Kreeger, L., and U. Elzur, "Generic Protocol Extension for VXLAN", draft-ietf-nvo3-vxlan-gpe-04 (work in progress), April 2017.

[I-D.ietf-sfc-nsh]
Quinn, P. and U. Elzur, "Network Service Header", draft-ietf-sfc-nsh-13 (work in progress), June 2017.

[I-D.kitamura-ipv6-record-route]
Kitamura, H., "Record Route for IPv6 (PR6) Hop-by-Hop Option Extension", draft-kitamura-ipv6-record-route-00 (work in progress), November 2000.

[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.

[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>.

[RFC7799] Morton, A., "Active and Passive Metrics and Methods (with Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799, May 2016, <http://www.rfc-editor.org/info/rfc7799>.

[RFC7820] Mizrahi, T., "UDP Checksum Complement in the One-Way Active Measurement Protocol (OWAMP) and Two-Way Active Measurement Protocol (TWAMP)", RFC 7820, DOI 10.17487/RFC7820, March 2016, <http://www.rfc-editor.org/info/rfc7820>.

[RFC7821] Mizrahi, T., "UDP Checksum Complement in the Network Time Protocol (NTP)", RFC 7821, DOI 10.17487/RFC7821, March 2016, <http://www.rfc-editor.org/info/rfc7821>.

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

John Leddy
Comcast

Email: John_Leddy@cable.comcast.com

Stephen Youell
JP Morgan Chase
25 Bank Street
London E14 5JP
United Kingdom

Email: stephen.youell@jpmorgan.com

Tal Mizrahi
Marvell
6 Hamada St.
Yokneam 2066721
Israel

Email: talmi@marvell.com

David Mozes
Mellanox Technologies Ltd.

Email: davidm@mellanox.com

Petr Lapukhov
Facebook
1 Hacker Way
Menlo Park, CA 94025
US

Email: petr@fb.com

Remy Chang
Barefoot Networks
2185 Park Boulevard
Palo Alto, CA 94306
US

Daniel
Bell Canada

Email: daniel.bernier@bell.ca

Network Working Group
Internet-Draft
Intended status: Informational
Expires: January 9, 2017

F. Brockners
S. Bhandari
S. Dara
C. Pignataro
Cisco
H. Gredler
RtBrick Inc.
July 8, 2016

Requirements for In-band OAM
draft-brockners-inband-oam-requirements-00

Abstract

This document discusses the motivation and requirements for including specific operational and telemetry information into data packets while the data packet traverses a path between two points in the network. This method is referred to as "in-band" Operations, Administration, and Maintenance (OAM), given that the OAM information is carried with the data packets as opposed to in "out-of-band" packets dedicated to OAM. In-band OAM complements other OAM mechanisms which use dedicated probe packets to convey OAM information.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction 2
2. Conventions 4
3. Motivation for In-band OAM 4
3.1. Path Congruency Issues with Dedicated OAM Packets 4
3.2. Results Sent to a System Other Than the Sender 5
3.3. Overlay and Underlay Correlation 5
3.4. SLA Verification 6
3.5. Analytics and Diagnostics 6
3.6. Frame Replication/Elimination Decision for Bi-casting /Active-active Networks 7
3.7. Proof of Transit 7
3.8. Use Cases 8
4. Considerations for In-band OAM 9
4.1. Type of Information to Be Recorded 9
4.2. MTU and Packet Size 10
4.3. Administrative Boundaries 10
4.4. Selective Enablement 11
4.5. Optimization of Node and Interface Identifiers 11
4.6. Loop Communication Path (IPv6-specifics) 11
5. Requirements for In-band OAM Data Types 12
5.1. Generic Requirements 12
5.2. In-band OAM Data with Per-hop Scope 13
5.3. In-band OAM with Selected Hop Scope 14
5.4. In-band OAM with End-to-end Scope 14
6. Security Considerations and Requirements 14
6.1. Proof of Transit 14
7. IANA Considerations 15
8. Acknowledgements 15
9. Informative References 16
Authors' Addresses 17

1. Introduction

This document discusses requirements for "in-band" Operations, Administration, and Maintenance (OAM) mechanisms. "In-band" OAM means to record OAM and telemetry information within the data packet while the data packet traverses a network or a particular network domain. The term "in-band" refers to the fact that the OAM and telemetry data is carried within data packets rather than being sent within packets specifically dedicated to OAM. In-band OAM mechanisms, which are sometimes also referred to as embedded network telemetry are a current topic of discussion. In-band network telemetry has been defined for P4 [P4]. The SPUD prototype [I-D.hildebrand-spud-prototype] uses a similar logic that allows network devices on the path between endpoints to participate explicitly in the tube outside the end-to-end context. Even the IPv4 route-record option defined in [RFC0791] can be considered an in-band OAM mechanism. In-band OAM complements "out-of-band" 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 characteristics or requirements, such as proving that a certain set of traffic takes a pre-defined path, strict congruency is desired, checking service level agreements 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. [RFC7276] presents an overview of OAM tools.

Compared to probably the most basic example of "in-band OAM" which is IPv4 route recording [RFC0791], an in-band OAM approach has the following capabilities:

a. A flexible data format to allow different types of information to be captured as part of an in-band OAM operation, including not only path tracing information, but additional operational and telemetry information such as timestamps, sequence numbers, or even generic data such as queue size, geo-location of the node that forwarded the packet, etc.

b. A data format to express node as well as link identifiers to record the path a packet takes with a fixed amount of added data.

c. The ability to detect whether any nodes were skipped while recording in-band OAM information (i.e., in-band OAM is not supported or not enabled on those nodes).

d. The ability to actively process information in the packet, for example to prove in a cryptographically secure way that a packet really took a pre-defined path using some traffic steering method such as service chaining or traffic engineering.

e. The ability to include OAM data beyond simple path information, such as timestamps or even generic data of a particular use case.

f. The ability to include OAM data in various different transport protocols.

2. Conventions

Abbreviations used in this document:

ECMP: Equal Cost Multi-Path

MTU: Maximum Transmit Unit

NFV: Network Function Virtualization

OAM: Operations, Administration, and Maintenance

PMTU: Path MTU

SLA: Service Level Agreement

SFC: Service Function Chain

SR: Segment Routing

This document defines in-band Operations, Administration, and Maintenance (in-band OAM), as the subset in which OAM information is carried along with data packets. This is as opposed to "out-of-band OAM", where specific packets are dedicated to carrying OAM information.

3. Motivation for In-band OAM

In several scenarios it is beneficial to make information about which path a packet took through the network available to the operator. This includes not only tasks like debugging, troubleshooting, as well as network planning and network optimization but also policy or service level agreement compliance checks. This section discusses the motivation to introduce new methods for enhanced in-band network diagnostics.

3.1. Path Congruency Issues with Dedicated OAM Packets

Mechanisms which add tracing information to the regular data traffic, sometimes also referred to as "in-band" or "passive OAM" can complement active, probe-based mechanisms such as ping or traceroute, which are sometimes considered as "out-of-band", because the messages are transported independently from regular data traffic. "In-band" mechanisms do not require extra packets to be sent and hence don't change the packet traffic mix within the network. Traceroute and ping for example use ICMP messages: New packets are injected to get tracing information. Those add to the number of messages in a network, which already might be highly loaded or suffering performance issues for a particular path or traffic type.

Packet scheduling algorithms, especially for balancing traffic across equal cost paths or links, often leverage information contained within the packet, such as protocol number, IP-address or MAC-address. Probe packets would thus either need to be sent from the exact same endpoints with the exact same parameters, or probe packets would need to be artificially constructed as "fake" packets and inserted along the path. Both approaches are often not feasible from an operational perspective, be it that access to the end-system is not feasible, or that the diversity of parameters and associated probe packets to be created is simply too large. An in-band mechanism is an alternative in those cases.

In-band mechanisms also don't suffer from implementations, where probe traffic is handled differently (and potentially forwarded differently) by a router than regular data traffic.

3.2. Results Sent to a System Other Than the Sender

Traditional ping and traceroute tools return the OAM results to the sender of the probe. Even when the ICMP messages that are used with these tools are enhanced, and additional telemetry is collected (e.g., ICMP Multi-Part [RFC4884] supporting MPLS information [RFC4950], Interface and Next-Hop Identification [RFC5837], etc.), it would be advantageous to separate the sending of an OAM probe from the receiving of the telemetry data. In this context, it is desired to not assume there is a bidirectional working path.

3.3. Overlay and Underlay Correlation

Several network deployments leverage tunneling mechanisms to create overlay or service-layer networks. Examples include VXLAN-GPE, GRE, or LISP. One often observed attribute of overlay networks is that they do not offer the user of the overlay any insight into the underlay network. This means that the path that a particular tunneled packet takes, nor other operational details such as the per-hop delay/jitter in the underlay are visible to the user of the overlay network, giving rise to diagnosis and debugging challenges in case of connectivity or performance issues. The scope of OAM tools like ping or traceroute is limited to either the overlay or the underlay which means that the user of the overlay has typically no access to OAM in the underlay, unless specific operational procedures are put in place. With in-band OAM the operator of the underlay can offer details of the connectivity in the underlay to the user of the overlay. The operator of the egress tunnel router could choose to share the recorded information about the path with the user of the overlay.

Coupled with mechanisms such as Segment Routing (SR) [I-D.ietf-spring-segment-routing], overlay network and underlay network can be more tightly coupled: The user of the overlay has detailed diagnostic information available in case of failure conditions. The user of the overlay can also use the path recording information as input to traffic steering or traffic engineering mechanisms, to for example achieve path symmetry for the traffic between two endpoints. [I-D.brockners-lisp-sr] is an example for how these methods can be applied to LISP.

3.4. SLA Verification

In-band OAM can help users of an overlay-service to verify that negotiated SLAs for the real traffic are met by the underlay network provider. Different from solutions which rely on active probes to test an SLA, in-band OAM based mechanisms avoid wrong interpretations and "cheating", which can happen if the probe traffic that is used to perform SLA-check is prioritized by the network provider of the underlay.

3.5. Analytics and Diagnostics

Network planners and operators benefit from knowledge of the actual traffic distribution in the network. When deriving an overall network connectivity traffic matrix one typically needs to correlate data gathered from each individual devices in the network. If the path of a packet is recorded while the packet is forwarded, the entire path that a packet took through the network is available to the egress system. This obviates the need to retrieve individual traffic statistics from every device in the network and correlate those statistics, or employ other mechanisms such as leveraging traffic engineering with null-bandwidth tunnels just to retrieve the appropriate statistics to generate the traffic matrix.

In addition, with individual path tracing, information is available at packet level granularity, rather than only at aggregate level - as is usually the case with IPFIX-style methods which employ flow-filters at the network elements. Data-center networks which use equal-cost multipath (ECMP) forwarding are one example where detailed statistics on flow distribution in the network are highly desired. If a network supports ECMP, one can create detailed statistics for the different paths packets take through the network at the egress system, without a need to correlate/aggregate statistics from every router in the system. Transit devices are off-loaded from the task of gathering packet statistics.

3.6. Frame Replication/Elimination Decision for Bi-casting/Active-active Networks

Bandwidth- and power-constrained, time-sensitive, or loss-intolerant networks (e.g., networks for industry automation/control, health care) require efficient OAM methods to decide when to replicate packets to a secondary path in order to keep the loss/error-rate for the receiver at a tolerable level - and also when to stop replication and eliminate the redundant flow. Many IoT networks are time sensitive and cannot leverage automatic retransmission requests (ARQ) to cope with transmission errors or lost packets. Transmitting the data over multiple disparate paths (often called bi-casting or live-live) is a method used to reduce the error rate observed by the receiver. TSN receive a lot of attention from the manufacturing industry as shown by a various standardization activities and industry forums being formed (see e.g., IETF 6TiSCH, IEEE P802.1CB, AVnu).

3.7. Proof of Transit

Several deployments use traffic engineering, policy routing, segment routing or Service Function Chaining (SFC) [RFC7665] to steer packets through a specific set of nodes. In certain cases regulatory obligations or a compliance policy require to prove that all packets that are supposed to follow a specific path are indeed being forwarded across the exact set of nodes specified. If a packet flow is supposed to go through a series of service functions or network nodes, it has to be proven that all packets of the flow actually went through the service chain or collection of nodes specified by the policy. In case the packets of a flow weren't appropriately processed, a verification device would be required to identify the policy violation and take corresponding actions (e.g., drop or redirect the packet, send an alert etc.) corresponding to the policy. In today's deployments, the proof that a packet traversed a particular service chain is typically delivered in an indirect way: Service appliances and network forwarding are in different trust domains. Physical hand-off-points are defined between these trust domains (i.e., physical interfaces). Or in other terms, in the "network forwarding domain" things are wired up in a way that traffic is delivered to the ingress interface of a service appliance and received back from an egress interface of a service appliance. This "wiring" is verified and trusted. The evolution to Network Function Virtualization (NFV) and modern service chaining concepts (using technologies such as LISP, NSH, Segment Routing, etc.) blurs the line between the different trust domains, because the hand-off-points are no longer clearly defined physical interfaces, but are virtual interfaces. Because of that very reason, networks operators require that different trust layers not to be mixed in the same device. For an NFV scenario a different proof is required. Offering a proof that a packet traversed a specific set of service functions would allow network operators to move away from the above described indirect methods of proving that a service chain is in place for a particular application.

A solution approach could be based on OAM data which is added to every packet for achieving Proof Of Transit. The OAM data is updated at every hop and is used to verify whether a packet traversed all required nodes. When the verifier receives each packet, it can validate whether the packet traversed the service chain correctly. The detailed mechanisms used for path verification along with the procedures applied to the OAM data carried in the packet for path verification are beyond the scope of this document. Details are addressed in [draft-brockners-proof-of-transit]. In this document the term "proof" refers to a discrete set of bits that represents an integer or string carried as OAM data. The OAM data is used to verify whether a packet traversed the nodes it is supposed to traverse.

3.8. Use Cases

In-band OAM could be leveraged for several use cases, including:

- Traffic Matrix: Derive the network traffic matrix: Traffic for a given time interval between any two edge nodes of a given domain. Could be performed for all traffic or per QoS-class.

- Flow Debugging: Discover which path(s) a particular set of traffic (identified by an n-tuple) takes in the network. Such a procedure is particularly useful in case traffic is balanced across multiple paths, like with link aggregation (LACP) or equal cost multi-pathing (ECMP).

- Loss Statistics per Path: Retrieve loss statistics per flow and path in the network.

- Path Heat Maps: Discover highly utilized links in the network.

- Trend Analysis on Traffic Patterns: Analyze if (and if so how) the forwarding path for a specific set of traffic changes over time (can give hints to routing issues, unstable links etc.).

- Network Delay Distribution: Show delay distribution across network by node or links. If enabled per application or for a specific flow then display the path taken along with the delay incurred at every hop.

- SLA Verification: Verify that a negotiated service level agreement (SLA), e.g., for packet drop rates or delay/jitter is conformed to by the actual traffic.

- Low-power Networks: Include application level OAM information (e.g., battery charge level, cache or buffer fill level) into data traffic to avoid sending extra OAM traffic which incur an extra cost on the devices. Using the battery charge level as example, one could avoid sending extra OAM packets just to communicate battery health, and as such would save battery on sensors.

- Path Verification or Service Function Path Verification: Proof and verification of packets traversing check points in the network, where check points can be nodes in the network or service functions.

- Geo-location Policy: Network policy implemented based on which path packets took. Example: Only if packets originated and stayed within the trading-floor department, access to specific applications or servers is granted.

4. Considerations for In-band OAM

The implementation of an in-band OAM mechanism needs to take several considerations into account, including administrative boundaries, how information is recorded, Maximum Transfer Unit (MTU), Path MTU discovery and packet size, etc.

4.1. Type of Information to Be Recorded

The information gathered for in-band OAM can be categorized into three main categories: Information with a per-hop scope, such as path tracing; information which applies to a specific set of nodes, such as path or service chain verification; information which only applies to the edges of a domain, such as sequence numbers.

- "edge to edge": Information that needs to be shared between network edges (the "edge" of a network could either be a host or a domain edge device): Edge to edge data e.g., packet and octet count of data entering a well-defined domain and leaving it is helpful in building traffic matrix, sequence number (also called "path packet counters") is useful for the flow to detect packet loss.

- o "selected hops": Information that applies to a specific set of nodes only. In case of path verification, only the nodes which are "check points" are required to interpret and update the information in the packet.

- o "per hop": Information that is gathered at every hop along the path a packet traverses within an administrative domain:

  * Hop by Hop information e.g., Nodes visited for path tracing, Timestamps at each hop to find delays along the path

  * Stats collection at each hop to optimize communication in resource constrained networks e.g., Battery, CPU, memory status of each node piggy backed in a data packet is useful in low power lossy networks where network nodes are mostly asleep and communication is expensive

4.2. MTU and Packet Size

The recorded data at every hop may lead to packet size exceeding the Maximum Transmit Unit (MTU). Based on the transport protocol used MTU is discovered as a configuration parameter or Path MTU (PMTU) is discovered dynamically. Example: IPv6 recommends PMTU discovery before data packets are sent to prevent packet fragmentation. It specifies 1280 octets as the default PDU to be carried in a IPv6 datagram. A detailed discussion of the implications of oversized IPv6 header chains if found in [RFC7112].

The Path MTU restricts the amount of data that can be recorded for purpose of OAM within a data packet. The total size of data to be recorded needs to be preset to avoid packet size exceeding the MTU. It is recommended to pre-calculate and configures network devices to limit the in-band OAM data that is attached to a packet.

4.3. Administrative Boundaries

There are challenges in enabling in-band OAM in the public Internet across administrative domains:

- o Deployment dependent, the data fields that in-band OAM requires as part of a specific transport protocol may not be supported across administrative boundaries.

- o Current OAM implementations are often done in the slow path, i.e., OAM packets are punted to router’s CPU for processing. This leads to performance and scaling issues and opens up routers for attacks such as Denial of Service (DoS) attacks.

- o Discovery of network topology and details of the network devices across administrative boundaries may open up attack vectors compromising network security.

- o Specifically on IPv6: At the administrative boundaries IPv6 packets with extension headers are dropped for several reasons described in [RFC7872]

The following considerations will be discussed in a future version of this document: If the packet is dropped due to the presence of the in-band OAM; If the policy failure is treated as feature disablement and any further recording is stopped but the packet itself is not dropped, it may lead to every node in the path to make this policy decision.

4.4. Selective Enablement

Deployment dependent, in-band OAM could either be used for all, or only a subset of the overall traffic. While it might be desirable to apply in-band OAM to all traffic and then selectively use the data gathered in case needed, it might not always be feasible. Depending on the forwarding infrastructure used, in-band OAM can have an impact on forwarding performance. The SPUD prototype for example uses the notion of "pipes" to describe the portion of the traffic that could be subject to in-path inspection. Mechanisms to decide which traffic would be subject to in-band OAM are outside the scope of this document.

4.5. Optimization of Node and Interface Identifiers

Since packets have a finite maximum size, the data recording or carrying capacity of one packet in which the in-band OAM meta data is present is limited. In-band OAM should use its own dedicated namespace (confined to the domain in-band OAM operates in) to represent node and interface IDs to save space in the header. Generic representations of node and interface identifiers which are globally unique (such as a UUID) would consume significantly more bits of in-band OAM data.

4.6. Loop Communication Path (IPv6-specifics)

When recorded data is required to be analyzed on a source node that issues a packet and inserts in-band OAM data, the recorded data needs to be carried back to the source node.

One way to carry the in-band OAM data back to the source is to utilize an ICMP Echo Request/Reply (ping) or ICMPv6 Echo Request/ Reply (ping6) mechanism. In order to run the in-band OAM mechanism appropriately on the ping/ping6 mechanism, the following two operations should be implemented by the ping/ping6 target node:

1. All of the in-band OAM fields would be copied from an Echo Request message to an Echo Reply message.

2. The Hop Limit field of the IPv6 header of these messages would be copied as a continuous sequence. Further considerations are addressed in a future version of this document.

5. Requirements for In-band OAM Data Types

The above discussed use cases require different types of in-band OAM data. This section details requirements for in-band OAM derived from the discussion above.

5.1. Generic Requirements

REQ-G1: Classification: It should be possible to enable in-band OAM on a selected set of traffic. The selected set of traffic can also be all traffic.

REQ-G2: Scope: If in-band OAM is used only within a specific domain, provisions need to be put in place to ensure that in-band OAM data stays within the specific domain only.

REQ-G3: Transport independence: Data formats for in-band OAM shall be defined in a transport independent way. In-band OAM applies to a variety of transport protocols. Encapsulations should be defined how the generic data formats are carried by a specific protocol.

REQ-G4: Layering: It should be possible to have in-band OAM information for different transport protocol layers be present in several fields within a single packet. This could for example be the case when tunnels are employed and in-band OAM information is to be gathered for both the underlay as well as the overlay network.

REQ-G5: MTU size: With in-band OAM information added, packets should not become larger than the path MTU.

REQ-G6: Data Structure Reusability: The data types and data formats defined and used for in-band OAM ought to be reusable for out-of-band OAM telemetry as well.

5.2. In-band OAM Data with Per-hop Scope

REQ-H1: Missing nodes detection: Data shall be present that allows a node to detect whether all nodes that should participate in in-band OAM operations have indeed participated.

REQ-H2: Node, instance or device identifier: Data shall be present that allows to retrieve the identity of the entity reporting telemetry information. The entity can be a device, or a subsystem/component within a device. The latter will allow for packet tracing within a device in much the same way as between devices.

REQ-H3: Ingress interface identifier: Data shall be present that allows the identification of the interface a particular packet was received from. The interface can be a logical or physical entity.

REQ-H4: Egress interface identifier: Data shall be present that allows the identification of the interface a particular packet was forwarded to. Interface can be a logical or physical entity.

REQ-H5: Time-related requirements

REQ-H5.1: Delay: Data shall be present that allows to retrieve the delay between two or more points of interest within the system. Those points can be within the same device or on different devices.

REQ-H5.2: Jitter: Data shall be present that allows to retrieve the jitter between two or more points of interest within the system. Those points can be within the same device or on different devices.

REQ-H5.3: Wall-clock time: Data shall be present that allows to retrieve the wall-clock time visited a particular point of interest in the system.

REQ-H5.4: Time precision: The precision of the time related data should be configurable. Use-case dependent, the required precision could e.g., be nano-seconds, micro-seconds, milli-seconds, or seconds.

REQ-H6: Generic data records (like e.g., GPS/Geo-location information): It should be possible to add user-defined OAM data at select hops to the packet. The semantics of the data are defined by the user.

5.3. In-band OAM with Selected Hop Scope

REQ-S1: Proof of transit: Data shall be present which allows to securely prove that a packet has visited or ore several particular points of interest (i.e., a particular set of nodes).

REQ-S1.1: In case "Shamir’s secret sharing scheme" is used for proof of transit, two data records, "random" and "cumulative" shall be present. The number of bits used for "random" and "cumulative" data records can vary between deployments and should thus be configurable.

5.4. In-band OAM with End-to-end Scope

REQ-E1: Sequence numbering:

REQ-E1.1: Reordering detection: It should be possible to detect whether packets have been reordered while traversing an in-band OAM domain.

REQ-E1.2: Duplicates detection: It should be possible to detect whether packets have been duplicated while traversing an in-band OAM domain.

REQ-E1.3: Detection of packet drops: It should be possible to detect whether packets have been dropped while traversing an in-band OAM domain.

6. Security Considerations and Requirements

General Security considerations will be addressed in a later version of this document. Security considerations for Proof of Transit alone are discussed below.

6.1. Proof of Transit

Threat Model: Attacks on the deployments could be due to malicious administrators or accidental misconfigurations resulting in bypassing of certain nodes. The solution approach should meet the following requirements:

REQ-SEC1: Sound Proof of Transit: A valid and verifiable proof that the packet definitively traversed through all the nodes as expected. Probabilistic methods to achieve this should be avoided, as the same could be exploited by an attacker.

REQ-SEC2: Tampering of meta data: An active attacker should not be able to insert or modify or delete meta data in whole or in parts and bypass few (or all) nodes. Any deviation from the expected path should be accurately determined.

REQ-SEC3: Replay Attacks: A attacker (active/passive) should not be able to reuse the proof of transit bits in the packet by observing the OAM data in the packet, packet characteristics (like IP addresses, octets transferred, timestamps) or even the proof bits themselves. The solution approach should consider usage of these parameters for deriving any secrets cautiously. Mitigating replay attacks beyond a window of longer duration could be intractable to achieve with fixed number of bits allocated for proof.

REQ-SEC4: Recycle Secrets: Any configuration of the secrets (like cryptographic keys, initialisation vectors etc.) either in the controller or service functions should be reconfigurable. Solution approach should enable controls, API calls etc. needed in order to perform such recycling. It is desirable to provide recommendations on the duration of rotation cycles needed for the secure functioning of the overall system.

REQ-SEC5: Secret storage and distribution: Secrets should be shared with the devices over secure channels. Methods should be put in place so that secrets cannot be retrieved by non authorized personnel from the devices.

7. IANA Considerations

[RFC Editor: please remove this section prior to publication.]

This document has no IANA actions.

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. Informative References

[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.

[I-D.brockners-lisp-sr]
Brockners, F., Bhandari, S., Maino, F., and D. Lewis, "LISP Extensions for Segment Routing", draft-brockners-lisp-sr-01 (work in progress), February 2014.

[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.ietf-spring-segment-routing]
Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", draft-ietf-spring-segment-routing-09 (work in progress), July 2016.

[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.

[RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, DOI 10.17487/RFC0791, September 1981, <http://www.rfc-editor.org/info/rfc791>.

[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, <http://www.rfc-editor.org/info/rfc4884>.

[RFC4950] Bonica, R., Gan, D., Tappan, D., and C. Pignataro, "ICMP Extensions for Multiprotocol Label Switching", RFC 4950, DOI 10.17487/RFC4950, August 2007, <http://www.rfc-editor.org/info/rfc4950>.

[RFC5837] Atlas, A., Ed., Bonica, R., Ed., Pignataro, C., Ed., Shen, N., and JR. Rivers, "Extending ICMP for Interface and Next-Hop Identification", RFC 5837, DOI 10.17487/RFC5837, April 2010, <http://www.rfc-editor.org/info/rfc5837>.

[RFC7112] Gont, F., Manral, V., and R. Bonica, "Implications of Oversized IPv6 Header Chains", RFC 7112, DOI 10.17487/RFC7112, January 2014, <http://www.rfc-editor.org/info/rfc7112>.

[RFC7276] Mizrahi, T., Sprecher, N., Bellagamba, E., and Y. Weingarten, "An Overview of Operations, Administration, and Maintenance (OAM) Tools", RFC 7276, DOI 10.17487/RFC7276, June 2014, <http://www.rfc-editor.org/info/rfc7276>.

[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>.

[RFC7872] Gont, F., Linkova, J., Chown, T., and W. Liu, "Observations on the Dropping of Packets with IPv6 Extension Headers in the Real World", RFC 7872, DOI 10.17487/RFC7872, June 2016, <http://www.rfc-editor.org/info/rfc7872>.

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

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

Email: sadara@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

Network Working Group
Internet-Draft
Intended status: Informational
Expires: September 14, 2017

F. Brockners
S. Bhandari
S. Dara
C. Pignataro
Cisco
H. Gredler
RtBrick Inc.
J. Leddy
Comcast
S. Youell
JMPC
D. Mozes
Mellanox Technologies Ltd.
T. Mizrahi
Marvell
P. Lapukhov
Facebook
R. Chang
Barefoot Networks
March 13, 2017

Requirements for In-situ OAM
draft-brockners-inband-oam-requirements-03

Abstract

This document discusses the motivation and requirements for including specific operational and telemetry information into data packets while the data packet traverses a path between two points in the network. This method is referred to as "in-situ" Operations, Administration, and Maintenance (OAM), given that the OAM information is carried with the data packets as opposed to in "out-of-band" packets dedicated to OAM. In situ OAM complements other OAM mechanisms which use dedicated probe packets to convey OAM information.

Status of This Memo

Table of Contents

1. Introduction 3
2. Conventions 4
3. Motivation for in-situ OAM 5
3.1. Path Congruency Issues with Dedicated OAM Packets 5
3.2. Results Sent to a System Other Than the Sender 6
3.3. Overlay and Underlay Correlation 6
3.4. SLA Verification 7
3.5. Analytics and Diagnostics 7
3.6. Frame Replication/Elimination Decision for Bi-casting /Active-active Networks 8
3.7. Proof of Transit 8
3.8. Use Cases 9
4. Considerations for In-situ OAM 11
4.1. Type of Information to be Recorded 11
4.2. MTU and Packet Size 12
4.3. Administrative Boundaries 13
4.3.1. Layered In-Situ OAM Domains 13
4.4. Selective Enablement 14
4.5. Forwarding Behavior 14
4.6. Optimization of Node and Interface Identifiers 14
4.7. Loop Communication Path (IPv6-specifics) 15
5. Requirements for In-situ OAM Data Types 15
5.1. Generic Requirements 15
5.2. In-situ OAM Data with Per-hop Scope 17

5.3. In-situ OAM with Selected Hop Scope 18
5.4. In-situ OAM with End-to-end Scope 18
6. Security Considerations and Requirements 19
6.1. General considerations 19
6.2. Proof of Transit 19
7. IANA Considerations 20
8. Acknowledgements 20
9. References 20
9.1. Normative References 20
9.2. Informative References 21
Authors' Addresses 22

1. Introduction

This document discusses requirements for "in-situ" Operations, Administration, and Maintenance (OAM) mechanisms. In this context, "in-situ OAM" refers to the concept of directly encoding telemetry information within the data packet as it traverses the network or telemetry domain. Mechanisms which add tracing or other types of telemetry information to the regular data traffic, sometimes also referred to as "in-band" OAM can complement active, probe-based mechanisms such as ping or traceroute, which are sometimes considered as "out-of-band", because the messages are transported independently from regular data traffic. In terms of "active" or "passive" OAM, "in-situ" OAM can be considered a hybrid OAM type. While no extra packets are sent, in-situ OAM adds information to the packets therefore cannot be considered passive. In terms of the classification given in [RFC7799] in-situ OAM could be portrayed as "hybrid OAM, type 1". "In-situ" mechanisms do not require extra packets to be sent and hence don't change the packet traffic mix within the network. Traceroute and ping for example use ICMP messages: New packets are injected to get tracing information. Those add to the number of messages in a network, which already might be highly loaded or suffering performance issues for a particular path or traffic type.

A number of in-situ as well as in-band OAM mechanisms have been discussed, such as the INT spec for the P4 programming language [P4] or the SPUD prototype [I-D.hildebrand-spud-prototype]. The SPUD prototype uses a similar logic that allows network devices on the path between endpoints to participate explicitly in the tube outside the end-to-end context. Even the IPv4 route-record option defined in [RFC0791] can be considered an in-situ OAM mechanism. Per what was already stated, in-situ OAM complements "out-of-band" mechanisms such as ping or traceroute, or more recent active probing mechanisms, as described in [I-D.lapukhov-dataplane-probe]. In-situ OAM mechanisms can be leveraged where current out-of-band mechanisms do not apply or do not offer the desired characteristics or requirements, such as proving that a certain set of traffic takes a pre-defined path, strict congruency between overlay and underlay transports is in place, checking service level agreements for the live data traffic, detailed statistics or verification of path selections within a domain, or scenarios where probe traffic is potentially handled differently from regular data traffic by the network devices. [RFC7276] presents an overview of OAM tools.

Compared to probably the most basic example of "in-situ OAM" which is IPv4 route recording [RFC0791], an in-situ OAM approach has the following capabilities:

a. A flexible data format to allow different types of information to be captured as part of an in-situ OAM operation, including but not limited to path tracing information, operational and telemetry information such as timestamps, sequence numbers, or even generic data such as queue size, geo-location of the node that forwarded the packet, etc.

b. A data format to express node as well as link identifiers to record the path a packet takes with a fixed amount of added data.

c. The ability to determine whether any nodes were skipped while recording in-situ OAM information (i.e., in-situ OAM is not supported or not enabled on those nodes).

d. The ability to actively process information in the packet, for example to prove in a cryptographically secure way that a packet really took a pre-defined path using some traffic steering method such as service chaining or traffic engineering.

e. The ability to include OAM data beyond simple path information, such as timestamps or even generic data of a particular use case.

f. The ability to carry in-situ OAM data in various different transport protocols.

2. Conventions

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

Abbreviations used in this document:

ECMP: Equal Cost Multi-Path

IOAM: In-situ Operations, Administration, and Maintenance

LISP: Locator/ID Separation Protocol

MTU: Maximum Transmit Unit

NSH: Network Service Header

NFV: Network Function Virtualization

OAM: Operations, Administration, and Maintenance

PMTU: Path MTU

SFC: Service Function Chain

SLA: Service Level Agreement

SR: Segment Routing

SID: Segment Identifier

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

This document defines in-situ Operations, Administration, and Maintenance (in-situ OAM), as the subset in which OAM information is carried along with data packets. This is as opposed to "out-of-band OAM", where specific packets are dedicated to carrying OAM information.

3. Motivation for in-situ OAM

In several scenarios it is beneficial to make information about the path a packet took through the network or through a network device as well as associated telemetry information available to the operator. This includes not only tasks like debugging, troubleshooting, as well as network planning and network optimization but also policy or service level agreement compliance checks. This section discusses the motivation to introduce new methods for enhanced in-situ network diagnostics.

3.1. Path Congruency Issues with Dedicated OAM Packets

Packet scheduling algorithms, especially for balancing traffic across equal cost paths or links, often leverage information contained within the packet, such as protocol number, IP-address or MAC-address. Probe packets would thus either need to be sent from the exact same endpoints with the exact same parameters, or probe packets would need to be artificially constructed as "fake" packets and inserted along the path. Both approaches are often not feasible from an operational perspective, be it that access to the end-system is not feasible, or that the diversity of parameters and associated probe packets to be created is simply too large. An in-situ mechanism is an alternative in those cases.

In-situ mechanisms are not impacted by differences in the handling of probe traffic compared to other data packets, where probe traffic is handled differently (and potentially forwarded differently) by a router than regular data traffic. This obviously assumes that the addition of in-situ information does not change the forwarding behavior of the packet. Note that in certain implementations, the addition information to a transport protocol changes the forwarding behavior. IPv6 extension header processing is one example. Some implementations process IPv6 packets with extension headers in the "slow" path of a router, as opposed to the "fast" path.

3.2. Results Sent to a System Other Than the Sender

Traditional ping and traceroute tools return the OAM results to the sender of the probe. Even when the ICMP messages that are used with these tools are enhanced, and additional telemetry is collected (e.g., ICMP Multi-Part [RFC4884] supporting MPLS information [RFC4950], Interface and Next-Hop Identification [RFC5837], etc.), it would be advantageous to separate the sending of an OAM probe from the receiving of the telemetry data. In this context, it is helpful to eliminate the requirement that there be a working bidirectional path.

3.3. Overlay and Underlay Correlation

Several network deployments leverage tunneling mechanisms to create overlay or service-layer networks. Examples include VXLAN-GPE, GRE, or LISP. One often observed attribute of overlay networks is that they do not offer the user of the overlay any insight into the underlay network. This means that the path that a particular tunneled packet takes, nor other operational details such as the per-hop delay/jitter in the underlay are visible to the user of the overlay network, giving rise to diagnosis and debugging challenges in case of connectivity or performance issues. The scope of OAM tools like ping or traceroute is limited to either the overlay or the underlay which means that the user of the overlay has typically no access to OAM in the underlay, unless specific operational procedures are put in place. With in-situ OAM the operator of the underlay can offer details of the connectivity in the underlay to the user of the overlay. This could include the ability to find out which underlay elements are shared by overlays and ability to know which overlays are mapped to the same underlay elements. Deployment dependent underlay transit nodes can be configured to update OAM information in the overlay transport encapsulation. The operator of the egress tunnel router could choose to share the recorded information about the path with the user of the overlay.

Coupled with mechanisms such as Segment Routing (SR) [I-D.ietf-spring-segment-routing], overlay network and underlay network can be more tightly coupled: The user of the overlay has detailed diagnostic information available in case of failure conditions. The user of the overlay can also use the path recording information as input to traffic steering or traffic engineering mechanisms, to for example achieve path symmetry for the traffic between two endpoints. [I-D.brockners-lisp-sr] is an example for how these methods can be applied to LISP.

3.4. SLA Verification

In-situ OAM can help users of an overlay-service to verify that negotiated SLAs for the real traffic are met by the underlay network provider. Different from solutions which rely on active probes to test an SLA, in-situ OAM based mechanisms avoid wrong interpretations and "cheating", which can happen if the probe traffic that is used to perform SLA-check is prioritized by the network provider of the underlay. In active/standby deployments in-situ OAM would only allow for SLA verification of the active path.

3.5. Analytics and Diagnostics

Network planners and operators benefit from knowledge of the actual traffic distribution in the network. When deriving an overall network connectivity traffic matrix one typically needs to correlate data gathered from each individual device in the network. If the path of a packet is recorded while the packet is forwarded, the entire path that a packet took through the network is available to the egress system. This obviates the need to retrieve individual traffic statistics from every device in the network and correlate those statistics, or employ other mechanisms such as leveraging traffic engineering with null-bandwidth tunnels just to retrieve the appropriate statistics to generate the traffic matrix.

In addition, with individual path tracing, information is available at packet level granularity, rather than only at aggregate level - as is usually the case with IPFIX-style methods which employ flow-filters at the network elements. Data-center networks which use equal-cost multipath (ECMP) forwarding are one example where detailed statistics on flow distribution in the network are highly desired. If a network supports ECMP, one can create detailed statistics for the different paths packets take through the network at the egress system, without a need to correlate/aggregate statistics from every router in the system. Transit devices are off-loaded from the task of gathering packet statistics.

In high-speed networks one can leverage and benefit from packet-accurate measurements with for example hardware-accurate timestamping (i.e., nanosecond-level verification) to support optimized packet scheduling and queuing mechanisms.

3.6. Frame Replication/Elimination Decision for Bi-casting/Active-active Networks

Bandwidth- and power-constrained, time-sensitive, or loss-intolerant networks (e.g., networks for industry automation/control, health care) require efficient OAM methods to decide when to replicate packets to a secondary path in order to keep the loss/error-rate for the receiver at a tolerable level - and also when to stop replication and eliminate the redundant flow. Many Internet of Things (IoT) networks are time sensitive and cannot leverage automatic retransmission requests (ARQ) to cope with transmission errors or lost packets. Transmitting the data over multiple disparate paths (often called bi-casting or live-live) is a method used to reduce the error rate observed by the receiver. Time sensitive networks (TSN) receive a lot of attention from the manufacturing industry as shown by a various standardization activities and industry forums being formed (see e.g., IETF 6TiSCH, IEEE P802.1CB, AVnu).

3.7. Proof of Transit

Several deployments use traffic engineering, policy routing, segment routing or Service Function Chaining (SFC) [RFC7665] to steer packets through a specific set of nodes. In certain cases regulatory obligations or a compliance policy require to prove that all packets that are supposed to follow a specific path are indeed being forwarded across the exact set of nodes specified. If a packet flow is supposed to go through a series of service functions or network nodes, it has to be proven that all packets of the flow actually went through the service chain or collection of nodes specified by the policy. In case the packets of a flow weren’t appropriately processed, a verification device would be required to identify the policy violation and take corresponding actions (e.g., drop or redirect the packet, send an alert etc.) corresponding to the policy. In today’s deployments, the proof that a packet traversed a particular service chain is typically delivered in an indirect way: Service appliances and network forwarding are in different trust domains. Physical hand-off-points are defined between these trust domains (i.e., physical interfaces). Or in other terms, in the "network forwarding domain" things are wired up in a way that traffic is delivered to the ingress interface of a service appliance and received back from an egress interface of a service appliance. This "wiring" is verified and trusted. The evolution to Network Function Virtualization (NFV) and modern service chaining concepts (using technologies such as Locator/ID Separation Protocol (LISP), Network Service Header (NSH), Segment Routing (SR), etc.) blurs the line between the different trust domains, because the hand-off-points are no longer clearly defined physical interfaces, but are virtual interfaces. Because of that very reason, networks operators require that different trust layers not to be mixed in the same device. For an NFV scenario a different proof is required. Offering a proof that a packet traversed a specific set of service functions would allow network operators to move away from the above described indirect methods of proving that a service chain is in place for a particular application.

Deployed service chains without the presence of a "proof of transit" mechanism are typically operated as fail-open system: The packets that arrive at the end of a service chain are processed. Adding "proof of transit" capabilities to a service chain allows an operator to turn a fail-open system into a fail-close system, i.e. packets that did not properly traverse the service chain can be blocked.

A solution approach could be based on OAM data which is added to every packet for achieving Proof Of Transit (POT).The OAM data is updated at every hop and is used to verify whether a packet traversed all required nodes. When the verifier receives each packet, it can validate whether the packet traversed the service chain correctly. The detailed mechanisms used for path verification along with the procedures applied to the OAM data carried in the packet for path verification are beyond the scope of this document. Details are addressed in [I-D.brockners-proof-of-transit]. In this document the term "proof" refers to a discrete set of bits that represents an integer or string carried as OAM data. The OAM data is used to verify whether a packet traversed the nodes it is supposed to traverse.

3.8. Use Cases

In-situ OAM could be leveraged for several use cases, including:

- Traffic Matrix: Derive the network traffic matrix: Traffic for a given time interval between any two edge nodes of a given domain. Could be performed for all traffic or on a per Quality of Service (QoS) class.

- Flow Debugging: Discover which path(s) a particular set of traffic (identified by an n-tuple) takes in the network. Such a procedure is particularly useful in case traffic is balanced across multiple paths, like with link aggregation (LACP) or equal cost multi-pathing (ECMP).

- o Loss Statistics per Path: Retrieve loss statistics per flow and path in the network.

- o Path Heat Maps: Discover highly utilized links in the network.

- o Trend Analysis on Traffic Patterns: Analyze if (and if so how) the forwarding path for a specific set of traffic changes over time (can give hints to routing issues, unstable links etc.)

- o Network Delay Distribution: Show delay distribution across network by node or links. If enabled per application or for a specific flow then display the path taken along with the delay incurred at every hop.

- o SLA Verification: Verify that a negotiated service level agreement (SLA), e.g., for packet drop rates or delay/jitter is conformed to by the actual traffic.

- o Low-power Networks: Include application level OAM information (e.g., battery charge level, cache or buffer fill level) into data traffic to avoid sending extra OAM traffic which incur an extra cost on the devices. Using the battery charge level as example, one could avoid sending extra OAM packets just to communicate battery health, and as such would save battery on sensors.

- o Path Verification or Service Function Path Verification: Proof and verification of packets traversing check points in the network, where check points can be nodes in the network or service functions.

- o Geo-location Policy: Network policy implemented based on which path packets took. Example: Only if packets originated and stayed within the trading-floor department, access to specific applications or servers is granted.

- o Device-level Troubleshooting and Optimization: In many cases, network operators could benefit from information specific to a single device. A non-exhaustive list of useful information includes: queue-depths, buffer utilization (either shared or per-port), packet latency measured from a known starting point, packet latency introduced by a single device, and resource utilization (CPU, memory, link bandwidth) of a given device or link. In some cases, this information changes over per-packet timescales (i.e., nanoseconds) and as such it is extremely challenging to collect and report this info in an accurate and scalable manner. By encoding the information from the forwarding element directly within a data packet (i.e., within the 'fast-path') this information can be added to some or all data packets and then collected and analyzed by human or machine tools. This type of information is particularly valuable for troubleshooting low-level device errors as well as providing a knowledge feedback loop for network and device optimization.

- o Custom Network Probing: Active network probing and in-situ OAM can be combined for customized and efficient network probing. This could for example be a customized traceroute.

4. Considerations for In-situ OAM

The implementation of an in-situ OAM mechanism needs to take several considerations into account, including administrative boundaries, how information is recorded, Maximum Transfer Unit (MTU), Path MTU Discovery (PMTUD) and packet size, etc.

4.1. Type of Information to be Recorded

The information gathered for in-situ OAM can be categorized into three main categories: Information with a per-hop scope, such as path tracing; information which applies to a specific set of hops, such as path or service chain verification; information which only applies to the edges of a domain, such as sequence numbers. Note that a single network device could comprise several in-situ OAM hops, for example in case one wants to trace the path of a packet through that device.

- o "edge to edge": Information that needs to be shared between network edges (the "edge" of a network could either be a host or a domain edge device): Edge to edge data e.g., packet and octet count of data entering a well-defined domain and leaving it is helpful in building traffic matrix, sequence number (also called "path packet counters") is useful for the flow to detect packet loss.

- o "selected hops": Information that applies to a specific set of nodes only. In case of path verification, only the nodes which are "check points" are required to interpret and update the information in the packet.

- o "per hop": Information that is gathered at every hop along the path a packet traverses within an administrative domain:

  * Hop by Hop information e.g., Nodes visited for path tracing, Timestamps at each hop to find delays along the path

* Stats collection at each hop to optimize communication in resource constrained networks e.g., battery, CPU, memory status of each node piggy backed in a data packet is useful in low power lossy networks where network nodes are mostly asleep and communication is expensive

4.2. MTU and Packet Size

The recorded data at every hop might lead to packet size exceeding the Maximum Transmit Unit (MTU). A detailed discussion of the implications of oversized IPv6 header chains is found in [RFC7112]. The Path MTU restricts the amount of data that can be recorded for purpose of OAM within a data packet.

If in-situ OAM data is inserted at the edge of the domain (e.g., by intermediate routers) then the MTU on all interfaces with the domain (MTU_INT) MUST be >= the maximum MTU on any "external" facing interfaces (MTU_EXT) and the total size of in-situ OAM data to be recorded MUST be <= (MTU_INT - MTU_EXT).

In-situ OAM comprises two approaches to insert OAM data fields in the packets:

- o Pre-allocated: In this case, the encapsulating node inserts empty data fields into the packet to cover the entire domain. The data fields will be incrementally updated/filled as the packet progresses through the network. With pre-allocation the packet size is only changed at the encapsulating node and is kept constant throughout the domain. The pre-allocated approach is beneficial for software data-plane implementations where allocating the required space only once and index into the array to populate the data during transit avoids copy operations at every hop.

- o Incremental: Every node that desires to include in-situ OAM information extends the packet as needed. The incremental approach is beneficial for hardware data-plane implementations as it eliminates the need for the transit nodes to read the full array and lookup the pointer in the option prior to updating the data fields contents.

The "incremental" or the "pre-allocated" approaches could even be combined in the same deployment - in which case two in-situ OAM headers would be present in the packet: One for the incremental approach and one for the pre-allocated approach. In such a case one would expect that nodes with a hardware data-plane would update the incremental header, whereas nodes with a software data-plane would process the pre-allocated header.

4.3. Administrative Boundaries

There are several challenges in enabling in-situ OAM in the public Internet as well as in corporate/enterprise networks across administrative domains, which include but are not limited to:

- o Deployment dependent, the data fields that in-situ OAM requires as part of a specific transport protocol may not be supported across administrative boundaries.

- o Current OAM implementations are often done in the slow path, i.e., OAM packets are punted to router’s CPU for processing. This leads to performance and scaling issues and opens up routers for attacks such as Denial of Service (DoS) attacks.

- o Discovery of network topology and details of the network devices across administrative boundaries may open up attack vectors compromising network security.

- o Specifically on IPv6: At the administrative boundaries IPv6 packets with extension headers are dropped for several reasons described in [RFC7872].

The following considerations will be discussed in a future version of this document: If the packet is dropped due to the presence of the in-situ OAM; If the policy failure is treated as feature disablement and any further recording is stopped but the packet itself is not dropped, it may lead to every node in the path to make this policy decision.

4.3.1. Layered In-Situ OAM Domains

Like any OAM domain, in-situ OAM domains could also be layered/ nested. Layering/nesting of in-situ OAM follows the general approach of OAM layering: An in-situ OAM domain consists of maintenance end-points (MEP) and maintenance intermediate points (MIP). MEP add to or remove the entire set of in-situ OAM data fields from the traffic, while only MIP update or add in-situ OAM data fields. When in-situ OAM layering is employed, a MEP of one layer becomes a MIP in the layer above, while MIP of the lower layer are not visible to the layer above - unless specifically configured otherwise.

Consider the following examples:

- o NSH over IPv6: In-situ OAM data fields could be present in both transport protocols: NSH and IPv6, with NSH forming the overlay network and IPv6 forming the underlay network. The network which deploys NSH would form an in-situ OAM domain. In addition each

IPv6 underlay network which connects two NSH nodes forms an in-situ OAM domain. The in-situ OAM domain with NSH as transport could be considered as layered on top of the different in-situ OAM domains which use IPv6 as transport.

- o NSH using an in-situ OAM aware transport: Consider a case where the underlay network would not natively support in-situ OAM, still the individual transport nodes would have the capability to "look deep into the packet" and update/add in-situ OAM information in the NSH header. The in-situ OAM domain with NSH as transport could be considered as layered on top of the different in-situ OAM domains which are in-situ OAM aware and connect the individual NSH nodes.

