MPLS Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 3, 2018

S. Bryant, Ed.
Huawei
A. Farrel, Ed.
J. Drake
Juniper Networks
J. Tantsura
Individual

October 30, 2017

MPLS Segment Routing in IP Networks
draft-bryant-mpls-unified-ip-sr-03

Abstract

Segment routing is a source routed forwarding method that allows packets to be steered through a network on paths other than the shortest path derived from the routing protocol. The approach uses information encoded in the packet header to partially or completely specify the route the packet takes through the network, and does not make use of a signaling protocol to pre-install paths in the network.

Two different encapsulations have been defined to enable segment routing in an MPLS network or in an IPv6 network. While acknowledging that there is a strong need to support segment routing in both environments, this document defines a mechanism to carry MPLS segment routing packets encapsulated in UDP. The resulting approach is applicable to both IPv4 and IPv6 networks without the need for any changes to the IP or segment routing specifications.

This document makes no changes to the segment routing architecture and builds on existing protocol mechanisms such as the encapsulation of MPLS within UDP defined in RFC 7510.

No new procedures are introduced, but existing mechanisms are combined to achieve the desired result.

Requirements Language

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

Status of This Memo

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

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

Table of Contents

1. Introduction

Segment routing (SR) [I-D.ietf-spring-segment-routing] is a source routed forwarding method that allows packets to be steered through a network on paths other than the shortest path derived from the routing protocol. SR also allows the packets to be steered through a set of packet processing functions along that path. SR uses information encoded in the packet header to partially or completely specify the route the packet takes through the network and does not make use of a signaling protocol to pre-install paths in the network.

The approach to segment routing in IPv6 networks is known as SRv6 and is described in [I-D.ietf-6man-segment-routing-header]. The mechanism described encodes the segment routing instruction list as an ordered list of 128-bit IPv6 addresses that is carried in a new IPv6 extension header: the Source Routing Header (SRH).

MPLS Segment Routing (MPLS-SR) [I-D.ietf-spring-segment-routing-mpls] encodes the route the packet takes through the network and the instructions to be applied to the packet as it transits the network by imposing a stack of MPLS label stack entries on the packet.

This document describes a method for running SR in IPv4 or IPv6 networks by using an MPLS-SR label stack carried in UDP. No change is made to the MPLS-SR encoding mechanism as described in [I-D.ietf-spring-segment-routing-mpls] where a sequence of 32 bit units, one for each instruction, called the Segment Routing Instruction Stack (SRIS) is used. Each basic unit is encoded as an MPLS label stack entry and the segment routing instructions (i.e., the Segment Identifiers, SIDs) are encoded in the 20 bit MPLS Label fields.

In summary, the processing described in this document is a combination of normal MPLS-over-UDP behavior as described in [RFC7510], MPLS-SR lookup and label-pop behavior as described in [I-D.ietf-spring-segment-routing-mpls], and normal IP forwarding. No new procedures are introduced, but existing mechanisms are combined to achieve the desired result.

The method defined is a complementary way of running SR in an IP network that can be used alongside or interchangeably with that defined in [I-D.ietf-6man-segment-routing-header]. Implementers and deployers should consider the benefits and drawbacks of each method and select the approach most suited to their needs.

2. The MPLS-SR-over-UDP Encoding Stack

The MPLS-SR-over-UDP encoding stack is shown in Figure 1.

