

Network Working Group
Internet-Draft
Intended status: Informational
Expires: January 21, 2016

C. Filsfils, Ed.
S. Previdi, Ed.
Cisco Systems, Inc.
J. Mitchell
Unaffiliated
E. Aries
P. Lapukhov
G. Nagarajan
Facebook
D. Afanasiev
Yandex
T. Laberge
E. Nkposong
M. Nanduri
Microsoft
J. Uttaro
ATT
S. Ray
Unaffiliated
July 20, 2015

BGP-Prefix Segment in large-scale data centers
draft-filsfils-spring-segment-routing-msdc-03

Abstract

This document describes the motivation and benefits for applying segment routing in the data-center. It describes the design to deploy segment routing in the data-center, for both the MPLS and IPv6 dataplanes.

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

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on January 21, 2016.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Large Scale Data Center Network Design Summary	3
2.1. Reference design	4
3. Some open problems in large data-center networks	5
4. Applying Segment Routing in the DC with MPLS dataplane	6
4.1. BGP Prefix Segment	6
4.2. eBGP Labeled Unicast (RFC3107)	7
4.2.1. Control Plane	8
4.2.2. Data Plane	9
4.2.3. Network Design Variation	10
4.2.4. Global BGP Prefix Segment through the fabric	10
4.2.5. Incremental Deployments	11
4.3. iBGP Labeled Unicast (RFC3107)	12
5. Applying Segment Routing in the DC with IPv6 dataplane	12
6. Communicating path information to the host	13
7. Addressing the open problems	14
7.1. Per-packet and flowlet switching	14
7.2. Performance-aware routing	15
7.3. Non-oblivious routing	16
7.4. Deterministic network probing	16
8. Additional Benefits	16
8.1. MPLS Dataplane with operational simplicity	16
8.2. Minimizing the FIB table	17
8.3. Egress Peer Engineering	17

8.4. Incremental Deployments	18
8.5. Anycast	18
9. Preferred SRGB Allocation	18
10. Alternative Options	19
11. IANA Considerations	20
12. Manageability Considerations	20
13. Security Considerations	20
14. Contributors	20
15. Acknowledgements	20
16. References	20
16.1. Normative References	20
16.2. Informative References	20
Authors' Addresses	21

1. Introduction

Segment Routing (SR), as described in [I-D.filsfils-spring-segment-routing] 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 local semantic to an SR node or global within an SR domain. SR allows to enforce a flow through any topological path and service chain while maintaining per-flow state only at the ingress node to the SR domain. Segment Routing can be applied to the MPLS and IPv6 data-planes.

The use-case use-cases described in this document should be considered in the context of the BGP-based large-scale data-center (DC) design described in [I-D.ietf-rtgwg-bgp-routing-large-dc]. We extend it by applying SR both with IPv6 and MPLS dataplane.

2. Large Scale Data Center Network Design Summary

This section provides a brief summary of the informational document [I-D.ietf-rtgwg-bgp-routing-large-dc] that outlines a practical network design suitable for data-centers of various scales:

- o Data-center networks have highly symmetric topologies with multiple parallel paths between two server attachment points. The well-known Clos topology is most popular among the operators. In a Clos topology, the minimum number of parallel paths between two elements is determined by the "width" of the middle stage. See Figure 1 below for an illustration of the concept.
- o Large-scale data-centers commonly use a routing protocol, such as BGP4 [RFC4271] in order to provide endpoint connectivity. Recovery after a network failure is therefore driven either by

local knowledge of directly available backup paths or by distributed signaling between the network devices.

- o Within data-center networks, traffic is load-shared using the Equal Cost Multipath (ECMP) mechanism. With ECMP, every network device implements a pseudo-random decision, mapping packets to one of the parallel paths by means of a hash function calculated over certain parts of the packet, typically a combination of various packet header fields.

The following is a schematic of a five-stage Clos topology, with four devices in the middle stage. Notice that number of paths between Node1 and Node12 equals to four: the paths have to cross all of Tier-1 devices. At the same time, the number of paths between Node1 and Node2 equals two, and the paths only cross Tier-2 devices. Other topologies are possible, but for simplicity we'll only look into the topologies that have a single path from Tier-1 to Tier-3. The rest could be treated similarly, with a few modifications to the logic.

2.1. Reference design

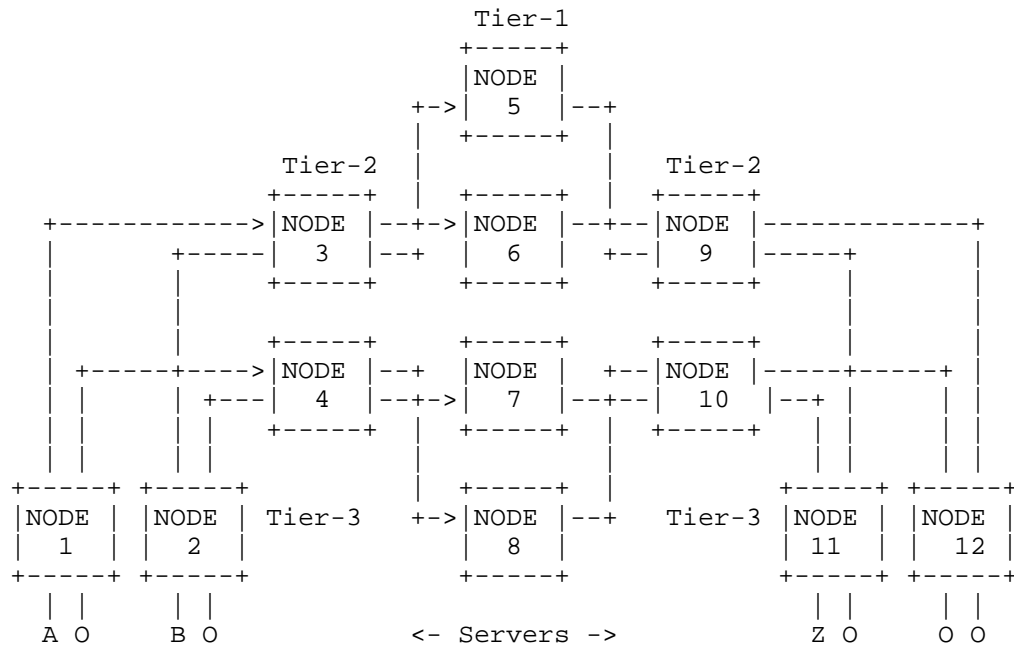


Figure 1: 5-stage Clos topology

In the reference topology illustrated in Figure 1, we assume:

- o Each node is its own AS (Node X has AS X)
 - * For simple and efficient route propagation filtering, Nodes 5, 6, 7 and 8 share the same AS, Nodes 3 and 4 share the same AS, Nodes 9 and 10 share the same AS.
 - * For efficient usage of the scarce 2-byte Private Use AS pool, different Tier-3 nodes might share the same AS.
 - * Without loss of generality, we will simplify these details in this document and assume that each node has its own AS.
- o Each node peers with its neighbors via BGP session
 - * If not specified, eBGP is assumed. In a specific use-case, iBGP will be used but this will be called out explicitly in that case.
- o Each node originates the IPv4 address of it's loopback interface into BGP and announces it to its neighbors.
 - * The loopback of Node X is 192.0.2.x/32.

In this document, we also refer to the Tier-1, Tier-2 and Tier-3 switches respectively as Spine, Leaf and ToR (top of rack) switches. When a ToR switch acts as a gateway to the "outside world", we call it a border switch.

3. Some open problems in large data-center networks

The data-center network design summarized above provides means for moving traffic between hosts with reasonable efficiency. There are few open performance and reliability problems that arise in such design:

- o ECMP routing is most commonly realized per-flow. This means that large, long-lived "elephant" flows may affect performance of smaller, short-lived "mouse" flows and reduce efficiency of per-flow load-sharing. In other words, per-flow ECMP that does not perform efficiently when flow life-time distribution is heavy-tailed. Furthermore, due to hash-function inefficiencies it is possible to have frequent flow collisions, where more flows get placed on one path over the others
- o Shortest-path routing with ECMP implements oblivious routing model, which is not aware of the network imbalances. If the network symmetry is broken, for example due to link failures, utilization hotspots may appear. For example, if a link fails

between Tier-1 and Tier-2 devices (e.g. "Node5" and "Node9"), Tier-3 devices "Node1" and "Node2" will not be aware of that, since there are other paths available from perspective of "Node3". They will continue sending roughly equal traffic to Node3 and Node4 as if the failure didn't exist which may cause a traffic hotspot.

- o Absence of path visibility leaves transport protocols, such as TCP, with a "blackbox" view of the network. Some TCP metrics, such as SRTT, MSS, CWND and few others could be inferred and cached based on past history, but those apply to destinations, regardless of the path that has been chosen to get there. Thus, for instance, TCP is not capable of remembering "bad" paths, such as those that exhibited poor performance in the past. This means that every new connection will be established obliviously (memory-less) with regards to the paths chosen before, or chosen by other nodes.
- o Isolating faults in the network with multiple parallel paths and ECMP-based routing is non-trivial due to lack of determinism. Specifically, the connections from HostA to HostB may take a different path every time a new connection is formed, thus making consistent reproduction of a failure much more difficult. This complexity scales linearly with the number of parallel paths in the network, and stems from the random nature of path selection by the network devices.

Further in this document, we are going to demonstrate how these problems could be addressed within the framework of Segment Routing.

First, we will explain how to apply SR in the DC, for MPLS and IPv6 data-planes.

4. Applying Segment Routing in the DC with MPLS dataplane

4.1. BGP Prefix Segment

A BGP-Prefix Segment is a segment associated with a BGP prefix. A BGP-Prefix Segment is a network-wide instruction to forward the packet along the ECMP-aware best path to the related prefix ([I-D.keyupate-idr-bgp-prefix-sid]).

In this document, we make the network design decision to assume that all the nodes are allocated the same SRGB, e.g. [16000, 23999]. This is important to fulfill the recommendation for operational simplification as explained in [I-D.filsfils-spring-segment-routing].

Note well that the use of a common SRGB in all nodes is not a requirement, one could use a different SRGB at every node. However, this would make the operation of the DC fabric more complex as the label allocated to the loopback of a remote switch is then different at every node. This also may increase the complexity of the centralized controller.

For illustration purpose, when considering an MPLS data-plane, we assume that the segment index allocated to prefix 192.0.2.x/32 is X. As a result, a local label 1600x is allocated for prefix 192.0.2.x/32 by each node throughout the DC fabric.

When IPv6 data-plane is considered, we assume that Node X is allocated IPv6 address (segment) 2001:DB8::X.

4.2. eBGP Labeled Unicast (RFC3107)

Referring to Figure 1 and [[I-D.ietf-rtgwg-bgp-routing-large-dc], the following design modifications are introduced:

- o Each node peers with its neighbors via eBGP3107 session
- o The forwarding plane at Tier-2 and Tier-1 is MPLS.
- o The forwarding plane at Tier-3 is either IP2MPLS (if the host sends IP traffic) or MPLS2MPLS (if the host sends MPLS-encapsulated traffic).

Figure 2 zooms on a path from server A to server Z within the topology of Figure 1.

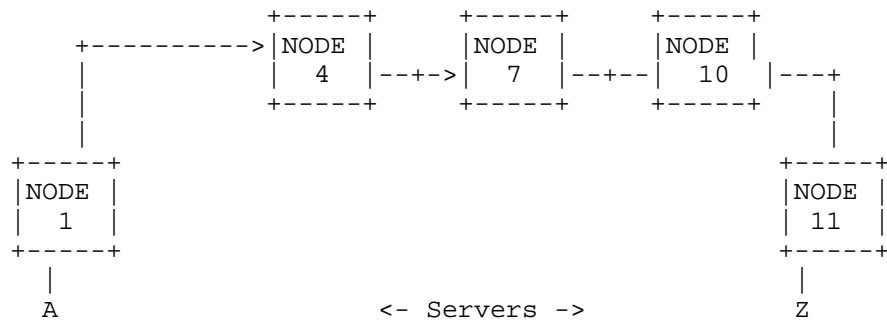


Figure 2: Path from A to Z via nodes 1, 4, 7, 10 and 11

Referring to Figure 1 and Figure 2 and assuming the IP address, AS and index allocation previously described, the following sections

detail the control plane operation and the data plane states for the prefix 192.0.2.11/32 (loopback of Node11)

4.2.1. Control Plane

Node11 originates 192.0.2.11/32 in BGP and allocates to it the BGP-Prefix Segment attribute (index11).

Node11 sends the following eBGP3107 update to Node10:

```
. NLRI: 192.0.2.11/32
. Label: Implicit-Null
. Next-hop: Node11's interface address on the link to Node10
. AS Path: {11}
. BGP-Prefix Attribute: Index 11
```

Node10 receives the above update. As it is SR capable, Node10 is able to interpret the BGP-Prefix Attribute and hence understands that it should allocate the label LOCAL-SRGB (16000) + "index" 11 (hence 16011) to the NLRI instead of allocating an nondeterministic label out of a dynamically allocated portion of the local label space. The implicit-null label in the NLRI tells Node10 that it is the penultimate hop and MUST pop the top label on the stack before forwarding traffic for this prefix to Node11.

Then, Node10 sends the following eBGP3107 update to Node7:

```
. NLRI: 192.0.2.11/32
. Label: 16011
. Next-hop: Node10's interface address on the link to Node7
. AS Path: {10, 11}
. BGP-Prefix Attribute: Index 11
```

Node7 receives the above update. As it is SR capable, Node7 is able to interpret the BGP-Prefix Attribute and hence allocates the local (incoming) label 16011 (16000 + 11) to the NLRI (instead of allocating a "dynamic" local label from its label manager). Node7 uses the label in the received eBGP3107 NLRI as the outgoing label (the index is only used to derive the local/incoming label).

Node7 sends the following eBGP3107 update to Node4:

```
. NLRI: 192.0.2.11/32
. Label: 16011
. Next-hop: Node7's interface address on the link to Node4
. AS Path: {7, 10, 11}
. BGP-Prefix Attribute: Index 11
```


Node4 receives the above update. As it is SR capable, Node4 is able to interpret the BGP-Prefix Attribute and hence allocates the local (incoming) label 16011 to the NLRI (instead of allocating a "dynamic" local label from its label manager). Node4 uses the label in the received eBGP3107 NLRI as outgoing label (the index is only used to derive the local/incoming label).

Node4 sends the following eBGP3107 update to Node1:

```
. NLRI: 192.0.2.11/32
. Label: 16011
. Next-hop: Node4's interface address on the link to Node1
. AS Path: {4, 7, 10, 11}
. BGP-Prefix Attribute: Index 11
```

Node1 receives the above update. As it is SR capable, Node1 is able to interpret the BGP-Prefix Attribute and hence allocates the local (incoming) label 16011 to the NLRI (instead of allocating a "dynamic" local label from its label manager). Node1 uses the label in the received eBGP3107 NLRI as outgoing label (the index is only used to derive the local/incoming label).

4.2.2. Data Plane

Referring to Figure 1 Referring to Figure 1, and assuming all nodes apply the same advertisement rules described above and all nodes have the same SRGB (16000-23999), here are the IP/MPLS forwarding tables for prefix 192.0.2.11/32 at Nodes 1, 4, 7 and 10.

Incoming label or IP destination	outgoing label	Outgoing Interface
16011	16011	ECMP{3, 4}
192.0.2.11/32	16011	ECMP{3, 4}

Figure 3: Node1 Forwarding Table

Incoming label or IP destination	outgoing label	Outgoing Interface
16011	16011	ECMP{7, 8}
192.0.2.11/32	16011	ECMP{7, 8}

Figure 4: Node4 Forwarding Table

Incoming label or IP destination	outgoing label	Outgoing Interface
16011	16011	10
192.0.2.11/32	16011	10

Figure 5: Node7 Forwarding Table

Incoming label or IP destination	outgoing label	Outgoing Interface
16011	POP	11
192.0.2.11/32	N/A	11

Nodel0 Forwarding Table

4.2.3. Network Design Variation

A network design choice could consist of switching all the traffic through Tier-1 and Tier-2 as MPLS traffic. In this case, one could filter away the IP entries at Nodes 4, 7 and 10. This might be beneficial in order to optimize the forwarding table size.

A network design choice could consist in allowing the hosts to send MPLS-encapsulated traffic (based on EPE use-case, [I-D.filsfils-spring-segment-routing-central-epe]). For example, applications at HostA would send their Z-destined traffic to Nodel with an MPLS label stack where the top label is 16011 and the next label is an EPE peer segment at Nodel1 directing the traffic to Z.

4.2.4. Global BGP Prefix Segment through the fabric

When the previous design is deployed, the operator enjoys global BGP prefix segment (label) allocation throughout the DC fabric.

A few examples follow:

- o Normal forwarding to Nodel1: a packet with top label 16011 received by any switch in the fabric will be forwarded along the ECMP-aware BGP best-path towards Nodel1 and the label 16011 is penultimate-popped at Nodel0.
- o Traffic-engineered path to Nodel1: an application on a host behind Nodel might want to restrict its traffic to paths via the Spine

switch Node5. The application achieves this by sending its packets with a label stack of {16005, 16011}. BGP Prefix segment 16005 directs the packet up to Node5 along the path (Node1, Node3, Node5). BGP Prefix Segment 16011 then directs the packet down to Node11 along the path (Node5, Node9, Node11).

4.2.5. Incremental Deployments

The design previously described can be deployed incrementally. Let us assume that Node7 does not support the BGP-Prefix Segment attribute and let us show how the fabric connectivity is preserved.

From a signaling viewpoint, nothing would change: if Node7 does not understand the BGP-Prefix Segment attribute, it does propagate the attribute unmodified to its neighbors.

From a label allocation viewpoint, the only difference is that Node7 would allocate a dynamic (random) label to the prefix 192.0.2.11/32 (e.g. 123456) instead of the "hinted" label as instructed by the BGP Prefix Segment attribute. The neighbors of Node7 adapt automatically as they always use the label in the BGP3107 NLRI as outgoing label.

Node4 does understand the BGP-Prefix Segment attribute and hence allocates the indexed label in the SRGB (16011) for 192.0.2.11/32.

As a result, all the data-plane entries across the network would be unchanged except the entries at Node7 and its neighbor Node4 as shown in the figures below.

The key point is that the end-to-end LSP is preserved because the outgoing label is always derived from the received label within the BGP3107 NLRI. The index in the BGP Prefix SID is only used as a hint on how to allocate the local label (the incoming label) but never for the outgoing label.

Incoming label or IP destination	outgoing label	Outgoing Interface
12345	16011	10

Figure 7: Node7 Forwarding Table

Incoming label or IP destination	outgoing label	Outgoing Interface
16011	12345	7

Figure 8: Node4 Forwarding Table

The BGP-Prefix Segment functionality can thus be deployed incrementally one node at a time.

When deployed together with a homogeneous SRGB (same SRGB across the fabric), the operator incrementally enjoys the global prefix segment benefits as the deployment progresses through the fabric.

4.3. iBGP Labeled Unicast (RFC3107)

The same exact design as eBGP3107 is used with the following modifications:

All switches share the same AS

iBGP3107 reflection with nhop-self is used instead of eBGP3107

For simple and efficient route propagation filtering, Nodes 5, 6, 7 and 8 share the same Cluster ID, Nodes 3 and 4 share the same Cluster ID, Nodes 9 and 10 share the same Cluster ID.

AIGP metric ([RFC7311]) is likely applied to the BGP prefix segments as part of a large-scale multi-domain design such as Seamless MPLS [I-D.ietf-mpls-seamless-mpls].

The control-plane behavior is mostly the same as described in the previous section: the only difference is that the eBGP3107 path propagation is simply replaced by an iBGP3107 path reflection with next-hop changed to self.