4.4. Selective Enablement

The ability to selectively enable in-situ OAM is valuable. While it may be desirable to enable data collection on all traffic or devices, this may not always be feasible. In-situ OAM collection may also come with a performance impact to forwarding rates or feature capabilities, which may be acceptable in only some locations. For example, the SPUD prototype uses the notion of "pipes" to describe the portion of the traffic that could be subject to in-path inspection. Mechanisms to decide which traffic would be subject to in-situ OAM are outside the scope of this document.

4.5. Forwarding Behavior

In-situ OAM adds additional data fields to live user traffic and as such changes the packet which is also why in-situ OAM is characterized as "hybrid, type 1" OAM. The effectiveness of in-situ OAM as a tool for operations depends on forwarding nodes not altering their forwarding behavior in case of in-situ OAM data fields being present in the packet. As a consequence, an implementation of in-situ OAM should not change the forwarding behavior of the packet, i.e. packets with or without in-situ OAM data fields should be handled the same way by a forwarding node (see also the associated requirement further below). Note that there are implementations where the addition of meta-data to live user traffic might cause the forwarding behavior of the packet to change, e.g. certain implementation handle IPv6 packets with or without extension headers differently (see [RFC7872]).

4.6. Optimization of Node and Interface Identifiers

Since packets have a finite maximum size, the data recording or carrying capacity of one packet in which the in-situ OAM metadata is present is limited. In-situ OAM should use its own dedicated namespace (confined to the domain in-situ OAM operates in) to represent node and interface IDs to save space in the header. Generic representations of node and interface identifiers which are globally unique (such as a UUID) would consume significantly more bits of in-situ OAM data.

4.7. Loop Communication Path (IPv6-specifics)

When recorded data is required to be analyzed on a source node that issues a packet and inserts in-situ OAM data, the recorded data needs to be carried back to the source node.

One way to carry the in-situ OAM data back to the source is to utilize an ICMP Echo Request/Reply (ping) or ICMPv6 Echo Request/Reply (ping6) mechanism. In order to run the in-situ OAM mechanism appropriately on the ping/ping6 mechanism, the following two operations should be implemented by the ping/ping6 target node:

1. All of the in-situ OAM fields would be copied from an Echo Request message to an Echo Reply message.

2. The Hop Limit field of the IPv6 header of these messages would be copied as a continuous sequence. Further considerations are addressed in a future version of this document.

5. Requirements for In-situ OAM Data Types

The above discussed use cases require different types of in-situ OAM data. This section details requirements for in-situ OAM derived from the discussion above.

5.1. Generic Requirements

REQ-G1: Classification: It should be possible to enable in-situ OAM on a selected set of traffic (e.g., per interface, based on an access control list specifying a specific set of traffic, etc.) The selected set of traffic can also be all traffic.

REQ-G2: Scope: If in-situ OAM is used only within a specific domain, provisions need to be put in place to ensure that in-situ OAM data stays within the specific domain only.

REQ-G3: Transport independence: Data formats for in-situ OAM shall be defined in a transport independent way. In-situ OAM applies to a variety of transport protocols. Encapsulations should be defined how the generic data formats are carried by a specific protocol.

REQ-G4: Layering: It should be possible to have in-situ OAM information for different transport protocol layers be present in several fields within a single packet. This could for example be the case when tunnels are employed and in-situ OAM information is to be gathered for both the underlay as well as the overlay network. Layering support should not be limited to just underlay and overlay, but include more than two layers.

REQ-G5: MTU size: With in-situ OAM information added, packets MUST NOT become larger than the path MTU.

REQ-G5.1: If due to some reason a packet which contains in situ OAM data fields cannot be forwarded due to the presence of in-situ OAM data fields, the node SHOULD remove the in situ OAM data fields and forward the packet, rather than drop the entire packet.

REQ-G5.2: If the encapsulating router is unable to insert in-situ OAM data fields into a packet, e.g., due to MTU issues, even though it is configured to do so, it should use some operational means to inform the operator (e.g., syslog) about the inability to add in-situ OAM data fields. Even if the in-situ OAM encapsulating node fails to add in-situ OAM data fields, it should forward the packet normally.

REQ-G5.3: MTU size consideration for in-situ OAM MUST take domain specifics into account, e.g., changes of the domain topology due to path protection mechanisms might extend the hop count of a path etc.

REQ-G6: Data structure reuse: The data fields and associated types defined and used for in-situ OAM ought to be reusable for out-of-band OAM telemetry as well.

REQ-G7: Data fields: It is desirable that the format of in-situ OAM data fields leverages already defined data formats for OAM as much as feasible.

REQ-G8: Combination with active OAM mechanisms: In-situ OAM should be usable for active network probing, like for example a customized version of traceroute. Decapsulating in-situ OAM nodes may have an ability to send the in-situ OAM information retrieved from the packet back to the source address of the packet or to the encapsulating node.

REQ-G9: Unaltered forwarding behavior of in-situ OAM nodes: The addition of in-situ OAM data fields should not change the way packets are forwarded within the in-situ OAM domain.

REQ-G10: Layering of in-situ OAM domains: It should be possible to layer in-situ OAM domains on each other. Layering should be supported within the same, as well as with different transport protocols which carry in-situ OAM data fields.

5.2. In-situ OAM Data with Per-hop Scope

REQ-H1: Missing nodes detection: Data shall be present that allows a node to detect whether all nodes that might participate in in-situ OAM operations have indeed participated.

REQ-H2: Node, instance or device identifier: Data shall be present that allows to retrieve the identity of the entity reporting telemetry information. The entity can be a device, or a subsystem/component within a device. The latter will allow for packet tracing within a device in much the same way as between devices.

REQ-H3: Ingress interface identifier: Data shall be present that allows the identification of the interface a particular packet was received from. The interface can be a logical and/or physical entity.

REQ-H4: Egress interface identifier: Data shall be present that allows the identification of the interface a particular packet was forwarded to. Interface can be a logical or physical entity.

REQ-H5: Time-related requirements

REQ-H5.1: Delay: Data shall be present that allows to retrieve the delay between two or more points of interest within the system. Those points can be within the same device or on different devices.

REQ-H5.2: Jitter: Data shall be present that allows to retrieve the jitter between two or more points of interest within the system. Those points can be within the same device or on different devices. Jitter can be derived from the different timestamps gathered and does not necessarily need to be an explicit data field.

REQ-H5.3: Wall-clock time: Data shall be present that allows to retrieve the wall-clock time visited a particular point of interest in the system.

REQ-H5.4: Time precision: Time with different precision should be supported. Use-case dependent, the required precision could e.g., be nanoseconds, microseconds, milliseconds, or seconds.

REQ-H6: Generic data fields (like e.g., GPS/Geo-location information): It should be possible to add user-defined OAM data at select hops to the packet. The semantics of the data are defined by the user.

5.3. In-situ OAM with Selected Hop Scope

REQ-S1: Proof of transit: Data shall be present which allows to securely prove that a packet has visited or ore several particular points of interest (i.e., a particular set of nodes).

REQ-S1.1: In case "Shamir’s secret sharing scheme" is used for proof of transit, two data fields, "random" and "cumulative" shall be present. The number of bits used for "random" and "cumulative" data fields can vary between deployments and should thus be configurable.

REQ-S1.2: Enable a fail-open service chaining system to be converted into a fail-closed service chaining system.

5.4. In-situ OAM with End-to-end Scope

REQ-E1: Sequence numbering:

REQ-E1.1: Reordering detection: It should be possible to detect whether packets have been reordered while traversing an in situ OAM domain.

REQ-E1.2: Duplicates detection: It should be possible to detect whether packets have been duplicated while traversing an in situ OAM domain.

REQ-E1.3: Detection of packet drops: It should be possible to detect whether packets have been dropped while traversing an in-situ OAM domain.

6. Security Considerations and Requirements

6.1. General considerations

General Security considerations will be expanded on in a later version of this document.

In-situ OAM is considered a "per domain" feature, where one or several operators decide on leveraging and configuring in-situ OAM according to their needs. Still operators need to properly secure the in-situ OAM domain to avoid malicious configuration and use, which could include injecting malicious in-situ OAM packets into a domain.

6.2. Proof of Transit

Threat Model: Attacks on the deployments could be due to malicious administrators or accidental misconfiguration resulting in bypassing of certain nodes. The solution approach should meet the following requirements:

REQ-SEC1: Sound Proof of Transit: A valid and verifiable proof that the packet definitively traversed through all the nodes as expected. Probabilistic methods to achieve this should be avoided, as the same could be exploited by an attacker.

REQ-SEC2: Tampering of meta data: An active attacker should not be able to insert or modify or delete meta data in whole or in parts and bypass few (or all) nodes. Any deviation from the expected path should be accurately determined.

REQ-SEC3: Replay Attacks: A attacker (active/passive) should not be able to reuse the POT bits in the packet by observing the OAM data in the packet, packet characteristics (like IP addresses, octets transferred, timestamps) or even the proof bits themselves. The solution approach should consider usage of these parameters for deriving any secrets cautiously. Mitigating replay attacks beyond a window of longer duration could be intractable to achieve with fixed number of bits allocated for proof.

REQ-SEC4: Pre-play Attacks: A active attacker should not be able to generate or reuse valid POT bits from legitimate packets, in order to prove to the verifier as valid packets. This slight variant of replay attacks. The attacker extracts POT bits from legitimate packets and ensure they do not reach the verifier. Subsequently reuse those POT bits in crafted packets.

REQ-SEC5: Recycle Secrets: Any configuration of the secrets (like cryptographic keys, initialization vectors etc.) either in the controller or service functions should be re-configurable. Solution approach should enable controls, API calls etc. needed in order to perform such recycling. It is desirable to provide recommendations on the duration of rotation cycles needed for the secure functioning of the overall system.

REQ-SEC6: Secret storage and distribution: Secrets should be shared with the devices over secure channels. Methods should be put in place so that secrets cannot be retrieved by non-authorized personnel from the devices.

7. IANA Considerations

[RFC Editor: please remove this section prior to publication.]

This document has no IANA actions.

8. Acknowledgements

The authors would like to thank Jen Linkova, LJ Wobker, Eric Vyncke, Nalini Elkins, Srihari Raghavan, Ranganathan T S, Karthik Babu Harichandra Babu, Akshaya Nadahalli, Ignas Bagdonas, LJ Wobker, Erik Nordmark, Vengada Prasad Govindan, and Andrew Yourtchenko for the comments and advice. This document leverages and builds on top of several concepts described in [I-D.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

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

9.2. Informative References

[I-D.brockners-lisp-sr]
Brockners, F., Bhandari, S., Maino, F., and D. Lewis, "LISP Extensions for Segment Routing", draft-brockners-lisp-sr-01 (work in progress), February 2014.

[I-D.brockners-proof-of-transit]
Brockners, F., Bhandari, S., Dara, S., Pignataro, C., Leddy, J., Youell, S., Mozes, D., and T. Mizrahi, "Proof of Transit", draft-brockners-proof-of-transit-02 (work in progress), October 2016.

[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.ietf-spring-segment-routing]
Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", draft-ietf-spring-segment-routing-10 (work in progress), November 2016.

[I-D.kitamura-ipv6-record-route]
Kitamura, H., "Record Route for IPv6 (PR6) Hop-by-Hop Option Extension", draft-kitamura-ipv6-record-route-00 (work in progress), November 2000.

[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.

[RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, DOI 10.17487/RFC0791, September 1981, <http://www.rfc-editor.org/info/rfc791>.

[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, <http://www.rfc-editor.org/info/rfc4884>.

[RFC4950] Bonica, R., Gan, D., Tappan, D., and C. Pignataro, "ICMP Extensions for Multiprotocol Label Switching", RFC 4950, DOI 10.17487/RFC4950, August 2007, <http://www.rfc-editor.org/info/rfc4950>.

[RFC5837] Atlas, A., Ed., Bonica, R., Ed., Pignataro, C., Ed., Shen, N., and JR. Rivers, "Extending ICMP for Interface and Next-Hop Identification", RFC 5837, DOI 10.17487/RFC5837, April 2010, <http://www.rfc-editor.org/info/rfc5837>.

[RFC7112] Gont, F., Manral, V., and R. Bonica, "Implications of Oversized IPv6 Header Chains", RFC 7112, DOI 10.17487/RFC7112, January 2014, <http://www.rfc-editor.org/info/rfc7112>.

[RFC7276] Mizrahi, T., Sprecher, N., Bellagamba, E., and Y. Weingarten, "An Overview of Operations, Administration, and Maintenance (OAM) Tools", RFC 7276, DOI 10.17487/RFC7276, June 2014, <http://www.rfc-editor.org/info/rfc7276>.

[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>.

[RFC7799] Morton, A., "Active and Passive Metrics and Methods (with Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799, May 2016, <http://www.rfc-editor.org/info/rfc7799>.

[RFC7872] Gont, F., Linkova, J., Chown, T., and W. Liu, "Observations on the Dropping of Packets with IPv6 Extension Headers in the Real World", RFC 7872, DOI 10.17487/RFC7872, June 2016, <http://www.rfc-editor.org/info/rfc7872>.

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

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

Email: sadara@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

John Leddy
Comcast

Email: John_Leddy@cable.comcast.com

Stephen Youell
JP Morgan Chase
25 Bank Street
London E14 5JP
United Kingdom

Email: stephen.youell@jpmorgan.com

David Mozes
Mellanox Technologies Ltd.

Email: davidm@mellanox.com

Tal Mizrahi
Marvell
6 Hamada St.
Yokneam 20692
Israel

Email: talmi@marvell.com

Petr Lapukhov
Facebook
1 Hacker Way
Menlo Park, CA 94025
USA

URI: petr@fb.com

Remy Chang
Barefoot Networks

Email: remy@barefootnetworks.com

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

Encapsulations for In-band OAM Data
draft-brockners-inband-oam-transport-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. In-band OAM is to complement current out-of-band OAM mechanisms based on ICMP or other types of probe packets. This document outlines how in-band OAM data records can be transported in protocols such as NSH, Segment Routing, VXLAN-GPE, native IPv6 (via extension header), and IPv4. Transport options are currently investigated as part of an implementation study. This document is intended to only serve informational purposes.

Table of Contents

1. Introduction 2
2. Conventions 3
3. In-Band OAM Metadata Transport in IPv6 4
  3.1. In-band OAM in IPv6 Hop by Hop Extension Header 4
    3.1.1. In-band OAM Hop by Hop Options 4
    3.1.2. Procedure at the Ingress Edge to Insert the In-band OAM Header 6
    3.1.3. Procedure at Intermediate Nodes 7
    3.1.4. Procedure at the Egress Edge to Remove the In-band OAM Header 7
4. In-band OAM Metadata Transport in VXLAN-GPE 7
5. In-band OAM Metadata Transport in NSH 9
6. In-band OAM Metadata Transport in Segment Routing 11
  6.1. In-band OAM in SR with IPv6 Transport 11
  6.2. In-band OAM in SR with MPLS Transport 11
7. IANA Considerations 12
8. Manageability Considerations 12
9. Security Considerations 12
10. Acknowledgements 12
11. References 12
  11.1. Normative References 12
  11.2. Informative References 12
Authors' Addresses 14

1. Introduction

This document discusses transport mechanisms for "in-band" operation, administration, and maintenance (OAM) data records. 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]. Data types and data formats for in-band OAM are defined in [draft-brockners-inband-oam-data].

This document outlines transport encapsulations for the in-band OAM data defined in [draft-brockners-inband-oam-data]. This document is to serve informational purposes only. As part of an in-band OAM implementation study different protocol encapsulations for in-band OAM data are being explored. Once data formats and encapsulation approaches are settled, protocol specific specifications for in-band OAM data transport will address the standardization aspect.

The data for in-band OAM defined in [draft-brockners-inband-oam-data] can be carried in a variety of protocols based on the deployment needs. This document discusses transport of in-band OAM data for the following protocols:

- o IPv6
- o VXLAN-GPE
- o NSH
- o Segment Routing (IPv6 and MPLS)

This list is non-exhaustive, as it is possible to carry the in-band OAM data in several other protocols and transports.

A feasibility study of in-band OAM is currently underway as part of the FD.io project [FD.io]. The in-band OAM implementation study should be considered as a "tool box" to showcase how "in-band" OAM can complement probe-packet based OAM mechanisms for different deployments and packet transport formats. For details, see the open source code in the FD.io [FD.io].

2. Conventions

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

Abbreviations used in this document:

MTU: Maximum Transmit Unit

OAM: Operations, Administration, and Maintenance

SR: Segment Routing

SID: Segment Identifier

NSH: Network Service Header

POT: Proof of Transit

SFC: Service Function Chain

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

3. In-Band OAM Metadata Transport in IPv6

This mechanisms of in-band OAM in IPv6 complement others proposed to enhance diagnostics of IPv6 networks, such as the IPv6 Performance and Diagnostic Metrics Destination Option described in [I-D.ietf-ippm-6man-pdm-option]. The IP Performance and Diagnostic Metrics Destination Option is destination focused and specific to IPv6, whereas in-band OAM is performed between end-points of the network or a network domain where it is enabled and used.

A historical note: The idea of IPv6 route recording was originally introduced by [draft-kitamura-ipv6-record-route] back in year 2000. With IPv6 now being generally deployed and new concepts such as Segment Routing [I-D.ietf-spring-segment-routing] being introduced, it is imperative to further mature the operations, administration, and maintenance mechanisms available to IPv6 networks.

The in-band OAM options translate into options for an IPv6 extension header. The extension header would be inserted by either a host source of the packet, or by a transit/domain-edge node.

3.1. In-band OAM in IPv6 Hop by Hop Extension Header

This section defines in-band OAM for IPv6 transport. In-band OAM data is transported as an IPv6 hop-by-hop extension header.

3.1.1. In-band OAM Hop by Hop Options

Brief recap of the IPv6 hop-by-hop header as well as the options used for carrying in-band OAM data:

```
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |  Next Header  |  Hdr Ext Len  |                               |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               +
 |                                                               |
 .                                                               .
 .                            Options                            .
 .                                                               .
 |                                                               |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+- - - - - - - - -
   |  Option Type  |  Opt Data Len |  Option Data
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+- - - - - - - - -
```

With 2 highest order bits of Option Type indicating the following:

00 - skip over this option and continue processing the header.

01 - discard the packet.

10 - discard the packet and, regardless of whether or not the packet's Destination Address was a multicast address, send an ICMP Parameter Problem, Code 2, message to the packet's Source Address, pointing to the unrecognized Option Type.

11 - discard the packet and, only if the packet's Destination Address was not a multicast address, send an ICMP Parameter Problem, Code 2, message to the packet's Source Address, pointing to the unrecognized Option Type.

3rd highest bit:

0 - Option Data does not change en-route

1 - Option Data may change en-route

In-band OAM data records are inserted as options in an IPv6 hop-by-hop extension header:

1. Tracing Option: The in-band OAM Tracing option defined in [draft-brockners-inband-oam-data] is represented as a IPv6 option in hop by hop extension header by allocating following type:

   Option Type: 001xxxxxx 8-bit identifier of the type of option. xxxxxx=TBD_IANA_TRACE_OPTION_IPV6.

2. Proof of Transit Option: The in-band OAM POT option defined in [draft-brockners-inband-oam-data] is represented as a IPv6 option in hop by hop extension header by allocating following type:

   Option Type: 001xxxxxx 8-bit identifier of the type of option. xxxxxx=TBD_IANA_POT_OPTION_IPV6.

3. Edge to Edge Option: The in-band OAM E2E option defined in [draft-brockners-inband-oam-data] is represented as a IPv6 option in hop by hop extension header by allocating following type:

   Option Type: 000xxxxxx 8-bit identifier of the type of option. xxxxxx=TBD_IANA_E2E_OPTION_IPV6.

3.1.2. Procedure at the Ingress Edge to Insert the In-band OAM Header

In an administrative domain where in-band OAM is used, insertion of the in-band OAM header is enabled at the required edge nodes by means of configuration.

Such a config SHOULD allow selective enablement of in-band OAM header insertion for a subset of traffic (e.g., one or several "pipes").

Further the ingress edge node should be aware of maximum size of the header that can be inserted. Details on how the maximum size/size of the in-band OAM domain are retrieved are outside the scope of this document.

```
Let n = max number of nodes to be allocated;
(Based on PMTU advertised in the domain)

Let k = number of node data that can be allocated by this node
Let node_data_size = size of each node_data based on in-band OAM type

if (packet matches traffic for which in-band OAM is enabled) {
    Create in-band OAM hbyh ext header with k node data preallocated
    Increment payload length in IPv6 header :
                      with size of in-band OAM hbyh ext header
    Populate node data at :
        (size of in-band OAM hbyh header = 8) + k * node_data_size
    from the beginning of the header
    Set segments left to : k - 1

 }
```

3.1.3. Procedure at Intermediate Nodes

If a network node receives a packet with an in-band OAM header and it is enabled to process in-band OAM data it performs the following:

```
k = number of node data that this node can allocate
if (in-band OAM ext hbyh header is present) {
    if (Segments Left > 0)) {
      populate node data at :
         node_data_start[Segments Left]
      Segments Left = Segments Left - 1
    }
}
```

3.1.4. Procedure at the Egress Edge to Remove the In-band OAM Header

```
egress_edge = list of interfaces where in-band OAM hbyh ext
               header is to be stripped
Before forwarding packet out of interfaces in egress_edge list:
if (in-band OAM hbyh ext header is present) {
   remove the in-band OAM hbyh ext header,
   possibly store the record along with additional
   fields for analysis and export
   Decrement Payload Length in IPv6 header
   by size of in-band OAM ext header
}
```

4. In-band OAM Metadata Transport in VXLAN-GPE

VXLAN-GPE [I-D.ietf-nvo3-vxlan-gpe] encapsulation is somewhat similar to IPv6 extension headers in that a series of headers can be contained in the header as a linked list. The different in-band OAM types are added as options within a new in-band OAM protocol header in VXLAN GPE.

In-band OAM header in VXLAN GPE header:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Outer Ethernet Header                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Outer IP Header                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Outer UDP Header                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ +
|R|R|Ver|I|P|R|O|           Reserved            | NP = i.b.OAM  |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ GPE
|     Virtual Network Identifier (VNI)          | Reserved      |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ +
| Type =i.b.OAM | i.b.OAM HDR len |  Reserved     | NP = IP/Eth |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+iOAM
|                     in-band OAM options                       |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ +
|                                                               |
|                                                               |
|                     Payload + Padding (L2/L3/ESP/...)         |
|                                                               |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The VXLAN-GPE header and fields are defined in [I-D.ietf-nvo3-vxlan-gpe]. in-band OAM specific fields and header are defined here:

Type: 8-bit unsigned integer defining in-band OAM header type

in-band OAM HDR len: 8-bit unsigned integer. Length of the in-band OAM HDR in 8-octet units

in-band OAM options: Variable-length field, of length such that the complete in-band OAM header is an integer multiple of 8 octets long. Contains one or more TLV-encoded options of the format:

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+- - - - - - - - -
|  Option Type  |  Opt Data Len |  Option Data
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+- - - - - - - - -
```

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.

Option Data Variable-length field. Option-Type-specific data.

The in-band OAM options defined in [draft-brockners-inband-oam-data] are encoded with an option type allocated in the new in-band OAM IANA registry - in-band OAM_PROTOCOL_OPTION_REGISTRY_IANA_TBD. In addition the following padding options are defined to be used when necessary to align subsequent options and to pad out the containing header to a multiple of 8 octets in length.

Pad1 option (alignment requirement: none)

```
+-+-+-+-+-+-+-+-+
|       0       |
+-+-+-+-+-+-+-+-+
```

NOTE: The format of the Pad1 option is a special case -- it does not have length and value fields.

The Pad1 option is used to insert one octet of padding into the Options area of a header. If more than one octet of padding is required, the PadN option, described next, should be used, rather than multiple Pad1 options.

PadN option (alignment requirement: none)

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+- - - - - - - - -
|       1       |  Opt Data Len |  Option Data
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+- - - - - - - - -
```

The PadN option is used to insert two or more octets of padding into the Options area of a header. For N octets of padding, the Opt Data Len field contains the value N-2, and the Option Data consists of N-2 zero-valued octets.

5. In-band OAM Metadata Transport in NSH

In Service Function Chaining (SFC) [RFC7665], the Network Service Header (NSH) [I-D.ietf-sfc-nsh] already includes path tracing capabilities [I-D.penno-sfc-trace], but currently does not offer a solution to securely prove that packets really traversed the service chain. The "Proof of Transit" capabilities (see [draft-brockners-inband-oam-requirements] and [draft-brockners-proof-of-transit]) of in-band OAM can be leveraged within NSH. Proof of transit in-band OAM data is added as NSH Type 2 metadata:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      TLV Class=Cisco (0x0009) |C|    Type=POT |F|R|R| Len=4   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|                            Random                             |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  S
|                         Random(contd)                         |  C
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  V
|                          Cumulative                           |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
|                          Cumulative (contd)                   |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
```

TLV Class: Describes the scope of the "Type" field. In some cases, the TLV Class will identify a specific vendor, in others, the TLV Class will identify specific standards body allocated types. POT is currently defined using the Cisco (0x0009) TLV class.

Type: The specific type of information being carried, within the scope of a given TLV Class. Value allocation is the responsibility of the TLV Class owner. Currently a type value of 0x94 is used for proof of transit

Reserved bits: Two reserved bit are present for future use. The reserved bits MUST be set to 0x0.

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

Length: Length of the variable metadata, in 4-octet words. Here the length is 4.

Random: 64-bit Per packet Random number.

Cumulative: 64-bit Cumulative that is updated by the Service Functions.

6. In-band OAM Metadata Transport in Segment Routing

6.1. In-band OAM in SR with IPv6 Transport

Similar to NSH, a service chain or path defined using Segment Routing for IPv6 can be verified using the in-band OAM "Proof of Transit" approach. The Segment Routing Header (SRH) for IPv6 offers the ability to transport TLV structured data, similar to what NSH does (see [I-D.ietf-6man-segment-routing-header]). A new "POT TLV" is defined for the SRH which is to carry proof of transit in-band OAM data.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Type     |    Length     |   RESERVED    |F|   Flags     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|                          Random                               |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  P
|                          Random(contd)                        |  O
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  T
|                          Cumulative                           |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
|                          Cumulative (contd)                   |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
```

Type: To be assigned by IANA.

Length: 18.

RESERVED: 8 bits. SHOULD be unset on transmission and MUST be ignored on receipt.

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.

Flags: 8 bits. No flags are defined in this document.

Random: 64-bit per packet random number.

Cumulative: 64-bit cumulative value that is updated at specific nodes that form the service path to be verified.

6.2. In-band OAM in SR with MPLS Transport

In-band OAM "Proof of Transit" data can also be carried as part of the MPLS label stack. Details will be addressed in a future version of this document.

7. IANA Considerations

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

8. Manageability Considerations

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

9. 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].

10. 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. For the IPv6 encapsulation, 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.

11. References

11.1. Normative References

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

11.2. Informative References

[draft-brockners-inband-oam-data]
Brockners, F., Bhandari, S., Pignataro, C., and H. Gredler, "Data Formats 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.ietf-6man-segment-routing-header]
Previdi, S., Filsfils, C., Field, B., Leung, I., Linkova, J., Aries, E., Kosugi, T., Vyncke, E., and D. Lebrun, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-01 (work in progress), March 2016.

[I-D.ietf-ippm-6man-pdm-option]
Elkins, N., Hamilton, R., and m. mackermann@bcbsm.com, "IPv6 Performance and Diagnostic Metrics (PDM) Destination Option", draft-ietf-ippm-6man-pdm-option-03 (work in progress), June 2016.

[I-D.ietf-nvo3-vxlan-gpe]
Kreeger, L. and U. Elzur, "Generic Protocol Extension for VXLAN", draft-ietf-nvo3-vxlan-gpe-02 (work in progress), April 2016.

[I-D.ietf-sfc-nsh]
Quinn, P. and U. Elzur, "Network Service Header", draft-ietf-sfc-nsh-05 (work in progress), May 2016.

[I-D.ietf-spring-segment-routing]
Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", draft-ietf-spring-segment-routing-09 (work in progress), July 2016.

[I-D.penno-sfc-trace]
Penno, R., Quinn, P., Pignataro, C., and D. Zhou, "Services Function Chaining Traceroute", draft-penno-sfc-trace-03 (work in progress), September 2015.

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

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

[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

ippm
Internet-Draft
Intended status: Informational
Expires: January 3, 2018

F. Brockners
S. Bhandari
V. Govindan
C. Pignataro
Cisco
H. Gredler
RtBrick Inc.
J. Leddy
Comcast
S. Youell
JMPC
T. Mizrahi
Marvell
D. Mozes
Mellanox Technologies Ltd.
P. Lapukhov
Facebook
R. Chang
Barefoot Networks
July 02, 2017

Encapsulations for In-situ OAM Data
draft-brockners-inband-oam-transport-05

Abstract

In-situ Operations, Administration, and Maintenance (OAM) records operational and telemetry information in the packet while the packet traverses a path between two points in the network. In-situ OAM is to complement current out-of-band OAM mechanisms based on ICMP or other types of probe packets. This document outlines how in-situ OAM data fields can be transported in protocols such as NSH, Segment Routing, VXLAN-GPE, native IPv6 (via extension headers), and IPv4. Transport options are currently investigated as part of an implementation study. This document is intended to only serve informational purposes.

Status of This Memo

Table of Contents

1. Introduction 3
2. Conventions 4
3. In-Situ OAM Metadata Transport in IPv6 4
3.1. In-situ OAM in IPv6 Hop by Hop Extension Header 5
3.1.1. In-situ OAM Hop by Hop Options 5
4. In-situ OAM Metadata Transport in IPv4 7
4.1. In-situ OAM Tracing in GRE 7
4.2. In-situ OAM POT in GRE 10
4.3. In-situ OAM End-to-End in GRE 12
5. In-situ OAM Metadata Transport in VXLAN-GPE 12
5.1. In-situ OAM Tracing in VXLAN-GPE 13
5.2. In-situ OAM POT in VXLAN-GPE 16
5.3. In-situ OAM Edge-to-Edge in VXLAN-GPE 18
6. In-situ OAM Metadata Transport in NSH 18
6.1. In-situ OAM Tracing in NSH 18
6.2. In-situ OAM POT in NSH 22
6.3. In-situ OAM Edge-to-Edge in NSH 24
7. In-situ OAM Metadata Transport in Segment Routing 25
7.1. In-situ OAM in SR with IPv6 Transport 25
7.2. In-situ OAM in SR with MPLS Transport 26
8. IANA Considerations 26
9. Manageability Considerations 26
10. Security Considerations 26
11. Acknowledgements 26

12. References 27
12.1. Normative References 27
12.2. Informative References 28
Authors' Addresses 28

1. Introduction

This document discusses transport mechanisms for "in-situ" Operations, Administration, and Maintenance (OAM) data fields. In-situ OAM records OAM information within the packet while the packet traverses a particular network domain. The term "in-situ" refers to the fact that the OAM data is added to the data packets rather than is being sent within packets specifically dedicated to OAM. A discussion of the motivation and requirements for in-situ OAM can be found in [I-D.brockners-inband-oam-requirements]. Data types and data formats for in-situ OAM are defined in [I-D.brockners-inband-oam-data].

This document outlines transport encapsulations for the in-situ OAM data defined in [I-D.brockners-inband-oam-data]. This document is to serve informational purposes only. As part of an in-situ OAM implementation study different protocol encapsulations for in-situ OAM data are being explored. Once data formats and encapsulation approaches are settled, protocol specific specifications for in-situ OAM data transport will address the standardization aspect.

The data for in-situ OAM defined in [I-D.brockners-inband-oam-data] can be carried in a variety of protocols based on the deployment needs. This document discusses transport of in-situ OAM data for the following protocols:

- IPv6
- IPv4
- VXLAN-GPE
- NSH
- Segment Routing (IPv6 and MPLS)

This list is non-exhaustive, as it is possible to carry the in-situ OAM data in several other protocols and transports.

A feasibility study of in-situ OAM is currently underway as part of the FD.io project [FD.io]. The in-situ OAM implementation study should be considered as a "tool box" to showcase how "in-situ" OAM can complement probe-packet based OAM mechanisms for different deployments and packet transport formats. For details, see the open source code in the FD.io [FD.io].

2. Conventions

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

Abbreviations used in this document:

IOAM: In-situ Operations, Administration, and Maintenance

MTU: Maximum Transmit Unit

NSH: Network Service Header

OAM: Operations, Administration, and Maintenance

POT: Proof of Transit

SFC: Service Function Chain

SID: Segment Identifier

SR: Segment Routing

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

3. In-Situ OAM Metadata Transport in IPv6

This mechanisms of in-situ OAM in IPv6 complement others proposed to enhance diagnostics of IPv6 networks, such as the IPv6 Performance and Diagnostic Metrics Destination Option described in [I-D.ietf-ippm-6man-pdm-option]. The IP Performance and Diagnostic Metrics Destination Option is destination focused and specific to IPv6, whereas in-situ OAM is performed between end-points of the network or a network domain where it is enabled and used.

A historical note: The idea of IPv6 route recording was originally introduced by [I-D.kitamura-ipv6-record-route] back in year 2000. With IPv6 now being generally deployed and new concepts such as Segment Routing [I-D.ietf-spring-segment-routing] being introduced, it is imperative to further mature the Operations, Administration, and Maintenance mechanisms available to IPv6 networks.

The in-situ OAM options translate into options for an IPv6 hop by hop extension header. The extension header would be inserted by either a host source of the packet, or by a transit/domain-edge node. If the addition of the in-situ OAM Hop-by-Hop Option header would lead to the packet exceeding the MTU of the domain an error should be reported. The methods and procedures of how the error is reported are outside the scope of this document. Likewise if an ICMPv6 forwarding error occurs between encapsulating and decapsulating nodes, the node generating the ICMPv6 error should strip the in-situ OAM Hop-by-Hop Option header before sending the ICMPv6 message to the source.

3.1. In-situ OAM in IPv6 Hop by Hop Extension Header

This section defines in-situ OAM for IPv6 transport. In-situ OAM Options are transported in IPv6 hop-by-hop extension header.

3.1.1. In-situ OAM Hop by Hop Options

IPv6 hop-by-hop option format for carrying in-situ OAM data fields:

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Option Type  |  Opt Data Len |         Reserved (MBZ)        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|                                                               |  |
.                                                               .  I
.                     Option Data                               .  O
.                                                               .  A
.                                                               .  M
.                                                               .  .
.                                                               .  O
.                                                               .  P
.                                                               .  T
.                                                               .  I
.                                                               .  O
.                                                               .  N
.                                                               .  |
|                                                               |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
```

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

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

Reserved (MBZ)   16-bit field MUST be filled with zeroes.

Option Data   Variable-length field. Option-Type-specific data.

In-situ OAM Options are inserted as Option data as follows:

1. Pre-allocated Tracing Option: The in-situ OAM Preallocated Tracing option defined in [I-D.brockners-inband-oam-data] is represented as a IPv6 option in hop by hop extension header by allocating following type:

   Option Type: 001xxxxxx 8-bit identifier of the type of option.
      xxxxxx=TBD_IANA_PRE_TRACE_OPTION_IPV6.

2. Incremental Tracing Option: The in-situ OAM Incremental Tracing option defined in [I-D.brockners-inband-oam-data] is represented as a IPv6 option in hop by hop extension header by allocating following type:

   Option Type: 001xxxxxx 8-bit identifier of the type of option.
      xxxxxx=TBD_IANA_INCR_TRACE_OPTION_IPV6.

3. Proof of Transit Option: The in-situ OAM POT option defined in [I-D.brockners-inband-oam-data] is represented as a IPv6 option in hop by hop extension header by allocating following type:

   Option Type: 001xxxxxx 8-bit identifier of the type of option. xxxxxx=TBD_IANA_POT_OPTION_IPV6.

4. Edge to Edge Option: The in-situ OAM E2E option defined in [I-D.brockners-inband-oam-data] is represented as a IPv6 option in hop by hop extension header by allocating following type:

   Option Type: 000xxxxxx 8-bit identifier of the type of option. xxxxxx=TBD_IANA_E2E_OPTION_IPV6.

4. In-situ OAM Metadata Transport in IPv4

Transport of in-situ OAM data in IPv4 will use GRE encapsulation.

GRE encapsulation is defined in [RFC2784]. IOAM is defined as a "set of Protocol Types" TBD_IANA_ETHERNET_NUMBER_IOAM_* and follows GRE header. These Protocol Types are defined in [RFC3232] as "ETHER TYPES" and in [ETYPES].

The different IOAM data fields defined in [I-D.brockners-inband-oam-data] are added as TLVs following the GRE header. In an administrative domain where IOAM is used, insertion of the IOAM protocol header in GRE is enabled at the GRE tunnel endpoints which also serve as IOAM encapsulating/decapsulating nodes by means of configuration.

For IOAM the following new GRE protocol types are requested:

1. IOAM_Trace_Preallocated: TBD_IANA_ETHERNET_NUMBER_IOAM_TRACE_PREALLOCATED

2. IOAM_Trace_Incremental: TBD_IANA_ETHERNET_NUMBER_IOAM_TRACE_INCREMENTAL

3. IOAM_POT: TBD_IANA_ETHERNET_NUMBER_IOAM_POT

4. IOAM_End-to_End: TBD_IANA_ETHERNET_NUMBER_IOAM_E2E

4.1. In-situ OAM Tracing in GRE

The packet formats of the pre-allocated IOAM trace and incremental IOAM trace when transported using GRE are defined as below. See [I-D.brockners-inband-oam-data] for details about pre-allocated and incremental IOAM trace options.

In-situ OAM Trace header following GRE header(Preallocated IOAM trace):

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|C|       Reserved0       | Ver | Protocol Type = IOAM_Trace    |  G
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  R
|      Checksum (optional)      |       Reserved1 (Optional)    |  E
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|     Type      |   IOAM HDR len|        Next Protocol          |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+IOAM
|          IOAM-Trace-Type      |NodeLen|  Flags  | Octets-left |Trace
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|                                                               |  |
|                        node data list [0]                     |  IOAM
|                                                               |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  D
|                                                               |  a
|                        node data list [1]                     |  t
|                                                               |  a
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~                             ...                               ~  S
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  p
|                                                               |  a
|                        node data list [n-1]                   |  c
|                                                               |  e
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
|                                                               |  |
|                        node data list [n]                     |  |
|                                                               |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|                                                               |
|                     Payload + Padding (L2/L3/ESP/...)         |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Pre-allocated Trace Option Data MUST be 4-octet aligned:

In-situ OAM Trace header following GRE header(Incremental IOAM trace):

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|C|       Reserved0       | Ver |Protocol Type = IOAM_Trace     |  G
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  R
|      Checksum (optional)      |       Reserved1 (Optional)    |  E
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|     Type      |   IOAM HDR len|        Next Protocol          |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+IOAM
|        IOAM-Trace-Type        |NodeLen|  Flags  | Max Length  |Trace
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|                                                               |  |
|                        node data list [0]                     |  IOAM
|                                                               |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  D
|                                                               |  a
|                        node data list [1]                     |  t
|                                                               |  a
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~                             ...                               ~  S
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  p
|                                                               |  a
|                        node data list [n-1]                   |  c
|                                                               |  e
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
|                                                               |  |
|                        node data list [n]                     |  |
|                                                               |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|                                                               |
|                     Payload + Padding (L2/L3/ESP/...)         |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

In-situ OAM Incremental Trace Option Data MUST be 4-octet aligned:

The GRE header and fields are defined in [RFC2784] with Protocol Type set to TBD_IANA_ETHERNET_NUMBER_IOAM_TRACE. IOAM specific fields and header are defined here:

Type: 8-bit unsigned integer defining IOAM header type IOAM_TRACE_Preallocated or IOAM_Trace_Incremental are defined here.

IOAM HDR Len: 8 bits Length field contains the length of the variable metadata octets.

Next Protocol: 16 bits Next Protocol Type field contains the protocol type of the packet following IOAM protocol header. These Protocol Types are defined in [RFC3232] as "ETHER TYPES" and in [ETYPES]. An implementation receiving a packet containing a Protocol Type which is not listed in [RFC3232] or [ETYPES] SHOULD discard the packet.

IOAM-Trace-Type: 16-bit identifier of IOAM Trace Type as defined in [I-D.brockners-inband-oam-data] IOAM-Trace-Types.

Node Data Length: 4-bit unsigned integer as defined in [I-D.brockners-inband-oam-data].

Flags: 5-bit field as defined in [I-D.brockners-inband-oam-data].

Octets-left: 7-bit unsigned integer as defined in [I-D.brockners-inband-oam-data].

Maximum-length: 7-bit unsigned integer as defined in [I-D.brockners-inband-oam-data].

Node data List [n]: Variable-length field as defined in [I-D.brockners-inband-oam-data].

4.2. In-situ OAM POT in GRE

In-situ OAM POT header following GRE header:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|C|       Reserved0       | Ver | Protocol Type = IOAM_POT      |  G
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  R
|      Checksum (optional)      |       Reserved1 (Optional)    |  E
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|IOAM POT Type|P|   IOAM HDR len|     Next Protocol             |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+IOAM
|                           Random                              |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  P
|                        Random(contd.)                         |  O
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  T
|                         Cumulative                            |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
|                    Cumulative (contd.)                        |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|                                                               |
|                     Payload + Padding (L2/L3/ESP/...)         |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The GRE header and fields are defined in [RFC2784] with Protocol Type set to TBD_IANA_ETHERNET_NUMBER_IOAM_POT. IOAM specific fields and header are defined here:

IOAM POT Type: 7-bit identifier of a particular POT variant that dictates the POT data that is included as defined in [I-D.brockners-inband-oam-data].

Profile to use (P): 1-bit as defined in [I-D.brockners-inband-oam-data] IOAM POT Option.

IOAM HDR Len: 8 bits Length field contains the length of the variable metadata octets.

Next Protocol: 16 bits Next Protocol Type field contains the protocol type of the packet following IOAM protocol header. These Protocol Types are defined in [RFC3232] as "ETHER TYPES" and in [ETYPES]. An implementation receiving a packet containing a Protocol Type which is not listed in [RFC3232] or [ETYPES] SHOULD discard the packet.

Random: 64-bit Per-packet random number.

Cumulative: 64-bit Cumulative value that is updated by the Service Functions.

4.3. In-situ OAM End-to-End in GRE

In-situ OAM End-to-End header following GRE header:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|C|       Reserved0       | Ver | Protocol Type = IOAM_E2E      |  G
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  R
|      Checksum (optional)      |       Reserved1 (Optional)    |  E
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|IOAM_E2E_Type  |   IOAM HDR len|     Next Protocol             |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+IOAM
|      E2E Option data field determined by IOAM-E2E-Type        |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|                                                               |
|                     Payload + Padding (L2/L3/ESP/...)         |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IOAM E2E Type: 8-bit identifier of a particular E2E variant that dictates the E2E data that is included as defined in [I-D.brockners-inband-oam-data].

IOAM HDR Len: 8 bits Length field contains the length of the variable metadata octets.

Next Protocol: 16 bits Next Protocol Type field contains the protocol type of the packet following IOAM protocol header. These Protocol Types are defined in [RFC3232] as "ETHER TYPES" and in [ETYPES]. An implementation receiving a packet containing a Protocol Type which is not listed in [RFC3232] or [ETYPES] SHOULD discard the packet.

E2E Option data field: Variable length field as defined in [I-D.brockners-inband-oam-data] IOAM E2E Option.

5. In-situ OAM Metadata Transport in VXLAN-GPE

VXLAN-GPE [I-D.ietf-nvo3-vxlan-gpe] encapsulation is somewhat similar to IPv6 extension headers in that a series of headers can be contained in the header as a linked list. The different iIOAM types are added as options within a new IOAM protocol header in VXLAN GPE. In an administrative domain where IOAM is used, insertion of the IOAM protocol header in VXLAN GPE is enabled at the VXLAN GPE tunnel endpoint which also serve as IOAM encapsulating/decapsulating nodes by means of configuration.

5.1. In-situ OAM Tracing in VXLAN-GPE

The packet formats of the pre-allocated IOAM trace and incremental IOAM trace when transported in VXLAN-GPE are defined as below. See [I-D.brockners-inband-oam-data] for details about pre-allocated and incremental IOAM trace options.

The VXLAN-GPE header and fields are defined in [I-D.ietf-nvo3-vxlan-gpe]. IOAM specific fields and header are defined here:

In-situ OAM Trace header following VXLAN GPE header
(Pre-allocated trace):

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Outer Ethernet Header                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Outer IP Header                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Outer UDP Header                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+--+
|R|R|Ver|I|P|R|O|          Reserved            |NP=IOAM_Trace  |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ GPE
|     Virtual Network Identifier (VNI)         | Reserved      |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+--+
|      Type     |   IOAM HDR len|    Reserved   | Next Protocol |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+IOAM
|          IOAM-Trace-Type      |NodeLen|  Flags  | Octets-left |Trace
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|                                                               |  |
|                       node data list [0]                      |  IOAM
|                                                               |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  D
|                                                               |  a
|                       node data list [1]                      |  t
|                                                               |  a
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~                             ...                               ~  S
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  p
|                                                               |  a
|                       node data list [n-1]                    |  c
|                                                               |  e
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
|                                                               |  |
|                       node data list [n]                      |  |
|                                                               |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-<--+
|                                                               |
|                                                               |
|                    Payload + Padding (L2/L3/ESP/...)          |
|                                                               |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Pre-allocated Trace Option Data MUST be 4-octet aligned:

In-situ OAM Trace header following VXLAN GPE header
(Incremental IOAM trace):

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Outer Ethernet Header                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Outer IP Header                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Outer UDP Header                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+---+
|R|R|Ver|I|P|R|O|           Reserved            | NP=IOAM_Trace |   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ GPE
|     Virtual Network Identifier (VNI)          | Reserved      |   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+---+
|      Type     |   IOAM HDR len|    Reserved   | Next Protocol |   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+IOAM
|         IOAM-Trace-Type       |NodeLen|  Flags  | Max Length  |Trace
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|                                                               |  |
|                         node data list [0]                    |  IOAM
|                                                               |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  D
|                                                               |  a
|                         node data list [1]                    |  t
|                                                               |  a
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~                              ...                              ~  S
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  p
|                                                               |  a
|                         node data list [n-1]                  |  c
|                                                               |  e
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
|                                                               |  |
|                         node data list [n]                    |  |
|                                                               |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-<--+
|                                                               |
|                                                               |
|                      Payload + Padding (L2/L3/ESP/...)        |
|                                                               |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
In-situ OAM Incremental Trace Option Data MUST be 4-octet aligned:

Type: 8-bit unsigned integer defining IOAM header type IOAM_TRACE_Preallocated or IOAM_Trace_Incremental are defined here.

IOAM HDR len: 8-bit unsigned integer. Length of the in-situ OAM HDR in 8-octet units.

Reserved: 8-bit reserved field MUST be set to zero.

Next Protocol: 8-bit unsigned integer that determines the type of header following IOAM protocol. The value is from the IANA registry setup for VXLAN GPE Next Protocol defined in [I-D.ietf-nvo3-vxlan-gpe].

IOAM-Trace-Type: 16-bit identifier of IOAM Trace Type as defined in [I-D.brockners-inband-oam-data] IOAM-Trace-Types.

Node Data Length: 4-bit unsigned integer as defined in [I-D.brockners-inband-oam-data].

Flags: 5-bit field as defined in [I-D.brockners-inband-oam-data].

Octets-left: 7-bit unsigned integer as defined in [I-D.brockners-inband-oam-data].

Maximum-length: 7-bit unsigned integer as defined in [I-D.brockners-inband-oam-data].

Node data List [n]: Variable-length field as defined in [I-D.brockners-inband-oam-data].

5.2. In-situ OAM POT in VXLAN-GPE

The VXLAN-GPE header and fields are defined in [I-D.ietf-nvo3-vxlan-gpe]. IOAM specific fields and header are defined here:

In-situ OAM POT header following VXLAN GPE header:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Outer Ethernet Header                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Outer IP Header                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Outer UDP Header                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+--+
|R|R|Ver|I|P|R|O|          Reserved(MBZ)        |NP = IOAM_POT  |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ GPE
|     Virtual Network Identifier (VNI)          | Reserved      |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+--+
|IOAM POT Type|P|   IOAM HDR len|    Reserved   | Next Protocol |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+IOAM
|                          Random                               |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  P
|                        Random(contd.)                         |  O
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  T
|                         Cumulative                            |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
|                    Cumulative (contd.)                        |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
```

IOAM POT Type: 7-bit identifier of a particular POT variant that dictates the POT data that is included as defined in [I-D.brockners-inband-oam-data].

Profile to use (P): 1-bit as defined in [I-D.brockners-inband-oam-data] IOAM POT Option.

IOAM HDR len: 8-bit unsigned integer. Length of the in-situ OAM HDR in 8-octet units

Reserved: 8-bit reserved field MUST be set to zero.