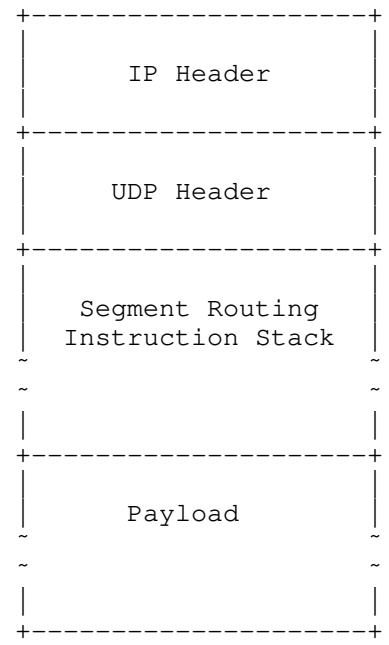


Figure 1: Packet Encapsulation

The payload may be of any type that, with an appropriate convergence layer, can be carried over a packet network. It is anticipated that the most common packet types will be IPv4, IPv6, native MPLS, and pseudowires [RFC3985].

Preceding the Payload is the Segment Routing Instruction Stack (SRIS) that carries the sequence of instructions to be executed on the packet as it traverses the network. This is the Segment Identifier (SID) stack that is the ordered list of segments described in [I-D.ietf-spring-segment-routing].

Preceding the SRIS is a UDP header. The UDP header is included to:

- Introduce entropy to allow equal-cost multi-path load balancing (ECMP) [RFC2992] in the IP layer [RFC7510].
- Provide a protocol multiplexing layer as an alternative to using a new IP type/next header.
- Allow transit through firewalls and other middleboxes.
- Provide disaggregation.

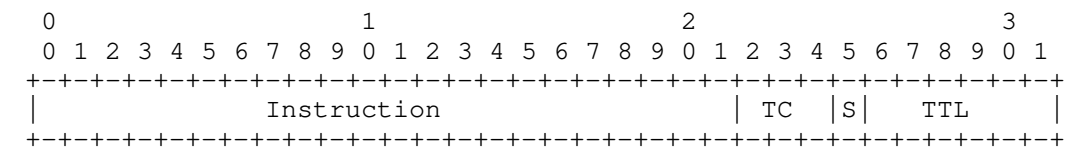
Preceding the UDP header is the IP header which may be IPv4 or IPv6.

3. The Segment Routing Instruction Stack

The Segment Routing Instruction Stack (SRIS) consists of a sequence of Segment Identifiers (SIDs) as described in [I-D.ietf-spring-segment-routing] encoded as an MPLS label stack as described in [I-D.ietf-spring-segment-routing-mpls].

The top SRIS entry is the next instruction to be executed. When the node to which this instruction is directed has processed the instruction it is removed (popped) from the SRIS, and the next instruction is processed.

Each instruction is encoded in a single Label Stack Entry (LSE) as shown in Figure 2. The structure of the LSE is unchanged from [RFC3032].



Instruction: Label Value, 20 bits
TC: Traffic Class, 3 bits
S: Bottom of Stack, 1 bit
TTL: Time to Live, 8 bits

Figure 2: SRIS Label Stack Entry

As with [I-D.ietf-spring-segment-routing-mpls] a 32 bit LSE is used to carry each SR instruction. The instruction itself is carried in the 20 bit Label Value field. The TC field has the normal meaning as

defined in [RFC3032] and modified in [RFC5462]. The S bit has bottom of stack semantics defined in [RFC3032]. TTL is discussed in Section 3.1.

3.1. TTL

The setting of the TTL is application specific, but the following operational consideration should be born in mind. In SR the size of the label stack may be increased within a single routing domain by various operations such as the pushing of a Binding SID. Furthermore, in SR packets are not necessarily constrained to travel on the shortest path within a routing domain. Therefore, considration has to be given to the possibility that there may be a forwarding loop. To mitigate against this it is RECOMMENDED that the TTL is decremented at each hop as the packet passes through the SR network regardless of any other changes to the network layer encapsulation.

Further discussion of the use of TTL during tunnelling can be found in [RFC4023].

4. UDP/IP Encapsulation

[RFC7510] specifies the values to be used in the UDP Source Port, Destination Port, and Checksum fields.

An administrative domain, or set of administrative domains that are sufficiently well managed and monitored to be able to safely use IP segment routing is likely to comply with the requirements called out in [RFC7510] to permit operation with a zero UDP checksum over IP. However each operator needs to validate the decision on whether or not to use a UDP checksum for themselves.