The data-plane tables are exactly the same.

5. Applying Segment Routing in the DC with IPv6 dataplane

The design described in I-D.ietf-rtgwg-bgp-routing-large-dc [I-D.ietf-rtgwg-bgp-routing-large-dc] is reused with one single modification. We highlight it using the example of the reachability to Nodell via spine switch Node5.

Spine5 originates 2001:DB8::5/128 with the attached BGP Prefix Attribute advertising the support of the Segment Routing extension

header (SRH, [I-D.previdi-6man-segment-routing-header]) for IPv6 packets destined to segment 2001:DB8::5.

Tor11 originates 2001:DB8::11/128 with the attached BGP Prefix Attribute advertising the support of the Segment Routing extension header (SRH, [I-D.previdi-6man-segment-routing-header]) for IPv6 packets destined to segment 2001:DB8::11.

The control-plane and data-plane processing of all the other nodes in the fabric is unchanged. Specifically, the routes to 2001:DB8::5 and 2001:DB8::11 are installed in the FIB along the eBGP best-path to Node5 (spine node) and Node11 (ToR node) respectively.

An application on HostA which needs to send traffic to HostZ via only Node5 (spine node) can do so by sending IPv6 packets with a SR extension header. The destination address and active segment is set to 2001:DB8::5. The next and last segment is set to 2001:DB8::11.

The application must only use IPv6 addresses that have been advertised as capable for SRv6 segment processing (e.g. for which the BGP prefix segment capability has been advertised). How applications learn this (e.g.: centralized controller and orchestration) is outside the scope of this document.

6. Communicating path information to the host

There are two general methods for communicating path information to the end-hosts: "proactive" and "reactive", aka "push" and "pull" models. There are multiple ways to implement either of these methods. Here, we note that one way could be using a centralized controller: the controller either tells the hosts of the prefix-to-path mappings beforehand and updates them as needed (network event driven push), or responds to the hosts making request for a path to specific destination (host event driven pull). It is also possible to use a hybrid model, i.e., pushing some state from the controller in response to particular network events, while the host pulls other state on demand.

We note, that when disseminating network-related data to the end-hosts a trade-off is made to balance the amount of information vs the level of visibility in the network state. This applies both to push and pull models. In the extreme case, the host would request path information on every flow, and keep no local state at all. On the other end of the spectrum, information for every prefix in the network along with available paths could be pushed and continuously updated on all hosts.

7. Addressing the open problems

This section demonstrates how the problems describe above could be solved using the segment routing concept. It is worth noting that segment routing signaling and data-plane are only parts of the solution. Additional enhancements, e.g. such as the centralized controller mentioned previously, and host networking stack support are required to implement the proposed solutions.

7.1. Per-packet and flowlet switching

With the ability to choose paths on the host, one may go from per-flow load-sharing in the network to per-packet or per-flowlet (see [KANDULA04] for information on flowlets). The host may select different segment routing instructions either per packet, or per flowlet, and route them over different paths. This allows for solving the "elephant flow" problem in the data-center and avoiding link imbalances.

Note that traditional ECMP routing could be easily simulated with on-host path selection, using method proposed in VL2 (see [GREENBERG09]). The hosts would randomly pick a Tier-2 or Tier-1 device to "bounce" the packet off of, depending on whether the destination is under the same Tier-2 switches, or has to be reached across Tier-1. The host would use a hash function that operates on per-flow invariants, to simulate per-flow load-sharing in the network.

Using Figure 1 as reference, let's illustrate this assuming that HostA has an elephant flow to Z called Flow-f.

Normally, a flow is hashed on to a single path. Let's assume HostA sends its packets associated with Flow-f with top label 16011 (the label for the remote ToR, Node11, where HostZ is connected) and Node1 would hash all the packets of Flow-F via the same nhop (e.g. Node3). Similarly, let's assume that leaf Node3 would hash all the packets of Flow-F via the same next-hop (e.g.: spine switch Node1). This normal operation would restrict the elephant flow on a small subset of the ECMP paths to HostZ and potentially create imbalance and congestion in the fabric.

Leveraging the flowlet proposal, assuming A is made aware of 4 disjoint paths via intermediate segment 16005, 16006, 16007 and 16008 (the BGP prefix SID's of the 4 spine switches) and also made aware of the prefix segment of the remote ToR connected to the destination (16011), then the application can break the elephant flow F into flowlets F1, F2, F3, F4 and associate each flowlet with one of the following 4 label stacks: {16005, 16011}, {16006, 16011}, {16007,

16011} and {16008, 16011}. This would spread the load of the elephant flow through all the ECMP paths available in the fabric and rebalance the load.

7.2. Performance-aware routing

Knowing the path associated with flows/packets, the end host may deduce certain characteristics of the path on its own, and additionally use the information supplied with path information pushed from the controller or received via pull request. The host may further share its path observations with the centralized agent, so that the latter may keep up-to-date network health map to assist other hosts with this information.

For example, an application A.1 at HostA may pin a TCP flow destined to HostZ via Spine switch Node5 using label stack {16005, 16011}. The application A.1 may collect information on packet loss, deduced from TCP retransmissions and other signals (e.g. RTT increases). A.1 may additionally publish this information to a centralized agent, e.g. after a flow completes, or periodically for longer lived flows. Next, using both local and/or global performance data, application A.1 as well as other applications sharing the same resources in the DC fabric may pick up the best path for the new flow, or update an existing path (e.g.: when informed of congestion on an existing path).

One particularly interesting instance of performance-aware routing is dynamic fault-avoidance. If some links or devices in the network start discarding packets due to a fault, the end-hosts could detect the path(s) being affected and steer their flows away from the problem spot. Similar logic applies to failure cases where packets get completely black-holed, e.g. when a link goes down.

For example, an application A.1 informed about 5 paths to Z {16005, 16011}, {16006, 16011}, {16007, 16011}, {16008, 16011} and {16011} might use the latter one by default (for simplicity). When performance is degrading, A.1 might then start to pin TCP flows to each of the 4 other paths (each via a distinct spine) and monitor the performance. It would then detect the faulty path and assign a negative preference to the faulty path to avoid further flows using it. Gradually, over time, it may re-assign flows on the faulty path to eventually detect the resolution of the trouble and start reusing the path.

7.3. Non-oblivious routing

By leveraging Segment Routing, one avoids issues associated with oblivious ECMP hashing. For example, if in the topology depicted on Figure 1 a link between spine switch Node5 and leaf node Node9 fails, HostA may exclude the segment corresponding to Node5 from the prefix matching the servers under Tier-2 devices Node9. In the push path discovery model, the affected path mappings may be explicitly pushed to all the servers for the duration of the failure. The new mapping would instruct them to avoid the particular Tier-1 switch until the link has recovered. Alternatively, in pull path, the centralized controller may start steering new flows immediately after it discovers the issue. Until then, the existing flows may recover using local detection of the path issues, as described in Section 7.2.

7.4. Deterministic network probing

Active probing is a well-known technique for monitoring network elements health, constituting of sending continuous packet streams simulating network traffic to the hosts in the data-center. Segment routing makes possible to prescribe the exact paths that each probe or series of probes would be taking toward their destination. This allows for fast correlation and detection of failed paths, by processing information from multiple actively probing agents. This complements the data collected from the hosts routing stacks as described in Section 7.2.

For example, imagine a probe agent sending packets to all machines in the data-center. For every host, it may send packets over each of the possible paths, knowing exactly which links and devices these packets will be crossing. Correlating results for multiple destinations with the topological data, it may automatically isolate possible problem to a link or device in the network.

8. Additional Benefits

8.1. MPLS Dataplane with operational simplicity

As required by [I-D.ietf-rtgwg-bgp-routing-large-dc], no new signaling protocol is introduced. The Prefix Segment is a lightweight extension to BGP Labelled Unicast (RFC3107 [RFC3107]). It applies either to eBGP or iBGP based designs.

Specifically, LDP and RSVP-TE are not used. These protocols would drastically impact the operational complexity of the Data Center and would not scale. This is in line with the requirements expressed in [I-D.ietf-rtgwg-bgp-routing-large-dc]

A key element of the operational simplicity is the deployment of the design with a single and consistent SRGB across the DC fabric.

At every node in the fabric, the same label is associated to a given BGP prefix segment and hence a notion of global prefix segment arises.

When a controller programs HostA to send traffic to HostZ via the normally available BGP ECMP paths, the controller uses label 16011 associated with the ToR switch connected to the HostZ. The controller does not need to pick the label based on the ToR that the source host is connected to.

In a classic BGP Labelled Unicast design applied to the DC fabric illustrated in Figure 1, the ToR Node1 connected to HostA would most likely allocate a different label for 192.0.2.11/32 than the one allocated by ToR Node2. As a consequence, the controller would need to adapt the SR policy to each host, based on the ToR switch that they are connected to. This adds state maintenance and synchronization problems. All of this unnecessary complexity is eliminated if a single consistent SRGB is utilized across the fabric.

8.2. Minimizing the FIB table

The designer may decide to switch all the traffic at Tier-1 and Tier-2's based on MPLS, hence drastically decreasing the IP table size at these nodes.

This is easily accomplished by encapsulating the traffic either directly at the host or at the source ToR switch by pushing the BGP-Prefix Segment of the destination ToR for intra-DC traffic or border switch for inter-DC or DC-to-outside-world traffic.

8.3. Egress Peer Engineering

It is straightforward to combine the design illustrated in this document with the Egress Peer Engineering (EPE) use-case described in [I-D.filsfils-spring-segment-routing-central-epe].

In such case, the operator is able to engineer its outbound traffic on a per host-flow basis, without incurring any additional state at intermediate points in the DC fabric.

For example, the controller only needs to inject a per-flow state on the HostA to force it to send its traffic destined to a specific Internet destination D via a selected border switch (say Node12 in Figure 1 instead of another border switch Node11) and a specific egress peer of Node12 (say peer AS 9999 of local PeerNode segment

9999 at Node12 instead of any other peer which provides a path to the destination D). Any packet matching this state at host A would be encapsulated with SR segment list (label stack) {16012, 9999}. 16012 would steer the flow through the DC fabric, leveraging any ECMP, along the best path to border switch Node12. Once the flow gets to border switch Node12, the active segment is 9999 (thanks to PHP on the upstream neighbor of Node12). This EPE PeerNode segment forces border switch Node12 to forward the packet to peer AS 9999, without any IP lookup at the border switch. There is no per-flow state for this engineered flow in the DC fabric. A benefit of segment routing is the per-flow state is only required at the source.

As well as allowing full traffic engineering control such a design also offers FIB table minimization benefits as the Internet- scale FIB at border switch Node12 is not required if all FIB lookups are avoided there by using EPE.

8.4. Incremental Deployments

As explained in Section 4.2.5, this design can be deployed incrementally.

8.5. Anycast

The design presented in this document preserves the availability and load-balancing properties of the base design presented in [I-D.filsfils-spring-segment-routing].

For example, one could assign an anycast loopback 192.0.2.20/32 and associate segment index 20 to it on the border switches 11 and 12 (in addition to their node-specific loopbacks). Doing so, the EPE controller could express a default "go-to-the- Internet via any border switch" policy as segment list {16020}. Indeed, from any host in the DC fabric or from any ToR switch, 16020 steers the packet towards the border switches 11 or 12 leveraging ECMP where available along the best paths to these switches.

9. Preferred SRGB Allocation

In the MPLS case, we do not recommend to use different SRGBs at each node.

Different SRGBs in each node likely increase the complexity of the solution both from an operation viewpoint and from a controller viewpoint.

From an operation viewpoint, it is much simpler to have the same global label at every node for the same destination (the MPLS

troubleshooting is then similar to the IPv6 troubleshooting where this global property is a given).

From a controller viewpoint, this allows to construct simple policies applicable across the fabric.

Let us consider two applications A and B respectively connected to ToR1 and ToR2. A has two flows FA1 and FA2 destined to Z. B has two flows FB1 and FB2 destined to Z. The controller wants FA1 and FB1 to be load-shared across the fabric while FA2 and FB2 must be respectively steered via Spine5 and spine 8.

Assuming a consistent unique SRGB across the fabric as described in the document, the controller can simply do it by instructing A and B to use {16011} respectively for FA1 and FB1 and by instructing A and B to use {16005 16011} and {16008 16011} respectively for FA2 and FB2.

Let us assume a design where the SRGB is different at every node: SRGB of Node K starts at value $K*1000$ and the SRGB length is 1000 (e.g. ToR1's SRGB is [1000, 1999], ToR2's SRGB is [2000, 2999]...).

In this case, not only the controller would need to collect and store all of these different SRGB's, furthermore it would need to adapt the policy for each host. Indeed, the controller would instruct A to use {1011} for FA1 while it would have to instruct B to use {2011} for FB1 (while with the same SRGB, both policies are the same {16011}).

Even worse, the controller would instruct A to use {1005, 5011} for FA1 while it would instruct B to use {2011, 8011} for FB1 (while with the same SRGB, the second segment is the same across both policies: 16011). When combining segments to create a policy, one need to carefully update the label of each segment. This is obviously more error-prone, more complex and more difficult to troubleshoot.

10. Alternative Options

In order to support all the requirements and get consensus, the BGP Prefix SID attribute has been extended to allow this design.

Specifically, the ORIGINATOR_SRGB TLV in the BGP Prefix SID signals the SRGB of the switch that originated the BGP Prefix Segment.

This allows to determine the local label allocated by any switch for any BGP Prefix Segment, despite the lack of a consistent unique SRGB in the domain.

11. IANA Considerations

TBD

12. Manageability Considerations

TBD

13. Security Considerations

TBD

14. Contributors

Benjamin Black, Arjun Arjun Sreekantiah and Keyur Patel have contributed to the content of this document.

15. Acknowledgements

The authors would like to thank Acee Lindem for his review.

16. References

16.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC3107] Rekhter, Y. and E. Rosen, "Carrying Label Information in BGP-4", RFC 3107, May 2001.
- [RFC4271] Rekhter, Y., Li, T., and S. Hares, "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, January 2006.
- [RFC7311] Mohapatra, P., Fernando, R., Rosen, E., and J. Uttaro, "The Accumulated IGP Metric Attribute for BGP", RFC 7311, August 2014.

16.2. Informative References

- [GREENBERG09] Greenberg, A., Hamilton, J., Jain, N., Kadula, S., Kim, C., Lahiri, P., Maltz, D., Patel, P., and S. Sengupta, "VL2: A Scalable and Flexible Data Center Network", 2009.

[I-D.filsfils-spring-segment-routing]

Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., Litkowski, S., Horneffer, M., Milojevic, I., Shakir, R., Ytti, S., Henderickx, W., Tantsura, J., and E. Crabbe, "Segment Routing Architecture", draft-filsfils-spring-segment-routing-04 (work in progress), July 2014.

[I-D.filsfils-spring-segment-routing-central-epe]

Filsfils, C., Previdi, S., Patel, K., Aries, E., shaw@fb.com, s., Ginsburg, D., and D. Afanasiev, "Segment Routing Centralized Egress Peer Engineering", draft-filsfils-spring-segment-routing-central-epe-04 (work in progress), July 2015.

[I-D.ietf-mpls-seamless-mpls]

Leymann, N., Decraene, B., Filsfils, C., Konstantynowicz, M., and D. Steinberg, "Seamless MPLS Architecture", draft-ietf-mpls-seamless-mpls-07 (work in progress), June 2014.

[I-D.ietf-rtgwg-bgp-routing-large-dc]

Lapukhov, P., Premji, A., and J. Mitchell, "Use of BGP for routing in large-scale data centers", draft-ietf-rtgwg-bgp-routing-large-dc-03 (work in progress), June 2015.

[I-D.keyupate-idr-bgp-prefix-sid]

Patel, K., Previdi, S., Filsfils, C., Sreekantiah, A., Ray, S., and H. Gredler, "Segment Routing Prefix SID extensions for BGP", draft-keyupate-idr-bgp-prefix-sid-04 (work in progress), July 2015.

[I-D.previdi-6man-segment-routing-header]

Previdi, S., Filsfils, C., Field, B., and I. Leung, "IPv6 Segment Routing Header (SRH)", draft-previdi-6man-segment-routing-header-06 (work in progress), May 2015.

[KANDULA04]

Sinha, S., Kandula, S., and D. Katabi, "Harnessing TCP's Burstiness with Flowlet Switching", 2004.

Authors' Addresses

Clarence Filsfils (editor)
Cisco Systems, Inc.
Brussels
BE

Email: cfilsfil@cisco.com

Stefano Previdi (editor)
Cisco Systems, Inc.
Via Del Serafico, 200
Rome 00142
Italy

Email: sprevidi@cisco.com

Jon Mitchell
Unaffiliated

Email: jrmitch@puck.nether.net

Ebben Aries
Facebook
US

Email: exa@fb.com

P. Lapukhov
Facebook
US

Email: petr@fb.com

G. Nagarajan
Facebook
US

Email: gaya@fb.com

Dmitry Afanasiev
Yandex
RU

Email: fl0w@yandex-team.ru

Tim Laberge
Microsoft

Email: tim.laberge@microsoft.com

Edet Nkposong
Microsoft

Email: edetn@microsoft.com

Mohan Nanduri
Microsoft

Email: mnanduri@microsoft.com

James Uttaro
ATT

Email: jul738@att.com

Saikat Ray
Unaffiliated

Email: raysaikat@gmail.com

SPRING WG
Internet-Draft
Intended status: Informational
Expires: September 10, 2015

F. Hu
ZTE Corporation
B. Khasnabish
ZTE TX Inc.
H. Cankaya
Fujitsu
March 9, 2015

SPRING OpenFlow Interworking Requirements
draft-khc-spring-openflow-interworking-req-01.txt

Abstract

This draft reviews the use cases and lists the requirements for interworking (IW) between OpenFlow (OF) and Segment routing (SR). Although the details and specifics of IW depend on both the architecture and framework, there are some common requirements. We specify those common requirements and show a simple architecture framework.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on September 10, 2015.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Conventions and Abbreviations	3
3. Data Center (DC) Inter-connection	3
3.1. List of Req. for DCI	4
4. OpenFlow Based WAN Control	5
4.1. List of Req. for OF-based SDN WAN	6
5. Interworking of OpenFlow and BGP Edge Routers	6
5.1. List of Req. for BGP-Edge and OF IW	7
5.2. Message Formats	8
5.2.1. Controller-to-Switch Messages	8
5.2.1.1. Asynchronous and Symmetric Messages	8
5.3. OF Controller and Route Reflector Messages	9
6. An SR OF IW Architecture Framework	9
6.1. Common SR OF IW Requirements	10
7. Security Considerations	10
8. Acknowledgements	10
9. IANA Considerations	10
10. Normative References	10
Authors' Addresses	11

1. Introduction

Segment Routing (SR) leverages the source routing paradigm. An ingress node steers a packet through a controlled set of instructions, called segments, by pre-pending the packet with an SR header. A segment can represent any instruction, topological or service-based. A segment can have a local semantic to an SR node or global within an SR domain. Segment Routing allows one to enforce a flow through any topological path and service chaining while maintaining per-flow state only at the ingress node to the Segment Routing domain

The Segment Routing architecture is described in ([I-D.filsfils-rtgwg-segment-routing]) The Segment Routing control plane is agnostic to the data plane, and hence it can be applied to both MPLS (and its many variants) and IPv6.

OpenFlow is a communications protocol and open interface defined between the control and forwarding layers([OpenFlow]). It allows direct access to and manipulation of the forwarding plane of network

devices such as switches and routers, both physical and virtual (hypervisor-based).

This document introduces several scenarios and discusses the interworking between segment routing and openflow.

2. Conventions and Abbreviations

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].