Next Protocol: 8-bit unsigned integer that determines the type of header following IOAM protocol. The value is from the IANA registry setup for VXLAN GPE Next Protocol defined in [I-D.ietf-nvo3-vxlan-gpe].

Random: 64-bit Per-packet random number.

Cumulative: 64-bit Cumulative value that is updated by the Service Functions.

5.3. In-situ OAM Edge-to-Edge in VXLAN-GPE

In-situ OAM Edge-to-Edge in VXLAN GPE header:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Outer Ethernet Header                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Outer IP Header                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Outer UDP Header                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+--+
|R|R|Ver|I|P|R|O|          Reserved              |NP = IOAM_E2E  |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ GPE
|     Virtual Network Identifier (VNI)          | Reserved      |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+--+
|Type=IOAM_E2E  |   IOAM HDR len  |  Reserved   | Next Protocol |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+IOAM
|      E2E Option data field determined by IOAM-E2E-Type        |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
```

Type: 8-bit identifier of a particular E2E variant that dictates the E2E data that is included as defined in [I-D.brockners-inband-oam-data].

IOAM HDR len: 8-bit unsigned integer. Length of the in-situ OAM HDR in 8-octet units

Reserved: 8-bit reserved field MUST be set to zero.

Next Protocol: 8-bit unsigned integer that determines the type of header following IOAM protocol. The value is from the IANA registry setup for VXLAN GPE Next Protocol defined in [I-D.ietf-nvo3-vxlan-gpe].

E2E Option data field: Variable length field as defined in [I-D.brockners-inband-oam-data] IOAM E2E Option.

6. In-situ OAM Metadata Transport in NSH

6.1. In-situ OAM Tracing in NSH

The packet formats of the pre-allocated IOAM trace and incremental IOAM trace when transported in NSH are defined as below. See [I-D.brockners-inband-oam-data] for details about pre-allocated and incremental IOAM trace options.

In Service Function Chaining (SFC) [RFC7665], the Network Service Header (NSH) [I-D.ietf-sfc-nsh] already includes path tracing capabilities [I-D.penno-sfc-trace]. Tracing information can be carried in-situ as IOAM data fields following NSH MDx metadata TLVs.

In-situ OAM Trace header following NSH MDx header
(Pre-allocated IOAM trace):

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|Ver|O|C|R|R|R|R|R|R|   Length  |  MD Type      | NP=IOAM_Trace |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  N
|          Service Path Identifer               | Service Index |  S
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  H
|                            ...                                |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|      Type     |  IOAM HDR len|    Reserved   | Next Protocol |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+IOAM
|         IOAM-Trace-Type       |NodeLen|  Flags  | Octets-left |Trace
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|                                                               |  |
|                        node data list [0]                     |IOAM
|                                                               |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  D
|                                                               |  a
|                        node data list [1]                     |  t
|                                                               |  a
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~                             ...                               ~  S
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  p
|                                                               |  a
|                        node data list [n-1]                   |  c
|                                                               |  e
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
|                                                               |  |
|                        node data list [n]                     |  |
|                                                               |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<--+
|                                                               |
|                                                               |
|                     Payload + Padding (L2/L3/ESP/...)         |
|                                                               |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

In-situ OAM Pre-allocated Trace Option Data MUST be 4-octet aligned:

In-situ OAM Trace header following NSH MDx header
(Incremental IOAM trace):

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|Ver|O|C|R|R|R|R|R|R|   Length  |  MD Type      | NP=IOAM_Trace |  N
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  S
|          Service Path Identifer               | Service Index |  H
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
|                            ...                                |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|      Type       |  IOAM HDR len |    Reserved   | Next Protocol |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+IOAM
|        IOAM-Trace-Type        |NodeLen|  Flags  | Max Length    |Trace
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|                                                               |  |
|                        node data list [0]                     |IOAM
|                                                               |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  D
|                                                               |  a
|                        node data list [1]                     |  t
|                                                               |  a
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~                             ...                               ~  S
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  p
|                                                               |  a
|                        node data list [n-1]                   |  c
|                                                               |  e
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
|                                                               |  |
|                        node data list [n]                     |  |
|                                                               |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-<--+
|                                                               |
|                                                               |
|                     Payload + Padding (L2/L3/ESP/...)         |
|                                                               |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

In-situ OAM Incremental Trace Option Data MUST be 4-octet aligned:

Next Protocol of NSH: TBD value for IOAM_Trace.

Type: 8-bit unsigned integer defining IOAM header type IOAM_TRACE_Preallocated or IOAM_Trace_Incremental are defined here.

IOAM HDR len: 8-bit unsigned integer. Length of the in-situ OAM HDR in 8-octet units.

Reserved bits and R bits: Reserved bits are present for future use. The reserved bits MUST be set to 0x0.

Next Protocol: 8-bit unsigned integer that determines the type of header following IOAM protocol.

IOAM-Trace-Type: 16-bit identifier of IOAM Trace Type as defined in [I-D.brockners-inband-oam-data] IOAM-Trace-Types.

Node Data Length: 4-bit unsigned integer as defined in [I-D.brockners-inband-oam-data].

Flags: 5-bit field as defined in [I-D.brockners-inband-oam-data].

Octets-left: 7-bit unsigned integer as defined in [I-D.brockners-inband-oam-data].

Maximum-length: 7-bit unsigned integer as defined in [I-D.brockners-inband-oam-data].

Node data List [n]: Variable-length field as defined in [I-D.brockners-inband-oam-data].

6.2. In-situ OAM POT in NSH

The "Proof of Transit" capabilities (see [I-D.brockners-inband-oam-requirements] and [I-D.brockners-proof-of-transit]) of in-situ OAM can be leveraged within NSH. In an administrative domain where in-situ OAM is used, insertion of the in-situ OAM data into the NSH header is enabled at the required nodes (i.e. at the in-situ OAM encapsulating/ decapsulating nodes) by means of configuration.

Proof of transit in-situ OAM data is added as NSH Type 2 metadata:

In-situ OAM POT header following NSH MDx header:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|Ver|O|C|R|R|R|R|R|R|   Length  |  MD Type      |NP = IOAM_POT  |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  N
|          Service Path Identifer               | Service Index |  S
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  H
|                            ...                                |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|IOAM_POT Type|P|   IOAM HDR len|    Reserved   | Next Protocol |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
|                          Random                               |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  P
|                       Random(contd.)                          |  O
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  T
|                        Cumulative                             |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
|                   Cumulative (contd.)                         |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
```

Next Protocol of NSH: TBD value for IOAM_POT.

IOAM POT Type: 7-bit identifier of a particular POT variant that dictates the POT data that is included as defined in [I-D.brockners-inband-oam-data].

Profile to use (P): 1-bit as defined in [I-D.brockners-inband-oam-data] IOAM POT Option.

IOAM HDR len: 8-bit unsigned integer. Length of the in-situ OAM HDR in 8-octet units

Reserved bits and R bits: Reserved bits are present for future use. The reserved bits MUST be set to 0x0.

Next Protocol: 8-bit unsigned integer that determines the type of header following IOAM protocol.

Random: 64-bit Per-packet random number.

Cumulative: 64-bit Cumulative value that is updated by the Service Functions.

6.3. In-situ OAM Edge-to-Edge in NSH

The "Edge-to-Edge" capabilities (see [I-D.brockners-inband-oam-requirements]) of in-situ OAM can be leveraged within NSH. In an administrative domain where in-situ OAM is used, insertion of the in-situ OAM data into the NSH header is enabled at the required nodes (i.e. at the in-situ OAM encapsulating/ decapsulating nodes) by means of configuration.

Edge-to-Edge in-situ OAM data is added as a TLV following NSH MDx metadata:

In-situ OAM E2E header following NSH MDx header:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|Ver|O|C|R|R|R|R|R|R|   Length  |  MD Type      |NP = IOAM_E2E  |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  N
|          Service Path Identifer               | Service Index |  S
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  H
|                            ...                                |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|IOAM_E2E_Type  |   IOAM HDR len|    Reserved   | Next Protocol | IOAM
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  E2E
|      E2E Option data field determined by IOAM-E2E-Type        |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
```

Next Protocol of NSH: TBD value for IOAM_E2E.

IOAM E2E Type: 8-bit identifier of a particular E2E variant that dictates the IOAM E2E data that is included as defined in [I-D.brockners-inband-oam-data].

IOAM HDR len: 8-bit unsigned integer. Length of the in-situ OAM HDR in 8-octet units

Reserved bits and R bits: Reserved bits are present for future use. The reserved bits MUST be set to 0x0.

Next Protocol: 8-bit unsigned integer that determines the type of header following IOAM protocol.

E2E Option data field: Variable length field as defined in [I-D.brockners-inband-oam-data] IOAM E2E Option.

7. In-situ OAM Metadata Transport in Segment Routing

7.1. In-situ OAM in SR with IPv6 Transport

Similar to NSH, a policy defined using Segment Routing for IPv6 can be verified using the in-situ OAM "Proof of Transit" approach. The Segment Routing Header (SRH) for IPv6 offers the ability to transport TLV structured data, similar to what NSH does (see [I-D.ietf-6man-segment-routing-header]). In an domain where in-situ OAM is used, insertion of the in-situ OAM data is enabled at the required edge nodes (i.e. at the in-situ OAM encapsulating/ decapsulating nodes) by means of configuration.

A new "POT TLV" is defined for the SRH which is to carry proof of transit in situ OAM data.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Type     |    Length     |   RESERVED    |F|   Flags     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|IOAM POT Type|P|                    Reserved (MBZ)             |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
|                              Random                           |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  P
|                           Random(contd.)                      |  O
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  T
|                            Cumulative                         |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  |
|                        Cumulative (contd.)                    |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
```

Type: To be assigned by IANA.

Length: 20.

RESERVED: 8 bits. SHOULD be unset on transmission and MUST be ignored on receipt.

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.

Flags: 8 bits. No flags are defined in this document.

IOAM POT Type: 7-bit identifier of a particular POT variant that dictates the POT data that is included as defined in [I-D.brockners-inband-oam-data].

Profile to use (P): 1-bit as defined in [I-D.brockners-inband-oam-data] IOAM POT Option.

Reserved (MBZ): 24-bit field MUST be filled with zeroes.

Random: 64-bit per-packet random number.

Cumulative: 64-bit cumulative value that is updated at specific nodes that form the service path to be verified.

7.2. In-situ OAM in SR with MPLS Transport

In-situ OAM "Proof of Transit" data can also be carried as part of the MPLS label stack. Details will be addressed in a future version of this document.

8. IANA Considerations

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

9. Manageability Considerations

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

10. Security Considerations

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

11. Acknowledgements

The authors would like to thank Eric Vyncke, Nalini Elkins, Srihari Raghavan, Ranganathan T S, Karthik Babu Harichandra Babu, Akshaya Nadahalli, Stefano Previdi, Hemant Singh, Erik Nordmark, LJ Wobker, and Andrew Yourtchenko for the comments and advice. The authors would like to acknowledge Craig Hill for contributing GRE IOAM encapsulation. For the IPv6 encapsulation, this document leverages and builds on top of several concepts described in [I-D.kitamura-ipv6-record-route]. The authors would like to acknowledge the work done by the author Hiroshi Kitamura and people involved in writing it.

12. References

12.1. Normative References

[ETYPES] "IANA Ethernet Numbers", <https://www.iana.org/assignments/ethernet-numbers/ethernet-numbers.xhtml>.

[I-D.brockners-inband-oam-data] Brockners, F., Bhandari, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Mozes, D., Lapukhov, P., <>, R., and d. daniel.bernier@bell.ca, "Data Fields for In-situ OAM", draft-brockners-inband-oam-data-05 (work in progress), May 2017.

[I-D.brockners-inband-oam-requirements] Brockners, F., Bhandari, S., Dara, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mozes, D., Mizrahi, T., <>, P., and r. remy@barefootnetworks.com, "Requirements for In-situ OAM", draft-brockners-inband-oam-requirements-03 (work in progress), March 2017.

[I-D.ietf-6man-segment-routing-header] Previdi, S., Filsfils, C., Raza, K., Leddy, J., Field, B., daniel.voyer@bell.ca, d., daniel.bernier@bell.ca, d., Matsushima, S., Leung, I., Linkova, J., Aries, E., Kosugi, T., Vyncke, E., Lebrun, D., Steinberg, D., and R. Raszuk, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-06 (work in progress), March 2017.

[I-D.ietf-nvo3-vxlan-gpe] Maino, F., Kreeger, L., and U. Elzur, "Generic Protocol Extension for VXLAN", draft-ietf-nvo3-vxlan-gpe-04 (work in progress), April 2017.

[I-D.ietf-sfc-nsh] Quinn, P. and U. Elzur, "Network Service Header", draft-ietf-sfc-nsh-13 (work in progress), June 2017.

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

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

[RFC3232] Reynolds, J., Ed., "Assigned Numbers: RFC 1700 is Replaced by an On-line Database", RFC 3232, DOI 10.17487/RFC3232, January 2002, <http://www.rfc-editor.org/info/rfc3232>.

12.2. Informative References

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

[I-D.brockners-proof-of-transit]
Brockners, F., Bhandari, S., Dara, S., Pignataro, C., Leddy, J., Youell, S., Mozes, D., and T. Mizrahi, "Proof of Transit", draft-brockners-proof-of-transit-03 (work in progress), March 2017.

[I-D.ietf-ippm-6man-pdm-option]
Elkins, N., Hamilton, R., and m. mackermann@bcbsm.com, "IPv6 Performance and Diagnostic Metrics (PDM) Destination Option", draft-ietf-ippm-6man-pdm-option-13 (work in progress), June 2017.

[I-D.ietf-spring-segment-routing]
Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", draft-ietf-spring-segment-routing-12 (work in progress), June 2017.

[I-D.kitamura-ipv6-record-route]
Kitamura, H., "Record Route for IPv6 (PR6) Hop-by-Hop Option Extension", draft-kitamura-ipv6-record-route-00 (work in progress), November 2000.

[I-D.penno-sfc-trace]
Penno, R., Quinn, P., Pignataro, C., and D. Zhou, "Services Function Chaining Traceroute", draft-penno-sfc-trace-03 (work in progress), 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

Vengada Prasad Govindan
Cisco Systems, Inc.

Email: venggovi@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

John Leddy
Comcast

Email: John_Leddy@cable.comcast.com

Stephen Youell
JP Morgan Chase
25 Bank Street
London E14 5JP
United Kingdom

Email: stephen.youell@jpmorgan.com

Tal Mizrahi
Marvell
6 Hamada St.
Yokneam 20692
Israel

Email: talmi@marvell.com

David Mozes
Mellanox Technologies Ltd.

Email: davidm@mellanox.com

Petr Lapukhov
Facebook
1 Hacker Way
Menlo Park, CA 94025
US

Email: petr@fb.com

Remy Chang
Barefoot Networks
2185 Park Boulevard
Palo Alto, CA 94306
US

Network Working Group
Internet-Draft
Intended status: Experimental
Expires: January 9, 2017

F. Brockners
S. Bhandari
S. Dara
C. Pignataro
Cisco

July 8, 2016

Proof of Transit
draft-brockners-proof-of-transit-00

Abstract

Several technologies such as traffic engineering, service function chaining, or policy based routing, are used to steer traffic through a specific, user-defined path. This document defines mechanisms to securely prove that traffic transited the defined path. The mechanisms allow to securely verify whether all packets traversed all those nodes of a given path that they are supposed to visit.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction 3
2. Conventions 4
3. Proof of Transit 4
  3.1. Basic Idea 4
  3.2. Solution Approach 5
    3.2.1. Setup 6
    3.2.2. In Transit 6
    3.2.3. Verification 6
  3.3. Example for Illustration 6
    3.3.1. Basic Version 6
      3.3.1.1. Secret Shares 7
      3.3.1.2. Lagrange Polynomials 7
      3.3.1.3. LPC Computation 7
      3.3.1.4. Reconstruction 8
      3.3.1.5. Verification 8
    3.3.2. Enhanced Version 8
      3.3.2.1. Random Polynomial 8
      3.3.2.2. Reconstruction 9
      3.3.2.3. Verification 9
  3.4. Operational Aspects 10
4. Sizing the Data for Proof of Transit 10
5. Node Configuration 11
  5.1. Procedure 11
  5.2. YANG Model 12
6. IANA Considerations 15
7. Manageability Considerations 15
8. Security Considerations 15
  8.1. Proof of Transit 15
  8.2. Anti Replay 16
  8.3. Anti Tampering 16
  8.4. Recycling 16
  8.5. Redundant Nodes and Failover 16
  8.6. Controller Operation 17
  8.7. Verification Scope 17
    8.7.1. Node Ordering 17
    8.7.2. Stealth Nodes 17
9. Acknowledgements 18
10. Normative References 18
Authors' Addresses 18

1. Introduction

Several deployments use traffic engineering, policy routing, segment routing or Service Function Chaining (SFC) [RFC7665] to steer packets through a specific set of nodes. In certain cases regulatory obligations or a compliance policy require operators to prove that all packets that are supposed to follow a specific path are indeed being forwarded across and exact set of pre-determined nodes.

If a packet flow is supposed to go through a series of service functions or network nodes, it has to be proven that indeed all packets of the flow followed the path or service chain or collection of nodes specified by the policy. In case some packets of a flow weren’t appropriately processed, a verification device should determine the policy violation and take corresponding actions corresponding to the policy (e.g., drop or redirect the packet, send an alert etc.). In today’s deployments, the proof that a packet traversed a particular path or service chain is typically delivered in an indirect way: Service appliances and network forwarding are in different trust domains. Physical hand-off-points are defined between these trust domains (i.e. physical interfaces). Or in other terms, in the "network forwarding domain" things are wired up in a way that traffic is delivered to the ingress interface of a service appliance and received back from an egress interface of a service appliance. This "wiring" is verified and then trusted upon. The evolution to Network Function Virtualization (NFV) and modern service chaining concepts (using technologies such as LISP, NSH, Segment Routing (SR), etc.) blurs the line between the different trust domains, because the hand-off-points are no longer clearly defined physical interfaces, but are virtual interfaces. As a consequence, different trust layers should not to be mixed in the same device. For an NFV scenario a different type of proof is required. Offering a proof that a packet indeed traversed a specific set of service functions or nodes allows operators to evolve from the above described indirect methods of proving that packets visit a predetermined set of nodes.

The solution approach presented in this document is based on a small portion of operational data added to every packet. This "in-band" operational data is also referred to as "proof of transit data", or POT data. The POT data is updated at every required node and is used to verify whether a packet traversed all required nodes. A particular set of nodes "to be verified" is either described by a set of secret keys, or a set of shares of a single secret. Nodes on the path retrieve their individual keys or shares of a key (using for e.g., Shamir’s Secret Sharing scheme) from a central controller. The complete key set is only known to the controller and a verifier node, which is typically the ultimate node on a path that performs verification. Each node in the path uses its secret or share of the secret to update the POT data of the packets as the packets pass through the node. When the verifier receives a packet, it uses its key(s) along with data found in the packet to validate whether the packet traversed the path correctly.

2. Conventions

Abbreviations used in this document:

MTU: Maximum Transmit Unit

SR: Segment Routing

NSH: Network Service Header

SFC: Service Function Chain

POT: Proof of Transit

POT-profile: Proof of Transit Profile that has the necessary data for nodes to participate in proof of transit

3. Proof of Transit

This section discusses methods and algorithms to provide for a "proof of transit" for packets traversing a specific path. A path which is to be verified consists of a set of nodes. Transit of the data packets through those nodes is to be proven. Besides the nodes, the setup also includes a Controller that creates secrets and secrets shares and configures the nodes for POT operations.

The methods how traffic is identified and associated to a specific path is outside the scope of this document. Identification could be done using a filter (e.g., 5-tupel classifier), or an identifier which is already present in the packet (e.g., path or service identifier, flow-label, etc.).

The solution approach is detailed in two steps. Initially the concept of the approach is explained. This concept is then further refined to make it operationally feasible.

3.1. Basic Idea

The method relies on adding POT data to all packets that traverse a path. The added POT data allows a verifying node (egress node) to check whether a packet traversed the identified set of nodes on a path correctly or not. Security mechanisms are natively built into the generation of the POT data to protect against misuse (i.e. configuration mistakes, malicious administrators playing tricks with routing, capturing, spoofing and replaying packets). The mechanism for POT leverages "Shamir's secret sharing scheme" [SSS].

Shamir's secret sharing base idea: A polynomial (represented by its co-efficients) is chosen as a secret by the controller. A polynomial represents a curve. A set of well defined points on the curve are needed to construct the polynomial. Each point of the polynomial is called "share" of the secret. A single secret is associated with a particular set of nodes, which typically represent the path, to be verified. Shares of the single secret (i.e., points on the curve) are securely distributed from a Controller to the network nodes. Nodes use their respective share to update a cumulative value in the POT data of each packet. Only a verifying node has access to the complete secret. The verifying node validates the correctness of the received POT data by reconstructing the curve.

The polynomial cannot be constructed if any of the points are missed or tampered. Per Shamir's Secret Sharing Scheme, any lesser points means one or more nodes are missed. Details of the precise configuration needed for achieving security are discussed further below.

While applicable in theory, a vanilla approach based on Shamir's secret sharing could be easily attacked. If the same polynomial is reused for every packet for a path a passive attacker could reuse the value. As a consequence, one could consider creating a different polynomial per packet. Such an approach would be operationally complex. It would be complex to configure and recycle so many curves and their respective points for each node. Rather than using a single polynomial, two polynomials are used for the solution approach: A secret polynomial which is kept constant, and a per-packet polynomial which is public. Operations are performed on the sum of those two polynomials - creating a third polynomial which is secret and per packet.

3.2. Solution Approach

Solution approach: The overall algorithm uses two polynomials: POLY-1 and POLY-2. POLY-1 is secret and constant. Each node gets a point on POLY-1 at setup-time and keeps it secret. POLY-2 is public, random and per packet. Each node generates a point on POLY-2 each time a packet crosses it. Each node then calculates (point on POLY-1 + point on POLY-2) to get a (point on POLY-3) and passes it to verifier by adding it to each packet. The verifier constructs POLY-3 from the points given by all the nodes and cross checks whether POLY-3 = POLY-1 + POLY-2. Only the verifier knows POLY-1. The solution leverages finite field arithmetic in a field of size "prime number".

Detailed algorithms are discussed next. A simple example is discussed in Section 3.3.

3.2.1. Setup

A controller generates a first polynomial (POLY-1) of degree k and k+1 points on the polynomial. The constant coefficient of POLY-1 is considered the SECRET. The non-constant coefficients are used to generate the Lagrange Polynomial Constants (LPC). Each of the k nodes (including verifier) are assigned a point on the polynomial i.e., shares of the SECRET. The verifier is configured with the SECRET. The Controller also generates coefficients (except the constant coefficient, called "RND", which is changed on a per packet basis) of a second polynomial POLY-2 of the same degree. Each node is configured with the LPC of POLY-2. Note that POLY-2 is public.

3.2.2. In Transit

For each packet, the source node generates a random number (RND). It is considered as the constant coefficient for POLY-2. A cumulative value (CML) is initialized to 0. Both RND, CML are carried as within the packet POT data. As the packet visits each node, the RND is retrieved from the packet and the respective share of POLY-2 is calculated. Each node calculates (Share(POLY-1)+Share(POLY-2)) and CML is updated with this sum. This step is performed by each node until the packet completes the path. The verifier also performs the step with its respective share.

3.2.3. Verification

The verifier cross checks whether CML = SECRET + RND. If this matches then the packet traversed the specified set of nodes in the path. This is due to the additive homomorphic property of Shamir's Secret Sharing scheme.

3.3. Example for Illustration

This section shows a simple example to illustrate step by step the approach described above.

3.3.1. Basic Version

Assumption: We like to verify that packets pass through 3 nodes. Consequently we need a polynomial of degree 2.

Choices: Prime = 53. POLY-1(x) = (3x^2 + 3x + 10) mod 53. The secret to be re-constructed is the constant coefficient of POLY-1, i.e., SECRET=10. It is important to note that all operations are done over a finite field (i.e., modulo prime).

3.3.1.1. Secret Shares

The shares of the secret are the points on POLY-1 chosen for the 3 nodes. Here we use x0=2, x1=4, x2=5.

```
POLY-1(2) = 28 => (x0,y0) = (2,28)

POLY-1(4) = 17 => (x1,y1) = (4,17)

POLY-1(5) = 47 => (x2,y2) = (5,47)
```

The three points above are the points on the curve which are considered the shares of the secret. They are assigned to three nodes respectively and are kept secret.

3.3.1.2. Lagrange Polynomials

Lagrange basis polynomials (or Lagrange polynomials) are used for polynomial interpolation. For a given set of points on the curve Lagrange polynomials (as defined below) are used to reconstruct the curve and thus reconstruct the complete secret.

```
l0(x) = (((x-x1)/(x0-x1))*((x-x2)/x0-x2))) mod 53 =
(((x-4)/(2-4))*((x-5)/2-5))) mod 53 =
(10/3 - 3x/2 + (1/6)x^2) mod 53

l1(x) = (((x-x0)/(x1-x0))*((x-x2)/x1-x2))) mod 53 =
(-5 + 7x/2 - (1/2)x^2) mod 53

l2(x) = (((x-x0)/(x2-x0))*((x-x1)/x2-x1))) mod 53 =
(8/3 - 2 + (1/3)x^2) mod 53
```

3.3.1.3. LPC Computation

Since x0=2, x1=4, x2=5 are chosen points. Given that computations are done over a finite arithmetic field ("modulo a prime number"), the Lagrange basis polynomial constants (LPC) are computed modulo 53. The Lagrange polynomial constant (LPC) would be 10/3 , -5 , 8/3.

```
LPC(x0) = (10/3) mod 53 = 21

LPC(x1) = (-5) mod 53 = 48
```

LPC(x2) = (8/3) mod 53 = 38

For a general way to compute the modular multiplicative inverse, see e.g., the Euclidean algorithm.

3.3.1.4. Reconstruction

Reconstruction of the polynomial is well defined as

POLY1(x) = l0(x)*y0 + l1(x)*y1 + l2(x)*y2.

Subsequently, the SECRET, which is the constant coefficient of POLY1(x) can be computed as below

SECRET = (y0*LPC(l0)+y1*LPC(l1)+y2*LPC(l2)) mod 53.

The secret can be easily reconstructed using the y-values and the LPC:

SECRET = (y0*LPC(l0) + y1*LPC(l1) + y2*LPC(l2)) mod 53 = mod (28 * 21 + 17 * 48 + 47 * 38) mod 53 = 3190 mod 53 = 10.

One observes that the secret reconstruction can easily be performed cumulatively hop by hop. CML represents the cumulative value. It is the POT data in the packet that is updated at each hop with the node’s respective (yi*LPC(i)), where i is their respective value.

3.3.1.5. Verification

Upon completion of the path, the resulting CML is retrieved by the verifier from the packet POT data. Recall that verifier is preconfigured with the original SECRET. It is cross checked with the CML by the verifier. Subsequent actions based on the verification failing or succeeding could be taken as per the configured policies.

3.3.2. Enhanced Version

As observed previously, the vanilla algorithm that involves a single secret polynomial is not secure. We enhance the solution with usage of a random second polynomial chosen per packet.

3.3.2.1. Random Polynomial

Let the second polynomial POLY-2 be (RND + 7x + 10 x^2). RND is a random number and is generated for each packet. Note that POLY-2 is public and need not be kept secret. The nodes can be pre-configured with the non-constant coefficients (for example, 7 and 10 in this case could be configured through the Controller on each node).

3.3.2.2. Reconstruction

Recall that each node is preconfigured with their respective Share(POLY-1). Each node calculates its respective Share(POLY-2) using the RND value retrieved from the packet. The CML reconstruction is enhanced as below. At every node, CML is updated as

CML = CML+(((Share(POLY-1)+ Share(POLY-2)) * LPC) mod Prime.

Lets observe the packet level transformations in detail. For the example packet here, let the value RND be 45. Thus POLY-2 would be (45 + 7x + 10x^2).

The shares that could be generated are (2,46), (4,21), (5,12).

At source: The fields RND = 45. CML = 0.

At node-1 (x0): Respective share of POLY-2 is generated i.e (2,46) because share index of node-1 is 2.

CML = 0 + ((28 + 46)* 21) mod 53 = 17.

At node-2 (x1): Respective share of POLY-2 is generated i.e (4,21) because share index of node-2 is 4.

CML = 17 + ((17 + 21)*48) mod 53 = 17 + 22 = 39.

At node-3 (x2), which is also the verifier: The respective share of POLY-2 is generated i.e (5,12) because the share index of the verifier is 12.

CML = 39 + ((47 + 12)*38) mod 53 = 39 + 16 = 55 mod 53 = 2

The verification using CML is discussed in next section.

3.3.2.3. Verification

As shown in the above example, for final verification, the verifier compares:

VERIFY = (SECRET + RND) mod Prime, with Prime = 53 here.

VERIFY = (RND-1 + RND-2) mod Prime = (10 + 45) mod 53 = 2.

Since VERIFY = CML the packet is proven to have gone through nodes 1, 2, and 3.

3.4. Operational Aspects

To operationalize this scheme, a central controller is used to generate the necessary polynomials, the secret share per node, the prime number, etc. and distributing the data to the nodes participating in proof of transit. The identified node that performs the verification is provided with the verification key. The information provided from the Controller to each of the nodes participating in proof of transit is referred to as a proof of transit profile (POT-profile).

To optimize the overall data amount of exchanged and the processing at the nodes the following optimizations are performed:

1. The points (x,y) for each of the nodes on the public and private polynomials are picked such that the x component of the points match. This lends to the LPC values which are used to calculate the cumulative value CML to be constant. Note that the LPC are only depending on the x components. The can be computed at the controller and communicated to the nodes. Otherwise, one would need to distributed the x components to all the nodes.

2. A pre-evaluated portion of the public polynomial for each of the nodes is calculated and added to the POT-profile. Without this all the coefficients of the public polynomial had to be added to the POT profile and each node had to evaluate them.

3. To provide flexibility on the size of the cumulative and random numbers carried in the POT data a field to indicate this is shared and interpreted at the nodes.

4. Sizing the Data for Proof of Transit

Proof of transit requires transport of two data records in every packet that should be verified:

1. RND: Random number (the constant coefficient of public polynomial)

2. CML: Cumulative

The size of the data records determines how often a new set of polynomials would need to be created. At maximum, the largest RND number that can be represented with a given number of bits determines the number of unique polynomials POLY-2 that can be created. The table below shows the maximum interval for how long a single set of polynomials could last for a variety of bit rates and RND sizes: When choosing 64 bits for RND and CML data records, the time between a renewal of secrets could be as long as 3,100 years, even when running at 100 Gbps.

Transfer rate	Secret/RND size	Max # of packets	Time RND lasts
1 Gbps	64	2^64 = approx. 2*10^19	approx. 310,000 years
10 Gbps	64	2^64 = approx. 2*10^19	approx. 31,000 years
100 Gbps	64	2^64 = approx. 2*10^19	approx. 3,100 years
1 Gbps	32	2^32 = approx. 4*10^9	2,200 seconds
10 Gbps	32	2^32 = approx. 4*10^9	220 seconds
100 Gbps	32	2^32 = approx. 4*10^9	22 seconds

Table assumes 64 octet packets

Table 1: Proof of transit data sizing

5. Node Configuration

A POT system consists of a number of nodes that participate in POT and a Controller, which serves as a control and configuration entity. The Controller is to create the required parameters (polynomials, prime number, etc.) and communicate those to the nodes. The sum of all parameters for a specific node is referred to as "POT-profile". This document does not define a specific protocol to be used between Controller and nodes. It only defines the procedures and the associated YANG data model.

5.1. Procedure

The Controller creates new POT-profiles at a constant rate and communicates the POT-profile to the nodes. The controller labels a POT-profile "even" or "odd" and the Controller cycles between "even" and "odd" labeled profiles. The rate at which the POT-profiles are communicated to the nodes is configurable and is more frequent than the speed at which a POT-profile is "used up" (see table above). Once the POT-profile has been successfully communicated to all nodes (e.g., all Netconf transactions completed, in case Netconf is used as a protocol), the controller sends an "enable POT-profile" request to the ingress node.

All nodes maintain two POT-profiles (an even and an odd POT-profile): One POT-profile is currently active and in use; one profile is standby and about to get used. A flag in the packet is indicating whether the odd or even POT-profile is to be used by a node. This is to ensure that during profile change the service is not disrupted. If the "odd" profile is active, the Controller can communicate the "even" profile to all nodes. Only if all the nodes have received the POT-profile, the Controller will tell the ingress node to switch to the "even" profile. Given that the indicator travels within the packet, all nodes will switch to the "even" profile. The "even" profile gets active on all nodes and nodes are ready to receive a new "odd" profile.

Unless the ingress node receives a request to switch profiles, it’ll continue to use the active profile. If a profile is "used up" the ingress node will recycle the active profile and start over (this could give rise to replay attacks in theory - but with 2^32 or 2^64 packets this isn’t really likely in reality).

5.2. YANG Model

This section defines that YANG data model for the information exchange between the Controller and the nodes.

```
module ietf-pot-profile {

  yang-version 1;

  namespace "urn:ietf:params:xml:ns:yang:ietf-pot-profile";

  prefix ietf-pot-profile;

  organization "IETF xxx Working Group";

  contact "";

  description "This module contains a collection of YANG
               definitions for proof of transit configuration
               parameters. The model is meant for proof of
               transit and is targeted for communicating the
               POT-profile between a controller and nodes
               participating in proof of transit.";

  revision 2016-06-15 {
    description
      "Initial revision.";
    reference
      "";
```

```
     }

     typedef profile-index-range {
       type int32 {
         range "0 .. 1";
       }
       description
         "Range used for the profile index. Currently restricted to
          0 or 1 to identify the odd or even profiles.";
     }


     grouping pot-profile {
       description "A grouping for proof of transit profiles.";
       list pot-profile-list {
         key "pot-profile-index";
         ordered-by user;
         description "A set of pot profiles.";

         leaf pot-profile-index {
           type profile-index-range;
           mandatory true;
           description
             "Proof of transit profile index.";
         }

         leaf prime-number {
           type uint64;
           mandatory true;
           description
             "Prime number used for module math computation";
         }

         leaf secret-share {
           type uint64;
           mandatory true;
           description
             "Share of the secret of polynomial 1 used in computation";
         }

         leaf public-polynomial {
           type uint64;
           mandatory true;
           description
             "Pre evaluated Public polynomial";
         }

         leaf lpc {
```

```
          type uint64;
          mandatory true;
          description
            "Lagrange Polynomial Coefficient";
        }

        leaf validator {
          type boolean;
          default "false";
          description
            "True if the node is a verifier node";
        }

        leaf validator-key {
          type uint64;
          description
            "Secret key for validating the path, constant of poly 1";
        }

        leaf bitmask {
          type uint64;
          default 4294967295;
          description
            "Number of bits as mask used in controlling the size of the
             random value generation. 32-bits of mask is default.";
        }
      }
    }

    container pot-profiles {
      description "A group of proof of transit profiles.";

      list pot-profile-set {
        key "pot-profile-name";
        ordered-by user;
        description
          "Set of proof of transit profiles that group parameters
           required to classify and compute proof of transit
           metadata at a node";

        leaf pot-profile-name {
          type string;
          mandatory true;
          description
            "Unique identifier for each proof of transit profile";
        }

        leaf active-profile-index {
```

```
           type profile-index-range;
           description
             "Proof of transit profile index that is currently active.
              Will be set in the first hop of the path or chain.
              Other nodes will not use this field.";
         }

         uses pot-profile;
       }
     /*** Container: end ***/
     }
   /*** module: end ***/
   }
```

6. IANA Considerations

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

7. Manageability Considerations

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

8. Security Considerations

Different security requirements achieved by the solution approach are discussed here.

8.1. Proof of Transit

Proof of correctness and security of the solution approach is per Shamir's Secret Sharing Scheme [SSS]. Cryptographically speaking it achieves information-theoretic security i.e., it cannot be broken by an attacker even with unlimited computing power. As long as the below conditions are met it is impossible for an attacker to bypass one or multiple nodes without getting caught.

- o If there are k+1 nodes in the path, the polynomials (POLY-1, POLY-2) should be of degree k. Also k+1 points of POLY-1 are chosen and assigned to each node respectively. The verifier can re-construct the k degree polynomial (POLY-3) only when all the points are correctly retrieved.

- o The Shares of the SECRET (i.e., points on POLY-1) are kept secret by individual nodes.

An attacker bypassing a few nodes will miss adding a respective point on POLY-1 to corresponding point on POLY-2 , thus the verifier cannot construct POLY-3 for cross verification.

8.2. Anti Replay

A passive attacker observing CML values across nodes (i.e., as the packets entering and leaving), cannot perform differential analysis to construct the points on POLY-1 as the operations are done modulo prime. The solution approach is flexible, one could use different points on POLY-1 or different polynomials as POLY-1 across different paths, traffic profiles or service chains.

Doing differential analysis across packets could be mitigated with POLY-2 being be random. Further an attacker could reuse a set of RND and all the intermediate CML values to bypass certain nodes in later packets. Such attacks could be avoided by carefully choosing POLY-2 as a timestamp concatenated with a random string. The verifier could use the timestamp to mitigate reuse within a time window.

8.3. Anti Tampering

An active attacker could not insert any arbitrary value for CML. This would subsequently fail the reconstruction of the POLY-3. Also an attacker could not update the CML with a previously observed value. This could subsequently be detected by using timestamps within the RND value as discussed above.

8.4. Recycling

The solution approach is flexible for recycling long term secrets like POLY-1. All the nodes could be periodically updated with shares of new SECRET as best practice. The table above could be consulted for refresh cycles (see Section 4).

8.5. Redundant Nodes and Failover

A "node" or "service" in terms of POT can be implemented by one or multiple physical entities. In case of multiple physical entities (e.g., for load-balancing, or business continuity situations - consider for example a set of firewalls), all physical entities which are implementing the same POT node are given that same share of the secret. This makes multiple physical entities represent the same POT node from an algorithm perspective.

8.6. Controller Operation

The Controller needs to be secured given that it creates and holds the secrets, as need to be the nodes. The communication between Controller and the nodes also needs to be secured. As secure communication protocol such as for example Netconf over SSH should be chosen for Controller to node communication.

The Controller only interacts with the nodes during the initial configuration and thereafter at regular intervals at which the operator chooses to switch to a new set of secrets. In case 64 bits are used for the data-records "CML" and "RND" which are carried within the data packet, the regular intervals are expected to be quite long (e.g., at 100 Gbps, a profile would only be used up after 3100 years) - see Section 4 above, thus even a "headless" operation without a Controller can be considered feasible. In such a case, the Controller would only be used for the initial configuration of the POT-profiles.

8.7. Verification Scope

The POT solution defined in this document verifies that a data-packet traversed or transited a specific set of nodes. From an algorithm perspective, a "node" is an abstract entity. It could be represented by one or multiple physical or virtual network devices, or is could be a component within a networking device or system. The latter would be the case if a forwarding path within a device would need to be securely verified.

8.7.1. Node Ordering

POT using Shamir's secret sharing scheme as discussed in this document provides for a means to verify that a set of nodes has been visited by a data packet. It does not verify the order in which the data packet visited the nodes. In case the order in which a data packet traversed a particular set of nodes needs to be verified as well, alternate schemes that e.g., rely on nested encryption could to be considered.

8.7.2. Stealth Nodes

The POT approach discussed in this document is to prove that a data packet traversed a specific set of "nodes". This set could be all nodes within a path, but could also be a subset of nodes in a path. Consequently, the POT approach isn't suited to detect whether "stealth" nodes which do not participate in proof-of-transit have been inserted into a path.

9. 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.

10. Normative References

[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>.

[SSS] "Shamir’s Secret Sharing", <https://en.wikipedia.org/wiki/Shamir%27s_Secret_Sharing>.

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

Sashank Dara
Cisco Systems, Inc.
Cessna Business Park, Sarjapura Marathalli Outer Ring Road
BANGALORE, Bangalore, KARNATAKA 560 087
INDIA

Email: sadara@cisco.com

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

Email: cpignata@cisco.com

Network Working Group
Internet-Draft
Intended status: Experimental
Expires: November 8, 2018

F. Brockners
S. Bhandari
S. Dara
C. Pignataro
Cisco
J. Leddy
Comcast
S. Youell
JPMC
D. Mozes

T. Mizrahi
Marvell
May 7, 2018

Proof of Transit
draft-brockners-proof-of-transit-05

Abstract

Several technologies such as Traffic Engineering (TE), Service Function Chaining (SFC), and policy based routing are used to steer traffic through a specific, user-defined path. This document defines mechanisms to securely prove that traffic transited said defined path. These mechanisms allow to securely verify whether, within a given path, all packets traversed all the nodes that they are supposed to visit.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction 3
2. Conventions 4
3. Proof of Transit 5
3.1. Basic Idea 5
3.2. Solution Approach 6
3.2.1. Setup 7
3.2.2. In Transit 7
3.2.3. Verification 7
3.3. Illustrative Example 7
3.3.1. Basic Version 7
3.3.1.1. Secret Shares 8
3.3.1.2. Lagrange Polynomials 8
3.3.1.3. LPC Computation 8
3.3.1.4. Reconstruction 9
3.3.1.5. Verification 9
3.3.2. Enhanced Version 9
3.3.2.1. Random Polynomial 9
3.3.2.2. Reconstruction 10
3.3.2.3. Verification 10
3.3.3. Final Version 11
3.4. Operational Aspects 11
3.5. Alternative Approach 12
3.5.1. Basic Idea 12
3.5.2. Pros 12
3.5.3. Cons 12
4. Sizing the Data for Proof of Transit 12
5. Node Configuration 13
5.1. Procedure 14
5.2. YANG Model 14
6. IANA Considerations 17
7. Manageability Considerations 17

8. Security Considerations 17
8.1. Proof of Transit 18
8.2. Cryptanalysis 18
8.3. Anti-Replay 19
8.4. Anti-Preplay 19
8.5. Anti-Tampering 20
8.6. Recycling 20
8.7. Redundant Nodes and Failover 20
8.8. Controller Operation 20
8.9. Verification Scope 21
8.9.1. Node Ordering 21
8.9.2. Stealth Nodes 21
9. Acknowledgements 21
10. References 21
10.1. Normative References 21
10.2. Informative References 22
Authors' Addresses 22

1. Introduction

Several deployments use Traffic Engineering, policy routing, Segment Routing (SR), and Service Function Chaining (SFC) [RFC7665] to steer packets through a specific set of nodes. In certain cases, regulatory obligations or a compliance policy require operators to prove that all packets that are supposed to follow a specific path are indeed being forwarded across and exact set of pre-determined nodes.

If a packet flow is supposed to go through a series of service functions or network nodes, it has to be proven that indeed all packets of the flow followed the path or service chain or collection of nodes specified by the policy. In case some packets of a flow weren't appropriately processed, a verification device should determine the policy violation and take corresponding actions corresponding to the policy (e.g., drop or redirect the packet, send an alert etc.) In today's deployments, the proof that a packet traversed a particular path or service chain is typically delivered in an indirect way: Service appliances and network forwarding are in different trust domains. Physical hand-off-points are defined between these trust domains (i.e. physical interfaces). Or in other terms, in the "network forwarding domain" things are wired up in a way that traffic is delivered to the ingress interface of a service appliance and received back from an egress interface of a service appliance. This "wiring" is verified and then trusted upon. The evolution to Network Function Virtualization (NFV) and modern service chaining concepts (using technologies such as Locator/ID Separation Protocol (LISP), Network Service Header (NSH), Segment Routing (SR), etc.) blurs the line between the different trust domains, because the hand-off-points are no longer clearly defined physical interfaces, but are virtual interfaces. As a consequence, different trust layers should not to be mixed in the same device. For an NFV scenario a different type of proof is required. Offering a proof that a packet indeed traversed a specific set of service functions or nodes allows operators to evolve from the above described indirect methods of proving that packets visit a predetermined set of nodes.

The solution approach presented in this document is based on a small portion of operational data added to every packet. This "in-situ" operational data is also referred to as "proof of transit data", or POT data. The POT data is updated at every required node and is used to verify whether a packet traversed all required nodes. A particular set of nodes "to be verified" is either described by a set of secret keys, or a set of shares of a single secret. Nodes on the path retrieve their individual keys or shares of a key (using for e.g., Shamir's Secret Sharing scheme) from a central controller. The complete key set is only known to the controller and a verifier node, which is typically the ultimate node on a path that performs verification. Each node in the path uses its secret or share of the secret to update the POT data of the packets as the packets pass through the node. When the verifier receives a packet, it uses its key(s) along with data found in the packet to validate whether the packet traversed the path correctly.

2. Conventions

Abbreviations used in this document:

HMAC: Hash based Message Authentication Code. For example, HMAC-SHA256 generates 256 bits of MAC

IOAM: In-situ Operations, Administration, and Maintenance

LISP: Locator/ID Separation Protocol

LPC: Lagrange Polynomial Constants

MTU: Maximum Transmit Unit

NFV: Network Function Virtualization

NSH: Network Service Header

POT: Proof of Transit

POT-profile: Proof of Transit Profile that has the necessary data for nodes to participate in proof of transit

RND: Random Bits generated per packet. Packet fields that donot change during the traversal are given as input to HMAC-256 algorithm. A minimum of 32 bits (left most) need to be used from the output if RND is used to verify the packet integrity. This is a standard recommendation by NIST.

SEQ_NO: Sequence number initialized to a predefined constant. This is used in concatenation with RND bits to mitigate different attacks discussed later.

SFC: Service Function Chain

SR: Segment Routing

3. Proof of Transit

This section discusses methods and algorithms to provide for a "proof of transit" for packets traversing a specific path. A path which is to be verified consists of a set of nodes. Transit of the data packets through those nodes is to be proven. Besides the nodes, the setup also includes a Controller that creates secrets and secrets shares and configures the nodes for POT operations.

The methods how traffic is identified and associated to a specific path is outside the scope of this document. Identification could be done using a filter (e.g., 5-tuple classifier), or an identifier which is already present in the packet (e.g., path or service identifier, NSH Service Path Identifier (SPI), flow-label, etc.)

The solution approach is detailed in two steps. Initially the concept of the approach is explained. This concept is then further refined to make it operationally feasible.

3.1. Basic Idea

The method relies on adding POT data to all packets that traverse a path. The added POT data allows a verifying node (egress node) to check whether a packet traversed the identified set of nodes on a path correctly or not. Security mechanisms are natively built into the generation of the POT data to protect against misuse (i.e. configuration mistakes, malicious administrators playing tricks with routing, capturing, spoofing and replaying packets). The mechanism for POT leverages "Shamir's Secret Sharing" scheme [SSS].

Shamir's secret sharing base idea: A polynomial (represented by its coefficients) is chosen as a secret by the controller. A polynomial represents a curve. A set of well-defined points on the curve are needed to construct the polynomial. Each point of the polynomial is called "share" of the secret. A single secret is associated with a particular set of nodes, which typically represent the path, to be verified. Shares of the single secret (i.e., points on the curve) are securely distributed from a Controller to the network nodes. Nodes use their respective share to update a cumulative value in the POT data of each packet. Only a verifying node has access to the complete secret. The verifying node validates the correctness of the received POT data by reconstructing the curve.

The polynomial cannot be constructed if any of the points are missed or tampered. Per Shamir's Secret Sharing Scheme, any lesser points means one or more nodes are missed. Details of the precise configuration needed for achieving security are discussed further below.

While applicable in theory, a vanilla approach based on Shamir's secret sharing could be easily attacked. If the same polynomial is reused for every packet for a path a passive attacker could reuse the value. As a consequence, one could consider creating a different polynomial per packet. Such an approach would be operationally complex. It would be complex to configure and recycle so many curves and their respective points for each node. Rather than using a single polynomial, two polynomials are used for the solution approach: A secret polynomial which is kept constant, and a per-packet polynomial which is public. Operations are performed on the sum of those two polynomials - creating a third polynomial which is secret and per packet.

3.2. Solution Approach

Solution approach: The overall algorithm uses two polynomials: POLY-1 and POLY-2. POLY-1 is secret and constant. Each node gets a point on POLY-1 at setup-time and keeps it secret. POLY-2 is public, random and per packet. Each node generates a point on POLY-2 each time a packet crosses it. Each node then calculates (point on POLY-1 + point on POLY-2) to get a (point on POLY-3) and passes it to verifier by adding it to each packet. The verifier constructs POLY-3 from the points given by all the nodes and cross checks whether POLY-3 = POLY-1 + POLY-2. Only the verifier knows POLY-1. The solution leverages finite field arithmetic in a field of size "prime number".

Detailed algorithms are discussed next. A simple example is discussed in Section 3.3.

3.2.1. Setup

A controller generates a first polynomial (POLY-1) of degree k and k+1 points on the polynomial. The constant coefficient of POLY-1 is considered the SECRET. The non-constant coefficients are used to generate the Lagrange Polynomial Constants (LPC). Each of the k nodes (including verifier) are assigned a point on the polynomial i.e., shares of the SECRET. The verifier is configured with the SECRET. The Controller also generates coefficients (except the constant coefficient, called "RND", which is changed on a per packet basis) of a second polynomial POLY-2 of the same degree. Each node is configured with the LPC of POLY-2. Note that POLY-2 is public.