The [RFC7510] UDP header may be carried over IPv4 or over IPv6.

The IP source address is the address of the encapsulating device. The IP destination address is implied by the instruction at the top of the instruction stack.

If IPv4 is in use, fragmentation is not permitted.

5. Elements of Procedure

Nodes that are SR capable can process MPLS-SR packets. Not all of the nodes in an SR domain are SR capable. Some nodes may be "legacy routers" that cannot handle SR packets but can forward IP packets. An SR capable node may advertise its capabilities using the IGP as described in Section 8. There are six types of node in an SR domain:

- Domain ingress nodes that receive packets and encapsulate them for transmission across the domain. Those packets may be any payload protocol including native IP packets or packets that are already MPLS encapsulated.

- Legacy transit nodes that are IP routers but that are not SR capable (i.e., are not able to perform segment routing).

- Transit nodes that are SR capable but that are not identified by a SID in the SID stack.

- Transit nodes that are SR capable and need to perform SR routing because they are identified by a SID in the SID stack.

- The penultimate SR capable node on the path that processes the last SID on the stack on behalf of the domain egress node.

- The domain egress node that forwards the payload packet for ultimate delivery.

The following sub-sections describe the processing behavior in each case.

In summary, the processing is a combination of normal MPLS-over-UDP behavior as described in [RFC7510], MPLS-SR lookup and label-pop behavior as described in [I-D.ietf-spring-segment-routing-mpls], and normal IP forwarding. No new procedures are introduced, but existing mechanisms ae combined to achieve the desired result.

The descriptions in the following sections represent the functional behavior. Optimizations on this behavior may be possible in implementations.

5.1. Domain Ingress Nodes

Domain ingress nodes receive packets from outside the domain and encapsulate them to be forwarded across the domain. Received packets may already be MPLS-SR packets (in the case of connecting two MPLS-SR networks across a native IP network), or may be nativee IP or MPLS packets.

In the latter case, the packet is classified by the domain ingress node and an MPLS-SR stack is imposed. In the former case the MPLS-SR stack is already in the packet. The top entry in the stack is popped from the stack and retained for use below.

The packet is then encapsulated in UDP with the destination port set to 6635 to indicate "MPLS-UDP" or to 6636 to indicate "MPLS-UDP-DTLS"

as described in [RFC7510]. The source UDP port is set randomly or to provide entropy as described in [RFC7510].

The packet is then encapsulated in IP for transmission across the network. The IP source address is set to the domain ingress node, and the destination address is set to the address corresponding to the label that was previously popped from the stack.

This processing is equivalent to sending the packet out of a virtual interface that corresponds to a virtual link between the ingress node and the next hop SR node realized by a UDP tunnel.

The packet is then sent into the IP network and is routed according to the local FIB and applying hashing to resolve any ECMP choices.

5.2. Legacy Transit Nodes

A legacy transit node is an IP router that has no SR capabilities. When such a router receives an MPLS-SR-in-UDP packet it will carry out normal TTL processing and if the packet is still live it will forward it as it would any other UDP-in-IP packet. The packet will be routed toward the destination indicated in the packet header using the local FIB and applying hashing to resolve any ECMP choices.

If the packet is mistakenly addressed to the legacy router, the UDP tunnel will be terminated and the packet will be discarded either because the MPLS-in-UDP port is not supported or because the uncovered top label has not been allocated. This is, however, a misconnection and should not occur unless there is a routing error.

5.3. On-Path Pass-Through SR Nodes

Just because a node is SR capable and receives an MPLS-SR-in-UDP packet does not mean that it performs SR processing on the packet. Only routers identified by SIDs in the SR stack need to do such processing.