3. Data Center (DC) Inter-connection

Modern day DCs are increasingly utilizing Openflow protocol now, while the WAN network keeps maintaining the traditional IP/MPLS network. Therefore, there exists a need for interworking between OpenFlow and the traditional IP/MPLS network. This scenario discusses how to control the inter-DC flow based on the SDN architecture and segment routing technology.

As shown in Figure 1, the data center is controlled by one or several OF controllers, and all the switches and routers support flow forwarding. The switches and routers communicate with OF controller by OpenFlow protocol. The router is the layer 3 gateway for that data center. The IP/MPLS network between the data centers support segment routing technology. The segment routing label stacks are encapsulated in the edge routers of the data center.

The security app and Traffic Engineering app are the application layer. They communicate with controller through north bound interface. If there are some policies, such as TE App, or security App and policy App for the inter-flow forwarding, the Apps download the service decision to the controller to result into forwarding instance, and the forwarding instances are downloaded to the gateway router of the data center to or from the forwarding label stack.

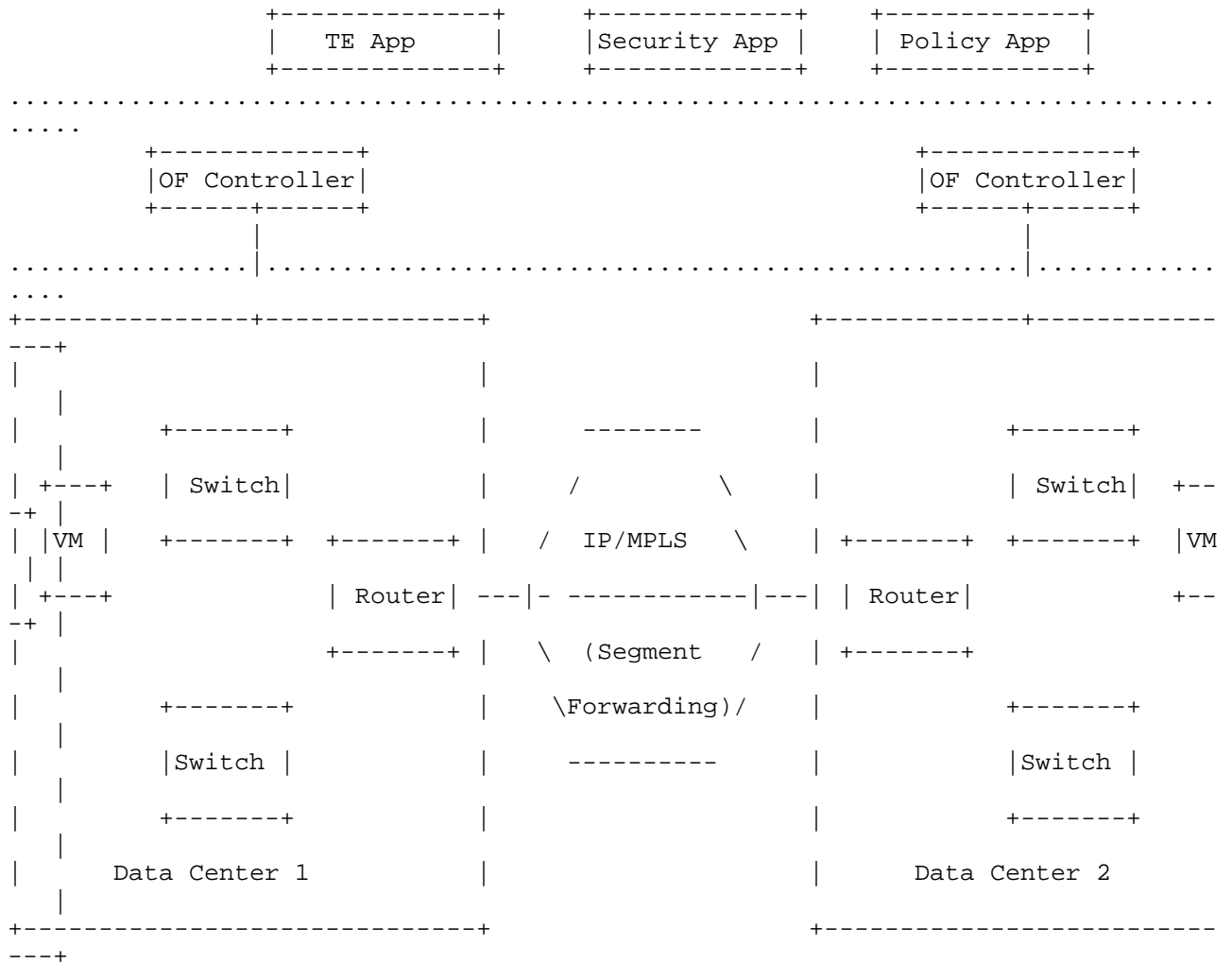


Figure 1: Data Center Inter-connection (DCI)

3.1. List of Req. for DCI

- o OF-based DCI-Req-1: All switches and routers in DCs should support flow forwarding.
- o OF-based DCI-Req-2: All switches and routers in DCs should support OpenFlow protocol.
- o OF-based DCI-Req-3: Routers in DCs should support IP/MPLS and SR technologies.
- o OF-based DCI-Req-4: Interworking function should be transparent at the edge routers of DCs that facilitate the label stacks and SIDs.
- o OF-based DCI-Req-5: The Interworking function should not introduce a considerable delay to the inter DC communication.
- o OF-based DCI-Req-6: The ECMP-based path placement and explicit path mechanisms should be supported.
- o OF-based DCI-Req-7: State information is only maintained at head-end. Tail-end and intermediate nodes do not maintain state info. Policy changes should happen at the head-end.

- o OF-based DCI-Req-8: The interworking function should support OAM functions that is needed to manage SPRING and OF.

4. OpenFlow Based WAN Control

Figure 2 shows an SDN based WAN control scenario. The controller is introduced to the WAN network. The controller should support an externally visible Discovery Service and a Routing Service. The Discovery Service is responsible for bootstrapping and configuring the network, discovering node-capabilities, discovering and maintaining the topology graph, providing statistics and troubleshooting services and finally implementing an API for the Routing Service as well as external requests. The Routing Service is responsible for default routing on the configured network using Segment Routing principles like Node Segments and ECMP. It should also support capabilities allowing for Policy Routing, Traffic Engineering and Steering.

The transit routers (Route 2 and Router 3) are only responsible for MPLS label forwarding. The SR forwarding tables could be built by the controller or the IGP protocols
 ([I-D.ietf-isis-segment-routing-extensions])
 ([I-D.ietf-ospf-segment-routing-extensions]).

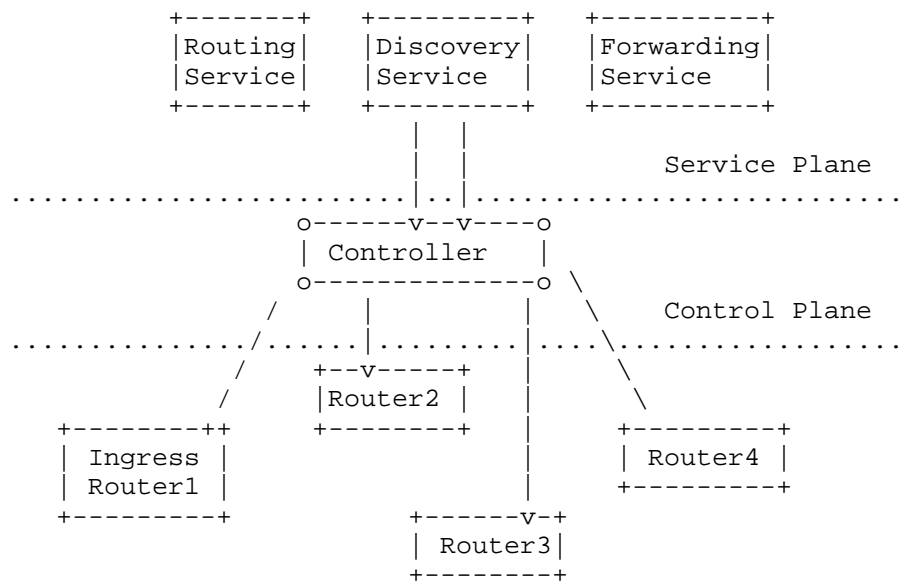


Figure 2: SDN-Based WAN Control

4.1. List of Req. for OF-based SDN WAN

- o OF-based-SDN-WAN-Req-1: Controller may be multiple in WAN.
- o OF-based-SDN-WAN-Req-2: The controller should support Discovery Service, Routing Service, and Forwarding Services.
- o OF-based-SDN-WAN-Req-3: The Discovery Service should support bootstrapping and configuring the network, discovering node-capabilities and topology, statistics and troubleshooting services, and an API for the Routing Service.
- o OF-based-SDN-WAN-Req-4: The Routing Service should support default routing on the configured network using Segment Routing principles like Node Segments and ECMP.
- o OF-based-SDN-WAN-Req-5: The Routing Service should also support Policy Routing, Traffic Engineering and Steering.
- o OF-based-SDN-WAN-Req-6: Forwarding tables should be maintained by the controller, but could be built by IGP protocol and the Routing Service.
- o OF-based-SDN-WAN-Req-7: Routing Service and Discovery Service should support high availability.
- o OF-based-SDN-WAN-Req-8: The ECMP-based path placement and explicit path mechanisms for WAN should be supported.

5. Interworking of OpenFlow and BGP Edge Routers

There are four PEs, two of them (PE1 and PE3) support Openflow protocol, and the other two (PE2 and PE4) support traditional BGP protocol as shown in Figure 3. The routes on the PE2 and PE4 are reflected to PE1 and PE3 through the BGP route reflector. For unified control and interoperability the OF controller needs to interpret the route control messages from the BGP route reflector/controller, and vice versa. The details of the interface and the messages that need to be exchanged between OF Controller and BGP Route Reflector/Controller need to be determined (future work).

We introduce an OF controller in the network and an application layer server. The Apps in the Server can have visibility to the routes from BGP RR as the figure shows. This makes it feasible to export the route to OF controller. The OF controller make its forwarding decision based on the route exported from application and the application policy. The forwarding decision is made of label stack

and downloaded to PE2 and PE4. The PE2 and PE3 are responsible for the segment routing encapsulation as ingress routers.

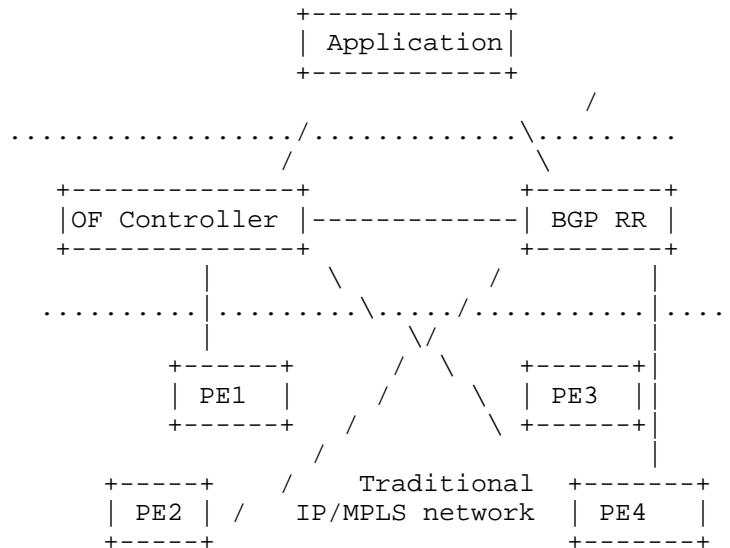


Figure 3: Interworking of OpenFlow and BGP Edge Routers

5.1. List of Req. for BGP-Edge and OF IW

- o BGP-Edge-OF-IW-Req-1: The OpenFlow controller should interpret the route control messages from the BGP route reflector.
- o BGP-Edge-OF-IW-Req-2: A routing application on the Application layer should coordinate routes between OpenFlow controller and BGP Route Reflector.
- o BGP-Edge-OF-IW-Req-3: Segment routing encapsulation and label stacking should take place in ingress routers.
- o BGP-Edge-OF-IW-Req-4: Forwarding decisions should be transparent to the traditional IP/MPLS network.
- o BGP-Edge-OF-IW-Req-5: The interworking between the OF Controller and BGP RR should not introduce considerable delay to the route convergence caused by routing changes.
- o BGP-Edge-OF-IW-Req-6: The proposed IW function solution should be scalable as network grows.

- o BGP-Edge-OF-IW-Req-7: All BGP extensions should be supported.
- o BGP-Edge-OF-IW-Req-8: Latest OF version should be supported.

5.2. Message Formats

There are two messaging systems; (1) between OF Controller and Switches/Routers, (2) between Of Controller and Route Reflector.

5.2.1. Controller-to-Switch Messages

These messages are originated by the controller and sent to the switches. Features request message: Controller sends this message to request the identification and capabilities of a certain switch. After the receipt, the switch responds to this message by reporting its identification and capabilities. Configuration message: Controller sets and queries the switch parameters with this message. The switch responds with its parameters, if the message was a query. Modify-State: The controller uses this message to set and manage the states of the switches. Usually it is used to add/delete/modify flow entities in the OF tables and set the switch port parameters. Read-State message: This message is used by the controller to request the current configuration and stats from the switch. Packet-out message: This message is used by the controller to specify a port for the packet to leave the switch. Barrier message: These messages used by the controller to accomplish message dependencies between switches. Role-request message: This message is used by the controller to either set or query the role of the OF channel to a specific switch. Asynch-config message: Using this message, a controller can set an additional filter that it wants to receive over the OF channel.

5.2.1.1. Asynchronous and Symmetric Messages

A switch can initiate a message to the controller without being solicited by a request message from the controller. These messages will include Packet-in message: this message transfers the control of a packet to the controller. Any event like packet loss or flow table miss is sent to the controller by the switch. Flow removed message: Switch informs the controller about the removal of the flow table entry by using this message. Port-status message: Switch informs the controller about a change on the port status. This change could be caused by a user who set the port down or a link failure. Error message: Switch let controller know about any error by using this message.

Symmetric Messages: These messages can be sent in both direction (switch to controller and controller to switch) without any requests. The messages include: Hollo message: hello messages are exchanged

between controller and switch at the connection startup. Echo message: These messages are exchanged in request/reply fashion to check if the connection between the controller and the switch is up.

5.3. OF Controller and Route Reflector Messages

OF Controller and Route Reflector (RR) belonging to the same cluster run IBGP between them to exchange route updates. OF Controller becomes a client and receives all routes from the Route Reflector [RFC4456]. Message formats (OPEN, UPDATE, and KEEPALIVE messages), message handling and Finite State Machines (FSM) mimic [RFC4271].

6. An SR OF IW Architecture Framework

In this section we discuss a simple SR OF IW Architecture Framework.

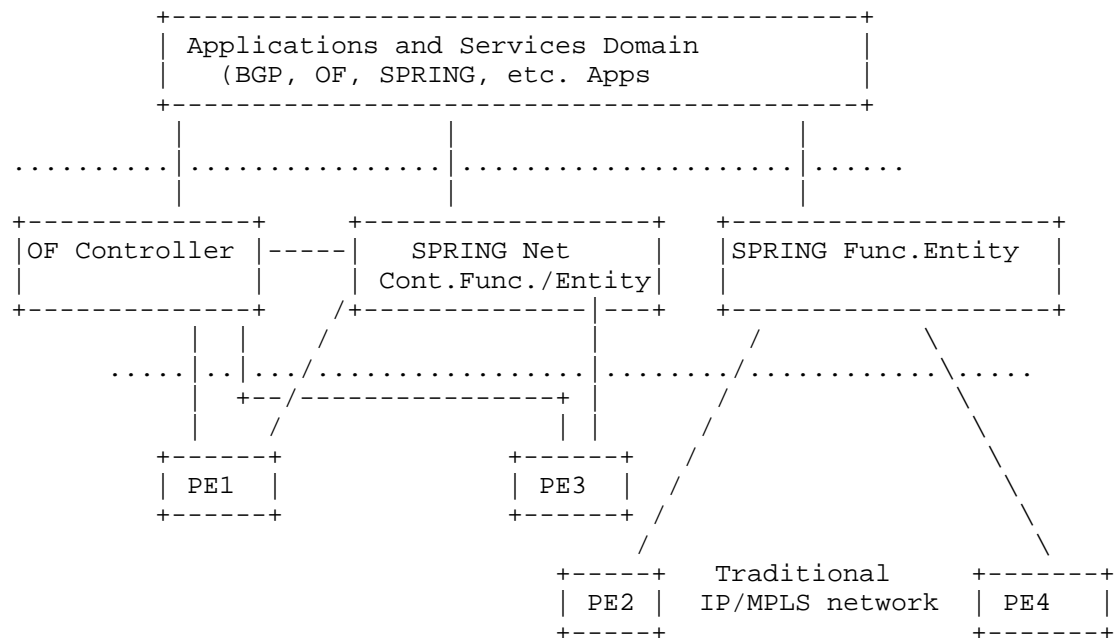


Figure 4: SR OF IW Architecture

6.1. Common SR OF IW Requirements

In this section we list the common SR OF interworking requirements.

- o Common SR-OF-IW-Req-1: In case of multiple OpenFlow controllers, consistency between controllers should be supported [OpenFlow].
- o Common SR-OF-IW-Req-2: Security threats should be addressed by the architecture and the Interworking Function.
- o Common SR-OF-IW-Req-3: The Interworking function should support OAM functions.

7. Security Considerations

TBD.

8. Acknowledgements

In progress.

9. IANA Considerations

IANA request for this document, if any, will be discussed in a future version of this draft.

10. Normative References

[I-D.filsfils-rtgwg-segment-routing]

Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., Litkowski, S., Horneffer, M., Milojevic, I., Shakir, R., Ytti, S., Henderickx, W., Tantsura, J., and E. Crabbe, "Segment Routing Architecture", draft-filsfils-rtgwg-segment-routing-01 (work in progress), October 2013.

[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-03 (work in progress), October 2014.

[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-04 (work in progress), February 2015.

[OpenFlow]

"OpenFlow Switch Specification, Version 1.3.4", March 2014.

[RFC2119]

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

Authors' Addresses

Fangwei Hu
ZTE Corporation
No.889 Bibo Rd
Shanghai 201203
China

Phone: +86 21 68897637
Email: hu.fangwei@zte.com.cn

Bhumip Khasnabish
ZTE TX Inc.
55 Madison Avenue, Suite 160
Morristown, New Jersey 07960
USA

Phone: +001-781-752-8003
Email: vumipl@gmail.com, bhumip.khasnabish@ztetx.com
URI: <http://tinyurl.com/bhumip/>

Hakki Cankaya
Fujitsu
2800 Telecom Parkway,
Richardson, TX 75080
USA

Email: Hakki.Cankaya@us.fujitsu.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: September 4, 2015

S. Kini, Ed.
Ericsson
K. Kompella
Juniper
S. Sivabalan
Cisco
S. Litkowski
Orange
R. Shakir
B.T.
X. Xu
Huawei
W. Hendrickx
Alcatel-Lucent
J. Tantsura
Ericsson
March 3, 2015

Entropy labels for source routed stacked tunnels
draft-kini-mpls-spring-entropy-label-03

Abstract

Source routed tunnel stacking is a technique that can be leveraged to provide a method to steer a packet through a controlled set of segments. This can be applied to the Multi Protocol Label Switching (MPLS) data plane. Entropy label (EL) is a technique used in MPLS to improve load balancing. This document examines and describes how ELs are to be applied to source routed stacked tunnels.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on September 4, 2015.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
1.1. Requirements Language	3
2. Abbreviations and Terminology	3
3. Use-case requiring multipath load balancing in source stacked tunnels	4
4. Recommended EL solution for SPRING	5
5. Options considered	6
5.1. Single EL at the bottom of the stack of tunnels	6
5.2. An EL per tunnel in the stack	7
5.3. A re-usable EL for a stack of tunnels	7
5.3.1. EL at top of stack	8
5.4. ELs at readable label stack depths	8
6. Acknowledgements	9
7. IANA Considerations	9
8. Security Considerations	9
9. References	9
9.1. Normative References	9
9.2. Informative References	10
Authors' Addresses	11

1. Introduction

The source routed stacked tunnels paradigm is leveraged by techniques such as Segment Routing (SR) [I-D.filsfils-spring-segment-routing] to steer a packet through a set of segments. This can be directly applied to the MPLS data plane, but it has implications on label stack depth.