3.2.2. In Transit

For each packet, the ingress node generates a random number (RND). It is considered as the constant coefficient for POLY-2. A cumulative value (CML) is initialized to 0. Both RND, CML are carried as within the packet POT data. As the packet visits each node, the RND is retrieved from the packet and the respective share of POLY-2 is calculated. Each node calculates (Share(POLY-1) + Share(POLY-2)) and CML is updated with this sum. This step is performed by each node until the packet completes the path. The verifier also performs the step with its respective share.

3.2.3. Verification

The verifier cross checks whether CML = SECRET + RND. If this matches then the packet traversed the specified set of nodes in the path. This is due to the additive homomorphic property of Shamir's Secret Sharing scheme.

3.3. Illustrative Example

This section shows a simple example to illustrate step by step the approach described above.

3.3.1. Basic Version

Assumption: It is to be verified whether packets passed through 3 nodes. A polynomial of degree 2 is chosen for verification.

Choices: Prime = 53. POLY-1(x) = (3x^2 + 3x + 10) mod 53. The secret to be re-constructed is the constant coefficient of POLY-1, i.e., SECRET=10. It is important to note that all operations are done over a finite field (i.e., modulo prime).

3.3.1.1. Secret Shares

The shares of the secret are the points on POLY-1 chosen for the 3 nodes. For example, let x0=2, x1=4, x2=5.

```
POLY-1(2) = 28 => (x0, y0) = (2, 28)

POLY-1(4) = 17 => (x1, y1) = (4, 17)

POLY-1(5) = 47 => (x2, y2) = (5, 47)
```

The three points above are the points on the curve which are considered the shares of the secret. They are assigned to three nodes respectively and are kept secret.

3.3.1.2. Lagrange Polynomials

Lagrange basis polynomials (or Lagrange polynomials) are used for polynomial interpolation. For a given set of points on the curve Lagrange polynomials (as defined below) are used to reconstruct the curve and thus reconstruct the complete secret.

```
l0(x) = (((x-x1) / (x0-x1)) * ((x-x2)/x0-x2))) mod 53 =
(((x-4) / (2-4)) * ((x-5)/2-5))) mod 53 =
(10/3 - 3x/2 + (1/6)x^2) mod 53

l1(x) = (((x-x0) / (x1-x0)) * ((x-x2)/x1-x2))) mod 53 =
(-5 + 7x/2 - (1/2)x^2) mod 53

l2(x) = (((x-x0) / (x2-x0)) * ((x-x1)/x2-x1))) mod 53 =
(8/3 - 2 + (1/3)x^2) mod 53
```

3.3.1.3. LPC Computation

Since x0=2, x1=4, x2=5 are chosen points. Given that computations are done over a finite arithmetic field ("modulo a prime number"), the Lagrange basis polynomial constants are computed modulo 53. The Lagrange Polynomial Constant (LPC) would be 10/3 , -5 , 8/3.

```
LPC(x0) = (10/3) mod 53 = 21

LPC(x1) = (-5) mod 53 = 48

LPC(x2) = (8/3) mod 53 = 38
```

For a general way to compute the modular multiplicative inverse, see e.g., the Euclidean algorithm.

3.3.1.4. Reconstruction

Reconstruction of the polynomial is well-defined as

POLY1(x) = l0(x) * y0 + l1(x) * y1 + l2(x) * y2

Subsequently, the SECRET, which is the constant coefficient of POLY1(x) can be computed as below

SECRET = (y0*LPC(l0)+y1*LPC(l1)+y2*LPC(l2)) mod 53

The secret can be easily reconstructed using the y-values and the LPC:

SECRET = (y0*LPC(l0) + y1*LPC(l1) + y2*LPC(l2)) mod 53 = mod (28 * 21 + 17 * 48 + 47 * 38) mod 53 = 3190 mod 53 = 10

One observes that the secret reconstruction can easily be performed cumulatively hop by hop. CML represents the cumulative value. It is the POT data in the packet that is updated at each hop with the node's respective (yi*LPC(i)), where i is their respective value.

3.3.1.5. Verification

Upon completion of the path, the resulting CML is retrieved by the verifier from the packet POT data. Recall that verifier is preconfigured with the original SECRET. It is cross checked with the CML by the verifier. Subsequent actions based on the verification failing or succeeding could be taken as per the configured policies.

3.3.2. Enhanced Version

As observed previously, the vanilla algorithm that involves a single secret polynomial is not secure. Therefore, the solution is further enhanced with usage of a random second polynomial chosen per packet.

3.3.2.1. Random Polynomial

Let the second polynomial POLY-2 be (RND + 7x + 10 x^2). RND is a random number and is generated for each packet. Note that POLY-2 is public and need not be kept secret. The nodes can be pre-configured with the non-constant coefficients (for example, 7 and 10 in this case could be configured through the Controller on each node). So precisely only RND value changes per packet and is public and the rest of the non-constant coefficients of POLY-2 kept secret.

3.3.2.2. Reconstruction

Recall that each node is preconfigured with their respective Share(POLY-1). Each node calculates its respective Share(POLY-2) using the RND value retrieved from the packet. The CML reconstruction is enhanced as below. At every node, CML is updated as

CML = CML+(((Share(POLY-1)+ Share(POLY-2)) * LPC) mod Prime

Let us observe the packet level transformations in detail. For the example packet here, let the value RND be 45. Thus POLY-2 would be (45 + 7x + 10x^2).

The shares that could be generated are (2, 46), (4, 21), (5, 12).

At ingress: The fields RND = 45. CML = 0.

At node-1 (x0): Respective share of POLY-2 is generated i.e., (2, 46) because share index of node-1 is 2.

CML = 0 + ((28 + 46)* 21) mod 53 = 17

At node-2 (x1): Respective share of POLY-2 is generated i.e., (4, 21) because share index of node-2 is 4.

CML = 17 + ((17 + 21)*48) mod 53 = 17 + 22 = 39

At node-3 (x2), which is also the verifier: The respective share of POLY-2 is generated i.e., (5, 12) because the share index of the verifier is 12.

CML = 39 + ((47 + 12)*38) mod 53 = 39 + 16 = 55 mod 53 = 2

The verification using CML is discussed in next section.

3.3.2.3. Verification

As shown in the above example, for final verification, the verifier compares:

VERIFY = (SECRET + RND) mod Prime, with Prime = 53 here

VERIFY = (RND-1 + RND-2) mod Prime = (10 + 45) mod 53 = 2

Since VERIFY = CML the packet is proven to have gone through nodes 1, 2, and 3.

3.3.3. Final Version

The enhanced version of the protocol is still prone to replay and preplay attacks. An attacker could reuse the POT metadata for bypassing the verification. So additional measures using packet integrity checks (HMAC) and sequence numbers (SEQ_NO) are discussed later "Security Considerations" section.

3.4. Operational Aspects

To operationalize this scheme, a central controller is used to generate the necessary polynomials, the secret share per node, the prime number, etc. and distributing the data to the nodes participating in proof of transit. The identified node that performs the verification is provided with the verification key. The information provided from the Controller to each of the nodes participating in proof of transit is referred to as a proof of transit profile (POT-profile). Also note that the set of nodes for which the transit has to be proven are typically associated to a different trust domain than the verifier. Note that building the trust relationship between the Controller and the nodes is outside the scope of this document. Techniques such as those described in [I-D.ietf-anima-autonomic-control-plane] might be applied.

To optimize the overall data amount of exchanged and the processing at the nodes the following optimizations are performed:

1. The points (x, y) for each of the nodes on the public and private polynomials are picked such that the x component of the points match. This lends to the LPC values which are used to calculate the cumulative value CML to be constant. Note that the LPC are only depending on the x components. They can be computed at the controller and communicated to the nodes. Otherwise, one would need to distributed the x components to all the nodes.

2. A pre-evaluated portion of the public polynomial for each of the nodes is calculated and added to the POT-profile. Without this all the coefficients of the public polynomial had to be added to the POT profile and each node had to evaluate them. As stated before, the public portion is only the constant coefficient RND value, the pre-evaluated portion for each node should be kept secret as well.

3. To provide flexibility on the size of the cumulative and random numbers carried in the POT data a field to indicate this is shared and interpreted at the nodes.

3.5. Alternative Approach

In certain scenarios preserving the order of the nodes traversed by the packet may be needed. An alternative, "nested encryption" based approach is described here for preserving the order

3.5.1. Basic Idea

1. The controller provisions all the nodes with their respective secret keys.

2. The controller provisions the verifier with all the secret keys of the nodes.

3. For each packet, the ingress node generates a random number RND and encrypts it with its secret key to generate CML value

4. Each subsequent node on the path encrypts CML with their respective secret key and passes it along

5. The verifier is also provisioned with the expected sequence of nodes in order to verify the order

6. The verifier receives the CML, RND values, re-encrypts the RND with keys in the same order as expected sequence to verify.

3.5.2. Pros

Nested encryption approach retains the order in which the nodes are traversed.

3.5.3. Cons

1. Standard AES encryption would need 128 bits of RND, CML. This results in a 256 bits of additional overhead is added per packet

2. In hardware platforms that do not support native encryption capabilities like (AES-NI). This approach would have considerable impact on the computational latency

4. Sizing the Data for Proof of Transit

Proof of transit requires transport of two data fields in every packet that should be verified:

1. RND: Random number (the constant coefficient of public polynomial)

2. CML: Cumulative

The size of the data fields determines how often a new set of polynomials would need to be created. At maximum, the largest RND number that can be represented with a given number of bits determines the number of unique polynomials POLY-2 that can be created. The table below shows the maximum interval for how long a single set of polynomials could last for a variety of bit rates and RND sizes: When choosing 64 bits for RND and CML data fields, the time between a renewal of secrets could be as long as 3,100 years, even when running at 100 Gbps.

Transfer rate	Secret/RND size	Max # of packets	Time RND lasts
1 Gbps	64	2^64 = approx. 2*10^19	approx. 310,000 years
10 Gbps	64	2^64 = approx. 2*10^19	approx. 31,000 years
100 Gbps	64	2^64 = approx. 2*10^19	approx. 3,100 years
1 Gbps	32	2^32 = approx. 4*10^9	2,200 seconds
10 Gbps	32	2^32 = approx. 4*10^9	220 seconds
100 Gbps	32	2^32 = approx. 4*10^9	22 seconds

Table assumes 64 octet packets

Table 1: Proof of transit data sizing

5. Node Configuration

A POT system consists of a number of nodes that participate in POT and a Controller, which serves as a control and configuration entity. The Controller is to create the required parameters (polynomials, prime number, etc.) and communicate those to the nodes. The sum of all parameters for a specific node is referred to as "POT-profile". This document does not define a specific protocol to be used between Controller and nodes. It only defines the procedures and the associated YANG data model.

5.1. Procedure

The Controller creates new POT-profiles at a constant rate and communicates the POT-profile to the nodes. The controller labels a POT-profile "even" or "odd" and the Controller cycles between "even" and "odd" labeled profiles. The rate at which the POT-profiles are communicated to the nodes is configurable and is more frequent than the speed at which a POT-profile is "used up" (see table above). Once the POT-profile has been successfully communicated to all nodes (e.g., all NETCONF transactions completed, in case NETCONF is used as a protocol), the controller sends an "enable POT-profile" request to the ingress node.

All nodes maintain two POT-profiles (an even and an odd POT-profile): One POT-profile is currently active and in use; one profile is standby and about to get used. A flag in the packet is indicating whether the odd or even POT-profile is to be used by a node. This is to ensure that during profile change the service is not disrupted. If the "odd" profile is active, the Controller can communicate the "even" profile to all nodes. Only if all the nodes have received the POT-profile, the Controller will tell the ingress node to switch to the "even" profile. Given that the indicator travels within the packet, all nodes will switch to the "even" profile. The "even" profile gets active on all nodes and nodes are ready to receive a new "odd" profile.

Unless the ingress node receives a request to switch profiles, it’ll continue to use the active profile. If a profile is "used up" the ingress node will recycle the active profile and start over (this could give rise to replay attacks in theory - but with 2^32 or 2^64 packets this isn’t really likely in reality).

5.2. YANG Model

This section defines that YANG data model for the information exchange between the Controller and the nodes.

```
<CODE BEGINS> file "ietf-pot-profile@2016-06-15.yang"
module ietf-pot-profile {

  yang-version 1;

  namespace "urn:ietf:params:xml:ns:yang:ietf-pot-profile";

  prefix ietf-pot-profile;

  organization "IETF xxx Working Group";
```

```
   contact "";

   description "This module contains a collection of YANG
                definitions for proof of transit configuration
                parameters. The model is meant for proof of
                transit and is targeted for communicating the
                POT-profile between a controller and nodes
                participating in proof of transit.";

   revision 2016-06-15 {
     description
       "Initial revision.";
     reference
       "";
   }

   typedef profile-index-range {
     type int32 {
       range "0 .. 1";
     }
     description
       "Range used for the profile index. Currently restricted to
        0 or 1 to identify the odd or even profiles.";
   }


   grouping pot-profile {
     description "A grouping for proof of transit profiles.";
     list pot-profile-list {
       key "pot-profile-index";
       ordered-by user;
       description "A set of pot profiles.";

       leaf pot-profile-index {
         type profile-index-range;
         mandatory true;
         description
           "Proof of transit profile index.";
       }

       leaf prime-number {
         type uint64;
         mandatory true;
         description
           "Prime number used for module math computation";
       }

       leaf secret-share {
```

```
            type uint64;
            mandatory true;
            description
              "Share of the secret of polynomial 1 used in computation";
          }

          leaf public-polynomial {
            type uint64;
            mandatory true;
            description
              "Pre evaluated Public polynomial";
          }

          leaf lpc {
            type uint64;
            mandatory true;
            description
              "Lagrange Polynomial Coefficient";
          }

          leaf validator {
            type boolean;
            default "false";
            description
              "True if the node is a verifier node";
          }

          leaf validator-key {
            type uint64;
            description
              "Secret key for validating the path, constant of poly 1";
          }

          leaf bitmask {
            type uint64;
            default 4294967295;
            description
              "Number of bits as mask used in controlling the size of the
               random value generation. 32-bits of mask is default.";
          }
        }
      }

      container pot-profiles {
        description "A group of proof of transit profiles.";

        list pot-profile-set {
          key "pot-profile-name";
```

```
         ordered-by user;
         description
           "Set of proof of transit profiles that group parameters
            required to classify and compute proof of transit
            metadata at a node";

         leaf pot-profile-name {
           type string;
           mandatory true;
           description
             "Unique identifier for each proof of transit profile";
         }

         leaf active-profile-index {
           type profile-index-range;
           description
             "Proof of transit profile index that is currently active.
              Will be set in the first hop of the path or chain.
              Other nodes will not use this field.";
         }

         uses pot-profile;
       }
     /*** Container: end ***/
     }
   /*** module: end ***/
   }
   <CODE ENDS>
```

6. IANA Considerations

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

7. Manageability Considerations

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

8. Security Considerations

Different security requirements achieved by the solution approach are discussed here.

8.1. Proof of Transit

Proof of correctness and security of the solution approach is per Shamir's Secret Sharing Scheme [SSS]. Cryptographically speaking it achieves information-theoretic security i.e., it cannot be broken by an attacker even with unlimited computing power. As long as the below conditions are met it is impossible for an attacker to bypass one or multiple nodes without getting caught.

- o If there are k+1 nodes in the path, the polynomials (POLY-1, POLY-2) should be of degree k. Also k+1 points of POLY-1 are chosen and assigned to each node respectively. The verifier can re-construct the k degree polynomial (POLY-3) only when all the points are correctly retrieved.

- o Precisely three values are kept secret by individual nodes. Share of SECRET (i.e. points on POLY-1), Share of POLY-2, LPC, P. Note that only constant coefficient, RND, of POLY-2 is public. x values and non-constant coefficient of POLY-2 are secret

An attacker bypassing a few nodes will miss adding a respective point on POLY-1 to corresponding point on POLY-2 , thus the verifier cannot construct POLY-3 for cross verification.

Also it is highly recommended that different polynomials should be used as POLY-1 across different paths, traffic profiles or service chains.

8.2. Cryptanalysis

A passive attacker could try to harvest the POT data (i.e., CML, RND values) in order to determine the configured secrets. Subsequently two types of differential analysis for guessing the secrets could be done.

- o Inter-Node: A passive attacker observing CML values across nodes (i.e., as the packets entering and leaving), cannot perform differential analysis to construct the points on POLY-1. This is because at each point there are four unknowns (i.e. Share(POLY-1), Share(Poly-2) LPC and prime number P) and three known values (i.e. RND, CML-before, CML-after).

- o Inter-Packets: A passive attacker could observe CML values across packets (i.e., values of PKT-1 and subsequent PKT-2), in order to predict the secrets. Differential analysis across packets could be mitigated using a good PRNG for generating RND. Note that if constant coefficient is a sequence number than CML values become quite predictable and the scheme would be broken.

8.3. Anti-Replay

A passive attacker could reuse a set of older RND and the intermediate CML values to bypass certain nodes in later packets. Such attacks could be avoided by carefully choosing POLY-2 as a (SEQ_NO + RND). For example, if 64 bits are being used for POLY-2 then first 16 bits could be a sequence number SEQ_NO and next 48 bits could be a random number.

Subsequently, the verifier could use the SEQ_NO bits to run classic anti-replay techniques like sliding window used in IPSEC. The verifier could buffer up to 2^16 packets as a sliding window. Packets arriving with a higher SEQ_NO than current buffer could be flagged legitimate. Packets arriving with a lower SEQ_NO than current buffer could be flagged as suspicious.

For all practical purposes in the rest of the document RND means SEQ_NO + RND to keep it simple.

The solution discussed in this memo does not currently mitigate replay attacks. An anti-replay mechanism may be included in future versions of the solution.

8.4. Anti-Preplay

An active attacker could try to perform a man-in-the-middle (MITM) attack by extracting the POT of PKT-1 and using it in PKT-2. Subsequently attacker drops the PKT-1 in order to avoid duplicate POT values reaching the verifier. If the PKT-1 reaches the verifier, then this attack is same as Replay attacks discussed before.

Preplay attacks are possible since the POT metadata is not dependent on the packet fields. Below steps are recommended for remediation:

- Ingress node and Verifier are configured with common pre shared key
- Ingress node generates a Message Authentication Code (MAC) from packet fields using standard HMAC algorithm.
- The left most bits of the output are truncated to desired length to generate RND. It is recommended to use a minimum of 32 bits.
- The verifier regenerates the HMAC from the packet fields and compares with RND. To ensure the POT data is in fact that of the packet.

If an HMAC is used, an active attacker lacks the knowledge of the pre-shared key, and thus cannot launch preplay attacks.

The solution discussed in this memo does not currently mitigate prereplay attacks. A mitigation mechanism may be included in future versions of the solution.

8.5. Anti-Tampering

An active attacker could not insert any arbitrary value for CML. This would subsequently fail the reconstruction of the POLY-3. Also an attacker could not update the CML with a previously observed value. This could subsequently be detected by using timestamps within the RND value as discussed above.

8.6. Recycling

The solution approach is flexible for recycling long term secrets like POLY-1. All the nodes could be periodically updated with shares of new SECRET as best practice. The table above could be consulted for refresh cycles (see Section 4).

8.7. Redundant Nodes and Failover

A "node" or "service" in terms of POT can be implemented by one or multiple physical entities. In case of multiple physical entities (e.g., for load-balancing, or business continuity situations - consider for example a set of firewalls), all physical entities which are implementing the same POT node are given that same share of the secret. This makes multiple physical entities represent the same POT node from an algorithm perspective.

8.8. Controller Operation

The Controller needs to be secured given that it creates and holds the secrets, as need to be the nodes. The communication between Controller and the nodes also needs to be secured. As secure communication protocol such as for example NETCONF over SSH should be chosen for Controller to node communication.

The Controller only interacts with the nodes during the initial configuration and thereafter at regular intervals at which the operator chooses to switch to a new set of secrets. In case 64 bits are used for the data fields "CML" and "RND" which are carried within the data packet, the regular intervals are expected to be quite long (e.g., at 100 Gbps, a profile would only be used up after 3100 years) - see Section 4 above, thus even a "headless" operation without a Controller can be considered feasible. In such a case, the

Controller would only be used for the initial configuration of the POT-profiles.

8.9. Verification Scope

The POT solution defined in this document verifies that a data-packet traversed or transited a specific set of nodes. From an algorithm perspective, a "node" is an abstract entity. It could be represented by one or multiple physical or virtual network devices, or is could be a component within a networking device or system. The latter would be the case if a forwarding path within a device would need to be securely verified.

8.9.1. Node Ordering

POT using Shamir’s secret sharing scheme as discussed in this document provides for a means to verify that a set of nodes has been visited by a data packet. It does not verify the order in which the data packet visited the nodes. In case the order in which a data packet traversed a particular set of nodes needs to be verified as well, alternate schemes that e.g., rely on "nested encryption" could to be considered.

8.9.2. Stealth Nodes

The POT approach discussed in this document is to prove that a data packet traversed a specific set of "nodes". This set could be all nodes within a path, but could also be a subset of nodes in a path. Consequently, the POT approach isn’t suited to detect whether "stealth" nodes which do not participate in proof-of-transit have been inserted into a path.

9. Acknowledgements

The authors would like to thank Eric Vyncke, Nalini Elkins, Srihari Raghavan, Ranganathan T S, Karthik Babu Harichandra Babu, Akshaya Nadahalli, Erik Nordmark, and Andrew Yourtchenko for the comments and advice.

10. References

10.1. Normative References

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

[SSS] "Shamir’s Secret Sharing", <https://en.wikipedia.org/wiki/Shamir%27s_Secret_Sharing>.

10.2. Informative References

[I-D.ietf-anima-autonomic-control-plane]
Behringer, M., Eckert, T., and S. Bjarnason, "An Autonomic Control Plane", draft-ietf-anima-autonomic-control-plane-03 (work in progress), July 2016.

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

Sashank Dara
Cisco Systems, Inc.
Cessna Business Park, Sarjapura Marathalli Outer Ring Road
BANGALORE, Bangalore, KARNATAKA 560 087
INDIA

Email: sadara@cisco.com

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

Email: cpignata@cisco.com

John Leddy
Comcast

Email: John_Leddy@cable.comcast.com

Stephen Youell
JP Morgan Chase
25 Bank Street
London E14 5JP
United Kingdom

Email: stephen.youell@jpmorgan.com

David Mozes

Email: mosesster@gmail.com

Tal Mizrahi
Marvell
6 Hamada St.
Yokneam 20692
Israel

Email: talmi@marvell.com

Networking Working Group L. Ginsberg
Internet-Draft P. Psenak
Intended status: Standards Track S. Previdi
Expires: December 24, 2016 Cisco Systems
M. Pilka
June 22, 2016

Segment Routing Conflict Resolution
draft-ietf-spring-conflict-resolution-01.txt

Abstract

In support of Segment Routing (SR) routing protocols advertise a variety of identifiers used to define the segments which direct forwarding of packets. In cases where the information advertised by a given protocol instance is either internally inconsistent or conflicts with advertisements from another protocol instance a means of achieving consistent forwarding behavior in the network is required. This document defines the policies used to resolve these occurrences.

Requirements Language

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

Table of Contents

1. Introduction 2
2. SR Global Block Inconsistency 3
3. SR-MPLS Segment Identifier Conflicts 5
3.1. Conflict Types 6
3.1.1. Prefix Conflict 6
3.1.2. SID Conflict 8
3.2. Processing conflicting entries 9
3.2.1. Policy: Ignore conflicting entries 9
3.2.2. Policy: Preference Algorithm/Quarantine 10
3.2.3. Policy: Preference algorithm/ignore overlap only . . 10
3.2.4. Preference Algorithm 10
3.2.5. Example Behavior - Single Topology/Algorithm 11
3.2.6. Example Behavior - Multiple Topologies 12
3.2.7. Evaluation of Policy Alternatives 13
3.2.8. Guaranteeing Database Consistency 14
4. Scope of SR-MPLS SID Conflicts 14
5. Security Considerations 15
6. IANA Consideration 15
7. Acknowledgements 15
8. References 15
8.1. Normative References 15
8.2. Informational References 16
Authors' Addresses 16

1. Introduction

Segment Routing (SR) as defined in [SR-ARCH] utilizes forwarding instructions called "segments" to direct packets through the network. Depending on the forwarding plane architecture in use, routing protocols advertise various identifiers which define the permissible values which can be used as segments, which values are assigned to specific prefixes, etc. Where segments have global scope it is necessary to have non-conflicting assignments - but given that the advertisements may originate from multiple nodes the possibility exists that advertisements may be received which are either internally inconsistent or conflicting with advertisements originated by other nodes. In such cases it is necessary to have consistent resolution of conflicts network-wide in order to avoid forwarding loops.

The problem to be addressed is protocol independent i.e., segment related advertisements may be originated by multiple nodes using different protocols and yet the conflict resolution MUST be the same on all nodes regardless of the protocol used to transport the advertisements.

The remainder of this document defines conflict resolution policies which meet these requirements. All protocols which support SR MUST adhere to the policies defined in this document.

2. SR Global Block Inconsistency

In support of an MPLS dataplane routing protocols advertise an SR Global Block (SRGB) which defines a set of label ranges reserved for use by the advertising node in support of SR. The details of how protocols advertise this information can be found in the protocol specific drafts e.g., [SR-OSPF], [SR-OSPFv3], and [SR-IS-IS]. However the protocol independent semantics are illustrated by the following example:

The originating router advertises the following ranges:

```
      Range 1: (100, 199)
      Range 2: (1000, 1099)
      Range 3: (500, 599)
```

The receiving routers concatenate the ranges and build the Segment Routing Global Block (SRGB) as follows:

```
   SRGB = (100, 199)
          (1000, 1099)
          (500, 599)
```

The indeces span multiple ranges:

```
      index=0 means label 100
      ...
      index 99 means label 199
      index 100 means label 1000
      index 199 means label 1099
      ...
      index 200 means label 500
      ...
```

Note that the ranges are an ordered set - what labels are mapped to a given index depends on the placement of a given label range in the set of ranges advertised.

For the set of ranges to be usable the ranges MUST be disjoint. Sender behavior is defined in various SR protocol drafts such as [SR-IS-IS] which specify that senders MUST NOT advertise overlapping ranges.

Receivers of SRGB ranges MUST validate the SRGB ranges advertised by other nodes. If the advertised ranges do not conform to the restrictions defined in the respective protocol specification receivers MUST ignore all advertised SRGB ranges from that node. Operationally the node is treated as though it did not advertise any SRGB ranges. [SR-MPLS] defines the procedures for mapping global SIDs to outgoing labels.

Note that utilization of local SIDs (e.g. adjacency SIDs) advertised by a node is not affected by the state of the advertised SRGB.

3. SR-MPLS Segment Identifier Conflicts

In support of an MPLS dataplane Segment identifiers (SIDs) are advertised and associated with a given prefix. SIDs may be advertised in the prefix reachability advertisements originated by a routing protocol (PFX) . SIDs may also be advertised by a Segment Routing Mapping Server (SRMS).

Mapping entries have an explicit context which includes the topology and the SR algorithm. A generalized mapping entry can be represented using the following definitions:

```
Src- PFX or SRMS
Pi - Initial prefix
Pe - End prefix
L  - Prefix length
Lx - Maximum prefix length (32 for IPv4, 128 for IPv6)
Si - Initial SID value
Se - End SID value
R  - Range value (See Note 1)
T  - Topology
A  - Algorithm

A Mapping Entry is then the tuple: (Src, Pi/L, Si, R, T, A)
Pe = (Pi + ((R-1) << (Lx-L))
Se = Si + (R-1)

NOTE 1: The SID advertised in a prefix reachability advertisement
        always has an implicit range of 1.
```

Conflicts in SID advertisements may occur as a result of misconfiguration. Conflicts may occur either in the set of advertisements originated by a single node or between advertisements originated by different nodes. Conflicts which occur within the set of advertisements (P-SID and SRMS) originated by a single node SHOULD be prevented by configuration validation on the originating node.

When conflicts occur, it is not possible for routers to know which of the conflicting advertisements is "correct". In order to avoid forwarding loops and/or blackholes, there is a need for all nodes to resolve the conflicts in a consistent manner. This in turn requires that all routers have identical sets of advertisements and that they all use the same selection algorithm. This document defines procedures to achieve these goals.

3.1. Conflict Types

Two types of conflicts may occur - Prefix Conflicts and SID Conflicts. Examples are provided in this section to illustrate these conflict types.

3.1.1. Prefix Conflict

When different SIDs are assigned to the same prefix we have a "prefix conflict". Prefix conflicts are specific to mapping entries sharing the same topology and algorithm.

Example PC1

```
(PFX, 192.0.2.120/32, 200, 1, 0, 0)
(PFX, 192.0.2.120/32, 30, 1, 0, 0)
```

The prefix 192.0.2.120/32 has been assigned two different SIDs:
  200 by the first advertisement
  30 by the second advertisement

Example PC2

```
(PFX, 2001:DB8::1/128, 400, 1, 2, 0)
(PFX, 2001:DB8::1/128, 50, 1, 2, 0)
```

The prefix 2001:DB8::1/128 has been assigned two different SIDs:
 400 by the first advertisement
 50 by the second advertisement

Prefix conflicts may also occur as a result of overlapping prefix ranges.

Example PC3

(SRMS, 192.0.2.1/32, 200, 200, 0, 0)
(SRMS, 192.0.2.121/32, 30, 10, 0, 0)

Prefixes 192.0.2.121/32 - 192.0.2.130/32 are assigned two different SIDs:
 320 through 329 by the first advertisement
 30 through 39 by the second advertisement

Example PC4
(SRMS, 2001:DB8::1/128, 400, 200, 2, 0)
(SRMS, 2001:DB8::121/128, 50, 10, 2, 0)

Prefixes 2001:DB8::121/128 - 2001:DB8::130/128 are assigned two different SIDs:
  420 through 429 by the first advertisement
  50 through 59 by the second advertisement

Examples PC3 and PC4 illustrate a complication - only part of the range advertised in the first advertisement is in conflict. It is logically possible to isolate the conflicting portion and try to use the non-conflicting portion(s) at the cost of increased implementation complexity.

A variant of the overlapping prefix range is a case where we have overlapping prefix ranges but no actual SID conflict.

Example PC5

(SRMS, 192.0.2.1/32, 200, 200, 0, 0)
(SRMS, 192.0.2.121/32, 320, 10, 0, 0)

(SRMS, 2001:DB8::1/128, 400, 200, 2, 0)
(SRMS, 2001:DB8::121/128, 520, 10, 2, 0)

Although there is prefix overlap between the two IPv4 entries (and the two IPv6 entries) the same SID is assigned to all of the shared prefixes by the two entries.

Given two mapping entries:

(SRC, P1/L1, S1, R1, T1, A1) and
(SRC, P2/L2, S2, R2, T2, A2)

where P1 <= P2

a prefix conflict exists if all of the following are true:

```
1)(T1 == T2) && (A1 == A2)
2)P1 <= P2
3)The prefixes are in the same address family.
2)L1 == L2
3)(P1e >= P2) && ((S1 + (P2 - P1)) != S2)
```

3.1.2. SID Conflict

When the same SID has been assigned to multiple prefixes we have a "SID conflict". SID conflicts are independent of address-family, independent of prefix len, independent of topology, and independent of algorithm. A SID conflict occurs when a mapping entry which has previously been checked to have no prefix conflict assigns one or more SIDs that are assigned by another entry which also has no prefix conflicts.

Example SC1

(PFX, 192.0.2.1/32, 200, 1, 0, 0)
(PFX, 192.0.2.222/32, 200, 1, 0, 0)
SID 200 has been assigned to 192.0.2.1/32 by the
first advertisement.
The second advertisement assigns SID 200 to 192.0.2.222/32.

Example SC2

(PFX, 2001:DB8::1/128, 400, 1, 2, 0)
(PFX, 2001:DB8::222/128, 400, 1, 2, 0)
SID 400 has been assigned to 2001:DB8::1/128 by the
first advertisement.
The second advertisement assigns SID 400 to 2001:DB8::222/128

SID conflicts may also occur as a result of overlapping SID ranges.

Example SC3

(SRMS, 192.0.2.1/32, 200, 200, 0, 0)
(SRMS, 198.51.100.1/32, 300, 10, 0, 0)

SIDs 300 - 309 have been assigned to two different prefixes.
The first advertisement assigns these SIDs
to 192.0.2.101/32 - 192.0.2.110/32.
The second advertisement assigns these SIDs to
198.51.100.1/32 - 198.51.100.10/32.

Example SC4
(SRMS, 2001:DB8::1/128, 400, 200, 2, 0)
(SRMS, 2001:DB8:1::1/128, 500, 10, 2, 0)

SIDs 500 - 509 have been assigned to two different prefixes.
The first advertisement assigns these SIDs to
2001:DB8::101/128 - 2001:DB8::10A/128.
The second advertisement assigns these SIDs to
2001:DB8:1::1/128 - 2001:DB8:1::A/128.

Examples SC3 and SC4 illustrate a complication - only part of the range advertised in the first advertisement is in conflict.

3.2. Processing conflicting entries

Two general approaches can be used to process conflicting entries.

1. Conflicting entries can be ignored

2. A standard preference algorithm can be used to choose which of the conflicting entries will be used

The following sections discuss these two approaches in more detail.

Note: This document does not discuss any implementation details i.e. what type of data structure is used to store the entries (trie, radix tree, etc.) nor what type of keys may be used to perform lookups in the database.

3.2.1. Policy: Ignore conflicting entries

In cases where entries are in conflict none of the conflicting entries are used i.e., the network operates as if the conflicting advertisements were not present.

Implementations are required to identify the conflicting entries and ensure that they are not used.

3.2.2. Policy: Preference Algorithm/Quarantine

For entries which are in conflict properties of the conflicting advertisements are used to determine which of the conflicting entries are used in forwarding and which are "quarantined" and not used. The entire quarantined entry is not used.

This approach requires that conflicting entries first be identified and then evaluated based on a preference rule. Based on which entry is preferred this in turn may impact what other entries are considered in conflict i.e. if A conflicts with B and B conflicts with C - it is possible that A does NOT conflict with C. Hence if as a result of the evaluation of the conflict between A and B, entry B is not used the conflict between B and C will not be detected.

3.2.3. Policy: Preference algorithm/ignore overlap only

A variation of the preference algorithm approach is to quarantine only the portions of the less preferred entry which actually conflicts. The original entry is split into multiple ranges. The ranges which are in conflict are quarantined. The ranges which are not in conflict are used in forwarding. This approach adds complexity as the relationship between the derived sub-ranges of the original mapping entry have to be associated with the original entry - and every time some change to the advertisement database occurs the derived sub-ranges have to be recalculated.

3.2.4. Preference Algorithm

The following algorithm is used to select the preferred mapping entry when a conflict exists. Evaluation is made in the order specified. Prefix conflicts are evaluated first. SID conflicts are then evaluated on the Active entries remaining after Prefix Conflicts have been resolved.

1. PFX source wins over SRMS source

2. Smaller range wins

3. IPv6 entry wins over IPv4 entry

4. Longer prefix length wins

5. Smaller algorithm wins

6. Smaller starting address (considered as an unsigned integer value) wins

7. Smaller starting SID wins

8. If topology IDs are NOT identical both entries MUST be ignored

Using smaller range as the highest priority tie breaker makes advertisements with a range of 1 the most preferred. This has the nice property that a single misconfiguration of an SRMS entry with a large range will not be preferred over a large number of advertisements with smaller ranges.

Since topology identifiers are locally scoped, it is not possible to make a consistent choice network wide when all elements of a mapping entry are identical except for the topology. This is why both entries MUST be ignored in such cases (Rule #8 above). Note that Rule #8 only applies when considering SID conflicts since Prefix conflicts are restricted to a single topology.

3.2.5. Example Behavior - Single Topology/Algorithm

The following mapping entries exist:in the database. For brevity, Topology/Algorithm is omitted and assumed to be (0,0) in all entries.

1. (PFX, 192.0.2.1/32, 100, 1)

2. (PFX, 192.0.2.101/32, 200, 1)

3. (SRMS, 192.0.2.1/32, 400, 255) !Prefix conflict with entries 1 and 2

4. (SRMS, 198.51.100.40/32, 200,1) !SID conflict with entry 2

The table below shows what mapping entries will be used in the forwarding plane (Active) and which ones will not be used (Excluded) under the three candidate policies:

Policy	Active Entries	Excluded Entries
Ignore		(PFX,192.0.2.1/32,100,1) (PFX,192.0.2.101/32,200,1) (SRMS,192.0.2.1/32,400,255) (SRMS,198.51.100.40/32,200,1)
Quarantine	(PFX,192.0.1.1/32,100,1) (PFX,192.0.2.101/32,200,1)	(SRMS,192.0.2.1/32,400,255) (SRMS,198.51.100.40/32,200,1)
Overlap-Only	(PFX,192.0.2.1/32,100,1) (PFX,192.0.2.101/32,200,1) *(SRMS,192.0.2.2/32,401,99) *(SRMS,192.0.2.102/32, 501,153)	(SRMS,198.51.100.40/32,200,1) *(SRMS,192.0.2.1/32,400,1) *(SRMS,192.0.2.101/32,500,1)

* Derived from (SRMS,192.0.2.1/32,400,300)

3.2.6. Example Behavior - Multiple Topologies

When using a preference rule the order in which conflict resolution is applied has an impact on what entries are usable when entries for multiple topologies (or algorithms) are present. The following mapping entries exist in the database:

1. (PFX, 192.0.2.1/32, 100, 1, 0, 0) !Topology 0

2. (PFX, 192.0.2.1/32, 200, 1, 0, 0) !Topology 0, Prefix Conflict with entry #1

3. (PFX, 198.51.100.40/32, 200,1,1,0) ! Topology 1, SID conflict with entry 2

The table below shows what mapping entries will be used in the forwarding plane (Active) and which ones will not be used (Excluded) under the Quarantine Policy based on the order in which conflict resolution is applied.

Order	Active Entries	Excluded Entries
Prefix-Conflict First	(PFX,192.0.2.1/32,100,1,0,0) (PFX,198.51.100.40/32,200,1,1,0)	(PFX,192.0.2.101/32,200,1,0)
SID-Conflict First	(PFX,192.0.2.1/32,100,1,0,0)	(PFX,192.0.2.101/32,200,1,0) (PFX,198.51.100.40/32,200,1,1,0)

This illustrates the advantage of evaluating prefix conflicts within a given topology (or algorithm) before evaluating topology (or algorithm) independent SID conflicts. It insures that entries which will be excluded based on intratopology preference will not prevent a SID assigned in another topology from being considered Active.

3.2.7. Evaluation of Policy Alternatives

The previous sections have defined three alternatives for resolving conflicts - ignore, quarantine, and ignore overlap-only.

The ignore policy impacts the greatest amount of traffic as forwarding to all destinations which have a conflict is affected.

Quarantine allows forwarding for some destinations which have a conflict to be supported.

Ignore overlap-only maximizes the destinations which will be forwarded as all destinations covered by some mapping entry (regardless of range) will be able to use the SID assigned by the winning range. This alternative increases implementation complexity as compared to quarantine. Mapping entries with a range greater than 1 which are in conflict with other mapping entries have to internally be split into 2 or more "derived mapping entries". The derived mapping entries then fall into two categories - those that are in conflict with other mapping entries and those which are NOT in conflict. The former are ignored and the latter are used. Each time the underived mapping database is updated the derived entries have to be recomputed based on the updated database. Internal data structures have to be maintained which maintain the relationship between the advertised mapping entry and the set of derived mapping entries. All nodes in the network have to achieve the same behavior regardless of implementation internals.

There is then a tradeoff between a goal of maximizing traffic delivery and the risks associated with increased implementation complexity.

It is the opinion of the authors that "quarantine" is the best alternative.

3.2.8. Guaranteeing Database Consistency

In order to obtain consistent active entries all nodes in a network MUST have the same mapping entry database. Mapping entries can be obtained from a variety of sources.

- o SIDs can be configured locally for prefixes assigned to interfaces on the router itself. Only SIDs which are advertised to protocol peers can be considered as part of the mapping entry database.

- o SIDs can be received in prefix reachability advertisements from protocol peers. These advertisements may originate from peers local to the area or be leaked from other areas and/or redistributed from other routing protocols.

- o SIDs can be received from SRMS advertisements - these advertisements can originate from routers local to the area or leaked from other areas

- o In cases where multiple routing protocols are in use mapping entries advertised by all routing protocols MUST be included.

4. Scope of SR-MPLS SID Conflicts

The previous section defines the types of SID conflicts and procedures to resolve such conflicts when using an MPLS dataplane. The mapping entry database used MUST be populated with entries for destinations for which the associated SID will be used to derive the labels installed in the forwarding plane of routers in the network. This consists of entries associated with intra-domain routes.

There are cases where destinations which are external to the domain are advertised by protocol speakers running within that network - and it is possible that those advertisements have SIDs associated with those destinations. However, if reachability to a destination is topologically outside the forwarding domain of the protocol instance then the SIDs for such destinations will never be installed in the forwarding plane of any router within the domain - so such advertisements cannot create a SID conflict within the domain. Such entries therefore MUST NOT be installed in the database used for intra-domain conflict resolution.

Consider the case of two sites "A and B" associated with a given [RFC4364] VPN. Connectivity between the sites is via a provider backbone. SIDs associated with destinations in Site A will never be installed in the forwarding plane of routers in Site B. Reachability between the sites (assuming SR is being used across the backbone) only requires using a SID associated with a gateway PE. So a destination in Site A MAY use the same SID as a destination in Site B without introducing any conflict in the forwarding plane of routers in Site A.

Such cases are handled by insuring that the mapping entries in the database used by the procedures defined in the previous section only include entries associated with advertisements within the site.

5. Security Considerations

TBD

6. IANA Consideration

This document has no actions for IANA.

7. Acknowledgements

The authors would like to thank Jeff Tantsura, Wim Henderickx, and Bruno Decraene for their careful review and content suggestions.

8. References

8.1. Normative References

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

[RFC4364] Rosen, E. and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, DOI 10.17487/RFC4364, February 2006, <http://www.rfc-editor.org/info/rfc4364>.

[SR-IS-IS] "IS-IS Extensions for Segment Routing, draft-ietf-isis-segment-routing-extensions-07(work in progress)", June 2016.

[SR-MPLS] "Segment Routing with MPLS dataplane, draft-ietf-spring-segment-routing-mpls-04(work in progress)", March 2016.

[SR-OSPF] "OSPF Extensions for Segment Routing, draft-ietf-ospf-segment-routing-extensions-08(work in progress)", May 2016.

[SR-OSPFv3] "OSPFv3 Extensions for Segment Routing, draft-ietf-ospf-ospfv3-segment-routing-extensions-05(work in progress)", March 2016.

8.2. Informational References

[SR-ARCH] "Segment Routing Architecture, draft-ietf-spring-segment-routing-08(work in progress)", May 2016.

Authors' Addresses

Les Ginsberg
Cisco Systems
510 McCarthy Blvd.
Milpitas, CA 95035
USA

Email: ginsberg@cisco.com

Peter Psenak
Cisco Systems
Apollo Business Center Mlynske nivy 43
Bratislava 821 09
Slovakia

Email: ppsenak@cisco.com

Stefano Previdi
Cisco Systems
Via Del Serafico 200
Rome 0144
Italy

Email: sprevidi@cisco.com

Martin Pilka

Email: martin@infobox.sk

Networking Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 3, 2018

L. Ginsberg
P. Psenak
S. Previdi
Cisco Systems
M. Pilka

July 2, 2017

Segment Routing MPLS Conflict Resolution
draft-ietf-spring-conflict-resolution-05.txt

Abstract

In support of Segment Routing (SR) for an MPLS data plane routing protocols advertise a variety of identifiers used to define the segments which direct forwarding of packets. In cases where the information advertised by a given protocol instance is either internally inconsistent or conflicts with advertisements from another protocol instance a means of achieving consistent forwarding behavior in the network is required. This document defines the policies used to resolve these occurrences.

Requirements Language

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

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction   3
2. SR Global Block Inconsistency   3
3. SR-MPLS Segment Identifier Conflicts   5
  3.1. SID Preference   6
  3.2. Conflict Types   7
    3.2.1. Prefix Conflict   7
    3.2.2. SID Conflict   9
  3.3. Preference rule for resolving conflicts 12
  3.4. Conflict Resolution Algorithm 13
  3.5. Example Behavior - Single Topology/Address Family/Algorithm 14
  3.6. Example Behavior - Multiple Topologies 15
  3.7. Guaranteeing Database Consistency 16
  3.8. Minimizing the occurence of conflicts 16
4. Scope of SR-MPLS SID Conflicts 16
5. Conflict Resolution and non-forwarding nodes 17
6. Security Considerations 17
7. IANA Consideration 18
8. Acknowledgements 18
9. References 18
  9.1. Normative References 18
  9.2. Informational References 19
Appendix A. Alternative SID Conflict Resolution Policy Discussion 19
  A.1. Policy: Ignore conflicting entries 19
  A.2. Policy: Preference Algorithm/Quarantine 19
  A.3. Policy: Preference algorithm/ignore overlap only 20
  A.4. Example Behavior - Single Topology/Address Family/Algorithm 20
  A.5. Evaluation of Policy Alternatives 21
Authors' Addresses 22

1. Introduction

Segment Routing (SR) as defined in [SR-ARCH] utilizes forwarding instructions called "segments" to direct packets through the network. Depending on the forwarding plane architecture in use, routing protocols advertise various identifiers which define the permissible values which can be used as segments, which values are assigned to specific prefixes, etc. Where segments have global scope it is necessary to have non-conflicting assignments - but given that the advertisements may originate from multiple nodes the possibility exists that advertisements may be received which are either internally inconsistent or conflicting with advertisements originated by other nodes. In such cases it is necessary to have consistent resolution of conflicts network-wide in order to avoid forwarding loops.

This document is limited to discussion of conflict resolution for identifiers used in an MPLS data plane.

The problem to be addressed is protocol independent i.e., segment related advertisements may be originated by multiple nodes using different protocols and yet the conflict resolution MUST be the same on all nodes regardless of the protocol used to transport the advertisements.

The remainder of this document defines conflict resolution policies which meet these requirements. All protocols which support SR MUST adhere to the policies defined in this document.

2. SR Global Block Inconsistency

In support of an MPLS dataplane [SR-MPLS] routing protocols advertise an SR Global Block (SRGB) which defines a set of label ranges reserved for use by the advertising node in support of SR. The details of how protocols advertise this information can be found in the protocol specific drafts e.g., [SR-OSPF], [SR-OSPFv3], [SR-IS-IS], and [SR-BGP]. However the protocol independent semantics are illustrated by the following example:

The originating router advertises the following ranges:

```
      Range 1: (100, 199)
      Range 2: (1000, 1099)
      Range 3: (500, 599)
```

The receiving routers concatenate the ranges and build the Segment Routing Global Block (SRGB) as follows:

```
   SRGB = (100, 199)
          (1000, 1099)
          (500, 599)
```

The indeces span multiple ranges:

```
      index=0 means label 100
      ...
      index 99 means label 199
      index 100 means label 1000
      index 199 means label 1099
      ...
      index 200 means label 500
      ...
```

Note that the ranges are an ordered set - what labels are mapped to a given index depends on the placement of a given label range in the set of ranges advertised.

For the set of ranges to be usable the ranges MUST be disjoint. Sender behavior is defined in various SR protocol drafts such as [SR-IS-IS] which specify that senders MUST NOT advertise overlapping ranges.

Receivers of SRGB ranges MUST validate the SRGB ranges advertised by other nodes. If the advertised ranges do not conform to the restrictions defined in the respective protocol specification receivers MUST ignore all advertised SRGB ranges from that node. Operationally the node is treated as though it did not advertise any SRGB ranges. [SR-MPLS] defines the procedures for mapping global SIDs to outgoing labels.

Note that utilization of local SIDs (e.g. adjacency SIDs) advertised by a node is not affected by the state of the advertised SRGB.