Routers that are not addressed by the destination address in the IP header simply treat the packet as a normal UDP-in-IP packet carrying out normal TTL processing and if the packet is still live routing the packet according to the local FIB and applying hashing to resolve any ECMP choices.

This is important because it means that the SR stack can be kept relatively small and the packet can be steered through the network using shortest path first routing between selected SR nodes.

5.4. SR Transit Nodes

An SR capable node that is addressed by the top most SID in the stack when that is not the last SID in the stack (i.e., the S bit is not set) is an SR transit node. When an SR transit node receives an MPLS-SR-in-UDP packet that is addressed to it, it acts as follows:

- Perform TTL processing as normal for an IP packet.
- Determine that the packet is addressed to the local node.
- Find that the payload is UDP and that the destination port indicates MPLS-in-UDP.
- Strip the IP and UDP headers.
- Pop the top label from the SID stack and retain it for use below.
- Encapsulate the packet in UDP with the destination port set to 6635 (or 6636 for DTLS) and the source port set for entropy. The entropy value SHOULD be retained from the received UDP header or MAY be freshly generated since this is a new UDP tunnel.
- Encapsulate the packet in IP with the IP source address set to this transit router, and the destination address set to the address corresponding to the next SID in the stack.
- Send the packet into the IP network routing the packet according to the local FIB and applying hashing to resolve any ECMP choices.

5.5. Penultimate SR Transit Nodes

The penultimate SR transit node is an SR transit node as described in Section 5.4 where the SID for the node is directly followed by the final SID (i.e., that of domain egress node). When a penultimate SR transit node receives an MPLS-SR-in-UDP packet that is addressed to it, it acts according to whether penultimate hop popping (PHP) is supported for the final SID. That information could be indicated using the control plane as described in Section 8. It is worth making some additional observations about PHP in SR: these are collected in Section 6.

If PHP is allowed the penultimate SR transit node acts as follows:

- Perform TTL processing as normal for an IP packet.
- Determine that the packet is addressed to the local node.

- Find that the payload is UDP and that the destination port indicates MPLS-in-UDP.
- Strip the IP and UDP headers.
- Pop the top label from the SID stack and retain it for use below.
- Pop the next label from the SID stack.
- Encapsulate the packet in UDP with the destination port set to 6635 (or 6636 for DTLS) and the source port set for entropy. The entropy value SHOULD be retained from the received UDP header or MAY be freshly generated since this is a new UDP tunnel.
- Encapsulate the packet in IP with the IP source address set to this transit router, and the destination address set to the domain egress node IP address corresponding to the label that was previously popped from the stack.
- Send the packet into the IP network routing the packet according to the local FIB and applying hashing to resolve any ECMP choices.

If PHP is not supported, the penultimate SR transit node just acts as a normal SR transit node just as described in Section 5.4. However, the penultimate SR transit node may be required to replace the final SID with an MPLS-SR label stack entry carrying an explicit null label value (0 for IPv4 and 2 for IPv6) before forwarding the packet. This requirement may also be indicated by the control plane as described in Section 8.

5.6. Domain Egress Nodes

The domain egress acts as follows:

- Perform TTL processing as normal for an IP packet.
- Determine that the packet is addressed to the local node.
- Find that the payload is UDP and that the destination port indicates MPLS-in-UDP.
- Strip the IP and UDP headers.
- Pop the outermost SID if present (i.e., if PHP was not performed as described in Section 5.5.

- Pop the explicit null label if it is present in the label stack as requested by the domain egress and communicated in the control plane as described in Section 8.

- Forward the payload packet according to its type and the local routing/forwarding mechanisms.

6. A Note on Segment Routing Paths and Penultimate Hop Popping

End-to-end SR paths are comprised of multiple segments. The end point of each segment is identified by a SID in the SID stack.

In normal SR processing a penultimate hop is the router that performs SR routing immediately prior to the end of segment router. Penultimate hop popping (PHP) is processing that applies at the penultimate router in a segment.