Clarifying statements on label stack depth have been provided in [RFC7325] but they do not address the case of source routed stacked MPLS tunnels as described in [I-D.gredler-spring-mpls] or

[I-D.filsfils-spring-segment-routing] where deeper label stacks are more prevalent.

Entropy label (EL) [RFC6790] is a technique used in the MPLS data plane to provide entropy for load balancing. When using LSP hierarchies there are implications on how [RFC6790] should be applied. One such issue is addressed by [I-D.ravisingh-mpls-el-for-seamless-mpls] but that is when different levels of the hierarchy are created at different LSRs. The current document addresses the case where the hierarchy is created at a single LSR as required by source stacked tunnels.

A use-case requiring load balancing with source stacked tunnels is given in Section 3. A recommended solution is described in Section 4 keeping in consideration the limitations of implementations when applying [RFC6790] to deeper label stacks. Options that were considered to arrive at the recommended solution are documented for historical purposes in Section 5.

1.1. 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].

Although this document is not a protocol specification, the use of this language clarifies the instructions to protocol designers producing solutions that satisfy the requirements set out in this document.

2. Abbreviations and Terminology

EL - Entropy Label

ELI - Entropy Label Identifier

ELC - Entropy Label Capability

SR - Segment Routing

ECMP - Equal Cost Multi Paths

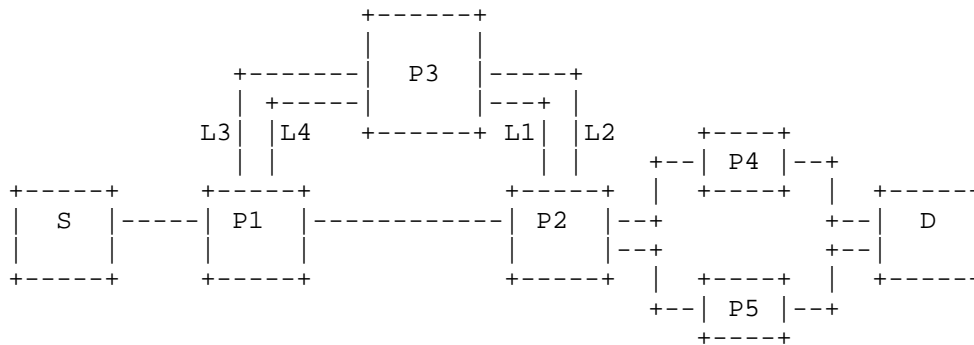
MPLS - Multiprotocol Label Switching

SID - Segment Identifier

RLD - Readable Label Depth

OAM - Operation, Administration and Maintenance

3. Use-case requiring multipath load balancing in source stacked tunnels



S=Source LSR, D=Destination LSR, P1,P2,P3,P4,P5=Transit LSRs,
L1,L2,L3,L4=Links

Figure 1: Traffic engineering use-case

Traffic-engineering (TE) is one of the applications of MPLS and is also a requirement for source stacked tunnels. Consider the topology shown in Figure 1. Lets say the LSR P1 has a limitation that it can only look four labels deep in the stack to do multipath decisions. All other transit LSRs in the figure can read deep label stacks and the LSR S can insert as many <ELI, EL> pairs as needed. The LSR S requires data to be sent to LSR D along a traffic-engineered path that goes over the link L1. Good load balancing is also required across equal cost paths (including parallel links). To engineer traffic along a path that takes link L1, the label stack that LSR S creates consists of a label to the node SID of LSR P3, stacked over the label for the adjacency SID of link L1 and that in turn is stacked over the label to the node SID of LSR D. For simplicity lets assume that all LSRs use the same label space for source stacked tunnels. Lets L_N-P denote the label to be used to reach the node SID of LSR P. Let L_A-Ln denote the label used for the adjacency SID for link Ln . The LSR S must use the label stack $\langle L_N-P3, L_A-L1, L_N-D \rangle$ for traffic-engineering. However to achieve good load balancing over the equal cost paths $P2-P4-D$, $P2-P5-D$ and the parallel links L3, L4, a mechanism such as Entropy labels [RFC6790] should be adapted for source stacked tunnels. Multiple ways to apply entropy labels were considered and are documented in Section 5 along with their tradeoffs. A recommended solution is described in Section 4.

4. Recommended EL solution for SPRING

The solution described in this section follows [RFC6790].

An LSR may have a limitation in its ability to read and process the label stack in order to do multipath load balancing. This limitation expressed in terms of the number of label stack entries that the LSR can read is henceforth referred to as the Readable Label Depth (RLD) capability of that LSR. If an EL does not occur within the RLD of an LSR in the label stack of the MPLS packet that it receives, then it would lead to poor load balancing at that LSR. The RLD of an LSR is a characteristic of the forwarding plane of that LSR's implementation and determining it is outside the scope of this document.

In order for the EL to occur within the RLD of LSRs along the path corresponding to a label stack, multiple <ELI, EL> pairs MAY be inserted in the label stack as long as the tunnel's label below which they are inserted are advertised with entropy label capability enabled. The LSR that inserts <ELI, EL> pairs MAY have limitations on the number of such pairs that it can insert and also the depth at which it can insert them. If due to any limitation, the inserted ELs are at positions such that an LSR along the path receives an MPLS packet without an EL in the label stack within that LSR's RLD, then the load balancing performed by that LSR would be poor. Special attention should be paid when a forwarding adjacency LSP (FA-LSP) [RFC4206] is used as a link along the path of a source stacked LSP, since the labels of the FA-LSP would additionally count towards the depth of the label stack when calculating the appropriate positions to insert the ELs. The recommendations for inserting <ELI, EL> pairs are:

- o An LSR that is limited in the number of <ELI, EL> pairs that it can insert SHOULD insert such pairs deeper in the stack.
- o An LSR SHOULD try to insert <ELI, EL> pairs at positions so that for the maximum number of transit LSRs, the EL occurs within the RLD of the incoming packet to that LSR.
- o An LSR SHOULD try to insert the minimum number of such pairs while trying to satisfy the above criteria.

A sample algorithm to insert ELs is shown below. Implementations can choose any algorithm as long as it follows the above recommendations.


```
Initialize the current EL insertion point to the
  bottommost label in the stack that is EL-capable
while (local-node can push more <ELI,EL> pairs OR
      insertion point is not above label stack) {
  insert an <ELI,EL> pair below current insertion point
  move new insertion point up from current insertion point until
    ((last inserted EL is below the RLD) AND (RLD > 2)
    AND
    (new insertion point is EL-capable))
  set current insertion point to new insertion point
}
```

Figure 2: Algorithm to insert <ELI, EL> pairs in a label stack

When this algorithm is applied to the example described in Section 3 it will result in ELs being inserted in two positions, one below the label L_N-D and another below L_N-P3. Thus the resulting label stack would be <L_N-P3, ELI, EL, L_A-L1, L_N-D, ELI, EL>

The RLD can be advertised via protocols and those extensions would be described in separate documents [I-D.xu-isis-mpls-elc] and [I-D.xu-ospf-mpls-elc].

The recommendations above are not expected to bring any additional OAM considerations beyond those described in section 6 of [RFC6790]. However, the OAM requirements and solutions for source stacked tunnels are still under discussion and future revisions of this document will address those if needed.

5. Options considered

5.1. Single EL at the bottom of the stack of tunnels

In this option a single EL is used for the entire label stack. The source LSR S encodes the entropy label (EL) below the labels of all the stacked tunnels. In the example described in Section 3 it will result in the label stack at LSR S to look like <L_N-P3, L_A-L1, L_N-D, ELI, EL> <remaining packet header>. Note that the notation in [RFC6790] is used to describe the label stack. An issue with this approach is that as the label stack grows due an increase in the number of SIDs, the EL goes correspondingly deeper in the label stack. Hence transit LSRs have to access a larger number of bytes in the packet header when making forwarding decisions. In the example described in Section 3 the LSR P1 would poorly load-balance traffic on the parallel links L3, L4 since the EL is below the RLD of the packet received by P1. A load balanced network design using this approach must ensure that all intermediate LSRs have the capability

to traverse the maximum label stack depth as required for that application that uses source routed stacking.

In the case where the hardware is capable of pushing a single <ELI, EL> pair at any depth, this option is the same as the recommended solution in Section 4.

This option was discounted since there exist a number of hardware implementations which have a low maximum readable label depth. Choosing this option can lead to a loss of load-balancing using EL in a significant part of the network but that is a critical requirement in a service provider network.

5.2. An EL per tunnel in the stack

In this option each tunnel in the stack can be given its own EL. The source LSR pushes an <ELI, EL> before pushing a tunnel label when load balancing is required to direct traffic on that tunnel. In the example described in Section 3, the source LSR S encoded label stack would be <L_N-P3, ELI, EL, L_A-L1, L_N-D, ELI, EL> where all the ELs can be the same. Accessing the EL at an intermediate LSR is independent of the depth of the label stack and hence independent of the specific application that uses source stacking on that network. A drawback is that the depth of the label stack grows significantly, almost 3 times as the number of labels in the label stack. The network design should ensure that source LSRs should have the capability to push such a deep label stack. Also, the bandwidth overhead and potential MTU issues of deep label stacks should be accounted for in the network design.

In the case where the RLD is the minimum value (3) for all LSRs, all LSRs are EL capable and the LSR that is inserting <ELI, EL> pairs has no limit on how many it can insert then this option is the same as the recommended solution in Section 4.

This option was discounted due to the existence of hardware implementations that can push a limited number of labels on the label stack. Choosing this option would result in a hardware requirement to push two additional labels per tunnel label. Hence it would restrict the number of tunnels that can form a LSP and constrain the types of LSPs that can be created. This was considered unacceptable.

5.3. A re-usable EL for a stack of tunnels

In this option an LSR that terminates a tunnel re-uses the EL of the terminated tunnel for the next inner tunnel. It does this by storing the EL from the outer tunnel when that tunnel is terminated and re-inserting it below the next inner tunnel label during the label swap

operation. The LSR that stacks tunnels SHOULD insert an EL below the outermost tunnel. It SHOULD NOT insert ELs for any inner tunnels. Also, the penultimate hop LSR of a segment MUST NOT pop the ELI and EL even though they are exposed as the top labels since the terminating LSR of that segment would re-use the EL for the next segment.

In Section 3 above, the source LSR S encoded label stack would be <L_N-P3, ELI, EL, L_A-L1, L_N-D>. At P1 the outgoing label stack would be <L_N-P3, ELI, EL, L_A-L1, L_N-D> after it has load balanced to one of the links L3 or L4. At P3 the outgoing label stack would be <L_N-D, ELI, EL>. At P2 the outgoing label stack would be <L_N-D, ELI, EL> and it would load balance to one of the nexthop LSRs P4 or P5. Accessing the EL at an intermediate LSR (e.g. P1) is independent of the depth of the label stack and hence independent of the specific use-case to which the stacked tunnels are applied.

This option was discounted due to the significant change in label swap operations that would be required for existing hardware.

5.3.1. EL at top of stack

A slight variant of the re-usable EL option is to keep the EL at the top of the stack rather than below the tunnel label. In this case each LSR that is not terminating a segment should continue to keep the received EL at the top of the stack when forwarding the packet along the segment. An LSR that terminates a segment should use the EL from the terminated segment at the top of the stack when forwarding onto the next segment.

This option was discounted due to the significant change in label swap operations that would be required for existing hardware.

5.4. ELs at readable label stack depths

In this option the source LSR inserts ELs for tunnels in the label stack at depths such that each LSR along the path that must load balance is able to access at least one EL. Note that the source LSR may have to insert multiple ELs in the label stack at different depths for this to work since intermediate LSRs may have differing capabilities in accessing the depth of a label stack. The label stack depth access value of intermediate LSRs must be known to create such a label stack. How this value is determined is outside the scope of this document. This value can be advertised using a protocol such as an IGP. For the same Section 3 above, if LSR P1 needs to have the EL within a depth of 4, then the source LSR S encoded label stack would be <L_N-P3, ELI, EL, L_A-L1, L_N-D, ELI, EL> where all the ELs would typically have the same value.

In the case where the RLD has different values along the path and the LSR that is inserting <ELI, EL> pairs has no limit on how many pairs it can insert, and it knows the appropriate positions in the stack where they should be inserted, then this option is the same as the recommended solution in Section 4.

A variant of this solution was selected which balances the number of labels that need to be pushed against the requirement for entropy.

6. Acknowledgements

The authors would like to thank John Drake, Loa Andersson, Curtis Villamizar, Greg Mirsky, Markus Jork, Kamran Raza and Nobo Akiya for their review comments and suggestions.

7. IANA Considerations

This memo includes no request to IANA.

8. Security Considerations

This document does not introduce any new security considerations beyond those already listed in [RFC6790].

9. References

9.1. Normative References

[I-D.filsfils-spring-segment-routing]

Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., Litkowski, S., Horneffer, M., Milojevic, I., Shakir, R., Ytti, S., Henderickx, W., Tantsura, J., and E. Crabbe, "Segment Routing Architecture", draft-filsfils-spring-segment-routing-04 (work in progress), July 2014.

[I-D.gredler-spring-mpls]

Gredler, H., Rekhter, Y., Jalil, L., Kini, S., and X. Xu, "Supporting Source/Explicitly Routed Tunnels via Stacked LSPs", draft-gredler-spring-mpls-06 (work in progress), May 2014.

[I-D.ravisingh-mpls-el-for-seamless-mpls]

Singh, R., Shen, Y., and J. Drake, "Entropy label for seamless MPLS", draft-ravisingh-mpls-el-for-seamless-mpls-04 (work in progress), October 2014.

[I-D.xu-isis-mpls-elc]

Xu, X., Kini, S., Sivabalan, S., Filsfils, C., and S. Litkowski, "Signaling Entropy Label Capability Using IS-IS", draft-xu-isis-mpls-elc-01 (work in progress), September 2014.

[I-D.xu-ospf-mpls-elc]

Xu, X., Kini, S., Sivabalan, S., Filsfils, C., and S. Litkowski, "Signaling Entropy Label Capability Using OSPF", draft-xu-ospf-mpls-elc-01 (work in progress), October 2014.

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

[RFC4206] Kompella, K. and Y. Rekhter, "Label Switched Paths (LSP) Hierarchy with Generalized Multi-Protocol Label Switching (GMPLS) Traffic Engineering (TE)", RFC 4206, October 2005.

[RFC6790] Kompella, K., Drake, J., Amante, S., Henderickx, W., and L. Yong, "The Use of Entropy Labels in MPLS Forwarding", RFC 6790, November 2012.

9.2. Informative References

[I-D.filsfils-spring-segment-routing-use-cases]

Filsfils, C., Francois, P., Previdi, S., Decraene, B., Litkowski, S., Horneffer, M., Milojevic, I., Shakir, R., Ytti, S., Henderickx, W., Tantsura, J., Kini, S., and E. Crabbe, "Segment Routing Use Cases", draft-filsfils-spring-segment-routing-use-cases-01 (work in progress), October 2014.

[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-03 (work in progress), October 2014.

[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-04 (work in progress), February 2015.

[RFC7325] Villamizar, C., Kompella, K., Amante, S., Malis, A., and C. Pignataro, "MPLS Forwarding Compliance and Performance Requirements", RFC 7325, August 2014.

Authors' Addresses

Sriganesh Kini (editor)
Ericsson

Email: sriganesh.kini@ericsson.com

Kireeti Kompella
Juniper

Email: kireeti@juniper.net

Siva Sivabalan
Cisco

Email: msiva@cisco.com

Stephane Litkowski
Orange

Email: stephane.litkowski@orange.com

Rob Shakir
B.T.

Email: rob.shakir@bt.com

Xiaohu Xu
Huawei

Email: xuxiaohu@huawei.com

Wim Hendrickx
Alcatel-Lucent

Email: wim.henderickx@alcatel-lucent.com

Jeff Tantsura
Ericsson

Email: jeff.tantsura@ericsson.com

MPLS Working Group
Internet-Draft
Intended status: Standards Track
Expires: February 9, 2016

G. Mirsky
J. Tantsura
Ericsson
I. Varlashkin
Google
M. Chen
Huawei
August 8, 2015

Bidirectional Forwarding Detection (BFD) Directed Return Path
draft-mirsky-mpls-bfd-directed-04

Abstract

Bidirectional Forwarding Detection (BFD) is expected to monitor bi-directional paths. When a BFD session monitors in its forward direction an explicitly routed path there is a need to be able to direct egress BFD peer to use specific path as reverse direction of the BFD session.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on February 9, 2016.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
1.1. Conventions used in this document	3
1.1.1. Terminology	3
1.1.2. Requirements Language	3
2. Problem Statement	3
3. Direct Reverse BFD Path	4
3.1. Case of MPLS Data Plane	4
3.1.1. BFD Reverse Path TLV	4
3.1.2. Static and RSVP-TE sub-TLVs	5
3.1.3. Segment Routing Tunnel sub-TLV	5
3.2. Case of IPv6 Data Plane	6
3.3. Bootstrapping BFD session with BFD Reverse Path over Segment Routed tunnel	6
3.4. Return Codes	7
4. Use Case Scenario	7
5. IANA Considerations	7
5.1. TLV	8
5.2. Sub-TLV	8
5.3. Return Codes	8
6. Security Considerations	9
7. Acknowledgements	9
8. Normative References	9
Authors' Addresses	10

1. Introduction

RFC 5880 [RFC5880], RFC 5881 [RFC5881], and RFC 5883 [RFC5883] established the BFD protocol for IP networks and RFC 5884 [RFC5884] set rules of using BFD asynchronous mode over IP/MPLS LSPs. All standards implicitly assume that the egress BFD peer will use the shortest path route regardless of route being used to send BFD control packets towards it. As result, if the ingress BFD peer sends its BFD control packets over explicit path that is diverging from the best route, then reverse direction of the BFD session is likely not to be on co-routed bi-directional path with the forward direction of the BFD session. And because BFD control packets are not guaranteed to cross the same links and nodes in both directions detection of Loss of Continuity (LoC) defect in forward direction may demonstrate positive negatives.

This document defines the extension to LSP Ping [RFC4379], BFD Reverse Path TLV, and proposes that it to be used to instruct the egress BFD peer to use explicit path for its BFD control packets associated with the particular BFD session. The TLV will be allocated from the TLV and sub-TLV registry defined by RFC 4379 [RFC4379]. As a special case, forward and reverse directions of the BFD session can form bi-directional co-routed associated channel.

1.1. Conventions used in this document

1.1.1. Terminology

BFD: Bidirectional Forwarding Detection

MPLS: Multiprotocol Label Switching

LSP: Label Switching Path

LoC: Loss of Continuity

1.1.2. 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 [RFC2119].