3. SR-MPLS Segment Identifier Conflicts

In support of an MPLS dataplane Segment Identifiers (SIDs) are advertised and associated with a given prefix. SIDs may be advertised in the prefix reachability advertisements originated by a routing protocol (PFX) . SIDs may also be advertised by a Segment Routing Mapping Server (SRMS). How this is done is defined in the protocol specific drafts e.g., [SR-OSPF], [SR-OSPFv3], [SR-IS-IS], and [SR-BGP]

Information in a SID advertisement is used to construct a mapping entry. A generalized mapping entry can be represented using the following definitions:

```
Prf - Preference Value (See Section 3.1)
Pi - Initial prefix
Pe - End prefix
L  - Prefix length
Lx - Maximum prefix length (32 for IPv4, 128 for IPv6)
Si - Initial SID value
Se - End SID value
R  - Range value (See Note 1)
T  - Topology
A  - Algorithm (see [SR-ARCH])

A Mapping Entry is then the tuple: (Prf, Pi/L, Si, R, T, A)
Pe = (Pi + ((R-1) << (Lx-L))
Se = Si + (R-1)

NOTE 1: The SID advertised in a prefix reachability advertisement
        always has an implicit range of 1.

NOTE 2: IPv4/IPv6 addresses can be viewed as 32/128 bit integers.
        Where operations such as addition, subtraction, and/or
        bit shifting are specified for prefixes this should be
        interpreted as operations on the integer representation
        of a prefix.
```

Note: Topology is a locally scoped identifier assigned by each router. Although it may have an association with Multitopology Identifiers (MTID) advertised by routing protocols it is NOT equivalent to these identifiers. MTIDs are scoped by a given routing protocol. MTID ranges are protocol specific and there may be standardized protocol specific MTID assignments for topologies of a specific type (e.g., an AFI specific topology). As mapping entries can be sourced from multiple protocols it is not possible to use a network scoped identifier for a topology when storing mapping entries in the local datbase.

Conflicts in SID advertisements may occur as a result of misconfiguration. When conflicts occur, it is not possible for routers to know which of the conflicting advertisements is "correct". In order to avoid forwarding loops and/or blackholes, there is a need for all nodes to resolve the conflicts in a consistent manner. This in turn requires that all routers have identical sets of advertisements and that they all use the same selection algorithm. This document defines procedures to achieve these goals.

3.1. SID Preference

If a node acts as an SRMS, it MAY advertise a preference to be associated with all SRMS SID advertisements sent by that node. The means of advertising the preference is defined in the protocol specific drafts e.g., [SR-OSPF], [SR-OSPFv3], and [SR-IS-IS]. The preference value is an unsigned 8 bit integer with the following properties:

- 0 - Reserved value indicating advertisements from that node MUST NOT be used.
- 1 - 255 Preference value

Advertisement of a preference value is optional. Nodes which do not advertise a preference value are assigned a preference value of 128.

All SIDs advertised in prefix reachability advertisements originated by an IGP implicitly have a preference value of 192.

All SIDs advertised in prefix reachability advertisements originated by BGP implicitly have a preference value of 64.

These preference values are deliberately chosen to favor SID advertisements originated within a domain (IGP and SRMS) over SID advertisements which may have been imported from other domains (BGP). In addition, as BGP originated advertisements may not be known on all nodes within a domain (because not every node will be a BGP speaker), the presence of a BGP originated mapping entry MUST NOT cause a mapping entry originated within the domain to become unusable as this would introduce inconsistency in the set of SIDs considered usable by a node which has the BGP originated mapping entries and the set considered usable by nodes without the BGP originated mapping entries.

3.2. Conflict Types

Two types of conflicts may occur - Prefix Conflicts and SID Conflicts. Examples are provided in this section to illustrate these conflict types and generic definitions of algorithms to determine when there is a conflict are presented.

3.2.1. Prefix Conflict

When different SIDs are assigned to the same prefix we have a "prefix conflict". Prefix conflicts are limited to mapping entries sharing the same topology, algorithm, address-family, and prefix length.

3.2.1.1. Prefix Conflict Examples

The simplest example is when two advertisements with a range of 1 assign different SIDs to the same prefix.

Example PC1

```
(192, 192.0.2.120/32, 200, 1, 0, 0)
(192, 192.0.2.120/32, 30, 1, 0, 0)
```

The prefix 192.0.2.120/32 has been assigned two different SIDs:
  200 by the first advertisement
  30 by the second advertisement

Example PC2

```
(192, 2001:DB8::1/128, 400, 1, 2, 0)
(192, 2001:DB8::1/128, 50, 1, 2, 0)
```

The prefix 2001:DB8::1/128 has been assigned two different SIDs:
 400 by the first advertisement
 50 by the second advertisement

Prefix conflicts may also occur as a result of overlapping prefix ranges.

Example PC3

(128, 192.0.2.1/32, 200, 200, 0, 0)
(128, 192.0.2.121/32, 30, 10, 0, 0)

Prefixes 192.0.2.121/32 - 192.0.2.130/32 are assigned two different SIDs:
 320 through 329 by the first advertisement
 30 through 39 by the second advertisement

Example PC4
(128, 2001:DB8::1/128, 400, 200, 2, 0)
(128, 2001:DB8::121/128, 50, 10, 2, 0)

Prefixes 2001:DB8::121/128 - 2001:DB8::130/128 are assigned two different SIDs:
  420 through 429 by the first advertisement
  50 through 59 by the second advertisement

Examples PC3 and PC4 illustrate a complication - only part of the range advertised in the first advertisement is in conflict. It is logically possible to consider the sub-range(s) which are in conflict as unusable while considering the sub-range(s) not in conflict as usable.

A variant of the overlapping prefix range is a case where we have overlapping prefix ranges but no actual prefix conflict.

Example PC5

(128, 192.0.2.1/32, 200, 200, 0, 0)
(128, 192.0.2.121/32, 320, 10, 0, 0)

(128, 2001:DB8::1/128, 400, 200, 2, 0)
(128, 2001:DB8::121/128, 520, 10, 2, 0)

Although there is prefix overlap between the two IPv4 entries (and the two IPv6 entries) the same SID is assigned to all of the shared prefixes by the two entries.

3.2.1.2. Prefix Conflict Generic Algorithm

The following generic algorithm can be used to determine when any two mapping entries have Prefix Conflicts and what the set of prefixes in conflict are.

Given two mapping entries:

```
(Prf, P1/L1, S1, R1, T1, A1) and
(Prf, P2/L2, S2, R2, T2, A2)
```

where P1 <= P2

a prefix conflict exists if all of the following are true:

1)Topologies, algorithms, and prefix lengths are identical

```
(T1 == T2) && (A1 == A2) && (L1 == L2)
```

2)The prefixes are in the same address-family.

3)If there are overlapping prefixes in the two ranges and
if there are different SIDs assigned to any of the prefixes
in the overlapping range

```
(P1e >= P2) && ((S1 + ((P2 - P1) >> (Lx-L1)) != S2)
```

Prefixes in the following range are in conflict:

```
P2 through MIN(P1e,P2e)
```

3.2.2. SID Conflict

When the same SID has been assigned to multiple prefixes we have a "SID conflict". SID conflicts are independent of address-family, independent of prefix len, independent of topology, and independent of algorithm.

3.2.2.1. SID Conflict Examples

The simplest example is when two mapping entries with a range of 1 assigns different SIDs to the same prefix.

Example SC1

(192, 192.0.2.1/32, 200, 1, 0, 0)
(192, 192.0.2.222/32, 200, 1, 0, 0)
SID 200 has been assigned to 192.0.2.1/32 by the
first advertisement.
The second advertisement assigns SID 200 to 192.0.2.222/32.

Example SC2

(192, 2001:DB8::1/128, 400, 1, 2, 0)
(192, 2001:DB8::222/128, 400, 1, 2, 0)
SID 400 has been assigned to 2001:DB8::1/128 by the
first advertisement.
The second advertisement assigns SID 400 to 2001:DB8::222/128

SID conflicts may also occur as a result of overlapping SID ranges.

Example SC3

(128, 192.0.2.1/32, 200, 200, 0, 0)
(128, 198.51.100.1/32, 300, 10, 0, 0)

SIDs 300 - 309 have been assigned to two different prefixes.
The first advertisement assigns these SIDs
to 192.0.2.101/32 - 192.0.2.110/32.
The second advertisement assigns these SIDs to
198.51.100.1/32 - 198.51.100.10/32.

Example SC4
(128, 2001:DB8::1/128, 400, 200, 2, 0)
(128, 2001:DB8:1::1/128, 500, 10, 2, 0)

SIDs 500 - 509 have been assigned to two different prefixes.
The first advertisement assigns these SIDs to
2001:DB8::101/128 - 2001:DB8::10A/128.
The second advertisement assigns these SIDs to
2001:DB8:1::1/128 - 2001:DB8:1::A/128.

Examples SC3 and SC4 illustrate a complication - only part of the range advertised in the first advertisement is in conflict.

SID conflicts may also occur because the same SID has been used in two different algorithms, two different topologies, two different address families, or prefixes with two different lengths.

Example SC5

(128, 192.0.2.1/32, 200, 1, 0, 0)
(128, 192.0.2.1/32, 200, 1, 0, 1)

SID 200 has been assigned to the same prefix with two different algorithms.

Example SC6
(128, 192.0.2.1/32, 200, 1, 0, 0)
(128, 2001:DB8::1/128, 200, 1, 0, 0)

SID 200 has been assigned to prefixes in two different address-families.

3.2.2.2. SID Conflict Generic Algorithm

The following generic algorithm can be used to determine when any two mapping entries have SID Conflicts and what the set of SIDs in conflict are.

Given two mapping entries:

(Prf, P1/L1, S1, R1, T1, A1) and
(Prf, P2/L2, S2, R2, T2, A2)

a SID conflict exists if all of the following are true:

1)If the SID ranges overlap

    (S1 <= S2) && (S1e >= S2)

2)If the same SID is assigned to prefixes with different address-families, prefix lengths, topologies, or algorithms or the same SID is assigned to two different prefixes for any of the prefixes in either range.

     P1 and P2 are NOT in the same address family OR
     L1 != L2 OR
     T1 != T2 OR
     A1 != A2 OR
     (P1 + ((S1e-S2) << (L1x-L1))) != P2

SIDs in the following range are in conflict:

     S2 through MIN(S1e,S2e)

3.3. Preference rule for resolving conflicts

When a conflict is detected the following algorithm is used to select the preferred mapping entry. Evaluation is made in the order specified. Prefix conflicts are evaluated first. SID conflicts are then evaluated on the Active entries remaining after Prefix Conflicts have been resolved.

1. Higher preference value wins

2. Smaller range wins

3. IPv6 entry wins over IPv4 entry

4. Longer prefix length wins

5. Smaller starting address (considered as an unsigned integer value) wins

6. Smaller algorithm wins

7. Smaller starting SID wins

8. If topology IDs are NOT identical both entries MUST be ignored

When applying the preference rule to prefix/SID pairs associated with an advertised mapping entry with a range greater than one, each prefix/SID pair in the range is considered as having the range associated with the advertised mapping entry. For example:

Advertised mapping entry: (128, 192.0.2.1/32, 200, 200, 0, 0)

The advertisement covers 200 prefix/SID pairs:
192.0.2.1/32 200
192.0.2.2/32 201
...
192.0.2.200/32 399

Each of these prefix/SID pairs is considered as having a range of 200 when applying Rule #2 above.

As SIDs associated with prefix reachability advertisements have a preference of 192 and an implied range of 1 while by default SRMS preference is 128, the default behavior is then to prefer SIDs advertised in prefix reachability advertisements over SIDs advertised by SRMSs, but an operator can choose to override this behavior by setting SRMS preference higher than 192.

Preferring advertisements with smaller range has the nice property that a single misconfiguration of an SRMS entry with a large range will not be preferred over a large number of advertisements with smaller ranges.

Since topology identifiers are locally scoped, it is not possible to make a consistent choice network wide when all elements of a mapping entry are identical except for the topology. This is why both entries MUST be ignored in such cases (Rule #8 above). Note that Rule #8 only applies when considering SID conflicts since Prefix conflicts are restricted to a single topology.

3.4. Conflict Resolution Algorithm

The following logical steps MUST be followed in the order specified when resolving conflicts.

Step 1: Resolve Prefix Conflicts (same topology/address family/ algorithm)

For each supported topology/address family/algorithm examine all qualifying mapping entries in the following order:

1)Preference (start w highest)
2)Range (start w smallest)
3)Prefix length (start w longest)
4)Address (start w smallest)
5)SID (start w smallest)

At each step if a prefix conflict is detected the losing prefix/SID pair is declared Inactive and is not considered in any subsequent steps. The remaining prefix/SID pairs are Active.

Mapping entries with Active prefix/SID pairs after completion of Step 1 are fed into ...

Step 2: SID Conflicts (across all topologies/address families/ algorithms)

Examine all Active prefix/SID pairs from Step #1 in the following order:

1)Preference (start w highest)
2)Range (start w smallest)
3)IPv6 entries
  a)Prefix length (start w longest)
  b)Address (start w smallest)
4)IPv4 entries
  a)Prefix Length (start w longest)
  b)Address (start w smallest)
5)Algorithm (start w smallest)
6)SID (start w smallest)

Prefix/SID pairs which are identical and are associated with the same topology are duplicates - both entries MUST be considered as Active.
Prefix/SID pairs which are identical and are associated with different topologies MUST both be considered Inactive.

Active Entries in the database may be used in forwarding. Inactive entries MUST NOT be used in forwarding.

Note that when the database of mapping entries changes the full set of logical steps MUST be reapplied to the entire database as conflict resolution is NOT transitive.

NOTE: Clever implementors may realize optimizations when rerunning the algorithm by evaluating changed entries as to whether they have potential conflicts with any of the existing entries in the database (both active and inactive). Such optimizations are outside the scope of this specification. The normative behavior is defined by the logical algorithm above.

3.5. Example Behavior - Single Topology/Address Family/Algorithm

The following mapping entries exist in the database. For brevity, Topology/Algorithm is omitted and assumed to be (0,0) in all entries.

1. (192, 192.0.2.1/32, 100, 1)

2. (192, 192.0.2.101/32, 200, 1)

3. (128, 192.0.2.1/32, 400, 255) !Prefix conflict with entries 1 and 2

4. (128, 198.51.100.40/32, 200,1) !SID conflict with entry 2

The table below shows what mapping entries will be used in the forwarding plane (Active) and which ones will not be used (Inactive)

```
+--------------------------------------------------------------+
|  Active Entries              |  Inactive Entries             |
+--------------------------------------------------------------+
| (192,192.0.2.1/32,100,1)     | (128,198.51.100.40/32,200,1)|
| (192,192.0.2.101/32,200,1)   |*(128,192.0.2.1/32,400,1)      |
|*(128,192.0.2.2/32,401,99)    |*(128,192.0.2.101/32,500,1)    |
|*(128,192.0.2.102/32,501,154) |                               |
+--------------------------------------------------------------+
```

* Derived from (128,192.0.2.1/32,400,255)

3.6. Example Behavior - Multiple Topologies

When using a preference rule the order in which conflict resolution is applied has an impact on what entries are Active when entries for multiple topologies (or algorithms) are present. The following mapping entries exist in the database:

1. (192, 192.0.2.1/32, 100, 1, 0, 0) !Topology 0

2. (192, 192.0.2.1/32, 200, 1, 0, 0) !Topology 0, Prefix Conflict with entry #1

3. (192, 198.51.100.40/32, 200,1,1,0) ! Topology 1, SID conflict with entry 2

The table below shows what mapping entries will be used in the forwarding plane (Active) and which ones will not be used (Inactive) based on the order in which conflict resolution is applied.

```
+--------------------------------------------------------------------+
|Order   | Active Entries              | Inactive Entries            |
+--------------------------------------------------------------------+
|Prefix- |(192,192.0.2.1/32,100,1,0,0)|(192,192.0.2.101/32,200,1,0)|
|Conflict|(192,198.51.100.40/32,200,1,|                              |
|First   |    1,0)                     |                              |
+--------------------------------------------------------------------+
|SID-    |(192,192.0.2.1/32,100,1,0,0)|(192,192.0.2.101/32,200,1,0)|
|Conflict|                             |(192,198.51.100.40/32,200,1,|
|First   |                             |    1,0)                     |
+--------------------------------------------------------------------+
```

This illustrates the advantage of evaluating prefix conflicts within a given topology (or algorithm) before evaluating topology (or algorithm) independent SID conflicts. It insures that entries which will be excluded based on intratopology preference will not prevent a SID assigned in another topology from being considered Active.

3.7. Guaranteeing Database Consistency

In order to obtain consistent active entries all nodes in a network MUST have the same mapping entry database. Mapping entries can be obtained from a variety of sources.

- o SIDs can be configured locally for prefixes assigned to interfaces on the router itself. Only SIDs which are advertised to protocol peers can be considered as part of the mapping entry database.
- o SIDs can be received in prefix reachability advertisements from protocol peers. These advertisements may originate from peers local to the area or be leaked from other areas and/or redistributed from other routing protocols.
- o SIDs can be received from SRMS advertisements - these advertisements can originate from routers local to the area or leaked from other areas
- o In cases where multiple routing protocols are in use mapping entries advertised by all routing protocols MUST be included.

3.8. Minimizing the occurence of conflicts

Conflicts in SID advertisements are always the result of a misconfiguration. Conflicts may occur either in the set of advertisements originated by a single node or between advertisements originated by different nodes.

Conflicts which occur within the set of advertisements (PFX and SRMS) originated by a single node SHOULD be prevented by configuration validation on the originating node.

It is possible to minimize the occurrence of conflicts between advertisements originated by different routers if new configuration is validated against the current state of the conflict resolution database before the configuration is advertised. How this is done is an implementation issue which is out of scope of this document.

4. Scope of SR-MPLS SID Conflicts

The previous section defines the types of SID conflicts and procedures to resolve such conflicts when using an MPLS dataplane. The mapping entry database used MUST be populated with entries for destinations for which the associated SID will be used to derive the labels installed in the forwarding plane of routers in the network. This consists of entries associated with intra-domain routes.

There are cases where destinations which are external to the domain are advertised by protocol speakers running within that network - and it is possible that those advertisements have SIDs associated with those destinations. However, if reachability to a destination is topologically outside the forwarding domain of the protocol instance then the SIDs for such destinations will never be installed in the forwarding plane of any router within the domain - so such advertisements cannot create a SID conflict within the domain. Such entries therefore MUST NOT be installed in the database used for intra-domain conflict resolution.

Consider the case of two sites "A and B" associated with a given [RFC4364] VPN. Connectivity between the sites is via a provider backbone. SIDs associated with destinations in Site A will never be installed in the forwarding plane of routers in Site B. Reachability between the sites (assuming SR is being used across the backbone) only requires using a SID associated with a gateway PE. So a destination in Site A MAY use the same SID as a destination in Site B without introducing any conflict in the forwarding plane of routers in Site A.

Such cases are handled by insuring that the mapping entries in the database used by the procedures defined in the previous section only include entries associated with advertisements within the site.

5. Conflict Resolution and non-forwarding nodes

The previous sections define conflict resolution behavior required of nodes which perform forwarding. But conflict resolution also impacts other entities e.g., controllers. If a controller were to define an explicit path using a SID in a way that is inconsistent with the set of Active entries produced by conflict resolution procedures used by the forwarding nodes then traffic following the explicit path may be misdelivered.

To prevent this such an entity MUST either implement the conflict resolution procedures defined above or implement an alternate form of conflict resolution which produces a subset of the Active entries which result from the conflict resolution procedures defined above. One such alternate form is to consider Inactive any mapping entry which has either a prefix conflict or a SID conflict with any other mapping entry.

6. Security Considerations

The ability to introduce SID conflicts into a deployment may compromise traffic forwarding. Protocol specific security mechanisms

SHOULD be used to insure that all SID advertisements originate from trusted sources.

7. IANA Consideration

This document has no actions for IANA.

8. Acknowledgements

The authors would like to thank Jeff Tantsura, Wim Henderickx, Bruno Decraene, and Stephane Litkowski for their careful review and content suggestions.

9. References

9.1. 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, <http://www.rfc-editor.org/info/rfc2119>.

[RFC4364] Rosen, E. and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, DOI 10.17487/RFC4364, February 2006, <http://www.rfc-editor.org/info/rfc4364>.

[SR-BGP] "Segment Routing Prefix SID extensions for BGP, draft-ietf-idr-bgp-prefix-sid-06(work in progress)", June 2017.

[SR-IS-IS] "IS-IS Extensions for Segment Routing, draft-ietf-isis-segment-routing-extensions-13(work in progress)", June 2017.

[SR-MPLS] "Segment Routing with MPLS dataplane, draft-ietf-spring-segment-routing-mpls-10(work in progress)", June 2017.

[SR-OSPF] "OSPF Extensions for Segment Routing, draft-ietf-ospf-segment-routing-extensions-17(work in progress)", June 2017.

[SR-OSPFv3] "OSPFv3 Extensions for Segment Routing, draft-ietf-ospf-ospfv3-segment-routing-extensions-09(work in progress)", March 2017.

9.2. Informational References

[SR-ARCH] "Segment Routing Architecture, draft-ietf-spring-segment-routing-12(work in progress)", June 2017.

Appendix A. Alternative SID Conflict Resolution Policy Discussion

A number of approaches to resolving SID conflicts were considered during the writing of this document. Two general approaches with a total of three policy alternatives were considered. This Appendix documents the alternatives considered. All content in this section is non-normative.

Two general approaches can be used to process conflicting entries.

1. Conflicting entries can be ignored

2. A standard preference algorithm can be used to choose which of the conflicting entries will be used

The following sections discuss these two approaches in more detail.

A.1. Policy: Ignore conflicting entries

In cases where entries are in conflict none of the conflicting entries are used i.e., the network operates as if the conflicting advertisements were not present.

Implementations are required to identify the conflicting entries and ensure that they are not used.

A.2. Policy: Preference Algorithm/Quarantine

For entries which are in conflict properties of the conflicting advertisements are used to determine which of the conflicting entries are used in forwarding and which are "quarantined" and not used. Losing mapping entries with ranges greater than 1 are quarantined in their entirety.

This approach requires that conflicting entries first be identified and then evaluated based on a preference rule. Based on which entry is preferred this in turn may impact what other entries are considered in conflict i.e. if A conflicts with B and B conflicts with C - it is possible that A does NOT conflict with C. Hence if as a result of the evaluation of the conflict between A and B, entry B is not used the conflict between B and C will not be detected.

A.3. Policy: Preference algorithm/ignore overlap only

A variation of the preference algorithm approach when applied to mapping entries with ranges greater than 1 is to quarantine only the portions of the less preferred entry which actually conflict. The original entry is logically considered as a set of entries with a range of 1, each of which inherits the range value of the original entry for purposes of applying the preference rule.

A.4. Example Behavior - Single Topology/Address Family/Algorithm

The following mapping entries exist in the database. For brevity, Topology/Algorithm is omitted and assumed to be (0,0) in all entries.

1. (192, 192.0.2.1/32, 100, 1)

2. (192, 192.0.2.101/32, 200, 1)

3. (128, 192.0.2.1/32, 400, 255) !Prefix conflict with entries 1 and 2

4. (128, 198.51.100.40/32, 200,1) !SID conflict with entry 2

The table below shows what mapping entries will be used in the forwarding plane (Active) and which ones will not be used (Inactive) under the three candidate policies:

Policy	Active Entries	Inactive Entries
Ignore		(192,192.0.2.1/32,100,1) (192,192.0.2.101/32,200,1) (128,192.0.2.1/32,400,255) (128,198.51.100.40/32,200,1)
Quarantine	(192,192.0.1.1/32,100,1) (192,192.0.2.101/32,200,1)	(128,192.0.2.1/32,400,255) (128,198.51.100.40/32,200,1)
Ignore-Overlap-Only	(192,192.0.2.1/32,100,1) (192,192.0.2.101/32,200,1) *(128,192.0.2.2/32,401,99) *(128,192.0.2.102/32, 501,153)	(128,198.51.100.40/32,200,1) *(128,192.0.2.1/32,400,1) *(128,192.0.2.101/32,500,1)

* Derived from (128,192.0.2.1/32,400,300)

A.5. Evaluation of Policy Alternatives

The previous sections have defined three alternatives for resolving conflicts - ignore, quarantine, and ignore overlap-only.

The ignore policy impacts the greatest number of mapping entriesas all prefix/SID pairs contained in an advertisement which has a conflict are considered Inactive.

Quarantine allows forwarding for some destinations which have a conflict to be supported - but losing mapping entries with ranges greater than 1 are declared Inactive in their entirety. This may result in not using individual prefix/SID entries contained within the quarantined advertisement which do not have a conflict.

Ignore-overlap-only maximizes the entries which may be Active as each prefix/SID pair contained within an advertised mapping entry with range greater than 1 is evaluated independent of the other entries within the same advertisement. To implement this alternative advertised mapping entries with a range greater than 1 which have a conflict with other advertised mapping entries have to logically be split into 2 or more "derived mapping entries". The derived mapping entries then fall into two categories - those that are in conflict with other mapping entries and have lost based on the preference rule and those which are either NOT in conflict or have won based on the preference rule. The former are considered Inactive while the latter are considered Active. Each time the underived mapping database is updated the derived entries have to be recomputed based on the updated database. Internal data structures have to be maintained which maintain the relationship between the advertised mapping entry and the set of derived mapping entries. All nodes in the network have to achieve the same behavior regardless of implementation internals.

There is then a tradeoff between a goal of maximizing advertised mapping entry usage and the risks associated with increased implementation complexity.

Consensus of the working group is that maximizing the use of the advertised prefix/SID pairs is the most important deployment consideration - therefore ignore-overlap-only has been specified as the standard policy which MUST be implemented by all nodes which support SR-MPLS.

Authors' Addresses

Les Ginsberg
Cisco Systems
821 Alder Drive
Milpitas, CA 95035
USA

Email: ginsberg@cisco.com

Peter Psenak
Cisco Systems
Apollo Business Center Mlynske nivy 43
Bratislava 821 09
Slovakia

Email: ppsenak@cisco.com

Stefano Previdi
Cisco Systems

Email: stefano@previdi.net

Martin Pilka

Email: martin@infobox.sk

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: July 29, 2018

C. Filsfils, Ed.
S. Previdi, Ed.
Cisco Systems, Inc.
L. Ginsberg
Cisco Systems, Inc
B. Decraene
S. Litkowski
Orange
R. Shakir
Google, Inc.

January 25, 2018

Segment Routing Architecture
draft-ietf-spring-segment-routing-15

Abstract

Segment Routing (SR) leverages the source routing paradigm. A node steers a packet through an ordered list of instructions, called segments. A segment can represent any instruction, topological or service-based. A segment can have a semantic local to an SR node or global within an SR domain. SR allows to enforce a flow through any topological path while maintaining per-flow state only at the ingress nodes to the SR domain.

Segment Routing can be directly applied to the MPLS architecture with no change on the forwarding plane. A segment is encoded as an MPLS label. An ordered list of segments is encoded as a stack of labels. The segment to process is on the top of the stack. Upon completion of a segment, the related label is popped from the stack.

Segment Routing can be applied to the IPv6 architecture, with a new type of routing header. A segment is encoded as an IPv6 address. An ordered list of segments is encoded as an ordered list of IPv6 addresses in the routing header. The active segment is indicated by the Destination Address of the packet. The next active segment is indicated by a pointer in the new routing header.

Requirements Language

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

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction 3
2. Terminology 5
3. Link-State IGP Segments 8
  3.1. IGP-Prefix Segment, Prefix-SID 8
    3.1.1. Prefix-SID Algorithm 9
    3.1.2. SR-MPLS 10
    3.1.3. SRv6 11
  3.2. IGP-Node Segment, Node-SID 12
  3.3. IGP-Anycast Segment, Anycast SID 12
    3.3.1. Anycast SID in SR-MPLS 12
  3.4. IGP-Adjacency Segment, Adj-SID 15
    3.4.1. Parallel Adjacencies 16
    3.4.2. LAN Adjacency Segments 17
  3.5. Inter-Area Considerations 18

4. BGP Peering Segments 19
4.1. BGP Prefix Segment 19
4.2. BGP Peering Segments 19
5. Binding Segment 20
5.1. IGP Mirroring Context Segment 20
6. Multicast 21
7. IANA Considerations 21
8. Security Considerations 21
8.1. SR-MPLS 21
8.2. SRv6 23
8.3. Congestion Control 24
9. Manageability Considerations 24
10. Contributors 25
11. Acknowledgements 26
12. References 26
12.1. Normative References 26
12.2. Informative References 27
Authors' Addresses 30

1. Introduction

Segment Routing (SR) leverages the source routing paradigm. A node steers a packet through an SR Policy instantiated as an ordered list of instructions called segments. A segment can represent any instruction, topological or service-based. A segment can have a semantic local to an SR node or global within an SR domain. SR supports per-flow explicit routing while maintaining per-flow state only at the ingress nodes to the SR domain.

A segment is often referred to by its Segment Identifier (SID).

A segment may be associated with a topological instruction. A topological local segment may instruct a node to forward the packet via a specific outgoing interface. A topological global segment may instruct an SR domain to forward the packet via a specific path to a destination. Different segments may exist for the same destination, each with different path objectives (e.g., which metric is minimized, what constraints are specified).

A segment may be associated with a service instruction (e.g. the packet should be processed by a container or VM associated with the segment). A segment may be associated with a QoS treatment (e.g., shape the packets received with this segment at x Mbps).

The SR architecture supports any type of instruction associated with a segment.

The SR architecture supports any type of control-plane: distributed, centralized or hybrid.

In a distributed scenario, the segments are allocated and signaled by IS-IS or OSPF or BGP. A node individually decides to steer packets on a source-routed policy (e.g., pre-computed local protection [I-D.ietf-spring-resiliency-use-cases]) . A node individually computes the source-routed policy.

In a centralized scenario, the segments are allocated and instantiated by an SR controller. The SR controller decides which nodes need to steer which packets on which source-routed policies. The SR controller computes the source-routed policies. The SR architecture does not restrict how the controller programs the network. Likely options are NETCONF, PCEP and BGP. The SR architecture does not restrict the number of SR controllers. Specifically multiple SR controllers may program the same SR domain. The SR architecture allows these SR controllers to discover which SID's are instantiated at which nodes and which sets of local (SRLB) and global labels (SRGB) are available at which node.

A hybrid scenario complements a base distributed control-plane with a centralized controller. For example, when the destination is outside the IGP domain, the SR controller may compute a source-routed policy on behalf of an IGP node. The SR architecture does not restrict how the nodes which are part of the distributed control-plane interact with the SR controller. Likely options are PCEP and BGP.

Hosts MAY be part of an SR Domain. A centralized controller can inform hosts about policies either by pushing these policies to hosts or responding to requests from hosts.

The SR architecture can be instantiated on various dataplanes. This document introduces two dataplane instantiations of SR: SR over MPLS (SR-MPLS) and SR over IPv6 (SRv6).

Segment Routing can be directly applied to the MPLS architecture with no change on the forwarding plane [I-D.ietf-spring-segment-routing-mpls] A segment is encoded as an MPLS label. An SR Policy is instantiated as a stack of labels. The segment to process (the active segment) is on the top of the stack. Upon completion of a segment, the related label is popped from the stack.

Segment Routing can be applied to the IPv6 architecture with a new type of routing header called the SR header (SRH) [I-D.ietf-6man-segment-routing-header] . An instruction is associated with a segment and encoded as an IPv6 address. An SRv6 segment is also called an SRv6 SID. An SR Policy is instantiated as an ordered list of SRv6 SID's in the routing header. The active segment is indicated by the Destination Address(DA) of the packet. The next active segment is indicated by the SegmentsLeft (SL) pointer in the SRH. When an SRv6 SID is completed, the SL is decremented and the next segment is copied to the DA. When a packet is steered on an SR policy, the related SRH is added to the packet.

In the context of an IGP-based distributed control-plane, two topological segments are defined: the IGP adjacency segment and the IGP prefix segment.

In the context of a BGP-based distributed control-plane, two topological segments are defined: the BGP peering segment and the BGP prefix segment.

The headend of an SR Policy binds a SID (called Binding segment or BSID) to its policy. When the headend receives a packet with active segment matching the BSID of a local SR Policy, the headend steers the packet into the associated SR Policy.

This document defines the IGP, BGP and Binding segments for the SR-MPLS and SRv6 dataplanes.

Note: This document defines the architecture for Segment Routing, including definitions of basic objects and functions and a description of the overall design. It does NOT define the means of implementing the architecture - that is contained in numerous referencing documents, some of which are mentioned in this document as a convenience to the reader.

2. Terminology

SR-MPLS: the instantiation of SR on the MPLS dataplane

SRv6: the instantiation of SR on the IPv6 dataplane.

Segment: an instruction a node executes on the incoming packet (e.g., forward packet according to shortest path to destination, or, forward packet through a specific interface, or, deliver the packet to a given application/service instance).

SID: a segment identifier. Note that the term SID is commonly used in place of the term Segment, though this is technically imprecise as it overlooks any necessary translation.

SR-MPLS SID: an MPLS label or an index value into an MPLS label space explicitly associated with the segment.

SRv6 SID: an IPv6 address explicitly associated with the segment.

Segment Routing Domain (SR Domain): the set of nodes participating in the source based routing model. These nodes may be connected to the same physical infrastructure (e.g., a Service Provider's network). They may as well be remotely connected to each other (e.g., an enterprise VPN or an overlay). If multiple protocol instances are deployed, the SR domain most commonly includes all of the protocol instances in a network. However, some deployments may wish to sub-divide the network into multiple SR domains, each of which includes one or more protocol instances. It is expected that all nodes in an SR Domain are managed by the same administrative entity.

Active Segment: the segment that is used by the receiving router to process the packet. In the MPLS dataplane it is the top label. In the IPv6 dataplane it is the destination address. [I-D.ietf-6man-segment-routing-header].

PUSH: the instruction consisting of the insertion of a segment at the top of the segment list. In SR-MPLS the top of the segment list is the topmost (outer) label of the label stack. In SRv6, the top of the segment list is represented by the first segment in the Segment Routing Header as defined in [I-D.ietf-6man-segment-routing-header].

NEXT: when the active segment is completed, NEXT is the instruction consisting of the inspection of the next segment. The next segment becomes active. In SR-MPLS, NEXT is implemented as a POP of the top label. In SRv6, NEXT is implemented as the copy of the next segment from the SRH to the Destination Address of the IPv6 header.

CONTINUE: the active segment is not completed and hence remains active. In SR-MPLS, CONTINUE instruction is implemented as a SWAP of the top label. [RFC3031] In SRv6, this is the plain IPv6 forwarding action of a regular IPv6 packet according to its Destination Address.

SR Global Block (SRGB): the set of global segments in the SR Domain. If a node participates in multiple SR domains, there is one SRGB for each SR domain. In SR-MPLS, SRGB is a local property of a node and identifies the set of local labels reserved for global segments. In SR-MPLS, using identical SRGBs on all nodes within the SR Domain is strongly recommended. Doing so eases operations and troubleshooting as the same label represents the same global segment at each node. In SRv6, the SRGB is the set of global SRv6 SIDs in the SR Domain.

SR Local Block (SRLB): local property of an SR node. If a node participates in multiple SR domains, there is one SRLB for each SR domain. In SR-MPLS, SRLB is a set of local labels reserved for local segments. In SRv6, SRLB is a set of local IPv6 addresses reserved for local SRv6 SID's. In a controller-driven network, some controllers or applications may use the control plane to discover the available set of local segments.

Global Segment: a segment which is part of the SRGB of the domain. The instruction associated to the segment is defined at the SR Domain level. A topological shortest-path segment to a given destination within an SR domain is a typical example of a global segment.

Local Segment: In SR-MPLS, this is a local label outside the SRGB. It may be part of the explicitly advertised SRLB. In SRv6, this can be any IPv6 address i.e., the address may be part of the SRGB but used such that it has local significance. The instruction associated to the segment is defined at the node level.

IGP Segment: the generic name for a segment attached to a piece of information advertised by a link-state IGP, e.g. an IGP prefix or an IGP adjacency.

IGP-Prefix Segment: an IGP-Prefix Segment is an IGP Segment representing an IGP prefix. When an IGP-Prefix Segment is global within the SR IGP instance/topology it identifies an instruction to forward the packet along the path computed using the routing algorithm specified in the algorithm field, in the topology and the IGP instance where it is advertised. Also referred to as Prefix Segment.

Prefix SID: the SID of the IGP-Prefix Segment.

IGP-Anycast Segment: an IGP-Anycast Segment is an IGP-Prefix Segment which identify an anycast prefix advertised by a set of routers.

Anycast-SID: the SID of the IGP-Anycast Segment.

IGP-Adjacency Segment: an IGP-Adjacency Segment is an IGP Segment attached to a unidirectional adjacency or a set of unidirectional adjacencies. By default, an IGP-Adjacency Segment is local (unless explicitly advertised otherwise) to the node that advertises it. Also referred to as Adjacency Segment.

Adj-SID: the SID of the IGP-Adjacency Segment.

IGP-Node Segment: an IGP-Node Segment is an IGP-Prefix Segment which identifies a specific router (e.g., a loopback). Also referred to as Node Segment.

Node-SID: the SID of the IGP-Node Segment.

SR Policy: an ordered list of segments. The headend of an SR Policy steers packets onto the SR policy. The list of segments can be specified explicitly in SR-MPLS as a stack of labels and in SRv6 as an ordered list of SRv6 SID's. Alternatively, the list of segments is computed based on a destination and a set of optimization objective and constraints (e.g., latency, affinity, SRLG, ...). The computation can be local or delegated to a PCE server. An SR policy can be configured by the operator, provisioned via NETCONF [RFC6241] or provisioned via PCEP [RFC5440] . An SR policy can be used for traffic-engineering, OAM or FRR reasons.

Segment List Depth: the number of segments of an SR policy. The entity instantiating an SR Policy at a node N should be able to discover the depth insertion capability of the node N. For example, the PCEP SR capability advertisement described in [I-D.ietf-pce-segment-routing] is one means of discovering this capability.

Forwarding Information Base (FIB): the forwarding table of a node

3. Link-State IGP Segments

Within an SR domain, an SR-capable IGP node advertises segments for its attached prefixes and adjacencies. These segments are called IGP segments or IGP SIDs. They play a key role in Segment Routing and use-cases as they enable the expression of any path throughout the SR domain. Such a path is either expressed as a single IGP segment or a list of multiple IGP segments.

Advertisement of IGP segments requires extensions in link-state IGP protocols. These extensions are defined in
[I-D.ietf-isis-segment-routing-extensions]
[I-D.ietf-ospf-segment-routing-extensions]
[I-D.ietf-ospf-ospfv3-segment-routing-extensions]

3.1. IGP-Prefix Segment, Prefix-SID

An IGP-Prefix segment is an IGP segment attached to an IGP prefix. An IGP-Prefix segment is global (unless explicitly advertised otherwise) within the SR domain. The context for an IGP-Prefix segment includes the prefix, topology, and algorithm. Multiple SIDs MAY be allocated to the same prefix so long as the tuple <prefix, topology, algorithm> is unique.

Multiple instances and topologies are defined in IS-IS and OSPF in: [RFC5120], [RFC8202], [RFC6549] and [RFC4915].

3.1.1. Prefix-SID Algorithm

Segment Routing supports the use of multiple routing algorithms i.e, different constraint based shortest path calculations can be supported. An algorithm identifier is included as part of a Prefix-SID advertisement. Specification of how an algorithm specific path calculation is done is required in the document defining the algorithm.

This document defines two algorithms:

- "Shortest Path": this algorithm is the default behavior. The packet is forwarded along the well known ECMP-aware SPF algorithm employed by the IGPs. However it is explicitly allowed for a midpoint to implement another forwarding based on local policy. The "Shortest Path" algorithm is in fact the default and current behavior of most of the networks where local policies may override the SPF decision.

- "Strict Shortest Path (Strict-SPF)": This algorithm mandates that the packet is forwarded according to ECMP-aware SPF algorithm and instructs any router in the path to ignore any possible local policy overriding the SPF decision. The SID advertised with Strict-SPF algorithm ensures that the path the packet is going to take is the expected, and not altered, SPF path. Note that Fast Reroute (FRR) [RFC5714] mechanisms are still compliant with the Strict Shortest Path. In other words, a packet received with a Strict-SPF SID may be rerouted through a FRR mechanism. Strict-SPF uses the same topology used by "Shortest Path". Obviously, nodes which do not support Strict-SPF will not install forwarding entries for this algorithm. Restricting the topology only to those nodes which support this algorithm will not produce the desired forwarding paths since the desired behavior is to follow the path calculated by "Shortest Path". Therefore, a source SR node MUST NOT use a source-routing policy containing a strict SPF segment if the path crosses a node not supporting the strict-SPF algorithm.

An IGP-Prefix Segment identifies the path, to the related prefix, computed as per the associated algorithm. A packet injected anywhere within the SR domain with an active Prefix-SID is expected to be forwarded along a path computed using the specified algorithm. For this to be possible, a fully connected topology of routers supporting the specified algorithm is required.

3.1.2. SR-MPLS

When SR is used over the MPLS dataplane SIDs are an MPLS label or an index into an MPLS label space (either SRGB or SRLB).

Where possible, it is recommended that identical SRGBs be configured on all nodes in an SR Domain. This simplifies troubleshooting as the same label will be associated with the same prefix on all nodes. In addition, it simplifies support for anycast as detailed in Section 3.3.

The following behaviors are associated with SR operating over the MPLS dataplane:

- o the IGP signaling extension for IGP-Prefix segment includes a flag to indicate whether directly connected neighbors of the node on which the prefix is attached should perform the NEXT operation or the CONTINUE operation when processing the SID. This behavior is equivalent to Penultimate Hop Popping (NEXT) or Ultimate Hop Popping (CONTINUE) in MPLS.

- o A Prefix-SID is allocated in the form of an MPLS label (or an index in the SRGB) according to a process similar to IP address allocation. Typically, the Prefix-SID is allocated by policy by the operator (or NMS) and the SID very rarely changes.

- o While SR allows to attach a local segment to an IGP prefix, it is specifically assumed that when the terms "IGP-Prefix Segment" and "Prefix-SID" are used, the segment is global (the SID is allocated from the SRGB or as an index into the advertised SRGB). This is consistent with all the described use-cases that require global segments attached to IGP prefixes.

- o The allocation process MUST NOT allocate the same Prefix-SID to different IP prefixes.

- o If a node learns a Prefix-SID having a value that falls outside the locally configured SRGB range, then the node MUST NOT use the Prefix-SID and SHOULD issue an error log reporting a misconfiguration.

- o If a node N advertises Prefix-SID SID-R for a prefix R that is attached to N, if N specifies CONTINUE as the operation to be performed by directly connected neighbors, N MUST maintain the following FIB entry:

```
      Incoming Active Segment: SID-R
      Ingress Operation: NEXT
      Egress interface: NULL
```

- o A remote node M MUST maintain the following FIB entry for any learned Prefix-SID SID-R attached to IP prefix R:

```
     Incoming Active Segment: SID-R
     Ingress Operation:
        If the next-hop of R is the originator of R
        and instructed to remove the active segment: NEXT
        Else: CONTINUE
     Egress interface: the interface towards the next-hop along the
                       path computed using the algorithm advertised with
                       the SID toward prefix R.
```

As Prefix-SIDs are specific to a given algorithm, if traffic associated with an algorithm arrives at a node which does not support that algorithm the traffic will be dropped as there will be no forwarding entry matching the incoming label.

3.1.3. SRv6

When SR is used over the IPv6 dataplane:

- o A Prefix-SID is an IPv6 address.

- o An operator MUST explicitly instantiate an SRv6 SID. IPv6 node addresses are not SRv6 SIDs by default.

A node N advertising an IPv6 address R usable as a segment identifier MUST maintain the following FIB entry:

```
      Incoming Active Segment: R
      Ingress Operation: NEXT
      Egress interface: NULL
```

Note that forwarding to R does not require an entry in the FIBs of all other routers for R. Forwarding can be and most often will be achieved by a shorter mask prefix which covers R.

Independent of Segment Routing support, any remote IPv6 node will maintain a plain IPv6 FIB entry for any prefix, no matter if the prefix represents a segment or not. This allows forwarding of packets to the node which owns the SID even by nodes which do not support Segment Routing.

Support of multiple algorithms applies to SRv6. Since algorithm specific SIDs are simply IPv6 addresses, algorithm specific forwarding entries can be achieved by assigning algorithm specific subnets to the (set of) algorithm specific SIDs which a node allocates.

Nodes which do not support a given algorithm may still have a FIB entry covering an algorithm specific address even though an algorithm specific path has not been calculated by that node. This is mitigated by the fact that nodes which do not support a given algorithm will not be included in the topology associated with that algorithm specific SPF and so traffic using the algorithm specific destination will normally not flow via the excluded node. If such traffic were to arrive and be forwarded by such a node, it will still progress towards the destination node. The nexthop will either be a node which supports the algorithm - in which case the packet will be forwarded along algorithm specific paths (or be dropped if none are available) - or the nexthop will be a node which does NOT support the algorithm - in which case the packet will continue to be forwarded along Algorithm 0 paths towards the destination node.

3.2. IGP-Node Segment, Node-SID

An IGP Node-SID MUST NOT be associated with a prefix that is owned by more than one router within the same routing domain.

3.3. IGP-Anycast Segment, Anycast SID

An "Anycast Segment" or "Anycast SID" enforces the ECMP-aware shortest-path forwarding towards the closest node of the anycast set. This is useful to express macro-engineering policies or protection mechanisms.

An IGP-Anycast segment MUST NOT reference a particular node.

Within an anycast group, all routers in an SR domain MUST advertise the same prefix with the same SID value.

3.3.1. Anycast SID in SR-MPLS

```
                               +--------------+
                               |   Group A    |
                               |192.0.2.10/32 |
                               |    SID:100   |
                               |              |
                        +-----------A1---A3----------+
                        |      |    | \ / |   |      |
             SID:10     |      |    |  /  |   |      |     SID:30
       203.0.113.1/32   |      |    | / \ |   |      |  203.0.113.3/32
               PE1------R1----------A2---A4---------R3------PE3
                 \     /|      |              |      |\     /
                  \   / |      +--------------+      | \   /
                   \ /  |                            |  \ /
                    /   |                            |   /
                   / \  |                            |  / \
                  /   \ |      +--------------+      | /   \
                 /     \|      |              |      |/     \
               PE2------R2----------B1---B3---------R4------PE4
       203.0.113.2/32   |      |    | \ / |   |      | 203.0.113.4/32
             SID:20     |      |    |  /  |   |      |     SID:40
                        |      |    | / \ |   |      |
                        +-----------B2---B4----------+
                               |              |
                               |   Group B    |
                               | 192.0.2.1/32 |
                               |    SID:200   |
                               +--------------+
```

Figure 1: Transit device groups

The figure above describes a network example with two groups of transit devices. Group A consists of devices {A1, A2, A3 and A4}. They are all provisioned with the anycast address 192.0.2.10/32 and the anycast SID 100.

Similarly, group B consists of devices {B1, B2, B3 and B4} and are all provisioned with the anycast address 192.0.2.1/32, anycast SID 200. In the above network topology, each PE device has a path to each of the groups A and B.

PE1 can choose a particular transit device group when sending traffic to PE3 or PE4. This will be done by pushing the anycast SID of the group in the stack.

Processing the anycast, and subsequent segments, requires special care.

```
                         +-------------------------+
                         |       Group A           |
                         |     192.0.2.10/32       |
                         |        SID:100          |
                         |-------------------------|
                         |                         |
                         |   SRGB:         SRGB:   |
      SID:10             |(1000-2000)   (3000-4000)|             SID:30
        PE1---+       +-------A1-------------A3-------+       +---PE3
               \     /   |    | \           / |    |   \     /
                \   /    |    |  +-----+   /  |    |    \   /
         SRGB:   \ /     |    |         \ /   |    |     \ /   SRGB:
      (7000-8000) R1     |    |          \    |    |      R3 (6000-7000)
                 / \     |    |         / \   |    |     / \
                /   \    |    |  +-----+   \  |    |    /   \
               /     \   |    | /           \ |    |   /     \
        PE2---+       +-------A2-------------A4-------+       +---PE4
      SID:20             |   SRGB:         SRGB:   |             SID:40
                         |(2000-3000)   (4000-5000)|
                         |                         |
                         +-------------------------+
```

Figure 2: Transit paths via anycast group A

Considering an MPLS deployment, in the above topology, if device PE1 (or PE2) requires to send a packet to the device PE3 (or PE4) it needs to encapsulate the packet in an MPLS payload with the following stack of labels.

o Label allocated by R1 for anycast SID 100 (outer label).

o Label allocated by the nearest router in group A for SID 30 (for destination PE3).

While the first label is easy to compute, in this case since there are more than one topologically nearest devices (A1 and A2), unless A1 and A2 allocated the same label value to the same prefix, determining the second label is impossible. Devices A1 and A2 may be devices from different hardware vendors. If both don’t allocate the same label value for SID 30, it is impossible to use the anycast group "A" as a transit anycast group towards PE3. Hence, PE1 (or PE2) cannot compute an appropriate label stack to steer the packet exclusively through the group A devices. Same holds true for devices PE3 and PE4 when trying to send a packet to PE1 or PE2.

To ease the use of anycast segment, it is recommended to configure identical SRGBs on all nodes of a particular anycast group. Using this method, as mentioned above, computation of the label following the anycast segment is straightforward.

Using anycast segment without configuring identical SRGBs on all nodes belonging to the same device group may lead to misrouting (in an MPLS VPN deployment, some traffic may leak between VPNs).