2. Problem Statement

BFD is best suited to monitor bi-directional co-routed paths. In most cases, given stable environments, the forward and reverse direction between two nodes is likely to be co-routed, this fulfilling the implicit BFD requirements. If BFD is used to monitor unidirectional explicitly routed paths, e.g. MPLS-TE LSPs, its control packets in forward direction would be in-band using the mechanism defined in [RFC5884] and [RFC5586]. But the reverse direction of the BFD session would still follow the shortest path route and that might lead to the following problems detecting failures on the unidirectional explicit path:

- o detection of a failure on the reverse path cannot reliably be interpreted as bi-directional defect and thus trigger, for example, protection switchover of the forward direction;
- o if a failure of the reverse path had been ignored, the ingress node would not receive indication of forward direction failure from its egress peer.

To address these challenges the egress BFD peer should be instructed to use specific path for its control packets.

3. Direct Reverse BFD Path

3.1. Case of MPLS Data Plane

LSP ping, defined in [RFC4379], uses BFD Discriminator TLV [RFC5884] to bootstrap a BFD session over an MPLS LSP. This document defines a new TLV, BFD Reverse Path TLV, that MUST contain a single sub-TLV that can be used to carry information about reverse path for the specified in BFD Discriminator TLV session.

3.1.1. BFD Reverse Path TLV

The BFD Reverse Path TLV is an optional TLV within the LSP ping protocol. However, if used, the BFD Discriminator TLV MUST be included in an Echo Request message as well. If the BFD Discriminator TLV is not present when the BFD Reverse Path TLV is included, then it MUST be treated as malformed Echo Request, as described in [RFC4379].

The BFD Reverse Path TLV carries the specified path that BFD control packets of the BFD session referenced in the BFD Discriminator TLV are required to follow. The format of the BFD Reverse Path TLV is as presented in Figure 1.

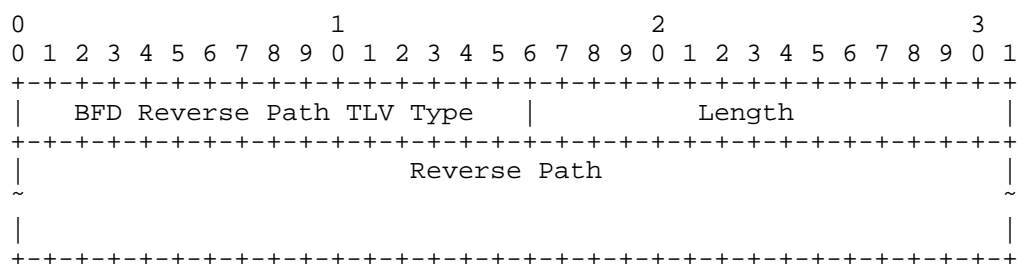


Figure 1: BFD Reverse Path TLV

BFD Reverse Path TLV Type is 2 octets in length and value to be assigned by IANA.

Length is 2 octets in length and defines the length in octets of the Reverse Path field.

Reverse Path field contains a sub-TLV. Any Target FEC sub-TLV, already or in the future defined, from IANA sub-registry Sub-TLVs for TLV Types 1, 16, and 21 of MPLS LSP Ping Parameters registry MAY be

used in this field. Only one sub-TLV MUST be included in the Reverse Path TLV. If more than one sub-TLVs are present in the Reverse Path TLV, then only the first sub-TLV MUST be used and the rest MUST be silently discarded.

If the egress LSR cannot find path specified in the Reverse Path TLV it MUST send Echo Reply with the received Reverse Path TLV and set the return code to "Failed to establish the BFD session. The specified reverse path was not found" Section 3.4. The egress LSR MAY establish the BFD session over IP network according to [RFC5884].

3.1.2. Static and RSVP-TE sub-TLVs

When explicit path on MPLS data plane set either as Static or RSVP-TE LSP respective sub-TLVs defined in [RFC7110] identify explicit return path.

3.1.3. Segment Routing Tunnel sub-TLV

In addition to Static and RSVP-TE, Segment Routing with MPLS data plane can be used to set explicit path. In this case a new sub-TLV is defined in this document as presented in Figure 2.

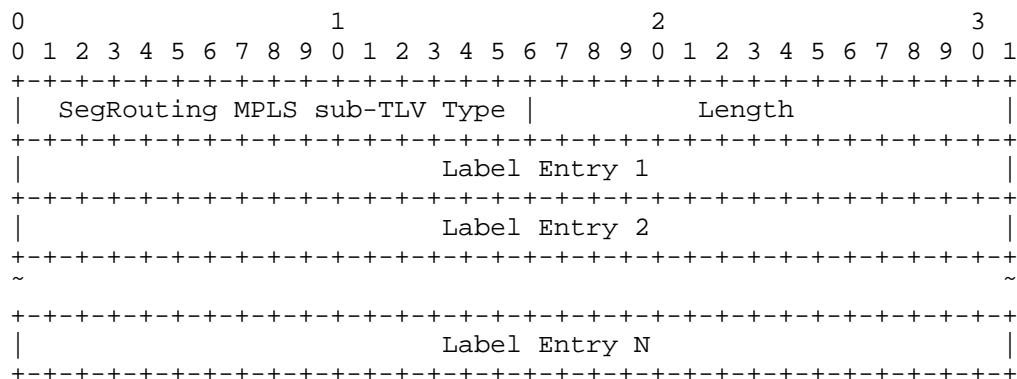


Figure 2: Segment Routing MPLS Tunnel sub-TLV

The Segment Routing Tunnel sub-TLV Type is two octets in length, and will be allocated by IANA.

The egress LSR MUST use the Value field as label stack for BFD control packets for the BFD session identified by source IP address and value in BFD Discriminator TLV.

The Segment Routing Tunnel sub-TLV MAY be used in Reply Path TLV defined in [RFC7110]

3.2. Case of IPv6 Data Plane

IPv6 can be data plane of choice for Segment Routed tunnels [I-D.previdi-6man-segment-routing-header]. In such networks the BFD Reverse Path TLV described in Section 3.1.1 can be used as well. To specify reverse path of a BFD session in IPv6 environment the BFD Discriminator TLV MUST be used along with the BFD Reverse Path TLV. The BFD Reverse Path TLV in IPv6 network MUST include sub-TLV.

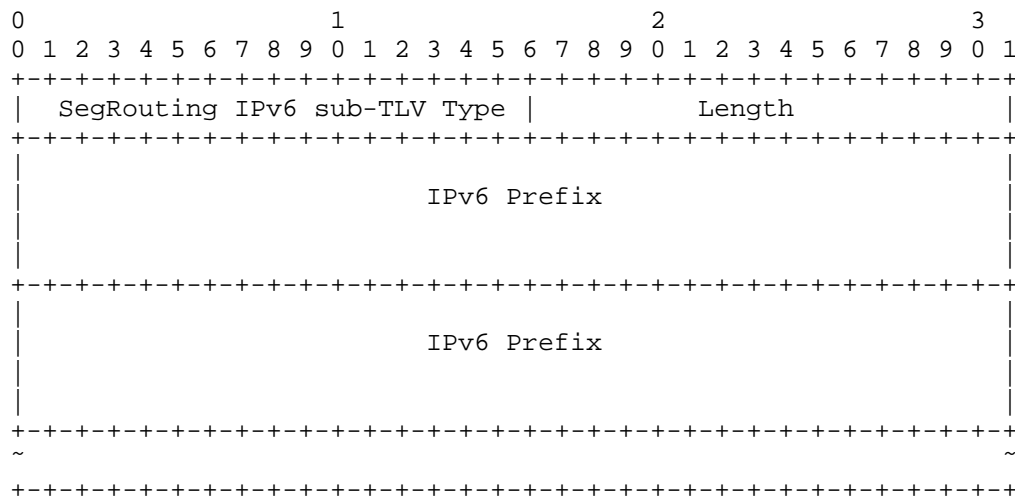


Figure 3: Segment Routing IPv6 Tunnel sub-TLV

3.3. Bootstrapping BFD session with BFD Reverse Path over Segment Routed tunnel

As discussed in [I-D.kumarkini-mpls-spring-lsp-ping] introduction of Segment Routing network domains with MPLS data plane adds three new sub-TLVs that may be used with Target FEC TLV. Section 6.1 addresses use of new sub-TLVs in Target FEC TLV in LSP ping and LSP traceroute. For the case of LSP ping the [I-D.kumarkini-mpls-spring-lsp-ping] states that:

"Initiator MUST include FEC(s) corresponding to the destination segment.

Initiator, i.e. ingress LSR, MAY include FECs corresponding to some or all of segments imposed in the label stack by the ingress LSR to communicate the segments traversed. "

When LSP ping is used to bootstrap BFD session this document updates this and defines that LSP Ping MUST include the FEC corresponding to the destination segment and SHOULD NOT include FECs corresponding to some or all of segment imposed by the ingress LSR. Operationally such restriction would not cause any problem or uncertainty as LSP ping with FECs corresponding to some or all segments or traceroute MAY precede the LSP ping that bootstraps the BFD session.

3.4. Return Codes

This document defines the following Return Codes:

- o "Failed to establish the BFD session. The specified reverse path was not found", (TBD4). When a specified reverse path is not available at the egress LSR, an Echo Reply with the return code set to "Failed to establish the BFD session. The specified reverse path was not found" MUST be sent back to the ingress LSR . (Section 3.1.1)

4. Use Case Scenario

In network presented in Figure 4 node A monitors two tunnels to node H: A-B-C-D-G-H and A-B-E-F-G-H. To bootstrap BFD session to monitor the first tunnel, node A MUST include BFD Discriminator TLV with Discriminator value foobar-1 and MAY include BFD Reverse Path TLV that references H-G-D-C-B-A tunnel. To bootstrap BFD session to monitor the second tunnel, node A MUST include BFD Discriminator TLV with Discriminator value foobar-2 [I-D.ietf-bfd-rfc5884-clarifications] and MAY include BFD Reverse Path TLV that references H-G-F-E-B-A tunnel.

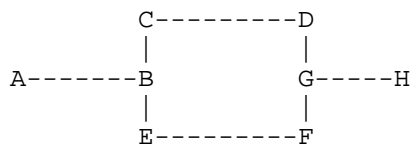


Figure 4: Use Case for BFD Reverse Path TLV

If an operator needs node H to monitor path to node A, e.g. H-G-D-C-B-A tunnel, then by looking up list of known Reverse Paths it MAY find and use existing BFD sessions.

5. IANA Considerations

5.1. TLV

The IANA is requested to assign a new value for BFD Reverse Path TLV from the "Multiprotocol Label Switching Architecture (MPLS) Label Switched Paths (LSPs) Ping Parameters - TLVs" registry, "TLVs and sub-TLVs" sub-registry.

Value	Description	Reference
X (TBD1)	BFD Reverse Path TLV	This document

Table 1: New BFD Reverse Type TLV

5.2. Sub-TLV

The IANA is requested to assign two new sub-TLV types from "Multiprotocol Label Switching Architecture (MPLS) Label Switched Paths (LSPs) Ping Parameters - TLVs" registry, "Sub-TLVs for TLV Types 1, 16, and 21" sub-registry.

Value	Description	Reference
X (TBD2)	Segment Routing MPLS Tunnel sub-TLV	This document
X (TBD3)	Segment Routing IPv6 Tunnel sub-TLV	This document

Table 2: New Segment Routing Tunnel sub-TLV

5.3. Return Codes

The IANA is requested to assign a new Return Code value from the "Multi-Protocol Label Switching (MPLS) Label Switched Paths (LSPs) Ping Parameters" registry, "Return Codes" sub-registry, as follows using a Standards Action value.

Value	Description	Reference
X (TBD4)	Failed to establish the BFD session. The specified reverse path was not found.	This document

Table 3: New Return Code

6. Security Considerations

Security considerations discussed in [RFC5880], [RFC5884], and [RFC4379], apply to this document.

7. Acknowledgements

8. Normative References

- [I-D.ietf-bfd-rfc5884-clarifications]
Govindan, V., Rajaraman, K., Mirsky, G., Akiya, N., and S. Aldrin, "Clarifications to RFC 5884", draft-ietf-bfd-rfc5884-clarifications-02 (work in progress), June 2015.
- [I-D.kumarkini-mpls-spring-lsp-ping]
Kumar, N., Swallow, G., Pignataro, C., Akiya, N., Kini, S., Gredler, H., and M. Chen, "Label Switched Path (LSP) Ping/Trace for Segment Routing Networks Using MPLS Dataplane", draft-kumarkini-mpls-spring-lsp-ping-04 (work in progress), July 2015.
- [I-D.previdi-6man-segment-routing-header]
Previdi, S., Filsfils, C., Field, B., Leung, I., Vyncke, E., and D. Lebrun, "IPv6 Segment Routing Header (SRH)", draft-previdi-6man-segment-routing-header-07 (work in progress), July 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>>.
- [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>>.
- [RFC5586] Bocci, M., Ed., Vigoureux, M., Ed., and S. Bryant, Ed., "MPLS Generic Associated Channel", RFC 5586, DOI 10.17487/RFC5586, June 2009, <<http://www.rfc-editor.org/info/rfc5586>>.
- [RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <<http://www.rfc-editor.org/info/rfc5880>>.

- [RFC5881] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD) for IPv4 and IPv6 (Single Hop)", RFC 5881, DOI 10.17487/RFC5881, June 2010, <<http://www.rfc-editor.org/info/rfc5881>>.
- [RFC5883] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD) for Multihop Paths", RFC 5883, DOI 10.17487/RFC5883, June 2010, <<http://www.rfc-editor.org/info/rfc5883>>.
- [RFC5884] Aggarwal, R., Kompella, K., Nadeau, T., and G. Swallow, "Bidirectional Forwarding Detection (BFD) for MPLS Label Switched Paths (LSPs)", RFC 5884, DOI 10.17487/RFC5884, June 2010, <<http://www.rfc-editor.org/info/rfc5884>>.
- [RFC7110] Chen, M., Cao, W., Ning, S., Jounay, F., and S. Delord, "Return Path Specified Label Switched Path (LSP) Ping", RFC 7110, DOI 10.17487/RFC7110, January 2014, <<http://www.rfc-editor.org/info/rfc7110>>.

Authors' Addresses

Greg Mirsky
Ericsson

Email: gregory.mirsky@ericsson.com

Jeff Tantsura
Ericsson

Email: jeff.tantsura@ericsson.com

Ilya Varlashkin
Google

Email: Ilya@nobulus.com

Mach(Guoyi) Chen
Huawei

Email: mach.chen@huawei.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 4, 2016

S. Previdi, Ed.
C. Filsfils
Cisco Systems, Inc.
B. Field
Comcast
I. Leung
Rogers Communications
J. Linkova
Google
E. Aries
Facebook
T. Kosugi
NTT
E. Vyncke
Cisco Systems, Inc.
D. Lebrun
Universite Catholique de Louvain
October 2, 2015

IPv6 Segment Routing Header (SRH)
draft-previdi-6man-segment-routing-header-08

Abstract

Segment Routing (SR) allows a node to steer a packet through a controlled set of instructions, called segments, by prepending a SR header to the packet. A segment can represent any instruction, topological or service-based. SR allows to enforce a flow through any path (topological, or application/service based) while maintaining per-flow state only at the ingress node to the SR domain.

Segment Routing can be applied to the IPv6 data plane with the addition of a new type of Routing Extension Header. This draft describes the Segment Routing Extension Header Type and how it is used by SR capable nodes.

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

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on April 4, 2016.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Segment Routing Documents	3
2. Introduction	3
2.1. Data Planes supporting Segment Routing	4
2.2. Segment Routing (SR) Domain	4
2.2.1. SR Domain in a Service Provider Network	5
2.2.2. SR Domain in a Overlay Network	6
2.3. Illustration	8
3. IPv6 Instantiation of Segment Routing	10
3.1. Segment Identifiers (SIDs)	10
3.1.1. Node-SID	10
3.1.2. Adjacency-SID	11
3.2. Segment Routing Extension Header (SRH)	11
3.2.1. SRH and RFC2460 behavior	14
4. SRH Procedures	15
4.1. Segment Routing Node Functions	15
4.1.1. Source SR Node	16
4.1.2. SR Domain Ingress Node	17
4.1.3. Transit Node	17
4.1.4. SR Segment Endpoint Node	17

5.	Security Considerations	18
5.1.	Threat model	19
5.1.1.	Source routing threats	19
5.1.2.	Applicability of RFC 5095 to SRH	19
5.1.3.	Service stealing threat	20
5.1.4.	Topology disclosure	20
5.1.5.	ICMP Generation	20
5.2.	Security fields in SRH	21
5.2.1.	Selecting a hash algorithm	22
5.2.2.	Performance impact of HMAC	22
5.2.3.	Pre-shared key management	23
5.3.	Deployment Models	23
5.3.1.	Nodes within the SR domain	23
5.3.2.	Nodes outside of the SR domain	24
5.3.3.	SR path exposure	24
5.3.4.	Impact of BCP-38	25
6.	IANA Considerations	25
7.	Manageability Considerations	25
8.	Contributors	25
9.	Acknowledgements	26
10.	References	26
10.1.	Normative References	26
10.2.	Informative References	26
	Authors' Addresses	28

1. Segment Routing Documents

Segment Routing terminology is defined in
[I-D.ietf-spring-segment-routing].

Segment Routing use cases are described in
[I-D.ietf-spring-problem-statement] and
[I-D.ietf-spring-ipv6-use-cases].

Segment Routing protocol extensions are defined in
[I-D.ietf-isis-segment-routing-extensions], and
[I-D.ietf-ospf-ospfv3-segment-routing-extensions].

2. Introduction

Segment Routing (SR), defined in [I-D.ietf-spring-segment-routing], allows a node to steer a packet through a controlled set of instructions, called segments, by prepending a SR header to the packet. A segment can represent any instruction, topological or service-based. SR allows to enforce a flow through any path (topological or service/application based) while maintaining per-flow state only at the ingress node to the SR domain. Segments can be derived from different components: IGP, BGP, Services, Contexts,

Locators, etc. The list of segment forming the path is called the Segment List and is encoded in the packet header.

SR allows the use of strict and loose source based routing paradigms without requiring any additional signaling protocols in the infrastructure hence delivering an excellent scalability property.

The source based routing model described in [I-D.ietf-spring-segment-routing] is inherited from the ones proposed by [RFC1940] and [RFC2460]. The source based routing model offers the support for explicit routing capability.

2.1. Data Planes supporting Segment Routing

Segment Routing (SR), can be instantiated over MPLS ([I-D.ietf-spring-segment-routing-mpls]) and IPv6. This document defines its instantiation over the IPv6 data-plane based on the use-cases defined in [I-D.ietf-spring-ipv6-use-cases].

This document defines a new type of Routing Header (originally defined in [RFC2460]) called the Segment Routing Header (SRH) in order to convey the Segment List in the packet header as defined in [I-D.ietf-spring-segment-routing]. Mechanisms through which segment are known and advertised are outside the scope of this document.

A segment is materialized by an IPv6 address. A segment identifies a topological instruction or a service instruction. A segment can be either:

- o global: a global segment represents an instruction supported by all nodes in the SR domain and it is instantiated through an IPv6 address globally known in the SR domain.
- o local: a local segment represents an instruction supported only by the node who originates it and it is instantiated through an IPv6 address that is known only by the local node.

2.2. Segment Routing (SR) Domain

We define the concept of the Segment Routing Domain (SR Domain) as the set of nodes participating into the source based routing model. These nodes may be connected to the same physical infrastructure (e.g.: a Service Provider's network) as well as nodes remotely connected to each other (e.g.: an enterprise VPN or an overlay).

A non-exhaustive list of examples of SR Domains is:

- o The network of an operator, service provider, content provider, enterprise including nodes, links and Autonomous Systems.
- o A set of nodes connected as an overlay over one or more transit providers. The overlay nodes exchange SR-enabled traffic with segments belonging solely to the overlay routers (the SR domain). None of the segments in the SR-enabled packets exchanged by the overlay belong to the transit networks

The source based routing model through its instantiation of the Segment Routing Header (SRH) defined in this document equally applies to all the above examples.

While the source routing model defined in [RFC2460] doesn't mandate which node is allowed to insert (or modify) the SRH, it is assumed in this document that the SRH is inserted in the packet by its source. For example:

- o At the node originating the packet (host, server).
- o At the ingress node of a SR domain where the ingress node receives an IPv6 packet and encapsulates it into an outer IPv6 header followed by a Segment Routing header.

2.2.1. SR Domain in a Service Provider Network

The following figure illustrates an SR domain consisting of an operator's network infrastructure.

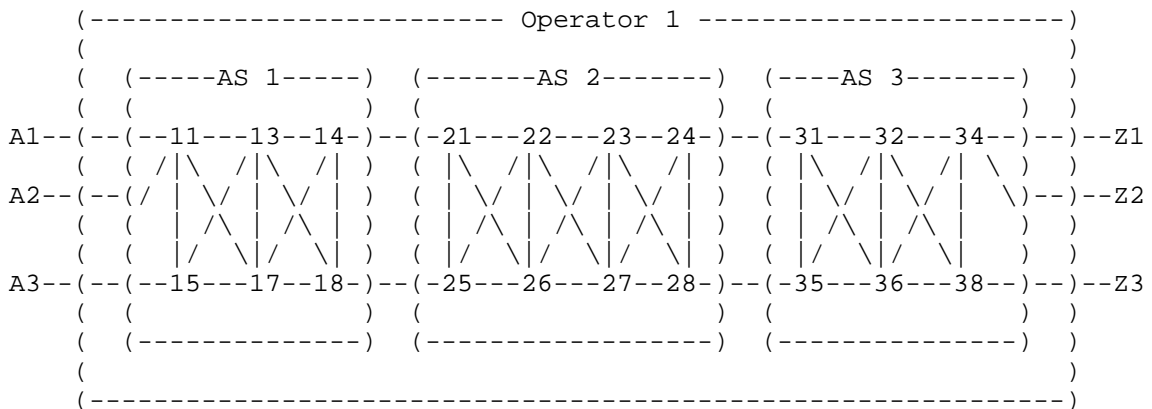


Figure 1: Service Provider SR Domain

Figure 1 describes an operator network including several ASes and delivering connectivity between endpoints. In this scenario, Segment

Routing is used within the operator networks and across the ASes boundaries (all being under the control of the same operator). In this case segment routing can be used in order to address use cases such as end-to-end traffic engineering, fast re-route, egress peer engineering, data-center traffic engineering as described in [I-D.ietf-spring-problem-statement], [I-D.ietf-spring-ipv6-use-cases] and [I-D.ietf-spring-resiliency-use-cases].

Typically, an IPv6 packet received at ingress (i.e.: from outside the SR domain), is classified according to network operator policies and such classification results into an outer header with an SRH applied to the incoming packet. The SRH contains the list of segment representing the path the packet must take inside the SR domain. Thus, the SA of the packet is the ingress node, the DA (due to SRH procedures described in Section 4) is set as the first segment of the path and the last segment of the path is the egress node of the SR domain.

The path may include intra-AS as well as inter-AS segments. It has to be noted that all nodes within the SR domain are under control of the same administration. When the packet reaches the egress point of the SR domain, the outer header and its SRH are removed so that the destination of the packet is unaware of the SR domain the packet has traversed.

The outer header with the SRH is no different from any other tunneling encapsulation mechanism and allows a network operator to implement traffic engineering mechanisms so to efficiently steer traffic across his infrastructure.

2.2.2. SR Domain in a Overlay Network

The following figure illustrates an SR domain consisting of an overlay network over multiple operator's networks.

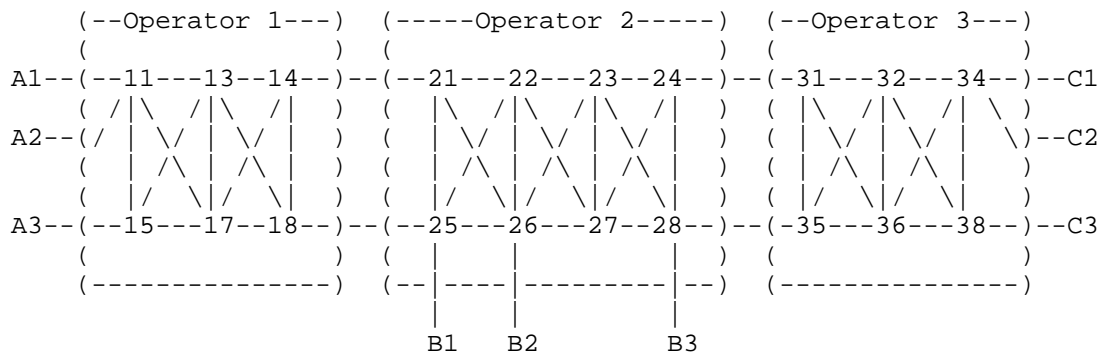


Figure 2: Overlay SR Domain

Figure 2 describes an overlay consisting of nodes connected to three different network operators and forming a single overlay network where Segment routing packets are exchanged.

The overlay consists of nodes A1, A2, A3, B1, B2, B3, C1, C2 and C3. These nodes are connected to their respective network operator and form an overlay network.

Each node may originate packets with an SRH which contains, in the segment list of the SRH or in the DA, segments identifying other overlay nodes. This implies that packets with an SRH may traverse operator's networks but, obviously, these SRHs cannot contain an address/segment of the transit operators 1, 2 and 3. The SRH originated by the overlay can only contain address/segment under the administration of the overlay (e.g. address/segments supported by A1, A2, A3, B1, B2, B3, C1, C2 or C3).

In this model, the operator network nodes are transit nodes and, according to [RFC2460], MUST NOT inspect the routing extension header since there are not the DA of the packet.

It is a common practice in operators networks to filter out, at ingress, any packet whose DA is the address of an internal node and it is also possible that an operator would filter out any packet destined to an internal address and having an extension header in it.

This common practice does not impact the SR-enabled traffic between the overlay nodes as the intermediate transit networks do never see a destination address belonging to their infrastructure. These SR-enabled overlay packets will thus never be filtered by the transit operators.

In all cases, transit packets (i.e.: packets whose DA is outside the domain of the operator's network) will be forwarded accordingly without introducing any security concern in the operator's network. This is similar to tunneled packets.

2.3. Illustration

In the context of Figure 3 we illustrate an example of how segment routing can be used within a SR domain in order to engineer traffic. Let's assume that the SR domain is configured as a single AS and the IGP (OSPF or IS-IS) is configured using the same cost on every link. Let's also assume that a packet P enters the SR domain at an ingress edge router I and that the operator requests the following requirements for packet P:

- o The local service S offered by node B must be applied to packet P.
- o The links AB and CE cannot be used to transport the packet P.
- o Any node N along the journey of the packet should be able to determine where the packet P entered the SR domain and where it will exit. The intermediate node should be able to determine the paths from the ingress edge router to itself, and from itself to the egress edge router.
- o Per-flow State for packet P should only be created at the ingress edge router.
- o The operator can forbid, for security reasons, anyone outside the operator domain to exploit its intra-domain SR capabilities.

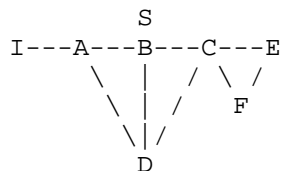


Figure 3: An illustration of SR properties

All these properties may be realized by instructing the ingress SR edge router I to create a SRH with the list of segments the packet must traverse: D, B, S, F, E. Therefore, the ingress router I creates an outer header where:

- o the SA is the IPv6 address of I

- o the final destination of the packet is the SR egress node E however, D being the first segment of the path, the DA is set to D IPv6 address.
- o the SRH is inserted with the segment list consisting of following IPv6 addresses: D, B, S, F, E

The SRH contains a source route encoded as a list of segments (D, B, S, F, E). The ingress and egress nodes are identified in the packet respectively by the SA and the last segment of the segment list.

The packet P reaches the ingress SR node I. Node I pushes the newly created outer header and SRH with the Segment List as illustrated above (D, B, S, F, E)

D is the IPv6 address of node D and it is recognized by all nodes in the SR domain as the forwarding instruction "forward to D according to D route in the IPv6 routing table". The routing table being built through IGPs (OSPF or IS-IS) it is equivalent to say "forward according to shortest path to D".

Once at D, the next segment is inspected and executed (segment B).

B is an instruction recognized by all the nodes in the SR domain which causes the packet to be forwarded along the shortest path to B.

Once at B, the next segment is executed (segment S).

S is an instruction only recognized by node B which causes the packet to receive service S.

Once the service S is applied, the next segment is executed (segment F) which causes the packet to be forwarded along the shortest path to F.

Once at F, the next segment is executed (segment E).

E is an instruction recognized by all the nodes in the SR domain which causes the packet to be forwarded along the shortest path to E.

E being the destination of the packet, removes the outer header and the SRH. Then, it inspects the inner packet header and forwards the packet accordingly.

All of the requirements are met:

- o First, the packet P has not used links AB and CE: the shortest-path from I to D is I-A-D, the shortest-path from D to B is D-B,

the shortest-path from B to F is B-C-F and the shortest-path from F to E is F-E, hence the packet path through the SR domain is I-A-D-B-C-F-E and the links AB and CE have been avoided.

- o Second, the service S supported by B has been applied on packet P.
- o Third, any node along the packet path is able to identify the service and topological journey of the packet within the SR domain by inspecting the SRH and SA/DA fields of the packet header.
- o Fourth, only node I maintains per-flow state for packet P. The entire program of topological and service instructions to be executed by the SR domain on packet P is encoded by the ingress edge router I in the SR header in the form of a list of segments where each segment identifies a specific instruction. No further per-flow state is required along the packet path. Intermediate nodes only hold states related to the global node segments and their local segments. These segments are not per-flow specific and hence scale very well. Typically, an intermediate node would maintain in the order of 100's to 1000's global node segments and in the order of 10's to 100 of local segments.
- o Fifth, the SR header (and its outer header) is inserted at the entrance to the domain and removed at the exit of the operator domain. For security reasons, the operator can forbid anyone outside its domain to use its intra-domain SR capability (e.g. configuring ACL that deny any packet with a DA towards its infrastructure segment).

3. IPv6 Instantiation of Segment Routing

3.1. Segment Identifiers (SIDs)

Segment Routing, as described in [I-D.ietf-spring-segment-routing], defines Node-SID and Adjacency-SID. When SR is used over IPv6 data-plane the following applies.

3.1.1. Node-SID

The Node-SID identifies a node. With SR-IPv6 the Node-SID is an IPv6 address that the operator configured on the node and that is used as the node identifier. Typically, in case of a router, this is the IPv6 address of the node loopback interface. Therefore, SR-IPv6 does not require any additional SID advertisement for the Node Segment. The Node-SID is in fact the IPv6 address of the node.

3.1.2. Adjacency-SID

Adjacency-SIDs can be either globally scoped IPv6 addresses or IPv6 addresses known locally by the node but not advertised in any control plane (in other words an Adjacency-SID may well be any 128-bit identifier). Obviously, in the latter case, the scope of the Adjacency-SID is local to the router and any packet with the a such Adjacency-SID would need first to reach the node through the node's Segment Identifier (i.e.: Node-SID) prior for the node to process the Adjacency-SID. In other words, two segments (SIDs) would then be required: the first is the node's Node-SID that brings the packet to the node and the second is the Adjacency-SID that will make the node to forward the packet through the interface the Adjacency-SID is allocated to.

In the SR architecture defined in [I-D.ietf-spring-segment-routing] a node may advertise one (or more) Adj-SIDs allocated to the same interface as well as a node can advertise the same Adj-SID for multiple interfaces. Use cases of Adj-SID advertisements are described in [I-D.ietf-spring-segment-routing] The semantic of the Adj-SID is:

Send out the packet to the interface this Adj-SID is allocated to.

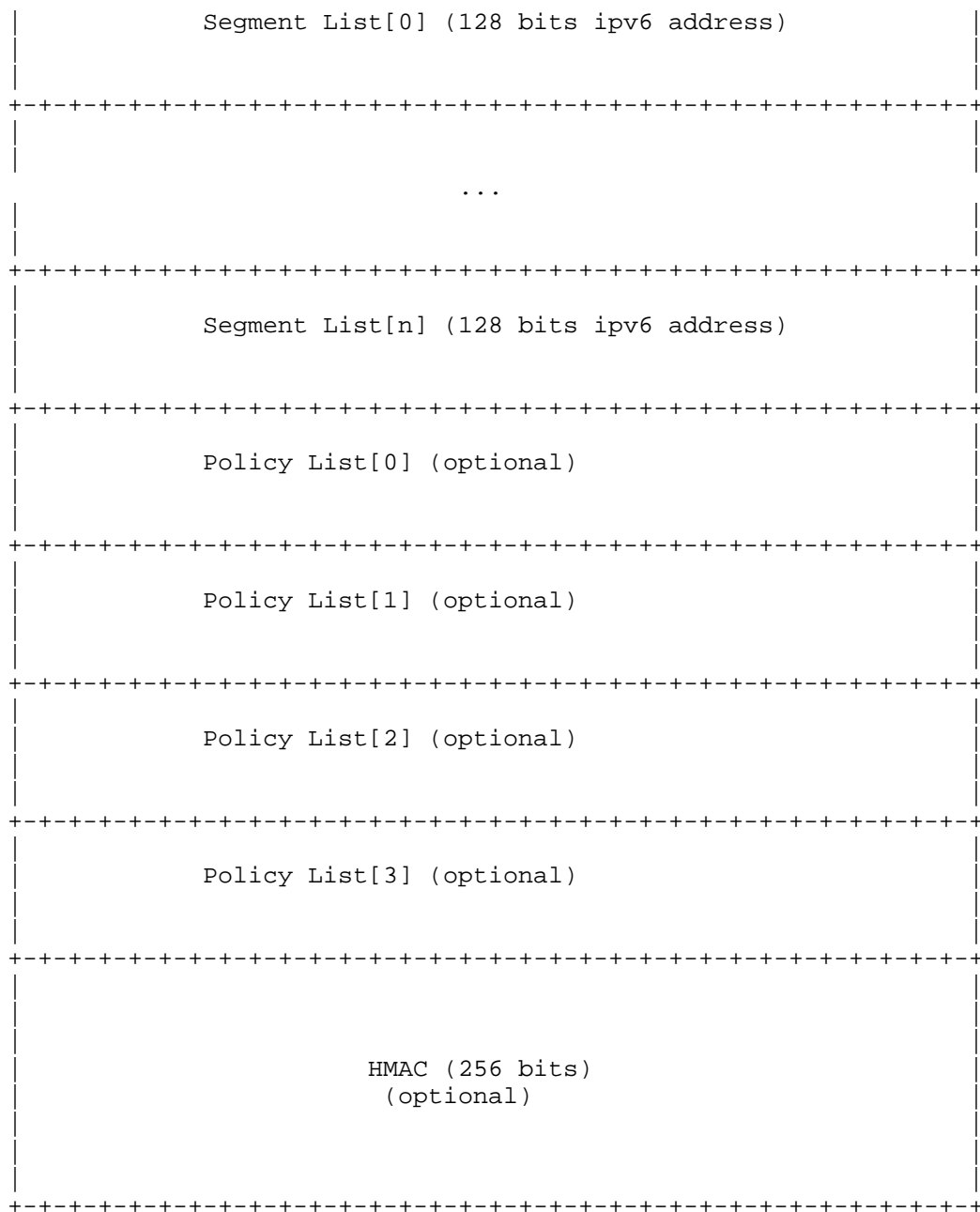
Advertisement of Adj-SID may be done using multiple mechanisms among which the ones described in ISIS and OSPF protocol extensions: [I-D.ietf-isis-segment-routing-extensions] and [I-D.ietf-ospf-ospfv3-segment-routing-extensions]. The distinction between local and global significance of the Adj-SID is given in the encoding of the Adj-SID advertisement.

3.2. Segment Routing Extension Header (SRH)

A new type of the Routing Header (originally defined in [RFC2460]) is defined: the Segment Routing Header (SRH) which has a new Routing Type, (suggested value 4) to be assigned by IANA.

The Segment Routing Header (SRH) 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
Next Header	Hdr Ext Len	Routing Type	Segments Left
First Segment	Flags	HMAC Key ID	



where:

- o Next Header: 8-bit selector. Identifies the type of header immediately following the SRH.
- o Hdr Ext Len: 8-bit unsigned integer, is the length of the SRH header in 8-octet units, not including the first 8 octets.
- o Routing Type: TBD, to be assigned by IANA (suggested value: 4).
- o Segments Left. Defined in [RFC2460], it contains the index, in the Segment List, of the next segment to inspect. Segments Left is decremented at each segment.
- o First Segment: contains the index, in the Segment List, of the first segment of the path which is in fact the last element of the Segment List.
- o Flags: 16 bits of flags. Following flags are defined:

```

                                1
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
+---+---+---+---+---+---+---+---+---+
|C|P|R|R|   Policy Flags   |
+---+---+---+---+---+---+---+---+

```

C-flag: Clean-up flag. Set when the SRH has to be removed from the packet when packet reaches the last segment.

P-flag: Protected flag. Set when the packet has been rerouted through FRR mechanism by a SR endpoint node.

R-flags. Reserved and for future use.

Policy Flags. Define the type of the IPv6 addresses encoded into the Policy List (see below). The following have been defined:

Bits 4-6: determine the type of the first element after the segment list.

Bits 7-9: determine the type of the second element.

Bits 10-12: determine the type of the third element.

Bits 13-15: determine the type of the fourth element.

The following values are used for the type:

0x0: Not present. If value is set to 0x0, it means the element represented by these bits is not present.

0x1: SR Ingress.

0x2: SR Egress.

0x3: Original Source Address.

0x4 to 0x7: currently unused and SHOULD be ignored on reception.

- o HMAC Key ID and HMAC field, and their use are defined in Section 5.
- o Segment List[n]: 128 bit IPv6 addresses representing the nth segment in the Segment List. The Segment List is encoded starting from the last segment of the path. I.e., the first element of the segment list (Segment List [0]) contains the last segment of the path while the last segment of the Segment List (Segment List[n]) contains the first segment of the path. The index contained in "Segments Left" identifies the current active segment.
- o Policy List. Optional addresses representing specific nodes in the SR path such as:

SR Ingress: a 128 bit generic identifier representing the ingress in the SR domain (i.e.: it needs not to be a valid IPv6 address).

SR Egress: a 128 bit generic identifier representing the egress in the SR domain (i.e.: it needs not to be a valid IPv6 address).

Original Source Address: IPv6 address originally present in the SA field of the packet.

The segments in the Policy List are encoded after the segment list and they are optional. If none are in the SRH, all bits of the Policy List Flags MUST be set to 0x0.

3.2.1. SRH and RFC2460 behavior

The SRH being a new type of the Routing Header, it also has the same properties:

SHOULD only appear once in the packet.

Only the router whose address is in the DA field of the packet header MUST inspect the SRH.

Therefore, Segment Routing in IPv6 networks implies that the segment identifier (i.e.: the IPv6 address of the segment) is moved into the DA of the packet.

The DA of the packet changes at each segment termination/completion and therefore the original DA of the packet MUST be encoded as the last segment of the path.

As illustrated in Section 2.3, nodes that are within the path of a segment will forward packets based on the DA of the packet without inspecting the SRH. This ensures full interoperability between SR-capable and non-SR-capable nodes.

4. SRH Procedures

In this section we describe the different procedures on the SRH.