3.4. IGP-Adjacency Segment, Adj-SID

The adjacency is formed by the local node (i.e., the node advertising the adjacency in the IGP) and the remote node (i.e., the other end of the adjacency). The local node MUST be an IGP node. The remote node may be an adjacent IGP neighbor or a non-adjacent neighbor (e.g., a Forwarding Adjacency, [RFC4206]).

A packet injected anywhere within the SR domain with a segment list {SN, SNL}, where SN is the Node-SID of node N and SNL is an Adj-SID attached by node N to its adjacency over link L, will be forwarded along the shortest-path to N and then be switched by N, without any IP shortest-path consideration, towards link L. If the Adj-SID identifies a set of adjacencies, then the node N load-balances the traffic among the various members of the set.

Similarly, when using a global Adj-SID, a packet injected anywhere within the SR domain with a segment list {SNL}, where SNL is a global Adj-SID attached by node N to its adjacency over link L, will be forwarded along the shortest-path to N and then be switched by N, without any IP shortest-path consideration, towards link L. If the Adj-SID identifies a set of adjacencies, then the node N does load-balance the traffic among the various members of the set. The use of global Adj-SID allows to reduce the size of the segment list when expressing a path at the cost of additional state (i.e.: the global Adj-SID will be inserted by all routers within the area in their forwarding table).

An "IGP Adjacency Segment" or "Adj-SID" enforces the switching of the packet from a node towards a defined interface or set of interfaces. This is key to theoretically prove that any path can be expressed as a list of segments.

The encodings of the Adj-SID include a set of flags supporting the following functionalities:

o Eligible for Protection (e.g., using IPFRR or MPLS-FRR). Protection allows that in the event the interface(s) associated with the Adj-SID are down, that the packet can still be forwarded via an alternate path. The use of protection is clearly a policy based decision i.e., for a given policy protection may or may not be desirable.

- o Indication whether the Adj-SID has local or global scope. Default scope SHOULD be Local.

- o Indication whether the Adj-SID is persistent across control plane restarts. Persistence is a key attribute in ensuring that an SR Policy does not temporarily result in misforwarding due to reassignment of an Adj-SID.

A weight (as described below) is also associated with the Adj-SID advertisement.

A node SHOULD allocate one Adj-SID for each of its adjacencies.

A node MAY allocate multiple Adj-SIDs for the same adjacency. An example is to support an Adj-SID which is eligible for protection and an Adj-SID which is NOT eligible for protection.

A node MAY associate the same Adj-SID to multiple adjacencies.

In order to be able to advertise in the IGP all the Adj-SIDs representing the IGP adjacencies between two nodes, parallel adjacency suppression MUST NOT be performed by the IGP.

When a node binds an Adj-SID to a local data-link L, the node MUST install the following FIB entry:

```
Incoming Active Segment: V
Ingress Operation: NEXT
Egress Interface: L
```

The Adj-SID implies, from the router advertising it, the forwarding of the packet through the adjacency(ies) identified by the Adj-SID, regardless of its IGP/SPF cost. In other words, the use of adjacency segments overrides the routing decision made by the SPF algorithm.

3.4.1. Parallel Adjacencies

Adj-SIDs can be used in order to represent a set of parallel interfaces between two adjacent routers.

A node MUST install a FIB entry for any locally originated adjacency segment (Adj-SID) of value W attached to a set of links B with:

Incoming Active Segment: W
Ingress Operation: NEXT
Egress interface: load-balance between any data-link within set B

When parallel adjacencies are used and associated to the same Adj-SID, and in order to optimize the load balancing function, a "weight" factor can be associated to the Adj-SID advertised with each adjacency. The weight tells the ingress (or an SDN/orchestration system) about the load-balancing factor over the parallel adjacencies. As shown in Figure 3, A and B are connected through two parallel adjacencies

```
              link-1
            +--------+
            |        |
        S---A        B---C
            |        |
            +--------+
              link-2
```

Figure 3: Parallel Links and Adj-SIDs

Node A advertises following Adj-SIDs and weights:

- o Link-1: Adj-SID 1000, weight: 1

- o Link-2: Adj-SID 1000, weight: 2

Node S receives the advertisements of the parallel adjacencies and understands that by using Adj-SID 1000 node A will load-balance the traffic across the parallel links (link-1 and link-2) according to a 1:2 ratio i.e., twice as many packets will flow over Link-2 as compared to Link-1.

3.4.2. LAN Adjacency Segments

In LAN subnetworks, link-state protocols define the concept of Designated Router (DR, in OSPF) or Designated Intermediate System (DIS, in IS-IS) that conduct flooding in broadcast subnetworks and that describe the LAN topology in a special routing update (OSPF Type2 LSA or IS-IS Pseudonode LSP).

The difficulty with LANs is that each router only advertises its connectivity to the DR/DIS and not to each of the individual nodes in the LAN. Therefore, additional protocol mechanisms (IS-IS and OSPF) are necessary in order for each router in the LAN to advertise an Adj-SID associated to each neighbor in the LAN.

3.5. Inter-Area Considerations

In the following example diagram it is assumed that the all areas are part of a single SR Domain.

The example here below assumes the IPv6 control plane with the MPLS dataplane.

```
                 !          !
                 !          !
        B------C-----F----G-----K
       /       |          |     |
 S---A/        |          |     |
      \        |          |     |
       \D------I----------J-----L----Z (2001:DB8::2:1/128, Node-SID 150)
               !          !
       Area-1  ! Backbone ! Area 2
               !   area   !
```

Figure 4: Inter-Area Topology Example

In area 2, node Z allocates Node-SID 150 to his local IPv6 prefix 2001:DB8::2:1/128.

Area Border Routers (ABR) G and J will propagate the prefix and its SIDs into the backbone area by creating a new instance of the prefix according to normal inter-area/level IGP propagation rules.

Nodes C and I will apply the same behavior when leaking prefixes from the backbone area down to area 1. Therefore, node S will see prefix 2001:DB8::2:1/128 with Prefix-SID 150 and advertised by nodes C and I.

It therefore results that a Prefix-SID remains attached to its related IGP Prefix through the inter-area process, which is the expected behavior in a single SR Domain.

When node S sends traffic to 2001:DB8::2:1/128, it pushes Node-SID(150) as active segment and forward it to A.

When packet arrives at ABR I (or C), the ABR forwards the packet according to the active segment (Node-SID(150)). Forwarding continues across area borders, using the same Node-SID(150), until the packet reaches its destination.

4. BGP Peering Segments

BGP segments may be allocated and distributed by BGP.

4.1. BGP Prefix Segment

A BGP-Prefix segment is a BGP segment attached to a BGP prefix.

A BGP-Prefix segment is global (unless explicitly advertised otherwise) within the SR domain.

The BGP Prefix SID is the BGP equivalent to the IGP Prefix Segment.

A likely use-case for the BGP Prefix Segment is an IGP-free hyper-scale spine-leaf topology where connectivity is learned solely via BGP [RFC7938]

4.2. BGP Peering Segments

In the context of BGP Egress Peer Engineering (EPE), as described in [I-D.ietf-spring-segment-routing-central-epe], an EPE enabled Egress PE node MAY advertise segments corresponding to its attached peers. These segments are called BGP peering segments or BGP peering SIDs. They enable the expression of source-routed inter-domain paths.

An ingress border router of an AS may compose a list of segments to steer a flow along a selected path within the AS, towards a selected egress border router C of the AS and through a specific peer. At minimum, a BGP peering Engineering policy applied at an ingress PE involves two segments: the Node SID of the chosen egress PE and then the BGP peering segment for the chosen egress PE peer or peering interface.

Three types of BGP peering segments/SIDs are defined: PeerNode SID, PeerAdj SID and PeerSet SID.

- o PeerNode SID: a BGP PeerNode segment/SID is a local segment. At the BGP node advertising it, its semantics is:

  * SR header operation: NEXT.

  * Next-Hop: the connected peering node to which the segment is related.

- o PeerAdj SID: a BGP PeerAdj segment/SID is a local segment. At the BGP node advertising it, the semantic is:

  * SR header operation: NEXT.

   * Next-Hop: the peer connected through the interface to which the segment is related.

- PeerSet SID. a BGP PeerSet segment/SID is a local segment. At the BGP node advertising it, the semantic is:

   * SR header operation: NEXT.

   * Next-Hop: load-balance across any connected interface to any peer in the related group.

   A peer set could be all the connected peers from the same AS or a subset of these. A group could also span across AS. The group definition is a policy set by the operator.

The BGP extensions necessary in order to signal these BGP peering segments are defined in [I-D.ietf-idr-bgpls-segment-routing-epe]

5. Binding Segment

In order to provide greater scalability, network opacity, and service independence, SR utilizes a Binding SID (BSID). The BSID is bound to an SR policy, instantiation of which may involve a list of SIDs. Any packets received with active segment = BSID are steered onto the bound SR Policy.

A BSID may either be a local or a global SID. If local, a BSID SHOULD be allocated from the SRLB. If global, a BSID MUST be allocated from the SRGB.

Use of a BSID allows the instantiation of the policy (the SID list) to be stored only on the node(s) which need to impose the policy. Direction of traffic to a node supporting the policy then only requires imposition of the BSID. If the policy changes, this also means that only the nodes imposing the policy need to be updated. Users of the policy are not impacted.

5.1. IGP Mirroring Context Segment

One use case for a Binding Segment is to provide support for an IGP node to advertise its ability to process traffic originally destined to another IGP node, called the Mirrored node and identified by an IP address or a Node-SID, provided that a "Mirroring Context" segment be inserted in the segment list prior to any service segment local to the mirrored node.

When a given node B wants to provide egress node A protection, it advertises a segment identifying node's A context. Such segment is called "Mirror Context Segment" and identified by the Mirror SID.

The Mirror SID is advertised using the binding segment defined in SR IGP protocol extensions [I-D.ietf-isis-segment-routing-extensions] .

In the event of a failure, a point of local repair (PLR) diverting traffic from A to B does a PUSH of the Mirror SID on the protected traffic. B, when receiving the traffic with the Mirror SID as the active segment, uses that segment and processes underlying segments in the context of A.

6. Multicast

Segment Routing is defined for unicast. The application of the source-route concept to Multicast is not in the scope of this document.

7. IANA Considerations

This document does not require any action from IANA.

8. Security Considerations

Segment Routing is applicable to both MPLS and IPv6 data planes.

Segment Routing adds some meta-data (instructions) to the packet, with the list of forwarding path elements (e.g., nodes, links, services, etc.) that the packet must traverse. It has to be noted that the complete source routed path may be represented by a single segment. This is the case of the Binding SID.

SR by default operates within a trusted domain. Traffic MUST be filtered at the domain boundaries.

The use of best practices to reduce the risk of tampering within the trusted domain is important. Such practices are discussed in [RFC4381] and are applicable to both SR-MPLS and SRv6.

8.1. SR-MPLS

When applied to the MPLS data plane, Segment Routing does not introduce any new behavior or any change in the way MPLS data plane works. Therefore, from a security standpoint, this document does not define any additional mechanism in the MPLS data plane.

SR allows the expression of a source routed path using a single segment (the Binding SID). Compared to RSVP-TE which also provides explicit routing capability, there are no fundamental differences in term of information provided. Both RSVP-TE and Segment Routing may express a source routed path using a single segment.

When a path is expressed using a single label, the syntax of the meta-data is equivalent between RSVP-TE [RFC3209] and SR.

When a source routed path is expressed with a list of segments additional meta-data is added to the packet consisting of the source routed path the packet must follow expressed as a segment list.

When a path is expressed using a label stack, if one has access to the meaning (i.e.: the Forwarding Equivalence Class) of the labels, one has the knowledge of the explicit path. For the MPLS data plane, as no data plane modification is required, there is no fundamental change of capability. Yet, the occurrence of label stacking will increase.

SR domain boundary routers MUST filter any external traffic destined to a label associated with a segment within the trusted domain. This includes labels within the SRGB of the trusted domain, labels within the SRLB of the specific boundary router, and labels outside either of these blocks. External traffic is any traffic received from an interface connected to a node outside the domain of trust.

From a network protection standpoint, there is an assumed trust model such that any node imposing a label stack on a packet is assumed to be allowed to do so. This is a significant change compared to plain IP offering shortest path routing but not fundamentally different compared to existing techniques providing explicit routing capability such as RSVP-TE. By default, the explicit routing information MUST NOT be leaked through the boundaries of the administered domain. Segment Routing extensions that have been defined in various protocols, leverage the security mechanisms of these protocols such as encryption, authentication, filtering, etc.

In the general case, a segment routing capable router accepts and install labels only if these labels have been previously advertised by a trusted source. The received information is validated using existing control plane protocols providing authentication and security mechanisms. Segment Routing does not define any additional security mechanism in existing control plane protocols.

Segment Routing does not introduce signaling between the source and the mid points of a source routed path. With SR, the source routed path is computed using SIDs previously advertised in the IP control plane. Therefore, in addition to filtering and controlled advertisement of SIDs at the boundaries of the SR domain, filtering in the data plane is also required. Filtering MUST be performed on the forwarding plane at the boundaries of the SR domain and may require looking at multiple labels/instruction.

For the MPLS data plane, there are no new requirements as the existing MPLS architecture already allows such source routing by stacking multiple labels. And for security protection, [RFC4381] and [RFC5920] already call for the filtering of MPLS packets on trust boundaries.

8.2. SRv6

When applied to the IPv6 data plane, Segment Routing does introduce the Segment Routing Header (SRH, [I-D.ietf-6man-segment-routing-header]) which is a type of Routing Extension header as defined in [RFC8200].

The SRH adds some meta-data to the IPv6 packet, with the list of forwarding path elements (e.g., nodes, links, services, etc.) that the packet must traverse and that are represented by IPv6 addresses. A complete source routed path may be encoded in the packet using a single segment (single IPv6 address).

SR domain boundary routers MUST filter any external traffic destined to an address within the SRGB of the trusted domain or the SRLB of the specific boundary router. External traffic is any traffic received from an interface connected to a node outside the domain of trust.

From a network protection standpoint, there is an assumed trust model such that any node adding an SRH to the packet is assumed to be allowed to do so. Therefore, by default, the explicit routing information MUST NOT be leaked through the boundaries of the administered domain. Segment Routing extensions that have been defined in various protocols, leverage the security mechanisms of these protocols such as encryption, authentication, filtering, etc.

In the general case, an SR IPv6 router accepts and install segments identifiers (in the form of IPv6 addresses), only if these SIDs are advertised by a trusted source. The received information is validated using existing control plane protocols providing authentication and security mechanisms. Segment Routing does not define any additional security mechanism in existing control plane protocols.

Problems which may arise when the above behaviors are not implemented or when the assumed trust model is violated (e.g., through a security breach) include:

- o Malicious looping
- o Evasion of access controls
- o Hiding the source of DOS attacks

Security concerns with source routing at the IPv6 data plane are more completely discussed in [RFC5095]. The new IPv6-based segment routing header is defined in [I-D.ietf-6man-segment-routing-header]. This document also discusses the above security concerns.

8.3. Congestion Control

SR does not introduce new requirements for congestion control. By default, traffic delivery is assumed to be best effort. Congestion control may be implemented at endpoints. Where SR policies are in use bandwidth allocation may be managed by monitoring incoming traffic associated with the binding SID identifying the SR policy. Other solutions such as [RFC8084] may be applicable.

9. Manageability Considerations

In SR enabled networks, the path the packet takes is encoded in the header. As the path is not signaled through a protocol, OAM mechanisms are necessary in order for the network operator to validate the effectiveness of a path as well as to check and monitor its liveness and performance. However, it has to be noted that SR allows to reduce substantially the number of states in transit nodes and hence the number of elements that a transit node has to manage is smaller.

SR OAM use cases for the MPLS data plane are defined in [I-D.ietf-spring-oam-usecase]. SR OAM procedures for the MPLS data plane are defined in [RFC8287].

SR routers receive advertisements of SIDs (index, label or IPv6 address) from the different routing protocols being extended for SR. Each of these protocols have monitoring and troubleshooting mechanisms to provide operation and management functions for IP addresses that must be extended in order to include troubleshooting and monitoring functions of the SID.

SR architecture introduces the usage of global segments. Each global segment MUST be bound to a unique index or address within an SR domain. The management of the allocation of such index or address by the operator is critical for the network behavior to avoid situations like mis-routing. In addition to the allocation policy/tooling that the operator will have in place, an implementation SHOULD protect the network in case of conflict detection by providing a deterministic resolution approach.

When a path is expressed using a label stack, the occurrence of label stacking will increase. A node may want to signal in the control plane its ability in terms of size of the label stack it can support.

A YANG data model [RFC6020] for segment routing configuration and operations has been defined in [I-D.ietf-spring-sr-yang].

When Segment Routing is applied to the IPv6 data plane, segments are identified through IPv6 addresses. The allocation, management and troubleshooting of segment identifiers is no different than the existing mechanisms applied to the allocation and management of IPv6 addresses.

The DA of the packet gives the active segment address. The segment list in the SRH gives the entire path of the packet. The validation of the source routed path is done through inspection of DA and SRH present in the packet header matched to the equivalent routing table entries.

In the context of SR over the IPv6 data plane, the source routed path is encoded in the SRH as described in [I-D.ietf-6man-segment-routing-header]. The SR IPv6 source routed path is instantiated into the SRH as a list of IPv6 address where the active segment is in the Destination Address (DA) field of the IPv6 packet header. Typically, by inspecting in any node the packet header, it is possible to derive the source routed path it belongs to. Similar to the context of SR over MPLS data plane, an implementation may originate path control and monitoring packets where the source routed path is inserted in the SRH and where each segment of the path inserts in the packet the relevant data in order to measure the end to end path and performance.

10. Contributors

The following people have substantially contributed to the definition of the Segment Routing architecture and to the editing of this document:

Ahmed Bashandy
Cisco Systems, Inc.
Email: bashandy@cisco.com

Martin Horneffer
Deutsche Telekom
Email: Martin.Horneffer@telekom.de

Wim Henderickx
Nokia
Email: wim.henderickx@nokia.com

Jeff Tantsura
Email: jefftant@gmail.com

Edward Crabbe
Email: edward.crabbe@gmail.com

Igor Milojevic
Email: milojevicigor@gmail.com

Saku Ytti
TDC
Email: saku@ytti.fi

11. Acknowledgements

We would like to thank Dave Ward, Peter Psenak, Dan Frost, Stewart Bryant, Pierre Francois, Thomas Telkamp, Ruediger Geib, Hannes Gredler, Pushpasis Sarkar, Eric Rosen, Chris Bowers and Alvaro Retana for their comments and review of this document.

12. References

12.1. Normative References

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

[RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, DOI 10.17487/RFC3031, January 2001, <https://www.rfc-editor.org/info/rfc3031>.

[RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <https://www.rfc-editor.org/info/rfc8200>.

12.2. Informative References

[I-D.ietf-6man-segment-routing-header]
Previdi, S., Filsfils, C., Raza, K., Dukes, D., Leddy, J., Field, B., daniel.voyer@bell.ca, d., daniel.bernier@bell.ca, d., Matsushima, S., Leung, I., Linkova, J., Aries, E., Kosugi, T., Vyncke, E., Lebrun, D., Steinberg, D., and R. Raszuk, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-08 (work in progress), January 2018.

[I-D.ietf-idr-bgpls-segment-routing-epe]
Previdi, S., Filsfils, C., Patel, K., Ray, S., and J. Dong, "BGP-LS extensions for Segment Routing BGP Egress Peer Engineering", draft-ietf-idr-bgpls-segment-routing-epe-14 (work in progress), December 2017.

[I-D.ietf-isis-segment-routing-extensions]
Previdi, S., Ginsberg, L., Filsfils, C., Bashandy, A., Gredler, H., Litkowski, S., Decraene, B., and J. Tantsura, "IS-IS Extensions for Segment Routing", draft-ietf-isis-segment-routing-extensions-15 (work in progress), December 2017.

[I-D.ietf-ospf-ospfv3-segment-routing-extensions]
Psenak, P., Previdi, S., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPFv3 Extensions for Segment Routing", draft-ietf-ospf-ospfv3-segment-routing-extensions-10 (work in progress), September 2017.

[I-D.ietf-ospf-segment-routing-extensions]
Psenak, P., Previdi, S., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPF Extensions for Segment Routing", draft-ietf-ospf-segment-routing-extensions-24 (work in progress), December 2017.

[I-D.ietf-pce-segment-routing]
Sivabalan, S., Filsfils, C., Tantsura, J., Henderickx, W., and J. Hardwick, "PCEP Extensions for Segment Routing", draft-ietf-pce-segment-routing-11 (work in progress), November 2017.

[I-D.ietf-spring-oam-usecase]
Geib, R., Filsfils, C., Pignataro, C., and N. Kumar, "A Scalable and Topology-Aware MPLS Dataplane Monitoring System", draft-ietf-spring-oam-usecase-10 (work in progress), December 2017.

[I-D.ietf-spring-resiliency-use-cases]
Filsfils, C., Previdi, S., Decraene, B., and R. Shakir, "Resiliency use cases in SPRING networks", draft-ietf-spring-resiliency-use-cases-12 (work in progress), December 2017.

[I-D.ietf-spring-segment-routing-central-epe]
Filsfils, C., Previdi, S., Dawra, G., Aries, E., and D. Afanasiev, "Segment Routing Centralized BGP Egress Peer Engineering", draft-ietf-spring-segment-routing-central-epe-10 (work in progress), December 2017.

[I-D.ietf-spring-segment-routing-mpls]
Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls-11 (work in progress), October 2017.

[I-D.ietf-spring-sr-yang]
Litkowski, S., Qu, Y., Sarkar, P., and J. Tantsura, "YANG Data Model for Segment Routing", draft-ietf-spring-sr-yang-08 (work in progress), December 2017.

[RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, DOI 10.17487/RFC3209, December 2001, <https://www.rfc-editor.org/info/rfc3209>.

[RFC4206] Kompella, K. and Y. Rekhter, "Label Switched Paths (LSP) Hierarchy with Generalized Multi-Protocol Label Switching (GMPLS) Traffic Engineering (TE)", RFC 4206, DOI 10.17487/RFC4206, October 2005, <https://www.rfc-editor.org/info/rfc4206>.

[RFC4381] Behringer, M., "Analysis of the Security of BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4381, DOI 10.17487/RFC4381, February 2006, <https://www.rfc-editor.org/info/rfc4381>.

[RFC4915] Psenak, P., Mirtorabi, S., Roy, A., Nguyen, L., and P. Pillay-Esnault, "Multi-Topology (MT) Routing in OSPF", RFC 4915, DOI 10.17487/RFC4915, June 2007, <https://www.rfc-editor.org/info/rfc4915>.

[RFC5095] Abley, J., Savola, P., and G. Neville-Neil, "Deprecation of Type 0 Routing Headers in IPv6", RFC 5095, DOI 10.17487/RFC5095, December 2007, <https://www.rfc-editor.org/info/rfc5095>.

[RFC5120] Przygienda, T., Shen, N., and N. Sheth, "M-ISIS: Multi Topology (MT) Routing in Intermediate System to Intermediate Systems (IS-ISs)", RFC 5120, DOI 10.17487/RFC5120, February 2008, <https://www.rfc-editor.org/info/rfc5120>.

[RFC5440] Vasseur, JP., Ed. and JL. Le Roux, Ed., "Path Computation Element (PCE) Communication Protocol (PCEP)", RFC 5440, DOI 10.17487/RFC5440, March 2009, <https://www.rfc-editor.org/info/rfc5440>.

[RFC5714] Shand, M. and S. Bryant, "IP Fast Reroute Framework", RFC 5714, DOI 10.17487/RFC5714, January 2010, <https://www.rfc-editor.org/info/rfc5714>.

[RFC5920] Fang, L., Ed., "Security Framework for MPLS and GMPLS Networks", RFC 5920, DOI 10.17487/RFC5920, July 2010, <https://www.rfc-editor.org/info/rfc5920>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/info/rfc6020>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC6549] Lindem, A., Roy, A., and S. Mirtorabi, "OSPFv2 Multi-Instance Extensions", RFC 6549, DOI 10.17487/RFC6549, March 2012, <https://www.rfc-editor.org/info/rfc6549>.

[RFC7938] Lapukhov, P., Premji, A., and J. Mitchell, Ed., "Use of BGP for Routing in Large-Scale Data Centers", RFC 7938, DOI 10.17487/RFC7938, August 2016, <https://www.rfc-editor.org/info/rfc7938>.

[RFC8084] Fairhurst, G., "Network Transport Circuit Breakers", BCP 208, RFC 8084, DOI 10.17487/RFC8084, March 2017, <https://www.rfc-editor.org/info/rfc8084>.

[RFC8202] Ginsberg, L., Previdi, S., and W. Henderickx, "IS-IS Multi-Instance", RFC 8202, DOI 10.17487/RFC8202, June 2017, <https://www.rfc-editor.org/info/rfc8202>.

[RFC8287] Kumar, N., Ed., Pignataro, C., Ed., Swallow, G., Akiya, N., Kini, S., and M. Chen, "Label Switched Path (LSP) Ping/Traceroute for Segment Routing (SR) IGP-Prefix and IGP-Adjacency Segment Identifiers (SIDs) with MPLS Data Planes", RFC 8287, DOI 10.17487/RFC8287, December 2017, <https://www.rfc-editor.org/info/rfc8287>.

Authors' Addresses

Clarence Filsfils (editor)
Cisco Systems, Inc.
Brussels
BE

Email: cfilsfil@cisco.com

Stefano Previdi (editor)
Cisco Systems, Inc.
Italy

Email: stefano@previdi.net

Les Ginsberg
Cisco Systems, Inc

Email: ginsberg@cisco.com

Bruno Decraene
Orange
FR

Email: bruno.decraene@orange.com

Stephane Litkowski
Orange
FR

Email: stephane.litkowski@orange.com

Rob Shakir
Google, Inc.
1600 Amphitheatre Parkway
Mountain View, CA 94043
US

Email: robjs@google.com

Network Working Group A. Bashandy, Ed.
Internet-Draft Individual
Intended status: Standards Track C. Filsfils, Ed.
Expires: March 6, 2019 S. Previdi
Cisco Systems, Inc.
B. Decraene
S. Litkowski
Orange
September 02, 2018

Segment Routing interworking with LDP
draft-ietf-spring-segment-routing-ldp-interop-15

Abstract

A Segment Routing (SR) node steers a packet through a controlled set of instructions, called segments, by prepending the packet with an SR header. A segment can represent any instruction, topological or service-based. SR allows to enforce a flow through any topological path while maintaining per-flow state only at the ingress node to the SR domain.

The Segment Routing architecture can be directly applied to the MPLS data plane with no change in the forwarding plane. This document describes how Segment Routing operates in a network where LDP is deployed and in the case where SR-capable and non-SR-capable nodes coexist.

Requirements Language

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

Status of This Memo

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 March 6, 2019.

Copyright Notice

Table of Contents

1. Introduction 3
2. SR/LDP Ships-in-the-night coexistence 3
2.1. MPLS2MPLS, MPLS2IP and IP2MPLS co-existence 5
3. SR and LDP Interworking 6
3.1. LDP to SR 7
3.1.1. LDP to SR Behavior 7
3.2. SR to LDP 7
3.2.1. Segment Routing Mapping Server (SRMS) 9
3.2.2. SR to LDP Behavior 10
3.2.3. Interoperability of Multiple SRMSes and Prefix-SID advertisements 11
4. SR/LDP Interworking Use Cases 12
4.1. SR Protection of LDP-based Traffic 12
4.2. Eliminating Targeted LDP Session 14
4.3. Guaranteed FRR coverage 15
4.4. Inter-AS Option C, Carrier's Carrier 17
5. IANA Considerations 17
6. Manageability Considerations 17
6.1. SR and LDP co-existence 17
6.2. Dataplane Verification 18
7. Security Considerations 18
8. Acknowledgements 18
9. Contributors' Addresses 19
10. References 19
10.1. Normative References 19
10.2. Informative References 20

Appendix A. Migration from LDP to SR 21
Authors' Addresses 22

1. Introduction

Segment Routing, as described in [I-D.ietf-spring-segment-routing], can be used on top of the MPLS data plane without any modification as described in [I-D.ietf-spring-segment-routing-mpls].

Segment Routing control plane can co-exist with current label distribution protocols such as LDP ([RFC5036]).

This document outlines the mechanisms through which SR interworks with LDP in cases where a mix of SR-capable and non-SR-capable routers co-exist within the same network and more precisely in the same routing domain.

Section 2 describes the co-existence of SR with other MPLS Control Plane protocols. Section 3 documents the interworking between SR and LDP in the case of non-homogeneous deployment. Section 4 describes how a partial SR deployment can be used to provide SR benefits to LDP-based traffic including a possible application of SR in the context of inter-domain MPLS use-cases. Appendix A documents a method to migrate from LDP to SR-based MPLS tunneling.

Typically, an implementation will allow an operator to select (through configuration) which of the described modes of SR and LDP co-existence to use.

2. SR/LDP Ships-in-the-night coexistence

"MPLS Control Plane Client (MCC)" refers to any control plane protocol installing forwarding entries in the MPLS data plane. SR, LDP [RFC5036], RSVP-TE [RFC3209], BGP [RFC8277], etc are examples of MCCs.

An MCC, operating at node N, must ensure that the incoming label it installs in the MPLS data plane of Node N has been uniquely allocated to himself.

Segment Routing makes use of the Segment Routing Global Block (SRGB, as defined in [I-D.ietf-spring-segment-routing]) for the label allocation. The use of the SRGB allows SR to co-exist with any other MCC.

This is clearly the case for the adjacency segment: it is a local label allocated by the label manager, as for any MCC.

This is clearly the case for the prefix segment: the label manager allocates the SRGB set of labels to the SR MCC client and the operator ensures the unique allocation of each global prefix segment/ label within the allocated SRGB set.

Note that this static label allocation capability of the label manager has existed for many years across several vendors and hence is not new. Furthermore, note that the label-manager ability's to statically allocate a range of labels to a specific application is not new either. This is required for MPLS-TP operation. In this case, the range is reserved by the label manager and it is the MPLS-TP ([RFC5960]) NMS (acting as an MCC) that ensures the unique allocation of any label within the allocated range and the creation of the related MPLS forwarding entry.

Let us illustrate an example of ship-in-the-night (SIN) coexistence.

```
     PE2          PE4
       \          /
PE1----A----B---C---PE3
```

Figure 1: SIN coexistence

The EVEN VPN service is supported by PE2 and PE4 while the ODD VPN service is supported by PE1 and PE3. The operator wants to tunnel the ODD service via LDP and the EVEN service via SR.

This can be achieved in the following manner:

The operator configures PE1, PE2, PE3, PE4 with respective loopbacks 192.0.2.201/32, 192.0.2.202/32, 192.0.2.203/32, 192.0.2.204/32. These PE's advertised their VPN routes with next-hop set on their respective loopback address.

The operator configures A, B, C with respective loopbacks 192.0.2.1/32, 192.0.2.2/32, 192.0.2.3/32.

The operator configures PE2, A, B, C and PE4 with SRGB [100, 300].

The operator attaches the respective Node Segment Identifiers (Node-SID's, as defined in [I-D.ietf-spring-segment-routing]): 202, 101, 102, 103 and 204 to the loopbacks of nodes PE2, A, B, C and PE4. The Node-SID's are configured to request penultimate-hop-popping.

PE1, A, B, C and PE3 are LDP capable.

PE1 and PE3 are not SR capable.

PE3 sends an ODD VPN route to PE1 with next-hop 192.0.2.203 and VPN label 10001.

From an LDP viewpoint: PE1 received an LDP label binding (1037) for a forwarding equivalence class (FEC) 192.0.2.203/32 from its next-hop A. A received an LDP label binding (2048) for that FEC from its next-hop B. B received an LDP label binding (3059) for that FEC from its next-hop C. C received implicit-null LDP binding from its next-hop PE3.

As a result, PE1 sends its traffic to the ODD service route advertised by PE3 to next-hop A with two labels: the top label is 1037 and the bottom label is 10001. Node A swaps 1037 with 2048 and forwards to B. B swaps 2048 with 3059 and forwards to C. C pops 3059 and forwards to PE3.

PE4 sends an EVEN VPN route to PE2 with next-hop 192.0.2.204 and VPN label 10002.

From an SR viewpoint: PE2 maps the IGP route 192.0.2.204/32 onto Node-SID 204; node A swaps 204 with 204 and forwards to B; B swaps 204 with 204 and forwards to C; C pops 204 and forwards to PE4.

As a result, PE2 sends its traffic to the VPN service route advertised by PE4 to next-hop A with two labels: the top label is 204 and the bottom label is 10002. Node A swaps 204 with 204 and forwards to B. B swaps 204 with 204 and forwards to C. C pops 204 and forwards to PE4.

The two modes of MPLS tunneling co-exist.

The ODD service is tunneled from PE1 to PE3 through a continuous LDP LSP traversing A, B and C.

The EVEN service is tunneled from PE2 to PE4 through a continuous SR node segment traversing A, B and C.

2.1. MPLS2MPLS, MPLS2IP and IP2MPLS co-existence

MPLS2MPLS refers to the forwarding behavior where a router receives a labeled packet and switches it out as a labeled packet. Several MPLS2MPLS entries may be installed in the data plane for the same prefix.

Let us examine A’s MPLS forwarding table as an example:

Incoming label: 1037

- outgoing label: 2048
- outgoing next-hop: B

Note: this entry is programmed by LDP for 192.0.2.203/32

Incoming label: 203

- outgoing label: 203
- outgoing next-hop: B

Note: this entry is programmed by SR for 192.0.2.203/32

These two entries can co-exist because their incoming label is unique. The uniqueness is guaranteed by the label manager allocation rules.

The same applies for the MPLS2IP forwarding entries. MPLS2IP is the forwarding behavior where a router receives a label IPv4/IPv6 packet with one label only, pops the label, and switches the packet out as IPv4/IPv6. For IP2MPLS coexistence, refer to Section 6.1.

3. SR and LDP Interworking

This section analyzes the case where SR is available in one part of the network and LDP is available in another part. It describes how a continuous MPLS tunnel can be built throughout the network.

```
          PE2            PE4
            \            /
       PE1----P5--P6--P7--P8---PE3
```

Figure 2: SR and LDP Interworking

Let us analyze the following example:

P6, P7, P8, PE4 and PE3 are LDP capable.

PE1, PE2, P5 and P6 are SR capable. PE1, PE2, P5 and P6 are configured with SRGB (100, 200) and respectively with node segments 101, 102, 105 and 106.

A service flow must be tunneled from PE1 to PE3 over a continuous MPLS tunnel encapsulation and hence SR and LDP need to interwork.

3.1. LDP to SR

In this section, a right-to-left traffic flow is analyzed.

PE3 has learned a service route whose next-hop is PE1. PE3 has an LDP label binding from the next-hop P8 for the FEC "PE1". Hence PE3 sends its service packet to P8 as per classic LDP behavior.

P8 has an LDP label binding from its next-hop P7 for the FEC "PE1" and hence P8 forwards to P7 as per classic LDP behavior.

P7 has an LDP label binding from its next-hop P6 for the FEC "PE1" and hence P7 forwards to P6 as per classic LDP behavior.

P6 does not have an LDP binding from its next-hop P5 for the FEC "PE1". However P6 has an SR node segment to the IGP route "PE1". Hence, P6 forwards the packet to P5 and swaps its local LDP-label for FEC "PE1" by the equivalent node segment (i.e. 101).

P5 pops 101 (assuming PE1 advertised its node segment 101 with the penultimate-pop flag set) and forwards to PE1.

PE1 receives the tunneled packet and processes the service label.

The end-to-end MPLS tunnel is built from an LDP LSP from PE3 to P6 and the related node segment from P6 to PE1.

3.1.1. LDP to SR Behavior

It has to be noted that no additional signaling or state is required in order to provide interworking in the direction LDP to SR.

A SR node having LDP neighbors MUST create LDP bindings for each Prefix-SID learned in the SR domain by treating SR learned labels as if they were learned through an LDP neighbot. In addition for each FEC, the SR node stitches the incoming LDP label to the outgoing SR label. This has to be done in both LDP independent and ordered label distribution control modes as defined in [RFC5036].

3.2. SR to LDP

In this section, the left-to-right traffic flow is analyzed.

This section defines the Segment Routing Mapping Server (SRMS). The SRMS is a IGP node advertising mapping between Segment Identifiers (SID) and prefixes advertised by other IGP nodes. The SRMS uses a dedicated IGP extension (IS-IS, OSPFv2 and OSPFv3) which is protocol specific and defined in [I-D.ietf-isis-segment-routing-extensions], [I-D.ietf-ospf-segment-routing-extensions], and [I-D.ietf-ospf-ospfv3-segment-routing-extensions].

The SRMS function of a SR capable router allows distribution of mappings for prefixes not locally attached to the advertising router and therefore allows advertisement of mappings on behalf of non-SR capable routers.

The SRMS is a control plane only function which may be located anywhere in the IGP flooding scope. At least one SRMS server MUST exist in a routing domain to advertise prefix-SIDs on behalf non-SR nodes, thereby allowing non-LDP routers to send and receive labeled traffic from LDP-only routers. Multiple SRMSs may be present in the same network (for redundancy). This implies that there are multiple ways a prefix-to-SID mapping can be advertised. Conflicts resulting from inconsistent advertisements are addressed by [I-D.ietf-spring-segment-routing-mpls].

The example diagram depicted in Figure 2 assumes that the operator configures P5 to act as a Segment Routing Mapping Server (SRMS) and advertises the following mappings: (P7, 107), (P8, 108), (PE3, 103) and (PE4, 104).

The mappings advertised by one or more SRMSs result from local policy information configured by the operator.

If PE3 had been SR capable, the operator would have configured PE3 with node segment 103. Instead, as PE3 is not SR capable, the operator configures that policy at the SRMS and it is the latter which advertises the mapping.

The mapping server advertisements are only understood by SR capable routers. The SR capable routers install the related node segments in the MPLS data plane exactly like the node segments had been advertised by the nodes themselves.

For example, PE1 installs the node segment 103 with next-hop P5 exactly as if PE3 had advertised node segment 103.

PE1 has a service route whose next-hop is PE3. PE1 has a node segment for that IGP route: 103 with next-hop P5. Hence PE1 sends its service packet to P5 with two labels: the bottom label is the service label and the top label is 103.

P5 swaps 103 for 103 and forwards to P6.

P6’s next-hop for the IGP route "PE3" is not SR capable (P7 does not advertise the SR capability). However, P6 has an LDP label binding from that next-hop for the same FEC (e.g. LDP label 1037). Hence, P6 swaps 103 for 1037 and forwards to P7.

P7 swaps this label with the LDP-label received from P8 and forwards to P8.

P8 pops the LDP label and forwards to PE3.

PE3 receives the tunneled packet and processes the service label.

The end-to-end MPLS tunnel is built from an SR node segment from PE1 to P6 and an LDP LSP from P6 to PE3.

SR mapping advertisement for a given prefix provides no information about the Penultimate Hop Popping. Other mechanisms, such as IGP specific mechanisms ([I-D.ietf-isis-segment-routing-extensions], [I-D.ietf-ospf-segment-routing-extensions] and [I-D.ietf-ospf-ospfv3-segment-routing-extensions]), MAY be used to determine the Penultimate Hop Popping in such case.

Note: In the previous example, Penultimate Hop Popping is not performed at the SR/LDP border for segment 103 (PE3), because none of the routers in the SR domain is Penultimate Hop for segment 103. In this case P6 requires the presence of the segment 103 such as to map it to the LDP label 1037.

3.2.1. Segment Routing Mapping Server (SRMS)

This section specifies the concept and externally visible functionality of a segment routing mapping server (SRMS).

The purpose of a SRMS functionality is to support the advertisement of prefix-SIDs to a prefix without the need to explicitly advertise such assignment within a prefix reachability advertisment. Examples of explicit prefix-SID advertisment are the prefix-SID sub-TLVs defined in ([I-D.ietf-isis-segment-routing-extensions], [I-D.ietf-ospf-segment-routing-extensions], and [I-D.ietf-ospf-ospfv3-segment-routing-extensions]).

The SRMS functionality allows assigning of prefix-SIDs to prefixes owned by non-SR-capable routers as well as to prefixes owned by SR capable nodes. It is the former capability which is essential to the SR-LDP interworking described later in this section

The SRMS functionality consists of two functional blocks: the Mapping Server (MS) and Mapping Client (MC).

A MS is a node that advertises an SR mappings. Advertisements sent by an MS define the assignment of a prefix-SID to a prefix independent of the advertisment of reachability to the prefix itself. An MS MAY advertise SR mappings for any prefix whether or not it advertises reachability for the prefix and irrespective of whether that prefix is advertised by or even reachable through any router in the network.

An MC is a node that receives and uses the MS mapping advertisments. Note that a node may be both an MS and an MC. An MC interprets the SR mapping advertisment as an assignment of a prefix-SID to a prefix. For a given prefix, if an MC receives an SR mapping advertisement from a mapping server and also has received a prefix-SID advertisement for that same prefix in a prefix reachability advertisement, then the MC MUST prefer the SID advertised in the prefix reachability advertisement over the mapping server advertisement i.e., the mapping server advertisment MUST be ignored for that prefix. Hence assigning a prefix-SID to a prefix using the SRMS functionality does not preclude assigning the same or different prefix-SID(s) to the same prefix using explict prefix-SID advertisement such as the aforementioned prefix-SID sub-TLVs.

For example consider an IPv4 prefix advertisement received by an IS-IS router in the extended IP reachability TLV (TLV 135). Suppose TLV 135 contained the prefix-SID sub-TLV. If the router that receives TLV 135 with the prefix-SID sub-TLV also received an SR mapping advertisement for the same prefix through the SID/label binding TLV, then the receiving router must prefer the prefix-SID sub-TLV over the SID/label binding TLV for that prefix. Refer to ([I-D.ietf-isis-segment-routing-extensions], for details about the prefix-SID sub-TLV and SID/label binding TLV.

3.2.2. SR to LDP Behavior

SR to LDP interworking requires a SRMS as defined above.

Each SR capable router installs in the MPLS data plane Node-SIDs learned from the SRMS exactly like if these SIDs had been advertised by the nodes themselves.

A SR node having LDP neighbors MUST stitch the incoming SR label (whose SID is advertised by the SRMS) to the outgoing LDP label.

It has to be noted that the SR to LDP behavior does not propagate the status of the LDP FEC which was signaled if LDP was configured to use the ordered mode.

It has to be noted that in the case of SR to LDP, the label binding is equivalent to the independent LDP Label Distribution Control Mode ([RFC5036]) where a label in bound to a FEC independently from the received binding for the same FEC.

3.2.3. Interoperability of Multiple SRMSes and Prefix-SID advertisements

In the case of SR/LDP interoperability through the use of a SRMS, mappings are advertised by one or more SRMS.

SRMS function is implemented in the link-state protocol (such as IS-IS and OSPF). Link-state protocols allow propagation of updates across area boundaries and therefore SRMS advertisements are propagated through the usual inter-area advertisement procedures in link-state protocols.

Multiple SRMSs can be provisioned in a network for redundancy. Moreover, a preference mechanism may also be used among SRMSs so to deploy a primary/secondary SRMS scheme allowing controlled modification or migration of SIDs.

The content of SRMS advertisement (i.e.: mappings) are a matter of local policy determined by the operator. When multiple SRMSs are active, it is necessary that the information (mappings) advertised by the different SRMSs is aligned and consistent. The following mechanism is applied to determine the preference of SRMS advertisements:

If a node acts as an SRMS, it MAY advertise a preference to be associated with all SRMS SID advertisements sent by that node. The means of advertising the preference is defined in the protocol specific drafts e.g.,[I-D.ietf-isis-segment-routing-extensions] , [I-D.ietf-ospf-segment-routing-extensions], and [I-D.ietf-ospf-ospfv3-segment-routing-extensions]. The preference value is an unsigned 8 bit integer with the following properties:

0 - Reserved value indicating advertisements from that node MUST NOT be used.

1 - 255 Preference value (255 is most preferred)

Advertisement of a preference value is optional. Nodes which do not advertise a preference value are assigned a preference value of 128.

A MCC on a node receiving one or more SRMS mapping advertisements applies them as follows

- For any prefix for which it did not receive a prefix-SID advertisement, the MCC applies the SRMS mapping advertisments with the highest preference. The mechanism by which a prefix-SID is advertised for a given prefix is defined in the protocol specification , [I-D.ietf-isis-segment-routing-extensions], [I-D.ietf-ospf-segment-routing-extensions] and [I-D.ietf-ospf-ospfv3-segment-routing-extensions]

- If there is an incoming label collision as specified in [I-D.ietf-spring-segment-routing-mpls] , apply the steps specified in [I-D.ietf-spring-segment-routing-mpls] to resolve the collision.

When the SRMS advertise mappings, an implementation should provide a mechanism through which the operator determines which of the IP2MPLS mappings are preferred among the one advertised by the SRMS and the ones advertised by LDP.

4. SR/LDP Interworking Use Cases

SR can be deployed such as to enhance LDP transport. The SR deployment can be limited to the network region where the SR benefits are most desired.

4.1. SR Protection of LDP-based Traffic

In Figure 4, let us assume:

All link costs are 10 except FG which is 30.

All routers are LDP capable.

X, Y and Z are PE’s participating to an important service S.

The operator requires 50msec link-based Fast Reroute (FRR) for service S.

A, B, C, D, E, F and G are SR capable.

X, Y, Z are not SR capable, e.g. as part of a staged migration from LDP to SR, the operator deploys SR first in a sub-part of the network and then everywhere.

```
              X
              |
        Y--A---B---E--Z
           |   |    \
           D---C--F--G
                   30
```

Figure 3: SR/LDP interworking example

The operator would like to resolve the following issues:

To protect the link BA along the shortest-path of the important flow XY, B requires a Remote Loop-Free alternate (RLFA, [RFC7490]) repair tunnel to D and hence a targeted LDP session from B to D. Typically, network operators prefer avoiding these dynamically established multi-hop LDP sessions in order to reduce the number of protocols running in the network and hence simplify network operations.

There is no LFA/RLFA solution to protect the link BE along the shortest path of the important flow XZ. The operator wants a guaranteed link-based FRR solution.

The operator can meet these objectives by deploying SR only on A, B, C, D, E, F and G:

The operator configures A, B, C, D, E, F and G with SRGB [100, 200] and respective node segments 101, 102, 103, 104, 105, 106 and 107.

The operator configures D as an SR Mapping Server with the following policy mapping: (X, 201), (Y, 202), (Z, 203).

Each SR node automatically advertises local adjacency segment for its IGP adjacencies. Specifically, F advertises adjacency segment 9001 for its adjacency FG.

A, B, C, D, E, F and G keep their LDP capability and hence the flows XY and XZ are transported over end-to-end LDP LSP’s.

For example, LDP at B installs the following MPLS data plane entries:

Incoming label: local LDP label bound by B for FEC Y
   Outgoing label: LDP label bound by A for FEC Y
   Outgoing next-hop: A

Incoming label: local LDP label bound by B for FEC Z
   Outgoing label: LDP label bound by E for FEC Z

Outgoing next-hop: E

The novelty comes from how the backup chains are computed for these LDP-based entries. While LDP labels are used for the primary next-hop and outgoing labels, SR information is used for the FRR construction. In steady state, the traffic is transported over LDP LSP. In transient FRR state, the traffic is backup thanks to the SR enhanced capabilities.

The RLFA paths are dynamically pre-computed as defined in [RFC7490]. Typically, implementations allow to enable RLFA mechanism through a simple configuration command that triggers both the pre-computation and installation of the repair path. The details on how RLFA mechanisms are implemented and configured is outside the scope of this document and not relevant to the aspects of SR/LDP interwork explained in this document.

This helps meet the requirements of the operator:

Eliminate targeted LDP session.

Guaranteed FRR coverage.

Keep the traffic over LDP LSP in steady state.

Partial SR deployment only where needed.

4.2. Eliminating Targeted LDP Session

B’s MPLS entry to Y becomes:

- Incoming label: local LDP label bound by B for FEC Y
  Outgoing label: LDP label bound by A for FEC Y
  Backup outgoing label: SR node segment for Y {202}
  Outgoing next-hop: A
  Backup next-hop: repair tunnel: node segment to D {104}
   with outgoing next-hop: C

It has to be noted that D is selected as Remote Loop-Free Alternate (RLFA) as defined in [RFC7490].

In steady-state, X sends its Y-destined traffic to B with a top label which is the LDP label bound by B for FEC Y. B swaps that top label for the LDP label bound by A for FEC Y and forwards to A. A pops the LDP label and forwards to Y.

Upon failure of the link BA, B swaps the incoming top-label with the node segment for Y (202) and sends the packet onto a repair tunnel to

D (node segment 104). Thus, B sends the packet to C with the label stack {104, 202}. C pops the node segment 104 and forwards to D. D swaps 202 for 202 and forwards to A. A’s next-hop to Y is not SR capable and hence node A swaps the incoming node segment 202 to the LDP label announced by its next-hop (in this case, implicit null).

After IGP convergence, B’s MPLS entry to Y will become:

- Incoming label: local LDP label bound by B for FEC Y
  Outgoing label: LDP label bound by C for FEC Y
  Outgoing next-hop: C

And the traffic XY travels again over the LDP LSP.

Conclusion: the operator has eliminated the need for targeted LDP sessions (no longer required) and the steady-state traffic is still transported over LDP. The SR deployment is confined to the area where these benefits are required.

Despite that in general, an implementation would not require a manual configuration of LDP Targeted sessions however, it is always a gain if the operator is able to reduce the set of protocol sessions running on the network infrastructure.

4.3. Guaranteed FRR coverage

As mentioned in Section 4.1 above, in the example topology described in Figure 4, there is no RLFA-based solution for protecting the traffic flow YZ against the failure of link BE because there is no intersection between the extended P-space and Q-space (see [RFC7490] for details). However:

- G belongs to the Q space of Z.

- G can be reached from B via a "repair SR path" {106, 9001} that is not affected by failure of link BE (The method by which G and the repair tunnel to it from B are identified are out of scope of this document.)

B’s MPLS entry to Z becomes:

- Incoming label: local LDP label bound by B for FEC Z
  Outgoing label: LDP label bound by E for FEC Z
  Backup outgoing label: SR node segment for Z {203}
  Outgoing next-hop: E
  Backup next-hop: repair tunnel to G: {106, 9001}

  G is reachable from B via the combination of a node segment to F {106} and an adjacency segment FG {9001}

  Note that {106, 107} would have equally work. Indeed, in many case, P’s shortest path to Q is over the link PQ. The adjacency segment from P to Q is required only in very rare topologies where the shortest-path from P to Q is not via the link PQ.

In steady-state, X sends its Z-destined traffic to B with a top label which is the LDP label bound by B for FEC Z. B swaps that top label for the LDP label bound by E for FEC Z and forwards to E. E pops the LDP label and forwards to Z.

Upon failure of the link BE, B swaps the incoming top-label with the node segment for Z (203) and sends the packet onto a repair tunnel to G (node segment 106 followed by adjacency segment 9001). Thus, B sends the packet to C with the label stack {106, 9001, 203}. C pops the node segment 106 and forwards to F. F pops the adjacency segment 9001 and forwards to G. G swaps 203 for 203 and forwards to E. E’s next-hop to Z is not SR capable and hence E swaps the incoming node segment 203 for the LDP label announced by its next-hop (in this case, implicit null).

After IGP convergence, B’s MPLS entry to Z will become:

- Incoming label: local LDP label bound by B for FEC Z
  Outgoing label: LDP label bound by C for FEC Z
  Outgoing next-hop: C

And the traffic XZ travels again over the LDP LSP.

Conclusions:

- the operator has eliminated its second problem: guaranteed FRR coverage is provided. The steady-state traffic is still transported over LDP. The SR deployment is confined to the area where these benefits are required.

- FRR coverage has been achieved without any signaling for setting up the repair LSP and without setting up a targeted LDP session between B and G.

4.4. Inter-AS Option C, Carrier's Carrier

In inter-AS Option C [RFC4364], two interconnected ASes sets up inter-AS MPLS connectivity. SR may be independently deployed in each AS.

```
PE1---R1---B1---B2---R2---PE2
<----------->   <----------->
     AS1             AS2
```

Figure 4: Inter-AS Option C

In Inter-AS Option C, B2 advertises to B1 a labeled BGP route [RFC8277] for PE2 and B1 reflects it to its internal peers, such as PE1. PE1 learns from a service route reflector a service route whose next-hop is PE2. PE1 resolves that service route on the labeled BGP route to PE2. That labeled BGP route to PE2 is itself resolved on the AS1 IGP route to B1.

If AS1 operates SR, then the tunnel from PE1 to B1 is provided by the node segment from PE1 to B1.

PE1 sends a service packet with three labels: the top one is the node segment to B1, the next-one is the label in the labeled BGP route provided by B1 for the route "PE2" and the bottom one is the service label allocated by PE2.

5. IANA Considerations

This document does not introduce any new codepoint.

6. Manageability Considerations

6.1. SR and LDP co-existence

When both SR and LDP co-exist, the following applies:

- If both SR and LDP propose an IP2MPLS entry for the same IP prefix, then by default the LDP route SHOULD be selected. This is because it is expected that SR is introduced into network that contain routers that do not support SR. Hence by having a behavior that prefers LDP over SR, traffic flow is unlikely to be disrupted

- A local policy on a router MUST allow to prefer the SR-provided IP2MPLS entry.

- Note that this policy MAY be locally defined. There is no requirement that all routers use the same policy.

6.2. Dataplane Verification

When Label switch paths (LSPs) are defined by stitching LDP LSPs with SR LSPs, it is necessary to have mechanisms allowing the verification of the LSP connectivity as well as validation of the path. These mechanisms are described in [RFC8287].

7. Security Considerations

This document does not introduce any change to the MPLS dataplane [RFC3031] and therefore no additional security of the MPLS dataplane is required.

This document introduces another form of label binding advertisements. The security associated with these advertisements is part of the security applied to routing protocols such as IS-IS [RFC5304] and OSPF [RFC5709] which both optionally make use of cryptographic authentication mechanisms. This form of advertisement is more centralized, on behalf of the node advertising the IP reachability, which presents a different risk profile. This document also specifies a mechanism by which the ill effects of advertising conflicting label bindings can be mitigated. In particular, advertisements from the node advertising the IP reachability is more preferred than the centralized one. Because this document recognizes that reachability, which presents a different risk profile. This document miscofiguration and/or programming may result in false or conflicting also specifies a mechanism by which the ill effects of advertising label binding advertisements, thereby compromising traffic conflicting label bindings can be mitigated. In particular, forwarding, the document recommends strict configuration/ advertisements from the node advertising the IP reachability is more programmability control as well as montoring the SID advertised and preferred than the centralized one. log/error messages by the operator to avoid or at least significantly minimize the possibility of such risk.

8. Acknowledgements

The authors would like to thank Pierre Francois, Ruediger Geib and Alexander Vainshtein for their contribution to the content of this document.

9. Contributors' Addresses

Edward Crabbe
Individual
Email: edward.crabbe@gmail.com

Igor Milojevic
Email: milojevicigor@gmail.com

Saku Ytti
TDC
Email: saku@ytti.fi

Rob Shakir
Google
Email: robjs@google.com

Martin Horneffer
Deutsche Telekom
Email: Martin.Horneffer@telekom.de

Wim Henderickx
Nokia
Email: wim.henderickx@nokia.com

Jeff Tantsura
Individual
Email: jefftant@gmail.com

Les Ginseberg
Cisco Systems
Email: ginsberg@cisco.com

10. References

10.1. Normative References

[I-D.ietf-spring-segment-routing]
Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", January 2018.

[I-D.ietf-spring-segment-routing-mpls]
Bashandy, A., Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls-13 (work in progress), April 2018.

[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>.

[RFC5036] Andersson, L., Ed., Minei, I., Ed., and B. Thomas, Ed., "LDP Specification", RFC 5036, DOI 10.17487/RFC5036, October 2007, <https://www.rfc-editor.org/info/rfc5036>.

10.2. Informative References

[I-D.ietf-isis-segment-routing-extensions] Previdi, S., Ginsberg, L., Filsfils, C., Bashandy, A., Gredler, H., Litkowski, S., Decraene, B., and J. Tantsura, "IS-IS Extensions for Segment Routing", draft-ietf-isis-segment-routing-extensions-19 (work in progress), July 2018.

[I-D.ietf-ospf-ospfv3-segment-routing-extensions] Psenak, P., Filsfils, C., Previdi, S., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPFv3 Extensions for Segment Routing", draft-ietf-ospf-ospfv3-segment-routing-extensions-11 (work in progress), January 2018.

[I-D.ietf-ospf-segment-routing-extensions] Psenak, P., Previdi, S., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPF Extensions for Segment Routing", draft-ietf-ospf-segment-routing-extensions-24 (work in progress), December 2017.

[RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, DOI 10.17487/RFC3031, January 2001, <https://www.rfc-editor.org/info/rfc3031>.

[RFC3209] Awduche, D., Berger, L., Gan, G., Li, T., Srinivasan, V., and G. Srinivasan, "RSVP-TE: Extensions to RSVP for LSP Tunnels", December 2001.

[RFC4364] Rosen, E. and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, DOI 10.17487/RFC4364, February 2006, <https://www.rfc-editor.org/info/rfc4364>.

[RFC5304] Li, T. and R. Atkinson, "IS-IS Cryptographic Authentication", RFC 5304, DOI 10.17487/RFC5304, October 2008, <https://www.rfc-editor.org/info/rfc5304>.

[RFC5709] Bhatia, M., Manral, V., Fanto, M., White, R., Barnes, M., Li, T., and R. Atkinson, "OSPFv2 HMAC-SHA Cryptographic Authentication", RFC 5709, DOI 10.17487/RFC5709, October 2009, <https://www.rfc-editor.org/info/rfc5709>.

[RFC5960] Frost, D., Ed., Bryant, S., Ed., and M. Bocci, Ed., "MPLS Transport Profile Data Plane Architecture", RFC 5960, DOI 10.17487/RFC5960, August 2010, <https://www.rfc-editor.org/info/rfc5960>.

[RFC7490] Bryant, S., Filsfils, C., Previdi, S., Shand, M., and N. So, "Remote Loop-Free Alternate (LFA) Fast Reroute (FRR)", RFC 7490, DOI 10.17487/RFC7490, April 2015, <https://www.rfc-editor.org/info/rfc7490>.

[RFC8277] Rosen, E., "Using BGP to Bind MPLS Labels to Address Prefixes", October 2017.

[RFC8287] Kumar, N., Pignataro, C., Swallow, G., Akiya, N., Kini, S., and M. Chen, "Label Switched Path (LSP) Ping/Traceroute for Segment Routing (SR) IGP-Prefix and IGP-Adjacency Segment Identifiers (SIDs) with MPLS Data Planes", December 2017.

[RFC8355] Filsfils, C., Previdi, S., Decraene, B., and R. Shakir, "Resiliency Use Cases in Source Packet Routing in Networking (SPRING) Networks", March 2018.

Appendix A. Migration from LDP to SR

```
      PE2        PE4
        \        /
PE1----P5--P6--P7---PE3
```

Figure 5: Migration

Several migration techniques are possible. The technique described here is inspired by the commonly used method to migrate from one IGP to another.

At time T0, all the routers run LDP. Any service is tunneled from an ingress PE to an egress PE over a continuous LDP LSP.

At time T1, all the routers are upgraded to SR. They are configured with the SRGB range [100, 300]. PE1, PE2, PE3, PE4, P5, P6 and P7 are respectively configured with the node segments 101, 102, 103, 104, 105, 106 and 107 (attached to their service-recursing loopback).

At this time, the service traffic is still tunneled over LDP LSP. For example, PE1 has an SR node segment to PE3 and an LDP LSP to PE3 but by default, as seen earlier, the LDP IP2MPLS encapsulation is preferred. However, it has to be noted that the SR infrastructure is usable, e.g. for Fast Reroute (FRR) or IGP Loop Free Convergence to protect existing IP and LDP traffic. FRR mechanisms are described in and [RFC8355].

At time T2, the operator enables the local policy at PE1 to prefer SR IP2MPLS encapsulation over LDP IP2MPLS.

The service from PE1 to any other PE is now riding over SR. All other service traffic is still transported over LDP LSP.

At time T3, gradually, the operator enables the preference for SR IP2MPLS encapsulation across all the edge routers.

All the service traffic is now transported over SR. LDP is still operational and services could be reverted to LDP.

At time T4, LDP is unconfigured from all routers.

Authors' Addresses

Ahmed Bashandy (editor)
Individual
USA

Email: abashandy.ietf@gmail.com

Clarence Filsfils (editor)
Cisco Systems, Inc.
Brussels
BE

Email: cfilsfil@cisco.com

Stefano Previdi
Cisco Systems, Inc.
IT

Email: stefano@previdi.net

Bruno Decraene
Orange
FR

Email: bruno.decraene@orange.com

Stephane Litkowski
Orange
FR

Email: stephane.litkowski@orange.com

Network Working Group A. Bashandy, Ed.
Internet Draft Arrcus
Intended status: Standards Track C. Filsfils, Ed.
Expires: November 2019 S. Previdi,
Cisco Systems, Inc.
B. Decraene
S. Litkowski
Orange
R. Shakir
Google
May 1, 2019

Segment Routing with MPLS data plane
draft-ietf-spring-segment-routing-mpls-22

Abstract

Segment Routing (SR) leverages the source routing paradigm. A node steers a packet through a controlled set of instructions, called segments, by prepending the packet with an SR header. In the MPLS dataplane, the SR header is instantiated through a label stack. This document specifies the forwarding behavior to allow instantiating SR over the MPLS dataplane.

Status of this Memo

Table of Contents

1. Introduction..3
   1.1. Requirements Language......................................4
2. MPLS Instantiation of Segment Routing...........................4
   2.1. Multiple Forwarding Behaviors for the Same Prefix.........4
   2.2. SID Representation in the MPLS Forwarding Plane...........5
   2.3. Segment Routing Global Block and Local Block..............5
   2.4. Mapping a SID Index to an MPLS label......................6
   2.5. Incoming Label Collision..................................7
      2.5.1. Tie-breaking Rules..................................10
      2.5.2. Redistribution between Routing Protocol Instances...13
         2.5.2.1. Illustration...................................13
         2.5.2.2. Illustration 2.................................14
   2.6. Effect of Incoming Label Collision on Outgoing Label
   Programming..14
   2.7. PUSH, CONTINUE, and NEXT.................................15
      2.7.1. PUSH..15
      2.7.2. CONTINUE..15
      2.7.3. NEXT..15
         2.7.3.1. Mirror SID.....................................16
   2.8. MPLS Label Downloaded to FIB for Global and Local SIDs...16
   2.9. Active Segment...16
   2.10. Forwarding behavior for Global SIDs.....................16
      2.10.1. Forwarding for PUSH and CONTINUE of Global SIDs....17
      2.10.2. Forwarding for NEXT Operation for Global SIDs......18
   2.11. Forwarding Behavior for Local SIDs......................18
      2.11.1. Forwarding for PUSH Operation on Local SIDs........19
      2.11.2. Forwarding for CONTINUE Operation for Local SIDs...19
      2.11.3. Outgoing label for NEXT Operation for Local SIDs...19
3. IANA Considerations..19

4. Manageability Considerations.................................20
5. Security Considerations......................................20
6. Contributors...20
7. Acknowledgements...20
8. References...21
8.1. Normative References.......................................21
8.2. Informative References.....................................22
9. Authors' Addresses...24
Appendix A. Examples..26
A.1. IGP Segments Example.......................................26
A.2. Incoming Label Collision Examples..........................28
A.2.1. Example 1..28
A.2.2. Example 2..29
A.2.3. Example 3..30
A.2.4. Example 4..30
A.2.5. Example 5..31
A.2.6. Example 6..31
A.2.7. Example 7..32
A.2.8. Example 8..32
A.2.9. Example 9..33
A.2.10. Example 10..33
A.2.11. Example 11..34
A.2.12. Example 12..35
A.2.13. Example 13..35
A.2.14. Example 14..36
A.3. Examples for the Effect of Incoming Label Collision on Outgoing Label...36
A.3.1. Example 1..36
A.3.2. Example 2..37

1. Introduction

The Segment Routing architecture RFC8402 can be directly applied to the MPLS architecture with no change in the MPLS forwarding plane. This document specifies the forwarding plane behavior to allow Segment Routing to operate on top of the MPLS data plane. This document does not address the control plane behavior. Control plane behavior is specified in other documents such as [I-D.ietf-isis-segment-routing-extensions], [I-D.ietf-ospf-segment-routing-extensions], and [I-D.ietf-ospf-ospfv3-segment-routing-extensions].

The Segment Routing problem statement is described in [RFC7855].

Co-existence of SR over MPLS forwarding plane with LDP [RFC5036] is specified in [I-D.ietf-spring-segment-routing-ldp-interop].

Policy routing and traffic engineering using segment routing can be found in [I-D.ietf-spring-segment-routing-policy]

1.1. Requirements Language

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

2. MPLS Instantiation of Segment Routing

MPLS instantiation of Segment Routing fits in the MPLS architecture as defined in [RFC3031] both from a control plane and forwarding plane perspective:

- From a control plane perspective, [RFC3031] does not mandate a single signaling protocol. Segment Routing makes use of various control plane protocols such as link state IGPs [I-D.ietf-isis-segment-routing-extensions], [I-D.ietf-ospf-segment-routing-extensions] and [I-D.ietf-ospf-ospfv3-segment-routing-extensions]. The flooding mechanisms of link state IGPs fit very well with label stacking on ingress. Future control layer protocol and/or policy/configuration can be used to specify the label stack.

- From a forwarding plane perspective, Segment Routing does not require any change to the forwarding plane because Segment IDs (SIDs) are instantiated as MPLS labels and the Segment routing header instantiated as a stack of MPLS labels.

We call "MPLS Control Plane Client (MCC)" any control plane entity installing forwarding entries in the MPLS data plane. Local configuration and policies applied on a router are examples of MCCs.

In order to have a node segment reach the node, a network operator SHOULD configure at least one node segment per routing instance, topology, or algorithm. Otherwise, the node is not reachable within the routing instance, topology or along the routing algorithm, which restrict its ability to be used by a SR policy, including for TI-LFA.

2.1. Multiple Forwarding Behaviors for the Same Prefix

The SR architecture does not prohibit having more than one SID for the same prefix. In fact, by allowing multiple SIDs for the same prefix, it is possible to have different forwarding behaviors (such as different paths, different ECMP/UCMP behaviors,...,etc) for the same destination.

Instantiating Segment routing over the MPLS forwarding plane fits seamlessly with this principle. An operator may assign multiple MPLS labels or indices to the same prefix and assign different forwarding behaviors to each label/SID. The MCC in the network downloads different MPLS labels/SIDs to the FIB for different forwarding behaviors. The MCC at the entry of an SR domain or at any point in the domain can choose to apply a particular forwarding behavior to a particular packet by applying the PUSH action to that packet using the corresponding SID.

2.2. SID Representation in the MPLS Forwarding Plane

When instantiating SR over the MPLS forwarding plane, a SID is represented by an MPLS label or an index [RFC8402].

A global segment is a label, or an index which may be mapped to an MPLS label within the Segment Routing Global Block (SRGB) of the node installing the global segment in its FIB/receiving the labeled packet. Section 2.4 specifies the procedure to map a global segment represented by an index to an MPLS label within the SRGB.

The MCC MUST ensure that any label value corresponding to any SID it installs in the forwarding plane follows the following rules:

- o The label value MUST be unique within the router on which the MCC is running. i.e. the label MUST only be used to represent the SID and MUST NOT be used to represent more than one SID or for any other forwarding purpose on the router.

- o The label value MUST NOT come from the range of special purpose labels [RFC7274].

Labels allocated in this document are considered per platform down-stream allocated labels [RFC3031].

2.3. Segment Routing Global Block and Local Block

The concepts of Segment Routing Global Block (SRGB) and global SID are explained in [RFC8402]. In general, the SRGB need not be a contiguous range of labels.

For the rest of this document, the SRGB is specified by the list of MPLS Label ranges [Ll(1),Lh(1)], [Ll(2),Lh(2)],..., [Ll(k),Lh(k)] where Ll(i) =< Lh(i).

The following rules apply to the list of MPLS ranges representing the SRGB

- o The list of ranges comprising the SRGB MUST NOT overlap.

- o Every range in the list of ranges specifying the SRGB MUST NOT cover or overlap with a reserved label value or range [RFC7274], respectively.

- o If the SRGB of a node does not conform to the structure specified in this section or to the previous two rules, then this SRGB MUST be completely ignored by all routers in the routing domain and the node MUST be treated as if it does not have an SRGB.

- o The list of label ranges MUST only be used to instantiate global SIDs into the MPLS forwarding plane

A Local segment MAY be allocated from the Segment Routing Local Block (SRLB) [RFC8402] or from any unused label as long as it does not use a special purpose label. The SRLB consists of the range of local labels reserved by the node for certain local segments. In a controller-driven network, some controllers or applications MAY use the control plane to discover the available set of local SIDs on a particular router [I-D.ietf-spring-segment-routing-policy]. The rules applicable to the SRGB are also applicable to the SRLB, except the rule that says that the SRGB MUST only be used to instantiate global SIDs into the MPLS forwarding plane. The recommended, minimum, or maximum size of the SRGB and/or SRLB is a matter of future study

2.4. Mapping a SID Index to an MPLS label

This sub-section specifies how the MPLS label value is calculated given the index of a SID. The value of the index is determined by an MCC such as IS-IS [I-D.ietf-isis-segment-routing-extensions] or OSPF [I-D.ietf-ospf-segment-routing-extensions]. This section only specifies how to map the index to an MPLS label. The calculated MPLS label is downloaded to the FIB, sent out with a forwarded packet, or both.

Consider a SID represented by the index "I". Consider an SRGB as specified in Section 2.3. The total size of the SRGB, represented by the variable "Size", is calculated according to the formula:

size = Lh(1)- Ll(1) + 1 + Lh(2)- Ll(2) + 1 + ... + Lh(k)- Ll(k) + 1

The following rules MUST be applied by the MCC when calculating the MPLS label value corresponding the SID index value "I".

o 0 =< I < size. If the index "I" does not satisfy the previous inequality, then the label cannot be calculated.

o The label value corresponding to the SID index "I" is calculated as follows

  o j = 1 , temp = 0

  o While temp + Lh(j)- Ll(j) < I

    . temp = temp + Lh(j)- Ll(j) + 1

    . j = j+1

  o label = I - temp + Ll(j)

An example for how a router calculates labels and forwards traffic based on the procedure described in this section can be found in Appendix A.1.

2.5. Incoming Label Collision

The MPLS Architecture [RFC3031] defines the term Forwarding Equivalence Class (FEC) as the set of packets with similar and / or identical characteristics which are forwarded the same way and are bound to the same MPLS incoming (local) label. In Segment-Routing MPLS, a local label serves as the SID for given FEC.

We define Segment Routing (SR) FEC as one of the following [RFC8402]:

- (Prefix, Routing Instance, Topology, Algorithm [RFC8402]), where a topology identifies a set of links with metrics. For the purpose of incoming label collision resolution, the same Topology numerical value SHOULD be used on all routers to identify the same set of links with metrics. For MCCs where the "Topology" and/or "Algorithm" fields are not defined, the numerical value of zero MUST be used for these two fields. For the purpose of incoming label collision resolution, a routing instance is identified by a single incoming label downloader to FIB. Two MCCs running on the same router are considered different routing instances if the only way the two instances can know about the other's incoming labels is through redistribution. The numerical value used to identify a routing instance MAY be derived from other configuration or MAY be explicitly configured. If it is derived from other configuration, then the same numerical value SHOULD be derived from the same configuration as long as the configuration survives router reload. If the derived numerical value varies for the same configuration, then an implementation SHOULD make numerical value used to identify a routing instance configurable.

- (next-hop, outgoing interface), where the outgoing interface is physical or virtual.

- (number of adjacencies, list of next-hops, list of outgoing interfaces IDs in ascending numerical order). This FEC represents parallel adjacencies [RFC8402]

- (Endpoint, Color) representing an SR policy [RFC8402]

- (Mirrored SID) The Mirrored SID [RFC8402, Section 5.1] is the IP address advertised by the advertising node to identify the mirror-SID. The IP address is encoded as specified in Section 2.5.1.

This section covers the RECOMMENDED procedure to handle the scenario where, because of an error/misconfiguration, more than one SR FEC as defined in this section, map to the same incoming MPLS label. Examples illustrating the behavior specified in this section can be found in Appendix A.2.

An incoming label collision occurs if the SIDs of the set of FECs {FEC1, FEC2,..., FECk} map to the same incoming SR MPLS label "L1".

Suppose an anycast prefix is advertised with a prefix-SID by some, but not all, of the nodes that advertise that prefix. If the prefix-SID sub-TLVs result in mapping that anycast prefix to the same incoming label, then the advertisement of the prefix-SID by some, but not all, of advertising nodes MUST NOT be treated as a label collision.

An implementation MUST NOT allow the MCCs belonging to the same router to assign the same incoming label to more than one SR FEC.

The objective of the following steps is to deterministically install in the MPLS Incoming Label Map, also known as label FIB, a single FEC with the incoming label "L1". By "deterministically install" we mean if the set of FECs {FEC1, FEC2,..., FECk} map to the same incoming SR MPLS label "L1", then the steps below assign the same FEC to the label "L1" irrespective of the order by which the mappings of this set of FECs to the label "L1" are received. For example, a first-come-first-serve tie-breaking is not allowed. The remaining FECs may be installed in the IP FIB without incoming label.

The procedure in this section relies completely on the local FEC and label database within a given router.

The collision resolution procedure is as follows

1. Given the SIDs of the set of FECs, {FEC1, FEC2,..., FECk} map to the same MPLS label "L1".

2. Within an MCC, apply tie-breaking rules to select one FEC only and assign the label to it. The losing FECs are handled as if no labels are attached to them. The losing FECs with algorithms other than the shortest path first [RFC8402] are not installed in the FIB.

   a. If the same set of FECs are attached to the same label "L1", then the tie-breaking rules MUST always select the same FEC irrespective of the order in which the FECs and the label "L1" are received. In other words, the tie-breaking rule MUST be deterministic.

3. If there is still collision between the FECs belonging to different MCCs, then re-apply the tie-breaking rules to the remaining FECs to select one FEC only and assign the label to that FEC

4. Install into the IP FIB the selected FEC and its incoming label in the label FIB.

5. The remaining FECs with the default algorithm (see the specification of prefix-SID algorithm [RFC8402]) may be installed in the FIB natively, such as pure IP entries in case of Prefix FEC, without any incoming labels corresponding to their SIDs. The remaining FECs with algorithms other than the shortest path first [RFC8402] are not installed in the FIB.

2.5.1. Tie-breaking Rules

The default tie-breaking rules are specified as follows:

1. if FECi has the lowest FEC administrative distance among the competing FECs as defined in this section below, filter away all the competing FECs with higher administrative distance.

2. if more than one competing FEC remains after step 1, select the smallest numerical FEC value. The numerical value of the FEC is determined according to the FEC encoding described later in this section.

These rules deterministically select the FEC to install in the MPLS forwarding plane for the given incoming label.

This document defines the default tie breaking rules that SHOULD be implemented. An implementation MAY choose to support different tie-breaking rules and MAY use one of the these instead of the default tie-breaking rules. To maximize MPLS forwarding consistency in case of SID configuration error, the network operator MUST deploy, within an IGP flooding area, routers implementing the same tie-breaking rules.

Each FEC is assigned an administrative distance. The FEC administrative distance is encoded as an 8-bit value. The lower the value, the better the administrative distance.

The default FEC administrative distance order starting from the lowest value SHOULD be:

- o Explicit SID assignment to a FEC that maps to a label outside the SRGB irrespective of the owner MCC. An explicit SID assignment is a static assignment of a label to a FEC such that the assignment survives router reboot.

  - o An example of explicit SID allocation is static assignment of a specific label to an adj-SID.

  - o An implementation of explicit SID assignment MUST guarantee collision freeness on the same router

- o Dynamic SID assignment:

  - o For all FEC types except for SR policy, the FEC types are ordered using the default administrative distance ordering defined by the implementation.

  - o Binding SID [RFC8402] assigned to SR Policy always has a higher default administrative distance than the default administrative distance of any other FEC type

To maximize MPLS forwarding consistency, If a same FEC is advertised in more than one protocol, a user MUST ensure that the administrative distance preference between protocols is the same on all routers of the IGP flooding domain. Note that this is not really new as this already applies to IP forwarding.

The numerical sort across FECs SHOULD be performed as follows:

- o Each FEC is assigned a FEC type encoded in 8 bits. The following are the type code point for each SR FEC defined at the beginning of this Section:

  - o 120: (Prefix, Routing Instance, Topology, Algorithm)

  - o 130: (next-hop, outgoing interface)

  - o 140: Parallel Adjacency [RFC8402]

  - o 150: an SR policy [RFC8402].

  - o 160: Mirror SID [RFC8402]

  - o The numerical values above are mentioned to guide implementation. If other numerical values are used, then the numerical values must maintain the same greater-than ordering of the numbers mentioned here.

- o The fields of each FEC are encoded as follows

  - o All fields in all FECs are encoded in big endian

- o Routing Instance ID represented by 16 bits. For routing instances that are identified by less than 16 bits, encode the Instance ID in the least significant bits while the most significant bits are set to zero

- o Address Family represented by 8 bits, where IPv4 encoded as 100 and IPv6 is encoded as 110. These numerical values are mentioned to guide implementations. If other numerical values are used, then the numerical value of IPv4 MUST be less than the numerical value for IPv6

- o All addresses are represented in 128 bits as follows

  - . IPv6 address is encoded natively

  - . IPv4 address is encoded in the most significant bits and the remaining bits are set to zero

- o All prefixes are represented by (8 + 128) bits.

  - . A prefix is encoded in the most significant bits and the remaining bits are set to zero.

  - . The prefix length is encoded before the prefix in a field of size 8 bits.

- o Topology ID is represented by 16 bits. For routing instances that identify topologies using less than 16 bits, encode the topology ID in the least significant bits while the most significant bits are set to zero

- o Algorithm is encoded in a 16 bits field.

- o The Color ID is encoded using 32 bits

o Choose the set of FECs of the smallest FEC type code point

o Out of these FECs, choose the FECs with the smallest address family code point

o Encode the remaining set of FECs as follows

- o (Prefix, Routing Instance, Topology, Algorithm) is encoded as (Prefix Length, Prefix, routing_instance_id, Topology, SR Algorithm)

  - o (next-hop, outgoing interface) is encoded as (next-hop, outgoing_interface_id)

  - o (number of adjacencies, list of next-hops in ascending numerical order, list of outgoing interface IDs in ascending numerical order). This encoding is used to encode a parallel adjacency [RFC8402]

  - o (Endpoint, Color) is encoded as (Endpoint_address, Color_id)

  - o (IP address): This is the encoding for a mirror SID FEC. The IP address is encoded as described above in this section

- o Select the FEC with the smallest numerical value

The numerical values mentioned in this section are for guidance only. If other numerical values are used then the other numerical values MUST maintain the same numerical ordering among different SR FECs.

2.5.2. Redistribution between Routing Protocol Instances

The following rule SHOULD be applied when redistributing SIDs with prefixes between routing protocol instances:

- o If the receiving instance's SRGB is the same as the SRGB of origin instance, then

  - o the index is redistributed with the route

- o Else

  - o the index is not redistributed and if the receiving instance decides to advertise an index with the redistributed route, it is the duty of the receiving instance to allocate a fresh index relative to its own SRGB. Note that in this case the receiving instance MUST compute the local label it assignes to the route according to section 2.4 and install it in FIB.

It is outside the scope of this document to define local node behaviors that would allow to map the original index into a new index in the receiving instance via the addition of an offset or other policy means.

2.5.2.1. Illustration

```
A----IS-IS----B---OSPF----C-192.0.2.1/32 (20001)
```

Consider the simple topology above.

- o A and B are in the IS-IS domain with SRGB [16000-17000]
- o B and C are in OSPF domain with SRGB [20000-21000]
- o B redistributes 192.0.2.1/32 into IS-IS domain
- o In that case A learns 192.0.2.1/32 as an IP leaf connected to B as usual for IP prefix redistribution
- o However, according to the redistribution rule above rule, B decides not to advertise any index with 192.0.2.1/32 into IS-IS because the SRGB is not the same.

2.5.2.2. Illustration 2

Consider the example in the illustration described in Section 2.5.2.1.

When router B redistributes the prefix 192.0.2.1/32, router B decides to allocate and advertise the same index 1 with the prefix 192.0.2.1/32

Within the SRGB of the IS-IS domain, index 1 corresponds to the local label 16001

- o Hence according to the redistribution rule above, router B programs the incoming label 16001 in its FIB to match traffic arriving from the IS-IS domain destined to the prefix 192.0.2.1/32.

2.6. Effect of Incoming Label Collision on Outgoing Label Programming

For the determination of the outgoing label to use, the ingress node pushing new segments, and hence a stack of MPLS labels, MUST use, for a given FEC, the same label that has been selected by the node receiving the packet with that label exposed as top label. So in case of incoming label collision on this receiving node, the ingress node MUST resolve this collision using this same "Incoming Label Collision resolution procedure", using the data of the receiving node.

In the general case, the ingress node may not have exactly the same data of the receiving node, so the result may be different. This is under the responsibility of the network operator. But in typical case, e.g. where a centralized node or a distributed link state IGP is used, all nodes would have the same database. However to minimize the chance of misforwarding, a FEC that loses its incoming label to the tie-breaking rules specified in Section 2.5 MUST NOT be installed in FIB with an outgoing segment routing label based on the SID corresponding to the lost incoming label.

Examples for the behavior specified in this section can be found in Appendix A.3.

2.7. PUSH, CONTINUE, and NEXT

PUSH, NEXT, and CONTINUE are operations applied by the forwarding plane. The specifications of these operations can be found in [RFC8402]. This sub-section specifies how to implement each of these operations in the MPLS forwarding plane.

2.7.1. PUSH

As described in [RFC8402], PUSH corresponds to pushing one or more labels on top of an incoming packet then sending it out of a particular physical interface or virtual interface, such as UDP tunnel [RFC7510] or L2TPv3 tunnel [RFC4817], towards a particular next-hop. When pushing labels onto a packet's label stack, the Time-to-Live (TTL) field ([RFC3032], [RFC3443]) and the Traffic Class (TC) field ([RFC3032], [RFC5462]) of each label stack entry must, of course, be set. This document does not specify any set of rules for setting these fields; that is a matter of local policy. Sections 2.10 and 2.11 specify additional details about forwarding behavior.

2.7.2. CONTINUE

As described in [RFC8402], the CONTINUE operation corresponds to swapping the incoming label with an outgoing label. The value of the outgoing label is calculated as specified in Sections 2.10 and 2.11.

2.7.3. NEXT

As described in [RFC8402], NEXT corresponds to popping the topmost label. The action before and/or after the popping depends on the instruction associated with the active SID on the received packet prior to the popping. For example suppose the active SID in the received packet was an Adj-SID [RFC8402], then on receiving the packet, the node applies NEXT operation, which corresponds to popping the top most label, and then sends the packet out of the physical or virtual interface (e.g. UDP tunnel [RFC7510] or L2TPv3 tunnel [RFC4817]) towards the next-hop corresponding to the adj-SID.

2.7.3.1. Mirror SID

If the active SID in the received packet was a Mirror SID [RFC8402, Section 5.1] allocated by the receiving router, then the receiving router applies NEXT operation, which corresponds to popping the top most label, then performs a lookup using the contents of the packet after popping the outer most label in the mirrored forwarding table. The method by which the lookup is made, and/or the actions applied to the packet after the lookup in the mirror table depends on the contents of the packet and the mirror table. Note that the packet exposed after popping the top most label may or may not be an MPLS packet. A mirror SID can be viewed as a generalization of the context label in [RFC5331] because a mirror SID does not make any assumptions about the packet underneath the top label.

2.8. MPLS Label Downloaded to FIB for Global and Local SIDs

The label corresponding to the global SID "Si" represented by the global index "I" downloaded to FIB is used to match packets whose active segment (and hence topmost label) is "Si". The value of this label is calculated as specified in Section 2.4.

For Local SIDs, the MCC is responsible for downloading the correct label value to FIB. For example, an IGP with SR extensions [I-D.ietf-isis-segment-routing-extensions, I-D.ietf-ospf-segment-routing-extensions] downloads the MPLS label corresponding to an Adj-SID [RFC8402].

2.9. Active Segment

When instantiated in the MPLS domain, the active segment on a packet corresponds to the topmost label on the packet that is calculated according to the procedure specified in Sections 2.10 and 2.11. When arriving at a node, the topmost label corresponding to the active SID matches the MPLS label downloaded to FIB as specified in Section 2.4.

2.10. Forwarding behavior for Global SIDs

This section specifies forwarding behavior, including the calculation of outgoing labels, that corresponds to a global SID when applying PUSH, CONTINUE, and NEXT operations in the MPLS forwarding plane.

This document covers the calculation of the outgoing label for the top label only. The case where the outgoing label is not the top label and is part of a stack of labels that instantiates a routing policy or a traffic engineering tunnel is outside the scope of this document and may be covered in other documents such as [I-D.ietf-spring-segment-routing-policy].

2.10.1. Forwarding for PUSH and CONTINUE of Global SIDs

Suppose an MCC on a router "R0" determines that PUSH or CONTINUE operation is to be applied to an incoming packet related to the global SID "Si" represented by the global index "I" and owned by the router Ri before sending the packet towards a neighbor "N" directly connected to "R0" through a physical or virtual interface such as UDP tunnel [RFC7510] or L2TPv3 tunnel [RFC4817].

The method by which the MCC on router "R0" determines that PUSH or CONTINUE operation must be applied using the SID "Si" is beyond the scope of this document. An example of a method to determine the SID "Si" for PUSH operation is the case where IS-IS [I-D.ietf-isis-segment-routing-extensions] receives the prefix-SID "Si" sub-TLV advertised with prefix "P/m" in TLV 135 and the destination address of the incoming IPv4 packet is covered by the prefix "P/m".

For CONTINUE operation, an example of a method to determine the SID "Si" is the case where IS-IS [I-D.ietf-isis-segment-routing-extensions] receives the prefix-SID "Si" sub-TLV advertised with prefix "P" in TLV 135 and the top label of the incoming packet matches the MPLS label in FIB corresponding to the SID "Si" on the router "R0".

The forwarding behavior for PUSH and CONTINUE corresponding to the SID "Si"

- o If the neighbor "N" does not support SR or advertises an invalid SRGB or a SRGB that is too small for the SID "Si"

  - o If it is possible to send the packet towards the neighbor "N" using standard MPLS forwarding behavior as specified in [RFC3031] and [RFC3032], then forward the packet. The method by which a router decides whether it is possible to send the packet to "N" or not is beyond the scope of this document. For example, the router "R0" can use the downstream label determined by another MCC, such as LDP [RFC5036], to send the packet.

o Else if there are other useable next-hops, then use other next-hops to forward the incoming packet. The method by which the router "R0" decides on the possibility of using other next-hops is beyond the scope of this document. For example, the MCC on "R0" may chose the send an IPv4 packet without pushing any label to another next-hop.

o Otherwise drop the packet.

o Else

o Calculate the outgoing label as specified in Section 2.4 using the SRGB of the neighbor "N"

o If the operation is PUSH

. Push the calculated label according to the MPLS label pushing rules specified in [RFC3032]

o Else

. swap the incoming label with the calculated label according to the label swapping rules in [RFC3032]

o Send the packet towards the neighbor "N"

2.10.2. Forwarding for NEXT Operation for Global SIDs

As specified in Section 2.7.3 NEXT operation corresponds to popping the top most label. The forwarding behavior is as follows

o Pop the topmost label

o Apply the instruction associated with the incoming label that has been popped

The action on the packet after popping the topmost label depends on the instruction associated with the incoming label as well as the contents of the packet right underneath the top label that got popped. Examples of NEXT operation are described in Appendix A.1.

2.11. Forwarding Behavior for Local SIDs

This section specifies the forwarding behavior for local SIDs when SR is instantiated over the MPLS forwarding plane.

2.11.1. Forwarding for PUSH Operation on Local SIDs

Suppose an MCC on a router "R0" determines that PUSH operation is to be applied to an incoming packet using the local SID "Si" before sending the packet towards a neighbor "N" directly connected to R0 through a physical or virtual interface such as UDP tunnel [RFC7510] or L2TPv3 tunnel [RFC4817].

An example of such local SID is an Adj-SID allocated and advertised by IS-IS [I-D.ietf-isis-segment-routing-extensions]. The method by which the MCC on "R0" determines that PUSH operation is to be applied to the incoming packet is beyond the scope of this document. An example of such method is backup path used to protect against a failure using TI-LFA [I-D.bashandy-rtgwg-segment-routing-ti-lfa].

As mentioned in [RFC8402], a local SID is specified by an MPLS label. Hence the PUSH operation for a local SID is identical to label push operation [RFC3032] using any MPLS label. The forwarding action after pushing the MPLS label corresponding to the local SID is also determined by the MCC. For example, if the PUSH operation was done to forward a packet over a backup path calculated using TI-LFA, then the forwarding action may be sending the packet to a certain neighbor that will in turn continue to forward the packet along the backup path

2.11.2. Forwarding for CONTINUE Operation for Local SIDs

A local SID on a router "R0" corresponds to a local label. In such scenario, the outgoing label towards a next-hop "N" is determined by the MCC running on the router "R0"and the forwarding behavior for CONTINUE operation is identical to swap operation [RFC3032] on an MPLS label.

2.11.3. Outgoing label for NEXT Operation for Local SIDs

NEXT operation for Local SIDs is identical to NEXT operation for global SIDs specified in Section 2.10.2.

3. IANA Considerations

This document does not make any request to IANA.

4. Manageability Considerations

This document describes the applicability of Segment Routing over the MPLS data plane. Segment Routing does not introduce any change in the MPLS data plane. Manageability considerations described in [RFC8402] applies to the MPLS data plane when used with Segment Routing. SR OAM use cases for the MPLS data plane are defined in [RFC8403]. SR OAM procedures for the MPLS data plane are defined in [RFC8287].

5. Security Considerations

This document does not introduce additional security requirements and mechanisms other than the ones described in [RFC8402].

6. Contributors

The following contributors have substantially helped the definition and editing of the content of this document:

Martin Horneffer
Deutsche Telekom
Email: Martin.Horneffer@telekom.de

Wim Henderickx
Nokia
Email: wim.henderickx@nokia.com

Jeff Tantsura
Email: jefftant@gmail.com
Edward Crabbe
Email: edward.crabbe@gmail.com

Igor Milojevic
Email: milojevicigor@gmail.com

Saku Ytti
Email: saku@ytti.fi

7. Acknowledgements

The authors would like to thank Les Ginsberg, Chris Bowers, Himanshu Shah, Adrian Farrel, Alexander Vainshtein, Przemyslaw Krol, Darren Dukes, Zafar Ali, and Martin Vigoureux for their valuable comments on this document.

This document was prepared using 2-Word-v2.0.template.dot.

8. References

8.1. Normative References

[RFC8402] Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402 July 2018, <http://www.rfc-editor.org/info/rfc8402>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 0.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, DOI 10.17487/RFC3031, January 2001, <http://www.rfc-editor.org/info/rfc3031>.

[RFC3032] Rosen, E., Tappan, D., Fedorkow, G., Rekhter, Y., Farinacci, D., Li, T., and A. Conta, "MPLS Label Stack Encoding", RFC 3032, DOI 10.17487/RFC3032, January 2001, <http://www.rfc-editor.org/info/rfc3032>.

[RFC3443] P. Agarwal, P. and Akyol, B. "Time To Live (TTL) Processing in Multi-Protocol Label Switching (MPLS) Networks", RFC 3443, DOI 10.17487/RFC3443, January 2003, <http://www.rfc-editor.org/info/rfc3443>.

[RFC5462] Andersson, L., and Asati, R., " Multiprotocol Label Switching (MPLS) Label Stack Entry: "EXP" Field Renamed to "Traffic Class" Field", RFC 5462, DOI 10.17487/RFC5462, February 2009, <http://www.rfc-editor.org/info/rfc5462>.

[RFC7274] K. Kompella, L. Andersson, and A. Farrel, "Allocating and Retiring Special-Purpose MPLS Labels", RFC7274 DOI 10.17487/RFC7274, May 2014 <http://www.rfc-editor.org/info/rfc7274>

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

8.2. Informative References

[I-D.ietf-isis-segment-routing-extensions] Previdi, S., Filsfils, C., Bashandy, A., Gredler, H., Litkowski, S., Decraene, B., and j. jefftant@gmail.com, "IS-IS Extensions for Segment Routing", draft-ietf-isis-segment-routing-extensions-13 (work in progress), June 2017.

[I-D.ietf-ospf-ospfv3-segment-routing-extensions] Psenak, P., Previdi, S., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPFv3 Extensions for Segment Routing", draft-ietf-ospf-ospfv3-segment-routing-extensions-09 (work in progress), March 2017.

[I-D.ietf-ospf-segment-routing-extensions] Psenak, P., Previdi, S., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPF Extensions for Segment Routing", draft-ietf-ospf-segment-routing-extensions-16 (work in progress), May 2017.

[I-D.ietf-spring-segment-routing-ldp-interop] Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., and S. Litkowski, "Segment Routing interworking with LDP", draft-ietf-spring-segment-routing-ldp-interop-08 (work in progress), June 2017.

[I-D.bashandy-rtgwg-segment-routing-ti-lfa], Bashandy, A., Filsfils, C., Decraene, B., Litkowski, S., Francois, P., Voyer, P. Clad, F., and Camarillo, P., "Topology Independent Fast Reroute using Segment Routing", draft-bashandy-rtgwg-segment-routing-ti-lfa-05 (work in progress), October 2018,

[RFC7855] Previdi, S., Ed., Filsfils, C., Ed., Decraene, B., Litkowski, S., Horneffer, M., and R. Shakir, "Source Packet Routing in Networking (SPRING) Problem Statement and Requirements", RFC 7855, DOI 10.17487/RFC7855, May 2016, <http://www.rfc-editor.org/info/rfc7855>.

[RFC5036] Andersson, L., Acreo, AB, Minei, I., Thomas, B., " LDP Specification", RFC5036, DOI 10.17487/RFC5036, October 2007, <https://www.rfc-editor.org/info/rfc5036>

[RFC5331] Aggarwal, R., Rekhter, Y., Rosen, E., " MPLS Upstream Label Assignment and Context-Specific Label Space", RFC5331 DOI 10.17487/RFC5331, August 2008, <http://www.rfc-editor.org/info/rfc5331>.

[RFC7510] Xu, X., Sheth, N., Yong, L., Callon, R., and D. Black, "Encapsulating MPLS in UDP", RFC 7510, DOI 10.17487/RFC7510, April 2015, <https://www.rfc-editor.org/info/rfc7510>.

[RFC4817] Townsley, M., Pignataro, C., Wainner, S., Seely, T., Young, T., "Encapsulation of MPLS over Layer 2 Tunneling Protocol Version 3", RFC4817, DOI 10.17487/RFC4817, March 2007, <https://www.rfc-editor.org/info/rfc4817>

[RFC8287] N. Kumar, C. Pignataro, G. Swallow, N. Akiya, S. Kini, and M. Chen " Label Switched Path (LSP) Ping/Traceroute for Segment Routing (SR) IGP-Prefix and IGP-Adjacency Segment Identifiers (SIDs) with MPLS Data Planes" RFC8287, DOI 10.17487/RFC8287, December 2017, https://www.rfc-editor.org/info/rfc8287

[RFC8403] R. Geib, C. Filsfils, C. Pignataro, N. Kumar, "A Scalable and Topology-Aware MPLS Data-Plane Monitoring System", RFC8403, DOI 10.17487/RFC8403, July 2018, <https://www.rfc-editor.org/info/rfc8403>

[I-D.ietf-spring-segment-routing-policy] Filsfils, C., Sivabalan, S., Raza, K., Liste, J. , Clad, F., Voyer, D., Bogdanov, A., Mattes, P., " Segment Routing Policy for Traffic Engineering", draft-ietf-spring-segment-routing-policy-01 (work in progress), June 2018

9. Authors' Addresses

Ahmed Bashandy (editor)
Arrcus

Email: abashandy.ietf@gmail.com

Clarence Filsfils (editor)
Cisco Systems, Inc.
Brussels
BE

Email: cfilsfil@cisco.com

Stefano Previdi
Cisco Systems, Inc.
Italy

Email: stefano@previdi.net

Bruno Decraene
Orange
FR

Email: bruno.decraene@orange.com

Stephane Litkowski
Orange
FR

Email: stephane.litkowski@orange.com

Rob Shakir
Google
US

Email: robjs@google.com

Appendix A. Examples

A.1. IGP Segments Example

Consider the network diagram of Figure 1 and the IP address and IGP Segment allocation of Figure 2. Assume that the network is running IS-IS with SR extensions [I-D.ietf-isis-segment-routing-extensions] and all links have the same metric. The following examples can be constructed.

```
                         +--------+
                        /          \
        R0-----R1-----R2----------R3-----R8
                      | \        / |
                      |  +--R4--+  |
                      |            |
                      +-----R5-----+
```

Figure 1: IGP Segments - Illustration

```
+--------------------------------------------------------------+
| IP address allocated by the operator:                        |
|                      192.0.2.1/32 as a loopback of R1        |
|                      192.0.2.2/32 as a loopback of R2        |
|                      192.0.2.3/32 as a loopback of R3        |
|                      192.0.2.4/32 as a loopback of R4        |
|                      192.0.2.5/32 as a loopback of R5        |
|                      192.0.2.8/32 as a loopback of R8        |
|              198.51.100.9/32 as an anycast loopback of R4    |
|              198.51.100.9/32 as an anycast loopback of R5    |
|                                                              |
| SRGB defined by the operator as 1000-5000                    |
|                                                              |
| Global IGP SID indices allocated by the operator:            |
|                      1 allocated to 192.0.2.1/32             |
|                      2 allocated to 192.0.2.2/32             |
|                      3 allocated to 192.0.2.3/32             |
|                      4 allocated to 192.0.2.4/32             |
|                      8 allocated to 192.0.2.8/32             |
|                   1009 allocated to 198.51.100.9/32          |
|                                                              |
| Local IGP SID allocated dynamically by R2                    |
|                     for its "north" adjacency to R3: 9001    |
|                     for its "east" adjacency to R3 : 9002    |
|                     for its "south" adjacency to R3: 9003    |
|                     for its only adjacency to R4   : 9004    |
|                     for its only adjacency to R1   : 9005    |
+--------------------------------------------------------------+
```

Figure 2: IGP Address and Segment Allocation - Illustration

Suppose R1 wants to send an IPv4 packet P1 to R8. In this case, R1 needs to apply PUSH operation to the IPv4 packet.