4.1. Segment Routing Node Functions

SR packets are forwarded to segments endpoints (i.e.: the segment endpoint is the node representing the segment and whose address is in the segment list and in the DA of the packet when traveling in the segment). The segment endpoint, when receiving a SR packet destined to itself, does:

- o Inspect the SRH.
- o Determine the next active segment.
- o Update the Segments Left field (or, if requested, remove the SRH from the packet).
- o Update the DA.
- o Forward the packet to the next segment.

The procedures applied to the SRH are related to the node function. Following nodes functions are defined:

Source SR Node.

SR Domain Ingress Node.

Transit Node.

SR Endpoint Node.

4.1.1.1. Source SR Node

A Source SR Node can be any node originating an IPv6 packet with its IPv6 and Segment Routing Headers. This include either:

A host originating an IPv6 packet

A SR domain ingress router encapsulating a received IPv6 packet into an outer IPv6 header followed by a SRH

The mechanism through which a Segment List is derived is outside of the scope of this document. As an example, the Segment List may be obtained through:

Local path computation.

Local configuration.

Interaction with a centralized controller delivering the path.

Any other mechanism.

The following are the steps of the creation of the SRH:

Next Header and Hdr Ext Len fields are set according to [RFC2460].

Routing Type field is set as TBD (SRH).

The Segment List is built with the FIRST segment of the path encoded in the LAST element of the Segment List. Subsequent segments are encoded on top of the first segment. Finally, the LAST segment of the path is encoded in the FIRST element of the Segment List. In other words, the Segment List is encoded in the reverse order of the path.

The original DA of the packet is encoded as the last segment of the path (encoded in the first element of the Segment List).

The DA of the packet is set with the value of the first segment (found in the last element of the segment list).

The Segments Left field is set to n-1 where n is the number of elements in the Segment List.

The First Segment field is set to n-1 where n is the number of elements in the Segment List.

The packet is sent out towards the first segment (i.e.: represented in the packet DA).

HMAC and HMAC Key ID may be set according to Section 5.

4.1.2. SR Domain Ingress Node

The SR Domain Ingress Node is the node where ingress policies are applied and where the packet path (and processing) is determined.

After policies are applied and packet classification is done, the result may be instantiated into a Segment List representing the path the packet should take. In such case, the SR Domain Ingress Node instantiate a new outer IPv6 header to which the SRH is appended (with the computed Segment List). The procedures for the creation and insertion of the new SRH are described in Section 4.1.1.

4.1.3. Transit Node

According to [RFC2460], the only node who is allowed to inspect the Routing Extension Header (and therefore the SRH), is the node corresponding to the DA of the packet. Any other transit node MUST NOT inspect the underneath routing header and MUST forward the packet towards the DA and according to the IPv6 routing table.

In the example case described in Section 2.2.2, when SR capable nodes are connected through an overlay spanning multiple third-party infrastructure, it is safe to send SRH packets (i.e.: packet having a Segment Routing Header) between each other overlay/SR-capable nodes as long as the segment list does not include any of the transit provider nodes. In addition, as a generic security measure, any service provider will block any packet destined to one of its internal routers, especially if these packets have an extended header in it.

4.1.4. SR Segment Endpoint Node

The SR segment endpoint node is the node whose address is in the DA. The segment endpoint node inspects the SRH and does:

1. IF DA = myself (segment endpoint)
2. IF Segments Left > 0 THEN
 decrement Segments Left
 update DA with Segment List[Segments Left]
3. IF Segments Left == 0 THEN
 IF Clean-up bit is set THEN remove the SRH
4. ELSE give the packet to next PID (application)
 End of processing.
5. Forward the packet out

5. Security Considerations

This section analyzes the security threat model, the security issues and mitigation techniques of SRH.

SRH is simply another type of the routing header as described in RFC 2460 [RFC2460] and is:

- o added to a new outer IP header by the ingress router when entering the SR domain or by the originating node itself. The source host can be outside the SR domain;
- o inspected and acted upon when reaching the destination address of the IP header per RFC 2460 [RFC2460].

Per RFC2460 [RFC2460], routers on the path that simply forward an IPv6 packet (i.e. the IPv6 destination address is none of theirs) will never inspect and process the content of any routing header (including SRH). Routers whose one interface IPv6 address equals the destination address field of the IPv6 packet MUST to parse the SRH and, if supported and if the local configuration allows it, MUST act accordingly to the SRH content.

According to RFC2460 [RFC2460], non SR-capable (or non SR-configured) router upon receipt of an IPv6 packet with SRH destined to an address of its:

- o must ignore the SRH completely if the Segment Left field is 0 and proceed to process the next header in the IPv6 packet;
- o must discard the IPv6 packet if Segment Left field is greater than 0 and send a Parameter Problem ICMP message back to the Source Address.

5.1. Threat model

5.1.1. Source routing threats

Using a SRH is a specific case of loose source routing, therefore it has some well-known security issues as described in RFC4942 [RFC4942] section 2.1.1 and RFC5095 [RFC5095]:

- o amplification attacks: where a packet could be forged in such a way to cause looping among a set of SR-enabled routers causing unnecessary traffic, hence a Denial of Service (DoS) against bandwidth;
- o reflection attack: where a hacker could force an intermediate node to appear as the immediate attacker, hence hiding the real attacker from naive forensic;
- o bypass attack: where an intermediate node could be used as a stepping stone (for example in a De-Militarized Zone) to attack another host (for example in the datacenter or any back-end server).

5.1.2. Applicability of RFC 5095 to SRH

First of all, the reader must remember this specific part of section 1 of RFC5095 [RFC5095], "A side effect is that this also eliminates benign RH0 use-cases; however, such applications may be facilitated by future Routing Header specifications.". In short, it is not forbidden to create new secure type of Routing Header; for example, RFC 6554 (RPL) [RFC6554] also creates a new Routing Header type for a specific application confined in a single network.

The main use case for SR consists of the single administrative domain (or cooperating administrative domains) where only trusted nodes with SR enabled and explicitly configured participate in SR: this is the same model as in RFC6554 [RFC6554]. All non-trusted nodes do not participate as either SR processing is not enabled by default or because they only process SRH from nodes within their domain.

Moreover, all SR routers SHOULD ignore SRH created by outsiders based on topology information (received on a peering or internal interface) or on presence and validity of the HMAC field. Therefore, if intermediate SR routers ONLY act on valid and authorized SRH (such as within a single administrative domain), then there is no security threat similar to RH-0. Hence, the RFC 5095 [RFC5095] attacks are not applicable.

5.1.3. Service stealing threat

Segment routing is used for added value services, there is also a need to prevent non-participating nodes to use those services; this is called 'service stealing prevention'.

5.1.4. Topology disclosure

The SRH may also contains IPv6 addresses of some intermediate SR routers in the path towards the destination, this obviously reveals those addresses to the potentially hostile attackers if those attackers are able to intercept packets containing SRH. On the other hand, if the attacker can do a traceroute whose probes will be forwarded along the SR path, then there is little learned by intercepting the SRH itself. The clean-bit of SRH can help by removing the SRH before forwarding the packet to potentially a non-trusted part of the network; if the attacker can force the generation of an ICMP message during the transit in the SR domain, then the ICMP will probably contain the SRH header (totally or partially) depending on the ICMP-generating router behavior.

5.1.5. ICMP Generation

Per section 4.4 of RFC2460 [RFC2460], when destination nodes (i.e. where the destination address is one of theirs) receive a Routing Header with unsupported Routing Type, the required behavior is:

- o If Segments Left is zero, the node must ignore the Routing header and proceed to process the next header in the packet.
- o If Segments Left is non-zero, the node must discard the packet and SHOULD send an ICMP Parameter Problem, Code 0, message to the packet's Source Address, pointing to the unrecognized Routing Type.

This required behavior could be used by an attacker to force the generation of ICMP message by any node. The attacker could send packets with SRH (with Segment Left different than 0) destined to a node not supporting SRH. Per RFC2460 [RFC2460], the destination node must then generate an ICMP message per RFC 2460, causing a local CPU utilization and if the source of the offending packet with SRH was spoofed could lead to a reflection attack without any amplification.

It must be noted that this is a required behavior for any unsupported Routing Type and not limited to SRH packets. So, it is not specific to SRH and the usual rate limiting for ICMP generation is required anyway for any IPv6 implementation and has been implemented and deployed for many years.

5.2. Security fields in SRH

This section summarizes the use of specific fields in the SRH. They are based on a key-hashed message authentication code (HMAC).

The security-related fields in SRH are:

- o HMAC Key-id, 8 bits wide;
- o HMAC, 256 bits wide (optional, exists only if HMAC Key-id is not 0).

The HMAC field is the output of the HMAC computation (per RFC 2104 [RFC2104]) using a pre-shared key and hashing algorithm identified by HMAC Key-id and of the text which consists of the concatenation of:

- o the source IPv6 address;
- o First Segment field;
- o an octet whose bit-0 is the clean-up bit flag and others are 0;
- o HMAC Key-id;
- o all addresses in the Segment List.

The purpose of the HMAC field is to verify the validity, the integrity and the authorization of the SRH itself. If an outsider of the SR domain does not have access to a current pre-shared secret, then it cannot compute the right HMAC field and the first SR router on the path processing the SRH and configured to check the validity of the HMAC will simply reject the packet.

The HMAC field is located at the end of the SRH simply because only the router on the ingress of the SR domain needs to process it, then all other SR nodes can ignore it (based on local policy) because they trust the upstream router. This is to speed up forwarding operations because SR routers which do not validate the SRH do not need to parse the SRH until the end.

The HMAC Key-id field allows for the simultaneous existence of several hash algorithms (SHA-256, SHA3-256 ... or future ones) as well as pre-shared keys. This allows for pre-shared key roll-over when two pre-shared keys are supported for a while when all SR nodes converged to a fresher pre-shared key. The HMAC Key-id field is opaque, i.e., it has neither syntax nor semantic except as an index to the right combination of pre-shared key and hash algorithm and except that a value of 0 means that there is no HMAC field. It could

also allow for interoperation among different SR domains if allowed by local policy and assuming a collision-free Key Id allocation which is out of scope of this memo.

When a specific SRH is linked to a time-related service (such as turbo-QoS for a 1-hour period), then it is important to refresh the shared-secret frequently as the HMAC validity period expires only when the HMAC Key-id and its associated shared-secret expires.

5.2.1. Selecting a hash algorithm

The HMAC field in the SRH is 256 bits wide. Therefore, the HMAC MUST be based on a hash function whose output is at least 256 bits. If the output of the hash function is 256, then this output is simply inserted in the HMAC field. If the output of the hash function is larger than 256 bits, then the output value is truncated to 256 by taking the least-significant 256 bits and inserting them in the HMAC field.

SRH implementations can support multiple hash functions but MUST implement SHA-2 [FIPS180-4] in its SHA-256 variant.

NOTE: SHA-1 is currently used by some early implementations used for quick interoperations testing, the 160-bit hash value must then be right-hand padded with 96 bits set to 0. The authors understand that this is not secure but is ok for limited tests.

5.2.2. Performance impact of HMAC

While adding a HMAC to each and every SR packet increases the security, it has a performance impact. Nevertheless, it must be noted that:

- o the HMAC field SHOULD be used only when SRH is inserted by a device (such as a home set-up box) which is outside of the segment routing domain. If the SRH is added by a router in the trusted segment routing domain, then, there is no need for a HMAC field, hence no performance impact.
- o when present, the HMAC field MUST be checked and validated only by the first router of the segment routing domain, this router is named 'validating SR router'. Downstream routers may not inspect the HMAC field.
- o this validating router can also have a cache of <IPv6 header + SRH, HMAC field value> to improve the performance. It is not the same use case as in IPsec where HMAC value was unique per packet, in SRH, the HMAC value is unique per flow.

- o Last point, hash functions such as SHA-2 have been optimized for security and performance and there are multiple implementations with good performance.

With the above points in mind, the performance impact of using HMAC is minimized.

5.2.3. Pre-shared key management

The field HMAC Key-id allows for:

- o key roll-over: when there is a need to change the key (the hash pre-shared secret), then multiple pre-shared keys can be used simultaneously. The validating routing can have a table of <HMAC Key-id, pre-shared secret, hash algorithm> for the currently active and future keys.
- o different algorithm: by extending the previous table to <HMAC Key-id, hash function, pre-shared secret>, the validating router can also support simultaneously several hash algorithms (see section Section 5.2.1)

The pre-shared secret distribution can be done:

- o in the configuration of the validating routers, either by static configuration or any SDN oriented approach;
- o dynamically using a trusted key distribution such as [RFC6407]

The intent of this document is NOT to define yet-another-key-distribution-protocol.

5.3. Deployment Models

5.3.1. Nodes within the SR domain

The routers inside a SR domain can be trusted to generate the outer IP header and the SRH and to process SRH received on interfaces that are part of the SR domain. These nodes MUST drop all SRH packets received on any interface that is not part of the SR domain and containing a SRH whose HMAC field cannot be validated by local policies. This includes obviously packet with a SRH generated by a non-cooperative SR domain.

If the validation fails, then these packets MUST be dropped, ICMP error messages (parameter problem) SHOULD be generated (but rate limited) and SHOULD be logged.

5.3.2. Nodes outside of the SR domain

Nodes outside of the SR domain cannot be trusted for physical security; hence, they need to obtain by some trusted means (outside of the scope of this document) a complete SRH for each new connection (i.e. new destination address). The received SRH MUST include a HMAC Key-id and HMAC field which has been computed correctly (see Section 5.2).

When a outside the SR domain sends a packet with a SRH and towards a SR domain ingress node, the packet MUST contain the HMAC Key-id and HMAC field and the destination address MUST be an address of a SR domain ingress node .

The ingress SR router, i.e., the router with an interface address equals to the destination address, MUST verify the HMAC field with respect to the HMAC Key-id.

If the validation is successful, then the packet is simply forwarded as usual for a SR packet. As long as the packet travels within the SR domain, no further HMAC check needs to be done. Subsequent routers in the SR domain MAY verify the HMAC field when they process the SRH (i.e. when they are the destination).

If the validation fails, then this packet MUST be dropped, an ICMP error message (parameter problem) SHOULD be generated (but rate limited) and SHOULD be logged.

5.3.3. SR path exposure

As the intermediate SR nodes addresses appears in the SRH, if this SRH is visible to an outsider then he/she could reuse this knowledge to launch an attack on the intermediate SR nodes or get some insider knowledge on the topology. This is especially applicable when the path between the source node and the first SR domain ingress router is on the public Internet.

The first remark is to state that 'security by obscurity' is never enough; in other words, the security policy of the SR domain SHOULD assume that the internal topology and addressing is known by the attacker.

IPsec Encapsulating Security Payload [RFC4303] cannot be use to protect the SRH as per RFC4303 the ESP header must appear after any routing header (including SRH).

When the SRH is not generated by the actual source node but by an SR domain ingress router, it is added after a new outer IP header, this

means that a normal traceroute will not reveal the routers in the SR domain (pretty much like in a MPLS network) and that if ICMP are generated by routers in the SR domain they will be sent to the ingress router of the SR domain without revealing anything to the outside of the SR domain.

To prevent a user to leverage the gained knowledge by intercepting SRH, it is recommended to apply an infrastructure Access Control List (iACL) at the edge of the SR domain. This iACL will drop all packets from outside the SR-domain whose destination is any address of any router inside the domain. This security policy should be tuned for local operations.

5.3.4. Impact of BCP-38

BCP-38 [RFC2827], also known as "Network Ingress Filtering", checks whether the source address of packets received on an interface is valid for this interface. The use of loose source routing such as SRH forces packets to follow a path which differs from the expected routing. Therefore, if BCP-38 was implemented in all routers inside the SR domain, then SR packets could be received by an interface which is not expected one and the packets could be dropped.

As a SR domain is usually a subset of one administrative domain, and as BCP-38 is only deployed at the ingress routers of this administrative domain and as packets arriving at those ingress routers have been normally forwarded using the normal routing information, then there is no reason why this ingress router should drop the SRH packet based on BCP-38. Routers inside the domain commonly do not apply BCP-38; so, this is not a problem.

6. IANA Considerations

TBD but should at least require a new type for routing header

7. Manageability Considerations

TBD should we talk about traceroute? about SRH in ICMP replies?

8. Contributors

The authors would like to thank Dave Barach, John Leddy, John Brzozowski, Pierre Francois, Nagendra Kumar, Mark Townsley, Christian Martin, Roberta Maglione, James Connolly, Aloys Augustin and Fred Baker for their contribution to this document.

9. Acknowledgements

TBD

10. References

10.1. Normative References

[FIPS180-4]

National Institute of Standards and Technology, "FIPS 180-4 Secure Hash Standard (SHS)", March 2012, <<http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf>>.

[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>>.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, DOI 10.17487/RFC2460, December 1998, <<http://www.rfc-editor.org/info/rfc2460>>.

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

[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, <<http://www.rfc-editor.org/info/rfc5095>>.

[RFC6407] Weis, B., Rowles, S., and T. Hardjono, "The Group Domain of Interpretation", RFC 6407, DOI 10.17487/RFC6407, October 2011, <<http://www.rfc-editor.org/info/rfc6407>>.

10.2. Informative References

[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-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-spring-ipv6-use-cases]
Brzozowski, J., Leddy, J., Leung, I., Previdi, S.,
Townsend, W., Martin, C., Filsfils, C., and R. Maglione,
"IPv6 SPRING Use Cases", draft-ietf-spring-ipv6-use-
cases-05 (work in progress), September 2015.
- [I-D.ietf-spring-problem-statement]
Previdi, S., Filsfils, C., Decraene, B., Litkowski, S.,
Horneffer, M., and R. Shakir, "SPRING Problem Statement
and Requirements", draft-ietf-spring-problem-statement-04
(work in progress), April 2015.
- [I-D.ietf-spring-resiliency-use-cases]
Francois, P., Filsfils, C., Decraene, B., and R. Shakir,
"Use-cases for Resiliency in SPRING", draft-ietf-spring-
resiliency-use-cases-01 (work in progress), March 2015.
- [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-05 (work in progress),
September 2015.
- [I-D.ietf-spring-segment-routing-mpls]
Filsfils, C., Previdi, S., Bashandy, A., Decraene, B.,
Litkowski, S., Horneffer, M., Shakir, R., Tantsura, J.,
and E. Crabbe, "Segment Routing with MPLS data plane",
draft-ietf-spring-segment-routing-mpls-01 (work in
progress), May 2015.
- [RFC1940] Estrin, D., Li, T., Rekhter, Y., Varadhan, K., and D.
Zappala, "Source Demand Routing: Packet Format and
Forwarding Specification (Version 1)", RFC 1940,
DOI 10.17487/RFC1940, May 1996,
<<http://www.rfc-editor.org/info/rfc1940>>.
- [RFC2104] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-
Hashing for Message Authentication", RFC 2104,
DOI 10.17487/RFC2104, February 1997,
<<http://www.rfc-editor.org/info/rfc2104>>.

- [RFC2827] Ferguson, P. and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing", BCP 38, RFC 2827, DOI 10.17487/RFC2827, May 2000, <<http://www.rfc-editor.org/info/rfc2827>>.
- [RFC4942] Davies, E., Krishnan, S., and P. Savola, "IPv6 Transition/ Co-existence Security Considerations", RFC 4942, DOI 10.17487/RFC4942, September 2007, <<http://www.rfc-editor.org/info/rfc4942>>.
- [RFC6554] Hui, J., Vasseur, JP., Culler, D., and V. Manral, "An IPv6 Routing Header for Source Routes with the Routing Protocol for Low-Power and Lossy Networks (RPL)", RFC 6554, DOI 10.17487/RFC6554, March 2012, <<http://www.rfc-editor.org/info/rfc6554>>.