Remember that the SID index "8" is a global IGP segment attached to the IP prefix 192.0.2.8/32. Its semantic is global within the IGP domain: any router forwards a packet received with active segment 8 to the next-hop along the ECMP-aware shortest-path to the related prefix.

R2 is the next-hop along the shortest path towards R8. By applying the steps in Section 2.8 the outgoing label downloaded to R1's FIB corresponding to the global SID index 8 is 1008 because the SRGB of R2 is [1000,5000] as shown in Figure 2.

Because the packet is IPv4, R1 applies the PUSH operation using the label value 1008 as specified in Section 2.10.1. The resulting MPLS header will have the "S" bit [RFC3032] set because it is followed directly by an IPv4 packet.

The packet arrives at router R2. Because the top label 1008 corresponds to the IGP SID "8", which is the prefix-SID attached to the prefix 192.0.2.8/32 owned by the node R8, then the instruction associated with the SID is "forward the packet using all ECMP/UCMP interfaces and all ECMP/UCMP next-hop(s) along the shortest/useable path(s) towards R8". Because R2 is not the penultimate hop, R2 applies the CONTINUE operation to the packet and sends it to R3 using one of the two links connected to R3 with top label 1008 as specified in Section 2.10.1.

R3 receives the packet with top label 1008. Because the top label 1008 corresponds to the IGP SID "8", which is the prefix-SID attached to the prefix 192.0.2.8/32 owned by the node R8, then the instruction associated with the SID is "send the packet using all ECMP interfaces and all next-hop(s) along the shortest path towards R8". Because R3 is the penultimate hop, we assume that R3 performs penumtimate hop popping, which corresponds to the NEXT operation, then sends the packet to R8. The NEXT operation results in popping the outer label and sending the packet as a pure IPv4 packet to R8.

In conclusion, the path followed by P1 is R1-R2--R3-R8. The ECMP-awareness ensures that the traffic be load-shared between any ECMP path, in this case the two links between R2 and R3.

A.2. Incoming Label Collision Examples

This section describes few examples to illustrate the handling of label collision described in Section 2.5.

For the examples in this section, we assume that Node A has the following:

- o OSPF default admin distance for implementation=50
- o ISIS default admin distance for implementation=60

A.2.1. Example 1

Illustration of incoming label collision resolution for the same FEC type using MCC administrative distance.

FEC1:

- o OSPF prefix SID advertisement from node B for 198.51.100.5/32 with index=5
- o OSPF SRGB on node A = [1000,1999]
- o Incoming label=1005

FEC2:

- o ISIS prefix SID advertisement from node C for 203.0.113.105/32 with index=5
- o ISIS SRGB on node A = [1000,1999]
- o Incoming label=1005

FEC1 and FEC2 both use dynamic SID assignment. Since neither ofthe FEC types is SR Policy, we use the default admin distances of 50 and 60 to break the tie. So FEC1 wins.

A.2.2. Example 2

Illustration of incoming label collision resolution for different FEC types using the MCC administrative distance.

FEC1:

- o Node A receives an OSPF prefix sid advertisement from node B for 198.51.100.6/32 with index=6
- o OSPF SRGB on node A = [1000,1999]
- o Hence the incoming label on node A corresponding to 198.51.100.6/32 is 1006

FEC2:
ISIS on node A assigns the label 1006 to the globally significant adj-SID (I.e. when advertised the "L" flag is clear in the adj-SID sub-TLV as described in [I-D.ietf-isis-segment-routing-extensions]) towards one of its neighbors. Hence the incoming label corresponding to this adj-SID 1006. Assume Node A allocates this adj-SID dynamically, and it may differ across router reboots.

FEC1 and FEC2 both use dynamic SID assignment. Since neither of the FEC types is SR Policy, we use the default admin distances of 50 and 60 to break the tie. So FEC1 wins.

A.2.3. Example 3

Illustration of incoming label collision resolution based on preferring static over dynamic SID assignment

FEC1:
OSPF on node A receives a prefix SID advertisement from node B for 198.51.100.7/32 with index=7. Assuming that the OSPF SRGB on node A is [1000,1999], then incoming label corresponding to 198.51.100.7/32 is 1007

FEC2:
The operator on node A configures ISIS on node A to assign the label 1007 to the globally significant adj-SID (I.e. when advertised the "L" flag is clear in the adj-SID sub-TLV as described in [I-D.ietf-isis-segment-routing-extensions]) towards one of its neighbor advertisement from node A with label=1007

Node A assigns this adj-SID explicitly via configuration, so the adj-SID survives router reboots.

FEC1 uses dynamic SID assignment, while FEC2 uses explicit SID assignment. So FEC2 wins.

A.2.4. Example 4

Illustration of incoming label collision resolution using FEC type default administrative distance

FEC1:
OSPF on node A receives a prefix SID advertisement from node B for 198.51.100.8/32 with index=8. Assuming that OSPF SRGB on node A = [1000,1999], the incoming label corresponding to 198.51.100.8/32 is 1008.

FEC2:
Suppose the SR Policy advertisement from controller to node A for the policy identified by (Endpoint = 192.0.2.208, color = 100) and

consisting of SID-List = <S1, S2> assigns the globally significant Binding-SID label 1008

From the point of view of node A, FEC1 and FEC2 both use dynamic SID assignment. Based on the default administrative distance outlined in Section 2.5.1, the binding SID has a higher administrative distance than the prefix-SID and hence FEC1 wins.

A.2.5. Example 5

Illustration of incoming label collision resolution based on FEC type preference

FEC1:
ISIS on node A receives a prefix SID advertisement from node B for 203.0.113.110/32 with index=10. Assuming that the ISIS SRGB on node A is [1000,1999], then incoming label corresponding to 203.0.113.110/32 is 1010.

FEC2:
ISIS on node A assigns the label 1010 to the globally significant adj-SID (I.e. when advertised the "L" flag is clear in the adj-SID sub-TLV as described in [I-D.ietf-isis-segment-routing-extensions]) towards one of its neighbors).

Node A allocates this adj-SID dynamically, and it may differ across router reboots. Hence both FEC1 and FEC2 both use dynamic SID assignment.

Since both FECs are from the same MCC, they have the same default admin distance. So we compare FEC type code-point. FEC1 has FEC type code-point=120, while FEC2 has FEC type code-point=130. Therefore, FEC1 wins.

A.2.6. Example 6

Illustration of incoming label collision resolution based on address family preference.

FEC1:
ISIS on node A receives prefix SID advertisement from node B for 203.0.113.111/32 with index 11. Assuming that the ISIS SRGB on node A is [1000,1999], the incoming label on node A for 203.0.113.111/32 is 1011.

FEC2:
ISIS on node A prefix SID advertisement from node C for 2001:DB8:1000::11/128 with index=11. Assuming that the ISIS SRGB on node A is [1000,1999], the incoming label on node A for 2001:DB8:1000::11/128 is 1011

FEC1 and FEC2 both use dynamic SID assignment. Since both FECs are from the same MCC, they have the same default admin distance. So we compare FEC type code-point. Both FECs have FEC type code-point=120. So we compare address family. Since IPv4 is preferred over IPv6, FEC1 wins.

A.2.7. Example 7

Illustration incoming label collision resolution based on prefix length.

FEC1:
ISIS on node A receives a prefix SID advertisement from node B for 203.0.113.112/32 with index 12. Assuming that ISIS SRGB on node A is [1000,1999], the incoming label for 203.0.113.112/32 on node A is 1012.

FEC2:
ISIS on node A receives a prefix SID advertisement from node C for 203.0.113.128/30 with index 12. Assuming that the ISIS SRGB on node A is [1000,1999], then incoming label for 203.0.113.128/30 on node A is 1012

FEC1 and FEC2 both use dynamic SID assignment. Since both FECs are from the same MCC, they have the same default admin distance. So we compare FEC type code-point. Both FECs have FEC type code-point=120. So we compare address family. Both are IPv4 address family, so we compare prefix length. FEC1 has prefix length=32, and FEC2 has prefix length=30, so FEC2 wins.

A.2.8. Example 8

Illustration of incoming label collision resolution based on the numerical value of the FECs.

FEC1:
ISIS on node A receives a prefix SID advertisement from node B for 203.0.113.113/32 with index 13. Assuming that ISIS SRGB on node A is

[1000,1999], then the incoming label for 203.0.113.113/32 on node A is 1013

FEC2:
ISIS on node A receives a prefix SID advertisement from node C for 203.0.113.213/32 with index 13. Assuming that ISIS SRGB on node A is [1000,1999], then the incoming label for 203.0.113.213/32 on node A is 1013

FEC1 and FEC2 both use dynamic SID assignment. Since both FECs are from the same MCC, they have the same default admin distance. So we compare FEC type code-point. Both FECs have FEC type code-point=120. So we compare address family. Both are IPv4 address family, so we compare prefix length. Prefix lengths are the same, so we compare prefix. FEC1 has the lower prefix, so FEC1 wins.

A.2.9. Example 9

Illustration of incoming label collision resolution based on routing instance ID.

FEC1:
ISIS on node A receives a prefix SID advertisement from node B for 203.0.113.114/32 with index 14. Assume that this ISIS instance on node A has the Routing Instance ID 1000 and SRGB [1000,1999]. Hence the incoming label for 203.0.113.114/32 on node A is 1014

FEC2:
ISIS on node A receives a prefix SID advertisement from node C for 203.0.113.114/32 with index=14. Assume that this is another instance of ISIS on node A with a different routing Instance ID 2000 but the same SRGB [1000,1999]. Hence incoming label for 203.0.113.114/32 on node A 1014

These two FECs match all the way through the prefix length and prefix. So Routing Instance ID breaks the tie, with FEC1 winning.

A.2.10. Example 10

Illustration of incoming label collision resolution based on topology ID.

FEC1:
ISIS on node A receives a prefix SID advertisement from node B for 203.0.113.115/32 with index=15. Assume that this ISIS instance on

node A has Routing Instance ID 1000. Assume that the prefix advertisement of 203.0.113.115/32 was received in ISIS Multi-topology advertisement with ID = 50. If the ISIS SRGB for this routing instance on node A is [1000,1999], then incoming label of 203.0.113.115/32 for topology 50 on node A is 1015

FEC2:
ISIS on node A receives a prefix SID advertisement from node C for 203.0.113.115/32 with index 15. Assume that it is the same routing Instance ID = 1000 but 203.0.113.115/32 was advertised with a different ISIS Multi-topology ID = 40. If the ISIS SRGB on node A is [1000,1999], then incoming label of 203.0.113.115/32 for topology 40 on node A is also 1015

These two FECs match all the way through the prefix length, prefix, and Routing Instance ID. We compare ISIS Multi-topology ID, so FEC2 wins.

A.2.11. Example 11

Illustration of incoming label collision for resolution based on algorithm ID.

FEC1:
ISIS on node A receives a prefix SID advertisement from node B for 203.0.113.116/32 with index=16 Assume that ISIS on node A has Routing Instance ID = 1000. Assume that node B advertised 203.0.113.116/32 with ISIS Multi-topology ID = 50 and SR algorithm = 0. Assume that the ISIS SRGB on node A = [1000,1999]. Hence the incoming label corresponding to this advertisement of 203.0.113.116/32 is 1016.

FEC2:
ISIS on node A receives a prefix SID advertisement from node C for 203.0.113.116/32 with index=16. Assume that it is the same ISIS instance on node A with Routing Instance ID = 1000. Also assume that node C advertised 203.0.113.116/32 with ISIS Multi-topology ID = 50 but with SR algorithm = 22. Since it is the same routing instance, the SRGB on node A = [1000,1999]. Hence the incoming label corresponding to this advertisement of 203.0.113.116/32 by node C is also 1016.

These two FECs match all the way through the prefix length, prefix, and Routing Instance ID, and Multi-topology ID. We compare SR algorithm ID, so FEC1 wins.

A.2.12. Example 12

Illustration of incoming label collision resolution based on FEC numerical value and independent of how the SID assigned to the colliding FECs.

FEC1:
ISIS on node A receives a prefix SID advertisement from node B for 203.0.113.117/32 with index 17. Assume that the ISIS SRGB on node A is [1000,1999], then the incoming label is 1017

FEC2:
Suppose there is an ISIS mapping server advertisement (SID/Label Binding TLV) from node D has Range 100 and Prefix = 203.0.113.1/32. Suppose this mapping server advertisement generates 100 mappings, one of which maps 203.0.113.17/32 to index 17. Assuming that it is the same ISIS instance, then the SRGB is [1000,1999] and hence the incoming label for 1017.

The fact that FEC1 comes from a normal prefix SID advertisement and FEC2 is generated from a mapping server advertisement is not used as a tie-breaking parameter. Both FECs use dynamic SID assignment, are from the same MCC, have the same FEC type code-point=120. Their prefix lengths are the same as well. FEC2 wins based on lower numerical prefix value, since 203.0.113.17 is less than 203.0.113.117.

A.2.13. Example 13

Illustration of incoming label collision resolution based on address family preference

FEC1:
SR Policy advertisement from controller to node A. Endpoint address=2001:DB8:3000::100, color=100, SID-List=<S1, S2> and the Binding-SID label=1020

FEC2:
SR Policy advertisement from controller to node A. Endpoint address=192.0.2.60, color=100, SID-List=<S3, S4> and the Binding-SID label=1020

The FECs match through the tie-breaks up to and including having the same FEC type code-point=140. FEC2 wins based on IPv4 address family being preferred over IPv6.

A.2.14. Example 14

Illustration of incoming label resolution based on numerical value of the policy endpoint.

FEC1:
SR Policy advertisement from controller to node A. Endpoint address=192.0.2.70, color=100, SID-List=<S1, S2> and Binding-SID label=1021

FEC2:
SR Policy advertisement from controller to node A Endpoint address=192.0.2.71, color=100, SID-List=<S3, S4> and Binding-SID label=1021

The FECs match through the tie-breaks up to and including having the same address family. FEC1 wins by having the lower numerical endpoint address value.

A.3. Examples for the Effect of Incoming Label Collision on Outgoing Label

This section presents examples to illustrate the effect of incoming label collision on the selection of the outgoing label described in Section 2.6.

A.3.1. Example 1

Illustration of the effect of incoming label resolution on the outgoing label

FEC1:
ISIS on node A receives a prefix SID advertisement from node B for 203.0.113.122/32 with index 22. Assuming that the ISIS SRGB on node A is [1000,1999] the corresponding incoming label is 1022.

FEC2:
ISIS on node A receives a prefix SID advertisement from node C for 203.0.113.222/32 with index=22 Assuming that the ISIS SRGB on node A is [1000,1999] the corresponding incoming label is 1022.

FEC1 wins based on lowest numerical prefix value. This means that node A installs a transit MPLS forwarding entry to SWAP incoming label 1022, with outgoing label N and use outgoing interface I. N is determined by the index associated with FEC1 (index 22) and the SRGB advertised by the next-hop node on the shortest path to reach 203.0.113.122/32.

Node A will generally also install an imposition MPLS forwarding entry corresponding to FEC1 for incoming prefix=203.0.113.122/32 pushing outgoing label N, and using outgoing interface I.

The rule in Section 2.6 means node A MUST NOT install an ingress MPLS forwarding entry corresponding to FEC2 (the losing FEC, which would be for prefix 203.0.113.222/32).

A.3.2. Example 2

Illustration of the effect of incoming label collision resolution on outgoing label programming on node A

FEC1:

o SR Policy advertisement from controller to node A

o Endpoint address=192.0.2.80, color=100, SID-List=<S1, S2>

o Binding-SID label=1023

FEC2:

o SR Policy advertisement from controller to node A

o Endpoint address=192.0.2.81, color=100, SID-List=<S3, S4>

o Binding-SID label=1023

FEC1 wins by having the lower numerical endpoint address value. This means that node A installs a transit MPLS forwarding entry to SWAP incoming label=1023, with outgoing labels and outgoing interface determined by the SID-List for FEC1.

In this example, we assume that node A receives two BGP/VPN routes:

o R1 with VPN label=V1, BGP next-hop = 192.0.2.80, and color=100,

o R2 with VPN label=V2, BGP next-hop = 192.0.2.81, and color=100, We also assume that A has a BGP policy which matches on color=100 that allows that its usage as SLA steering information. In this case, node A will install a VPN route with label stack = <S1,S2,V1> (corresponding to FEC1).

The rule described in section 2.6 means that node A MUST NOT install a VPN route with label stack = <S3,S4,V1> (corresponding to FEC2.)

spring R. Leipnitz, Ed.
Internet-Draft R. Geib
Intended status: Informational Deutsche Telekom
Expires: December 24, 2016 June 22, 2016

A scalable and topology aware MPLS data plane monitoring system
draft-leipnitz-spring-pms-implementation-report-00

Abstract

This document reports round-trip delay measurements captured by a single MPLS Path Monitoring System (PMS) compared with results of an IPPM conformant measurement system, consisting of three different Measurement Agents. The measurements were made in a research backbone with an LDP control plane. The packets of the MPLS PMS use label stacks similar to those to be used by a segment routing MPLS PMS. The measurement packets of the MPLS PMS remained in the network data plane.

Table of Contents

1. Introduction 2
2. Measurement system implementation 3
  2.1. A PMS based round-trip delay measurement system 3
  2.2. Perfas+ IPPM measurement system 4
3. Test set up 4
4. Measurement Result Evaluation 6
5. Measurement results 6
  5.1. Round-trip delay measurement and ADK test results 6
  5.2. PMS delay measurements with IP-address variation 9
6. Error Calibration 10
7. Summary 11
8. Acknowledgements 12
9. IANA Considerations 12
10. Security Considerations 12
11. References 12
  11.1. Normative References 12
  11.2. Informative References 13
Appendix A. ADK2 Test Source Code 13
Authors’ Addresses 22

1. Introduction

Deutsche Telekom has implemented an MPLS Path Monitoring System (PMS). The PMS operates on MPLS networks with LDP control plane. Forwarding follows the principles of Segment Routing, i.e. the packets sent by the PMS use stacked transport labels to execute a combination of MPLS paths and finally return to the PMS. The PMS is connected to a research backbone of Deutsche Telekom spanning parts of Germany. One of the new network monitoring features enabled by Segment Routing are round-trip delay measurements purely executed in data plane. Deutsche Telekom captured delays between three IPPM standard conformant Measurement Agents and compared these with delays measured along identical backbone paths by a single PMS. To prove that the same delays were measured the IPPM results were then compared with the PMS results by applying IPPM methodology as specified by [RFC6576]. Some results passed this test, while others did not. The results of both systems seemed to differ by very small and relatively stable latencies. As the research network only offered single paths between the involved routers, processing of different flows in parallel forwarding instances of the routers along the paths offered an explanation. The PMS was used to execute some measurements whose results at least are not contradicting that assumption.

The results reported here show that a PMS [I-D.ietf-spring-oam-usecase] can be built and operated (also as part of an LDP based MPLS network). To set up packets with proper label stacks, the PMS needs to be aware of the MPLS topology of the network. MPLS topology awareness within an LDP based network requires reasonable effort. Segment Routing will significantly simplify detection of the MPLS topology. Delay measurements where picked here to give an example of a feature which can be supported by a PMS. Others are possible, like checking continuity of arbitrary segmented routed MPLS paths [I-D.ietf-spring-oam-usecase].

The remaining document is organized as follows: Section 2 briefly informs about the PMS and IPPM measurement system implementation. Section 3 introduces the measurement set up within the research network. Section 4 briefly discusses the test by which the measurements were compared. Section 5 informs about the test results and Section 6 about an IPPM error calibration. Section 7 sums up the document.

2. Measurement system implementation

Deutsche Telekom operates an IPPM standard conformant performance measurement system called Perfas+. Deutsche Telekom intends deployment of an MPLS PMS to monitor the IP performance in network segments connecting roughly 1000 edge routers to the IP-backbone. 11 MPLS PMS are supposed to execute backbone to edge performance monitoring. Had the monitoring system been based on IPPM, one IPPM system had been required per edge router.

2.1. A PMS based round-trip delay measurement system

Deutsche Telekom has implemented an MPLS PMS. The PMS is part of an MPLS research and development backbone of Deutsche Telekom. This backbone only supports LDP routing. The PMS works with an LDP control plane. Detecting the MPLS topology of an LDP based MPLS network is more complex, than doing this by Segment Routing. The PMS consists of the following logical components:

- o An MPLS Label detection system. It is collecting MPLS routing information from all MPLS routers of the MPLS network by management plane access (see e.g. [LDP-TE], [BCP-TX])

- o An MPLS topology database.

- o A measurement system able to compose packets executing any combination MPLS Label Switched Paths (MPLS LSP) which are part of the MPLS topology database. The measurement system further is able to measure delays, if the final address information of the measurement packet directs the packet back to the PMS after the MPLS LSPs to be measured have been passed.

- o An IGP topology detection system. It is passively listening to IGP routing.

- o A measurement system which is complying to [RFC4379].

Note that the final two MPLS PMS functionalities are required if ECMP routed paths should be detected and addressed by [RFC4379] functions. No ECMP routed paths are present between the sites involved in the measurement set up. The role of these components is reduced to detection of operational issues, should the measurement not work as expected.

While the control plane of the network monitored by the PMS is LDP based, the measurement packets used to execute MPLS LSPs apply the forwarding mechanisms as within a Segment Routing network.

2.2. Perfas+ IPPM measurement system

IPPM conformant one-way delay measurements were performed by Perfas+ Measurement Agents. Three Perfas+ Measurement Agents are connected to edge routers at three different sites of the research network. Perfas+ is one of the few IPPM implementations with proven conformance to some standard IPPM metrics, like one-way delay [RFC6808]. Two of the Perfas+ Measurement Agents were synchronized by NTP only. Due to this restriction, the comparison with the PMS measurements are limited to round-trip times (round-trip delays, RTD). As no ECMP routed paths are active between the sites used for test execution, two back and forth Perfas+ one-way delay measurements between two sites were added to result in an RTD value.

3. Test set up

The test set up is shown in the figure below. The PMS and Perfas+ Measurement Agent 1 (PerfMA 1) are connected to the same LER.

```
                          +--------+
                          |PerfMA 1|
                          +--------+
                              |
                  +---+   +-----+
                  |PMS|---|LER 1|
                  +---+   +-----+
                              |
                          ~~~~~~~
                         /       \   +-----+  +--------+
                        (  MPLS   )--|LER 2|--|PerfMA 2|
                        ( Network )  +-----+  +--------+
                         \       /
                          ~~~~~~~
                              |
                          +-----+  +--------+
                          |LER 3|--|PerfMA 3|
                          +-----+  +--------+
```

Figure 1: Test set up

The Perfas+ Measurement Agents (MAs) measure the one-way delay to each of the remote Perfas+ MAs. The PMS measures the round-trip delay from LER 1 to LER 2 and back as well the round-trip delay from LER 1 to LER 3 and back. The measurements start and terminate at the PMS, but this segment is omitted here. The round-trip delay from LER 2 to LER 3 is measured along two path combinations by the PMS. The first measurement path is LER 1 to LER 2 to LER 3 and back exactly that way. The round-trip delay LER 1 to LER 2 captured earlier by the PMS is subtracted from the result. The other measurement is LER 1 to LER 3 to LER 2 and back exactly that way. Here, the PMS round-trip delay LER 1 to LER 3 is subtracted to receive the round-trip delay LER 2 to LER 3.

There is a small LAN section causing limited additional latencies for the IPPM measurement. The measurements were executed with an IP packet size of 64 Byte. Perfas is attached by an IP-VPN. The PMS label stack is differing slightly. The assumption is that both differences have minor impact. Note that IPPM metrics expect similar results if differences in measurement set up can be neglected. The sending interval is 10 seconds periodic. A measurement mean is calculated from 10 consecutive measurement packets. The measurements were repeated for 8 hours, resulting in 288 mean values collected per round-trip delay measurement path and measurement system.

The resulting round-trip delays are divided by two and indicate the one-way delay. This seems sound, as there is no path diversity in the research network and the low standard deviation of the results (single digit [us] figures in all cases, see test results below) indicate that no link was congested.

4. Measurement Result Evaluation

IPPM WG applies the Anderson-Darling-K-Sample (ADK) test to compare up to which temporal resolution the results of two measurements share the same statistical distribution [RFC6576]. To decide, whether Perfas+ and the PMS were measuring identical data, the round-trip delays captured along identical measurement paths were compared by an ADK test. (The ADK test source code is given at Appendix A). Note that the ADK test does not judge accuracy (i.e. it does not test whether the result is close to the true value?), ADK rather judges precision (that the test estimates whether the same value was measured by repeated measurements). As applied here, an RTD sample of Perfas+ was compared with one of the PMS captured along the same path.

To illustrate, how sensible the ADK test is to changes in a measurement environment, a PMS round-trip delay test was set up where all configurations were identical and only packet size was variable. Obviously all paths are identical, so any difference in results is caused by the packet size only (64, 128 and 256 Byte were picked). The ADK test indicated a reasonably high probability that results do not follow the same distribution in roughly half of the cases (i.e. ADK test said that the distribution of round-trip delays captured with packet size of 64 bytes follows a different distribution than the round-trip delays captured with a packet size of 128 Byte).

5. Measurement results

5.1. Round-trip delay measurement and ADK test results

The one-way delays between Perfas MA 1 and Perfas MA 2 calculated on basis of the round-trip Delay and the ADK test results comparing them to the measurement results captured by the PMS are shown in Table 1.

Test metric	PERFAS+	PMS
minimum [us]	691.5	695.5
maximum [us]	701	704.5
mean [us]	695.4	699.6
median [us]	695.5	699.5
standard deviation [us]	1.4	1.7
ADK value		278.445
ADK value with adjustment of mean		1.701
ADK value with adjustment of median		1.982

Perfas+ and PMS OWD measurement results for path LER 1 to LER 2 and ADK test results

Table 1: Perfas+ and PMS OWD measurement results for path LER 1 to LER 2 and ADK test results

The ADK test result is surprisingly good and was not expected a priori. As mentioned, ADK is a very sensible test. When IPPM WG worked on [RFC6808], the packets used by two different IPPM implementation only passed ADK after a network emulator was inserted into the measurement path. As IPPM puts more emphasis on precision than on accuracy, correcting tests samples to result by the same mean for small and constant differences is plausible. Still, the smallest temporal resolution of the standard deviation by which ADK was passed when used to compare two IPPM implementations for [RFC6808] was single digit milliseconds. No network emulator has been used when comparing Perfas+ and the PMS. After adjusting the means, ADK is passed by a temporal resolution of the standard deviation of single digit microseconds!

The one-way delays between Perfas MA 1 and Perfas MA 3 calculated on basis of the round-trip Delay and the ADK test results comparing them to the measurement results as captured by the PMS are shown in Table 2.

Test metric	PERFAS+	PMS
minimum [us]	2991.5	2983
maximum [us]	3008.5	2994.5
mean [us]	2995.7	2988.1
median [us]	2995.5	2988
standard deviation [us]	1.9	2.1
ADK value		231.638
ADK value with adjustment of mean		1.886
ADK value with adjustment of median		2.026

Perfas+ and PMS OWD measurement results for path LER 1 to LER 3 and ADK test results

Table 2: Perfas+ and PMS OWD measurement results for path LER 1 to LER 3 and ADK test results

After adjustment of the means values, also here the ADK test is passed. Comparing Table 1 with Table 2 readers figure can see, that once mean the one-way delay measured by Perfas+ is lower, while in the other case the mean one-way delay captured by the PMS is lower. This behavior was visible in all our measurements. The delays measured per path by one system were always bigger than that of the other along the same path (for all single 10 sample mean values of the time series).

We now compare the one-way delays between Perfas MA 2 and Perfas MA 3 calculated on basis of the round-trip delay and the ADK test results comparing them to the measurement results as captured by the PMS are shown in Table 3.

Test metric	PERFAS+	PMS over LER 2	PMS over LER 3
minimum [us]	3606.5	3551	3542.5
maximum [us]	3659	3568	3558
mean [us]	3611.9	3560.1	3549,8
median [us]	3609	3560	3549,5
standard deviation [us]	8.3	2.9	2.9
ADK value		231.144	231.094
ADK value with adjustment of mean		54.591	56.589
ADK value with adjustment of median		8.915	10.054

Perfas+ and PMS OWD measurement results for path LER 2 to LER 3 and ADK test results

Table 3: Perfas+ and PMS OWD measurement results for path LER 2 to LER 3 and ADK test results

In this case, the ADK test fails (the cause is the difference of the standard deviation, not the mean or median difference). Note that in terms of mean values the difference is around 50 us between Perfas and PMS. The relative error is 1,75%. While ADK indicates that both distributions deviate, human perception may confirm that both results capture delays along the same path.

It is interesting however, that the two PMS measurements deviate in the mean values. And again, the one showing the lower delay does so sample mean measurements. A brief test investigating this symptom was performed. Test and results follow in the next section.

5.2. PMS delay measurements with IP-address variation

The PMS allows to send measurement packets with different destination IP-addresses (routing based on IP-addresses only occurs from LER 1 to PMS and only in this direction). While the IP-address varied, the MPLS Label stack and thus the MPLS path was kept identical. This measurement can only be configured by CLI configuration. Per IP destination address, the mean-value of 10 round-trip delay times was captured. After some measurements the IP-addresses showing the biggest round-trip delay difference were selected for further testing. With these IP-addresses, the test was repeated at different days and daytimes. Overall we had at least 10 more measurement values of every of these IP-addresses. The PMS is connected with two interfaces to two different LERs of the same site. Both interfaces and LERs respectively were used to perform the measurements. As has been mentioned already, the network does not have ECMP-paths. Table 4 shows the results of the two measurements with the biggest difference in results. The mean delays measured with IP-address a.b.c.0 were the smallest. They were always smaller than those delays captured with IP-address a.b.c.32, which were the biggest. The difference of the mean values from the measurement over the first interface was 19.5 us and 14.4 us over the second interface.

Interface / IP-address	mean [us]	median [us]
one / a.b.c.0	1413.2	1412
one / a.b.c.32	1432.7	1433
two / a.b.c.0	1446.4	1446
two / a.b.c.32	1460.8	1460.5

Table 4: Destination-IP-address variation

Parallel hardware processing within some or all of the routers passed on the measurement paths may be a plausible explanation. Investigating the cause for this behavior was however not the main aim of the test activities documented here. Further activities related to this issue are left to interested research.

6. Error Calibration

Section 3.7. and following of [RFC2679] recommend an error calibration of the (IPPM) measurement clients. The one-way delay of a back-to-back connection of two PERFAS+ clients is measured. Table 5 shows the characteristics of this calibration measurement. The negative values for the one-way delay shown in the table, are physically impossible. The standard deviation is very high. It was decided to calibrate with the round-trip delay which is shown in Table 6. Referring to section 3.7.3 of [RFC2679] there is a systematic error and a random error. The systematic error is the median of the measurement with 49.5 us. The random error is the difference between the median and the 2.5% percentile, which is 17 us. (The random error is the larger absolute value between the median and the 2.5% percentile and the 97.5% percentile; the calculation is |49.5 - 32.5| > |49.5 - 59.5|). The resolution of the PERFAS+ Measurement Agents is 1 us, so the absolute random error is 19 us. So measurement error is 49.5 +/- 19 us. (The synchronization error is 0, as two one-way delays are added, making this error disappear). There was no possibility to calibrate the PMS. The error is assumed to be the same like that of PERAS+, because the PMS is based on the same hardware (and possibly the same host-system).

Test metric	PERFAS+
minimum [us]	-55
maximum [us]	39
mean [us]	-38
median [us]	-23.1
standard deviation [us]	29.4

Table 5: measurement results of one-way delay of back-to-back connection from two PERFAS+ clients at 64 Bytes

Test metric	PERFAS+
minimum [us]	26
maximum [us]	205
mean [us]	49.1
median [us]	49.5
standard deviation [us]	7.6
2.5% percentile [us]	32.5
97.5% percentile [us]	59.5

Table 6: measurement results of both one-way delays of back-to-back connection between two PERFAS+ clients at 64 Bytes

7. Summary

By an IPPM measurement system like PERFAS+ three physical measurement clients are needed to measure the round-trip delay between all sites. With the PMS the same measurements can be performed with only one client. In theory one PMS could monitor a whole MPLS-enabled backbone. The GPS receivers of two IPPM measurement agents were not available, hence the one-way delay could not be captured with the IPPM system PERFAS+. Otherwise a direct comparison with calculated one-way delay values based on the PMS measured values would have been possible. This could be done in future. The results shown in Section 4 indicate, that the PMS measurements equal those captured by an IPPM conformant measurement system. The ADK test is successful by comparing the measurement values of the round-trip delays for packets with a size of 64 bytes. The network does not include an impairment generator (which was required within a test set up to compare independent IPPM implementations, see [RFC6808]). An impairment generator as part of the test set up will have a positive effect on the measurements and the measurements with bigger packet size will also succeed at a temporal resolution above [us] level.

8. Acknowledgements

Joachim Mende, Marc Wieland, Ralf Widera and Jens Wyduba helped to implement and operate the LDP PMS in our research network. In memoriam of Holger Zarwel, who gave our project unconditional support.

9. IANA Considerations

This memo includes no request to IANA.

10. Security Considerations

A PMS monitoring packet should never leave the domain where it originated. It therefore should never use stale MPLS or IGP routing information. If the Label Switch Path is broken, a packet with the destination address 127.0.0.0/26 should not be routed, it should be discarded. The PMS must be configured with a measurement interval (or sum of all measurement stream intervals) that does not overload the network. Too many measurement streams with a big packet size could overload a link.

11. References

11.1. Normative References

[RFC2679] Almes, G., Kalidindi, S., and M. Zekauskas, "A One-way Delay Metric for IPPM", RFC 2679, DOI 10.17487/RFC2679, September 1999, <http://www.rfc-editor.org/info/rfc2679>.

[RFC4379] Kompella, K. and G. Swallow, "Detecting Multi-Protocol Label Switched (MPLS) Data Plane Failures", RFC 4379, DOI 10.17487/RFC4379, February 2006, <http://www.rfc-editor.org/info/rfc4379>.

[RFC6576] Geib, R., Ed., Morton, A., Fardid, R., and A. Steinmitz, "IP Performance Metrics (IPPM) Standard Advancement Testing", BCP 176, RFC 6576, DOI 10.17487/RFC6576, March 2012, <http://www.rfc-editor.org/info/rfc6576>.

[RFC6808] Ciavattone, L., Geib, R., Morton, A., and M. Wieser, "Test Plan and Results Supporting Advancement of RFC 2679 on the Standards Track", RFC 6808, DOI 10.17487/RFC6808, December 2012, <http://www.rfc-editor.org/info/rfc6808>.

11.2. Informative References

[BCP-TX]   NANOG, "Best Practices for Determining Traffic Matrices in IP Networks V 4.0", 2008.

[I-D.ietf-spring-oam-usecase]
Geib, R., Filsfils, C., Pignataro, C., and N. Kumar, "A Scalable and Topology-Aware MPLS Dataplane Monitoring System", draft-ietf-spring-oam-usecase-03 (work in progress), April 2016.

[LDP-TE]   VDE-Verlag, "Traffic Matrices for MPLS Networks with LDP Traffic Statistics", 2004.

Appendix A. ADK2 Test Source Code

The following C++ source code is a modified version of the Code at [RFC6576]. This version allows to test two files containing values with the ADK2. It is not necessary that the values are sorted, because in the first step the values get sorted.

```
/*
Copyright (c) 2012 IETF Trust and the persons identified
as authors of the code.  All rights reserved.

Redistribution and use in source and binary forms, with
or without modification, is permitted pursuant to, and subject
to the license terms contained in, the Simplified BSD License
set forth in Section 4.c of the IETF Trust's Legal Provisions
Relating to IETF Documents (http://trustee.ietf.org/license-info).
*/

/* Routines for computing the Anderson-Darling 2 sample
* test statistic.
*
* Implemented based on the description in
* "Anderson-Darling K Sample Test" Heckert, Alan and
* Filliben, James, editors, Dataplot Reference Manual,
* Chapter 15 Auxiliary, NIST, 2004.
* Official Reference by 2010
* Heckert, N. A. (2001).  Dataplot website at the
* National Institute of Standards and Technology:
* http://www.itl.nist.gov/div898/software/dataplot.html/
* June 2001.
*/

// this code is a modified version of the code in RFC6576
```

```
// use '-std=c++11' for compiling

#include <iostream>
#include <fstream>
#include <vector>
#include <sstream>
#include <iterator>

#include <algorithm>


using namespace std;

/* This function reads the values and sorts this in an ascending
* order.
* The format is: one value per line followed by a line break.
* A blank line at the end of the file will crash the program.
*/
vector<double> read_file_sort (string filename) {
    vector<double> vec;
    // variable for one line of the file and the value
    string line;
    double tmp;

    ifstream file;
    file.open(filename, ios::in);
    if (!file) {
        cout << "Error in file " << filename << endl;
    }
    else {
        // read file in a vector
        while(!file.eof()) {
            getline (file, line);
            tmp = stod (line);
            vec.push_back(tmp);
        }
        // sort the vector ascending
        sort(vec.begin(), vec.end());
    }
    file.close();
    return vec;
}

 int main(int argn, char *argv[]) {

    if (argn != 1 && argn != 3) {
        cout << "wrong invocation" << endl;
        cout << "start with " << argv[0] << " file1 file2" << endl;
```

```
            cout << "start with " << argv[0] << " without parameter, if \
            the files are named file1.csv and file2.csv" << endl;
            return 1;
        }

        vector<double> vec1, vec2;
        double adk_result;
        static int k, val_st_z_samp1, val_st_z_samp2,
                   val_eq_z_samp1, val_eq_z_samp2,
                   j, n_total, n_sample1, n_sample2, L,
                   max_number_samples, line, maxnumber_z;
        static int column_1, column_2;
        static double adk, n_value, z, sum_adk_samp1,
                      sum_adk_samp2, z_aux;
        static double H_j, F1j, hj, F2j, denom_1_aux, denom_2_aux;
        static bool next_z_sample2, equal_z_both_samples;
        static int stop_loop1, stop_loop2, stop_loop3,old_eq_line2,
                   old_eq_line1;

        static double adk_criterium = 1.993;

        string filename1 = "file1.csv";
        string filename2 = "file2.csv";

        // if called with filenames
        if (argn == 3) {
            filename1 = argv[1];
            filename2 = argv[2];
        }

        // sort the two files i a vector
        vec1 = read_file_sort(filename1);
        vec2 = read_file_sort(filename2);

        k = 2;
        n_sample1 = vec1.size() - 1;
        n_sample2 = vec2.size() - 1;

        // -1 because vec[0] is a dummy value
        n_total = n_sample1 + n_sample2;

        /* value equal to the line with a value = zj in sample 1.
         * Here j=1, so the line is 1.
         */
        val_eq_z_samp1 = 1;

        /* value equal to the line with a value = zj in sample 2.
         * Here j=1, so the line is 1.
```

```
         */
        val_eq_z_samp2 = 1;

        /* value equal to the last line with a value < zj
         * in sample 1.  Here j=1, so the line is 0.
         */
        val_st_z_samp1 = 0;

        /* value equal to the last line with a value < zj
         * in sample 1.  Here j=1, so the line is 0.
         */
        val_st_z_samp2 = 0;

        sum_adk_samp1 = 0;
        sum_adk_samp2 = 0;
        j = 1;

        // as mentioned above, j=1
        equal_z_both_samples = false;

        next_z_sample2 = false;

        // assuming the next z to be of sample 1
        stop_loop1 = n_sample1 + 1;

        // + 1 because vec[0] is a dummy, see n_sample1 declaration
        stop_loop2 = n_sample2 + 1;
        stop_loop3 = n_total + 1;

        /* The required z values are calculated until all values
         * of both samples have been taken into account.  See the
         * lines above for the stoploop values.  Construct required
         * to avoid a mathematical operation in the while condition.
        */
        while (((stop_loop1 > val_eq_z_samp1)

               || (stop_loop2 > val_eq_z_samp2)) && stop_loop3 > j) {
            if (val_eq_z_samp1 < n_sample1+1) {
                /* here, a preliminary zj value is set.
                 * See below how to calculate the actual zj.
                 */
                z = vec1[val_eq_z_samp1];

                /* this while sequence calculates the number of values
                 * equal to z.
                 */
                while ((val_eq_z_samp1+1 < n_sample1)
                    && z == vec1[val_eq_z_samp1+1] ) {
```

```
                val_eq_z_samp1++;
                }
        }
        else {
        val_eq_z_samp1 = 0;
        val_st_z_samp1 = n_sample1;

        // this should be val_eq_z_samp1 - 1 = n_sample1
        }

        if (val_eq_z_samp2 < n_sample2+1) {
            z_aux = vec2[val_eq_z_samp2];

            /* this while sequence calculates the number of values
             * equal to z_aux
             */

            while ((val_eq_z_samp2+1 < n_sample2)
                    && z_aux == vec2[val_eq_z_samp2+1] ) {
                val_eq_z_samp2++;
            }

            /* the smaller of the two actual data values is picked
             * as the next zj.
             */

            if(z > z_aux) {
                z = z_aux;
                next_z_sample2 = true;
            }
            else {
                if (z == z_aux) {
                equal_z_both_samples = true;
                }

                /* This is the case if the last value of column1 is
                 * smaller than the remaining values of column2.
                 */
               if (val_eq_z_samp1 == 0) {
                z = z_aux;
                next_z_sample2 = true;
                }
            }
        }
        else {
            val_eq_z_samp2 = 0;
            val_st_z_samp2 = n_sample2;
```

```
            // this should be val_eq_z_samp2 - 1 = n_sample2
        }

         /* in the following, sum j = 1 to L is calculated for
          * sample 1 and sample 2.
          */
        if (equal_z_both_samples) {

            /* hj is the number of values in the combined sample
             * equal to zj
             */
            hj = val_eq_z_samp1 - val_st_z_samp1
              + val_eq_z_samp2 - val_st_z_samp2;

            /* H_j is the number of values in the combined sample
             * smaller than zj plus one half the number of
             * values in the combined sample equal to zj
             * (that's hj/2).
             */
            H_j = val_st_z_samp1 + val_st_z_samp2 + hj / 2;

            /* F1j is the number of values in the 1st sample
             * that are less than zj plus one half the number
             * of values in this sample that are equal to zj.
             */

            F1j = val_st_z_samp1 + (double)
                  (val_eq_z_samp1 - val_st_z_samp1) / 2;

            /* F2j is the number of values in the 1st sample
             * that are less than zj plus one half the number
             * of values in this sample that are equal to zj.
             */
            F2j = val_st_z_samp2 + (double)
                 (val_eq_z_samp2 - val_st_z_samp2) / 2;

            /* set the line of values equal to zj to the
             * actual line of the last value picked for zj.
             */
            val_st_z_samp1 = val_eq_z_samp1;

            /* Set the line of values equal to zj to the actual
             * line of the last value picked for zj of each
             * sample.  This is required as data smaller than zj
             * is accounted differently than values equal to zj.
             */
            val_st_z_samp2 = val_eq_z_samp2;
```

```
               /* next the lines of the next values z, i.e., zj+1
                * are addressed.
                */
               val_eq_z_samp1++;

               /* next the lines of the next values z, i.e.,
                * zj+1 are addressed
                */
               val_eq_z_samp2++;
           }
           else {

               /* the smaller z value was contained in sample 2;
                * hence, this value is the zj to base the following
                * calculations on.
                */
               if (next_z_sample2){
                   /* hj is the number of values in the combined
                    * sample equal to zj; in this case, these are
                    * within sample 2 only.
                    */
                   hj = val_eq_z_samp2 - val_st_z_samp2;

                   /* H_j is the number of values in the combined sample
                    * smaller than zj plus one half the number of
                    * values in the combined sample equal to zj
                    * (that's hj/2).
                    */
                   H_j = val_st_z_samp1 + val_st_z_samp2 + hj / 2;

                 /* F1j is the number of values in the 1st sample that
                * are less than zj plus one half the number of values in
                  * this sample that are equal to zj.
                  * As val_eq_z_samp2 < val_eq_z_samp1, these are the
                  * val_st_z_samp1 only.
                  */
                   F1j = val_st_z_samp1;

                 /* F2j is the number of values in the 1st sample that
                * are less than zj plus one half the number of values in
                * this sample that are equal to zj.  The latter are from
                  * sample 2 only in this case.
                  */

                   F2j = val_st_z_samp2 + (double)
                        (val_eq_z_samp2 - val_st_z_samp2) / 2;

                /* Set the line of values equal to zj to the actual line
```

```
                   * of the last value picked for zj of sample 2 only in
                   * this case.
                   */
                   val_st_z_samp2 = val_eq_z_samp2;

                  /* next the line of the next value z, i.e., zj+1 is
                   * addressed.  Here, only sample 2 must be addressed.
                   */

                    val_eq_z_samp2++;
                    if (val_eq_z_samp1 == 0) {
                        val_eq_z_samp1 = stop_loop1;
                    }
                }
               /* the smaller z value was contained in sample 2;
                * hence, this value is the zj to base the following
                * calculations on.
                */

                else {

                    /* hj is the number of values in the combined
                     * sample equal to zj; in this case, these are
                     * within sample 1 only.
                     */
                    hj = val_eq_z_samp1 - val_st_z_samp1;

                    /* H_j is the number of values in the combined
                     * sample smaller than zj plus one half the number
                     * of values in the combined sample equal to zj
                     * (that's hj/2).
                     */

                    H_j = val_st_z_samp1 + val_st_z_samp2 + hj / 2;

                /* F1j is the number of values in the 1st sample that
                 * are less than zj plus; in this case, these are within
                 * sample 1 only one half the number of values in this
                 * sample that are equal to zj.  The latter are from
                 * sample 1 only in this case.
                 */

                    F1j = val_st_z_samp1 + (double)
                        (val_eq_z_samp1 - val_st_z_samp1) / 2;

                /* F2j is the number of values in the 1st sample that
                 * are less than zj plus one half the number of values
                 * in this sample that are equal to zj.  As
```

```
                * val_eq_z_samp1 < val_eq_z_samp2, these are the
                * val_st_z_samp2 only.
                */

                   F2j = val_st_z_samp2;

               /* Set the line of values equal to zj to the actual line
                * of the last value picked for zj of sample 1 only in
                * this case.
                */

                   val_st_z_samp1 = val_eq_z_samp1;
                   /* next the line of the next value z, i.e., zj+1 is
                    * addressed.  Here, only sample 1 must be addressed.
                    */
                   val_eq_z_samp1++;

                   if (val_eq_z_samp2 == 0) {
                       val_eq_z_samp2 = stop_loop2;
                   }
               }
           }

           denom_1_aux = n_total * F1j - n_sample1 * H_j;
           denom_2_aux = n_total * F2j - n_sample2 * H_j;

           sum_adk_samp1 = sum_adk_samp1 + hj
                           * (denom_1_aux * denom_1_aux) /
                           (H_j * (n_total - H_j)
                           - n_total * hj / 4);
           sum_adk_samp2 = sum_adk_samp2 + hj
                           * (denom_2_aux * denom_2_aux) /
                           (H_j * (n_total - H_j)
                           - n_total * hj / 4);

           next_z_sample2 = false;
           equal_z_both_samples = false;

           /* index to count the z.  It is only required to prevent
            * the while slope to execute endless
            */
           j++;
       }

       // calculating the adk value is the final step.
       adk_result = (double) (n_total - 1) / (n_total
              * n_total * (k - 1))
               * (sum_adk_samp1 / n_sample1
```

```
               + sum_adk_samp2 / n_sample2);


       /* if(adk_result <= adk_criterium)
        * adk_2_sample test is passed
        */
       //return adk_result <= adk_criterium;
       cout << "Result: " << adk_result << endl;
   }
```

Authors' Addresses

Raik Leipnitz (editor)
Deutsche Telekom
Olgastr. 67
Ulm 89073
Germany

Email: r.leipnitz@telekom.de

Ruediger Geib
Deutsche Telekom
Heinrich Hertz Str. 3-7
Darmstadt 64295
Germany

Phone: +49 6151 5812747
Email: Ruediger.Geib@telekom.de