Authors' Addresses

Stefano Previdi (editor)
Cisco Systems, Inc.
Via Del Serafico, 200
Rome 00142
Italy

Email: sprevidi@cisco.com

Clarence Filsfils
Cisco Systems, Inc.
Brussels
BE

Email: cfilsfil@cisco.com

Brian Field
Comcast
4100 East Dry Creek Road
Centennial, CO 80122
US

Email: Brian_Field@cable.comcast.com

Ida Leung
Rogers Communications
8200 Dixie Road
Brampton, ON L6T 0C1
CA

Email: Ida.Leung@rci.rogers.com

Jen Linkova
Google
1600 Amphitheatre Parkway
Mountain View, CA 94043
US

Email: furry@google.com

Ebben Aries
Facebook
US

Email: exa@fb.com

Tomoya Kosugi
NTT
3-9-11, Midori-Cho Musashino-Shi,
Tokyo 180-8585
JP

Email: kosugi.tomoya@lab.ntt.co.jp

Eric Vyncke
Cisco Systems, Inc.
De Kleetlaann 6A
Diegem 1831
Belgium

Email: evyncke@cisco.com

David Lebrun
Universite Catholique de Louvain
Place Ste Barbe, 2
Louvain-la-Neuve, 1348
Belgium

Email: david.lebrun@uclouvain.be

6man Group
Internet-Draft
Intended status: Standards Track
Expires: August 29, 2015

E. Vyncke, Ed.
S. Previdi
Cisco Systems, Inc.
D. Lebrun
Universite Catholique de Louvain
February 25, 2015

IPv6 Segment Routing Security Considerations
draft-vyncke-6man-segment-routing-security-02

Abstract

Segment Routing (SR) allows a node to steer a packet through a controlled set of instructions, called segments, by prepending a SR header to the packet. A segment can represent any instruction, topological or service-based. SR allows to enforce a flow through any path (topological, or application/service based) while maintaining per-flow state only at the ingress node to the SR domain.

Segment Routing can be applied to the IPv6 data plane with the addition of a new type of Routing Extension Header. This document analyzes the security aspects of the Segment Routing Extension Header (SRH) and how it is used by SR capable nodes to deliver a secure service.

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

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on August 29, 2015.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
1.1. Segment Routing Documents	3
2. Threat model	3
2.1. Source routing threats	4
2.2. Applicability of RFC 5095 to SRH	4
2.3. Service stealing threat	5
2.4. Topology disclosure	5
2.5. ICMP Generation	5
3. Security fields in SRH	6
3.1. Selecting a hash algorithm	7
3.2. Performance impact of HMAC	7
3.3. Pre-shared key management	8
4. Deployment Models	9
4.1. Nodes within the SR domain	9
4.2. Nodes outside of the SR domain	9
4.3. SR path exposure	10
4.4. Impact of BCP-38	10
5. IANA Considerations	10
6. Manageability Considerations	11
7. Security Considerations	11
8. Acknowledgements	11
9. References	11
9.1. Normative References	11
9.2. Informative References	11
Authors' Addresses	13

1. Introduction

This document analyzes the security threat model, the security issues and proposed solutions related to the new routing header for segment routing with an IPv6 data plane.

The Segment Routing Header (SRH) is simply another type of the routing header as described in RFC 2460 [RFC2460] and is:

- o inserted by a SR edge router when entering the segment routing domain or by the originating host itself. The source host can even be outside the SR domain;
- o inspected and acted upon when reaching the destination address of the IP header per RFC 2460 [RFC2460].

Per RFC2460 [RFC2460], routers on the path that simply forward an IPv6 packet (i.e. the IPv6 destination address is none of theirs) will never inspect and process the content of SRH. Routers whose one interface IPv6 address equals the destination address field of the IPv6 packet MUST to parse the SRH and, if supported and if the local configuration allows it, MUST act accordingly to the SRH content.

According to RFC2460 [RFC2460], the default behavior of a non SR-capable router upon receipt of an IPv6 packet with SRH destined to an address of its, is to:

- o ignore the SRH completely if the Segment Left field is 0 and proceed to process the next header in the IPv6 packet;
- o discard the IPv6 packet if Segment Left field is greater than 0, it MAY send a Parameter Problem ICMP message back to the Source Address.

1.1. Segment Routing Documents

Segment Routing terminology is defined in [I-D.ietf-spring-segment-routing] and in [I-D.ietf-spring-problem-statement]. Segment Routing use cases are described in [I-D.filsfils-spring-segment-routing-use-cases]. Segment Routing protocol extensions are defined in [I-D.ietf-isis-segment-routing-extensions], and [I-D.ietf-ospf-ospfv3-segment-routing-extensions].

Segment Routing IPv6 use cases are described in [I-D.ietf-spring-ipv6-use-cases]. And the IPv6 Segment Routing header is described in [I-D.previdi-6man-segment-routing-header].

2. Threat model

2.1. Source routing threats

Using a SRH is similar to source routing, therefore it has some well-known security issues as described in RFC4942 [RFC4942] section 2.1.1 and RFC5095 [RFC5095]:

- o amplification attacks: where a packet could be forged in such a way to cause looping among a set of SR-enabled routers causing unnecessary traffic, hence a Denial of Service (DoS) against bandwidth;
- o reflection attack: where a hacker could force an intermediate node to appear as the immediate attacker, hence hiding the real attacker from naive forensic;
- o bypass attack: where an intermediate node could be used as a stepping stone (for example in a De-Militarized Zone) to attack another host (for example in the datacenter or any back-end server).

2.2. Applicability of RFC 5095 to SRH

First of all, the reader must remember this specific part of section 1 of RFC5095 [RFC5095], "A side effect is that this also eliminates benign RH0 use-cases; however, such applications may be facilitated by future Routing Header specifications.". In short, it is not forbidden to create new secure type of Routing Header; for example, RFC 6554 (RPL) [RFC6554] also creates a new Routing Header type for a specific application confined in a single network.

In the segment routing architecture described in [I-D.ietf-spring-segment-routing] there are basically two kinds of nodes (routers and hosts):

- o nodes within the SR domain, which is within one single administrative domain, i.e., where all nodes are trusted anyway else the damage caused by those nodes could be worse than amplification attacks: traffic interception, man-in-the-middle attacks, more server DoS by dropping packets, and so on.
- o nodes outside of the SR domain, which is outside of the administrative segment routing domain hence they cannot be trusted because there is no physical security for those nodes, i.e., they can be replaced by hostile nodes or can be coerced in wrong behaviors.

The main use case for SR consists of the single administrative domain where only trusted nodes with SR enabled and configured participate

in SR: this is the same model as in RFC6554 [RFC6554]. All non-trusted nodes do not participate as either SR processing is not enabled by default or because they only process SRH from nodes within their domain.

Moreover, all SR nodes ignore SRH created by outsiders based on topology information (received on a peering or internal interface) or on presence and validity of the HMAC field. Therefore, if intermediate nodes ONLY act on valid and authorized SRH (such as within a single administrative domain), then there is no security threat similar to RH-0. Hence, the RFC 5095 [RFC5095] attacks are not applicable.

2.3. Service stealing threat

Segment routing is used for added value services, there is also a need to prevent non-participating nodes to use those services; this is called 'service stealing prevention'.

2.4. Topology disclosure

The SRH may also contains IPv6 addresses of some intermediate SR-nodes in the path towards the destination, this obviously reveals those addresses to the potentially hostile attackers if those attackers are able to intercept packets containing SRH. On the other hand, if the attacker can do a traceroute whose probes will be forwarded along the SR path, then there is little learned by intercepting the SRH itself. Also the clean-bit of SRH can help by removing the SRH before forwarding the packet to potentially a non-trusted part of the network.

2.5. ICMP Generation

Per section 4.4 of RFC2460 [RFC2460], when destination nodes (i.e. where the destination address is one of theirs) receive a Routing Header with unsupported Routing Type, the required behavior is:

- o If Segments Left is zero, the node must ignore the Routing header and proceed to process the next header in the packet.
- o If Segments Left is non-zero, the node must discard the packet and send an ICMP Parameter Problem, Code 0, message to the packet's Source Address, pointing to the unrecognized Routing Type.

This required behavior could be used by an attacker to force the generation of ICMP message by any node. The attacker could send packets with SRH (with Segment Left set to 0) destined to a node not supporting SRH. Per RFC2460 [RFC2460], the destination node could

generate an ICMP message, causing a local CPU utilization and if the source of the offending packet with SRH was spoofed could lead to a reflection attack without any amplification.

It must be noted that this is a required behavior for any unsupported Routing Type and not limited to SRH packets. So, it is not specific to SRH and the usual rate limiting for ICMP generation is required anyway for any IPv6 implementation and has been implemented and deployed for many years.

3. Security fields in SRH

This section summarizes the use of specific fields in the SRH; they are integral part of [I-D.previdi-6man-segment-routing-header] and they are again described here for reader's sake. They are based on a key-hashed message authentication code (HMAC).

The security-related fields in SRH are:

- o HMAC Key-id, 8 bits wide;
- o HMAC, 256 bits wide (optional, exists only if HMAC Key-id is not 0).

The HMAC field is the output of the HMAC computation (per RFC 2104 [RFC2104]) using a pre-shared key identified by HMAC Key-id and of the text which consists of the concatenation of:

- o the source IPv6 address;
- o First Segment field;
- o an octet whose bit-0 is the clean-up bit flag and others are 0;
- o HMAC Key-id;
- o all addresses in the Segment List.

The purpose of the HMAC field is to verify the validity, the integrity and the authorization of the SRH itself. If an outsider of the SR domain does not have access to a current pre-shared secret, then it cannot compute the right HMAC field and the first SR router on the path processing the SRH and configured to check the validity of the HMAC will simply reject the packet.

The HMAC field is located at the end of the SRH simply because only the router on the ingress of the SR domain needs to process it, then all other SR nodes can ignore it (based on local policy) because they

trust the upstream router. This is to speed up forwarding operations because SR routers which do not validate the SRH do not need to parse the SRH until the end.

The HMAC Key-id field allows for the simultaneous existence of several hash algorithms (SHA-256, SHA3-256 ... or future ones) as well as pre-shared keys. This allows for pre-shared key roll-over when two pre-shared keys are supported for a while when all SR nodes converged to a fresher pre-shared key. The HMAC Key-id field is opaque, i.e., it has neither syntax nor semantic except as an index to the right combination of pre-shared key and hash algorithm and except that a value of 0 means that there is no HMAC field. It could also allow for interoperation among different SR domains if allowed by local policy and assuming a collision-free Key Id allocation.

When a specific SRH is linked to a time-related service (such as turbo-QoS for a 1-hour period) where the DA, Segment ID (SID) are identical, then it is important to refresh the shared-secret frequently as the HMAC validity period expires only when the HMAC Key-id and its associated shared-secret expires.

3.1. Selecting a hash algorithm

The HMAC field in the SRH is 256 bit wide. Therefore, the HMAC MUST be based on a hash function whose output is at least 256 bits. If the output of the hash function is 256, then this output is simply inserted in the HMAC field. If the output of the hash function is larger than 256 bits, then the output value is truncated to 256 by taking the least-significant 256 bits and inserting them in the HMAC field.

SRH implementations can support multiple hash functions but MUST implement SHA-2 [FIPS180-4] in its SHA-256 variant.

NOTE: SHA-1 is currently used by some early implementations used for quick interoperations testing, the 160-bit hash value must then be right-hand padded with 96 bits set to 0. The authors understand that this is not secure but is ok for limited tests.

3.2. Performance impact of HMAC

While adding a HMAC to each and every SR packet increases the security, it has a performance impact. Nevertheless, it must be noted that:

- o the HMAC field is used only when SRH is inserted by a device (such as a home set-up box) which is outside of the segment routing domain. If the SRH is added by a router in the trusted segment

routing domain, then, there is no need for a HMAC field, hence no performance impact.

- o when present, the HMAC field MUST only be checked and validated by the first router of the segment routing domain, this router is named 'validating SR router'. Downstream routers MAY NOT inspect the HMAC field.
- o this validating router can also have a cache of <IPv6 header + SRH, HMAC field value> to improve the performance. It is not the same use case as in IPsec where HMAC value was unique per packet, in SRH, the HMAC value is unique per flow.
- o Last point, hash functions such as SHA-2 have been optimized for security and performance and there are multiple implementations with good performance.

With the above points in mind, the performance impact of using HMAC is minimized.

3.3. Pre-shared key management

The field HMAC Key-id allows for:

- o key roll-over: when there is a need to change the key (the hash pre-shared secret), then multiple pre-shared keys can be used simultaneously. The validating routing can have a table of <HMAC Key-id, pre-shared secret> for the currently active and future keys.
- o different algorithm: by extending the previous table to <HMAC Key-id, hash function, pre-shared secret>, the validating router can also support simultaneously several hash algorithms (see section Section 3.1)

The pre-shared secret distribution can be done:

- o in the configuration of the validating routers, either by static configuration or any SDN oriented approach;
- o dynamically using a trusted key distribution such as [RFC6407]

The intent of this document is NOT to define yet-another-key-distribution-protocol.

4. Deployment Models

4.1. Nodes within the SR domain

A SR domain is defined as a set of interconnected routers where all routers at the perimeter are configured to insert and act on SRH. Some routers inside the SR domain can also act on SRH or simply forward IPv6 packets.

The routers inside a SR domain can be trusted to generate SRH and to process SRH received on interfaces that are part of the SR domain. These nodes MUST drop all SRH packets received on an interface that is not part of the SR domain and containing a SRH whose HMAC field cannot be validated by local policies. This includes obviously packet with a SRH generated by a non-cooperative SR domain.

If the validation fails, then these packets MUST be dropped, ICMP error messages (parameter problem) SHOULD be generated (but rate limited) and SHOULD be logged.

4.2. Nodes outside of the SR domain

Nodes outside of the SR domain cannot be trusted for physical security; hence, they need to request by some trusted means (outside of the scope of this document) a complete SRH for each new connection (i.e. new destination address). The received SRH MUST include a HMAC Key-id and HMAC field which is computed correctly (see Section 3).

When an outside node sends a packet with an SRH and towards a SR domain ingress node, the packet MUST contain the HMAC Key-id and HMAC field and the the destination address MUST be an address of a SR domain ingress node .

The ingress SR router, i.e., the router with an interface address equals to the destination address, MUST verify the HMAC field with respect to the HMAC Key-id.

If the validation is successful, then the packet is simply forwarded as usual for a SR packet. As long as the packet travels within the SR domain, no further HMAC check needs to be done. Subsequent routers in the SR domain MAY verify the HMAC field when they process the SRH (i.e. when they are the destination).

If the validation fails, then this packet MUST be dropped, an ICMP error message (parameter problem) SHOULD be generated (but rate limited) and SHOULD be logged.

4.3. SR path exposure

As the intermediate SR nodes addresses appears in the SRH, if this SRH is visible to an outsider then he/she could reuse this knowledge to launch an attack on the intermediate SR nodes or get some insider knowledge on the topology. This is especially applicable when the path between the source node and the first SR domain ingress router is on the public Internet.

The first remark is to state that 'security by obscurity' is never enough; in other words, the security policy of the SR domain MUST assume that the internal topology and addressing is known by the attacker. A simple traceroute will also give the same information (with even more information as all intermediate nodes between SID will also be exposed). IPsec Encapsulating Security Payload [RFC4303] cannot be use to protect the SRH as per RFC4303 the ESP header must appear after any routing header (including SRH).

To prevent a user to leverage the gained knowledge by intercepting SRH, it is recommended to apply an infrastructure Access Control List (iACL) at the edge of the SR domain. This iACL will drop all packets from outside the SR-domain whose destination is any address of any router inside the domain. This security policy should be tuned for local operations.

4.4. Impact of BCP-38

BCP-38 [RFC2827], also known as "Network Ingress Filtering", checks whether the source address of packets received on an interface is valid for this interface. The use of loose source routing such as SRH forces packets to follow a path which differs from the expected routing. Therefore, if BCP-38 was implemented in all routers inside the SR domain, then SR packets could be received by an interface which is not expected one and the packets could be dropped.

As a SR domain is usually a subset of one administrative domain, and as BCP-38 is only deployed at the ingress routers of this administrative domain and as packets arriving at those ingress routers have been normally forwarded using the normal routing information, then there is no reason why this ingress router should drop the SRH packet based on BCP-38. Routers inside the domain commonly do not apply BCP-38; so, this is not a problem.

5. IANA Considerations

There are no IANA request or impact in this document.

6. Manageability Considerations

TBD

7. Security Considerations

Security mechanisms applied to Segment Routing over IPv6 networks are detailed in Section 3.

8. Acknowledgements

The authors would like to thank Dave Barach and Stewart Bryant for their contributions to this document.

9. References

9.1. Normative References

- [FIPS180-4] National Institute of Standards and Technology, "FIPS 180-4 Secure Hash Standard (SHS)", March 2012, <<http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf>>.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.
- [RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, December 2005.
- [RFC5095] Abley, J., Savola, P., and G. Neville-Neil, "Deprecation of Type 0 Routing Headers in IPv6", RFC 5095, December 2007.
- [RFC6407] Weis, B., Rowles, S., and T. Hardjono, "The Group Domain of Interpretation", RFC 6407, October 2011.

9.2. Informative References

- [I-D.filsfils-spring-segment-routing-use-cases]
Filsfils, C., Francois, P., Previdi, S., Decraene, B., Litkowski, S., Horneffer, M., Milojevic, I., Shakir, R., Ytti, S., Henderickx, W., Tantsura, J., Kini, S., and E. Crabbe, "Segment Routing Use Cases", draft-filsfils-spring-segment-routing-use-cases-01 (work in progress), October 2014.
- [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-03 (work in progress), October 2014.
- [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-02 (work in progress), February 2015.
- [I-D.ietf-spring-ipv6-use-cases]
Brzozowski, J., Leddy, J., Leung, I., Previdi, S., Townsley, W., Martin, C., Filsfils, C., and R. Maglione, "IPv6 SPRING Use Cases", draft-ietf-spring-ipv6-use-cases-03 (work in progress), November 2014.
- [I-D.ietf-spring-problem-statement]
Previdi, S., Filsfils, C., Decraene, B., Litkowski, S., Horneffer, M., and R. Shakir, "SPRING Problem Statement and Requirements", draft-ietf-spring-problem-statement-03 (work in progress), October 2014.
- [I-D.ietf-spring-segment-routing]
Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., Litkowski, S., Horneffer, M., Shakir, R., Tantsura, J., and E. Crabbe, "Segment Routing Architecture", draft-ietf-spring-segment-routing-01 (work in progress), February 2015.
- [I-D.previdi-6man-segment-routing-header]
Previdi, S., Filsfils, C., Field, B., and I. Leung, "IPv6 Segment Routing Header (SRH)", draft-previdi-6man-segment-routing-header-05 (work in progress), January 2015.
- [RFC2104] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, February 1997.

- [RFC2827] Ferguson, P. and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing", BCP 38, RFC 2827, May 2000.
- [RFC4942] Davies, E., Krishnan, S., and P. Savola, "IPv6 Transition/ Co-existence Security Considerations", RFC 4942, September 2007.
- [RFC6554] Hui, J., Vasseur, JP., Culler, D., and V. Manral, "An IPv6 Routing Header for Source Routes with the Routing Protocol for Low-Power and Lossy Networks (RPL)", RFC 6554, March 2012.

Authors' Addresses

Eric Vyncke (editor)
Cisco Systems, Inc.
De Kleetlaann 6A
Diegem 1831
Belgium

Email: evyncke@cisco.com

Stefano Previdi
Cisco Systems, Inc.
Via Del Serafico, 200
Rome 00142
Italy

Email: sprevidi@cisco.com

David Lebrun
Universite Catholique de Louvain
Place Ste Barbe, 2
Louvain-la-Neuve, 1348
Belgium

Email: david.lebrun@uclouvain.be