With MPLS-SR-in-UDP encapsulation, each SR segment is achieved using using an MPLS-in-UDP tunnel that runs the full length of the segment. The SR SID stack on a packet is only examined at the head and tail of this segment. Thus, each segment is effectively one hop long in the SR overlay network and if there is any PHP processing it takes place at the head-end of the segment.

However, in order to simplify processing at each MPLS-SR-in-UDP end point, it is RECOMMENDED that PHP processing is only used for the final segment in an SR path as described in Section 5.5.

7. Modes of Deployment

As previously noted, the procedures described in this document may be used to connect islands of SR functionality across an IP backbone, or can provide SR function within a native IP network. This section briefly expounds upon those two deployment modes.

7.1. Interconnection of SR Domains

Figure 3 shows two SR domains interconnected by an IP network. The procedures described in this document are deployed at border routers R1 and R2 and packets are carried across the backbone network in a UDP tunnel.

R1 acts as the domain ingress as described in Section 5.1. It takes the MPLS-SR packet from the SR domain, pops the top label and uses it to identify its peer border router R2. R1 then encapsulates the packet in UDP in IP and sends it toward R2.

Routers within the IP network simply forward the packet using normal IP routing.

R2 acts as a domain egress router as described in Section 5.6. It receives a packet that is addressed to it, strips the IP and UDP headers, and acts on the payload SR label stack to continue to route the packet.

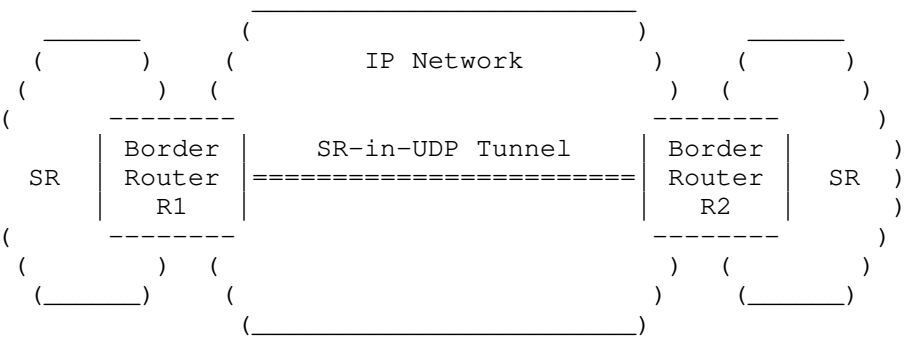


Figure 3: SR in UDP to Tunnel Between SR Sites

7.2. SR Within an IP Network

Figure 4 shows the procedures defined in this document to provide SR function across an IP network.

R1 receives a native packet and classifies it, determining that it should be sent on the SR path R2-R3-R4-R5. It imposes a label stack accordingly and then acts as a domain ingress as described in Section 5.1. It pops the label for R2, and encapsulates the packet in UDP in IP, sets the IP source to R1 and the IP destination to R2, and sends the packet into the IP network.

Routers Ra and Rb are transit routers that simply forward the packets using normal IP forwarding. They may be legacy transit routers (see Section 5.2) or on-path pass-through SR nodes (see Section 5.3).

R2 is an SR transit nodes as described in Section 5.4. It receives a packet addressed to it, strips the IP and UDP headers, and processes the SR label stack. It pops the top label and uses it to identify the next SR hop which is R3. R2 then encapsulates the packet in UDP in IP setting the IP source to R2 and the IP destination to R3.

Rc, Rd, and Re are transit routers and perform as Ra and Rb.

R3 is an SR transit node and performs as R2.

R4 is a penultimate SR transit node as described in Section 5.5. It receives a packet addressed to it, strips the IP and UDP headers, and processes the SR label stack. It pops the top label and uses it to identify the next SR hop which is R5.

R5 is the domain egress as described in Section 5.6. It receives a packet addressed to it, strips the IP and UDP headers.

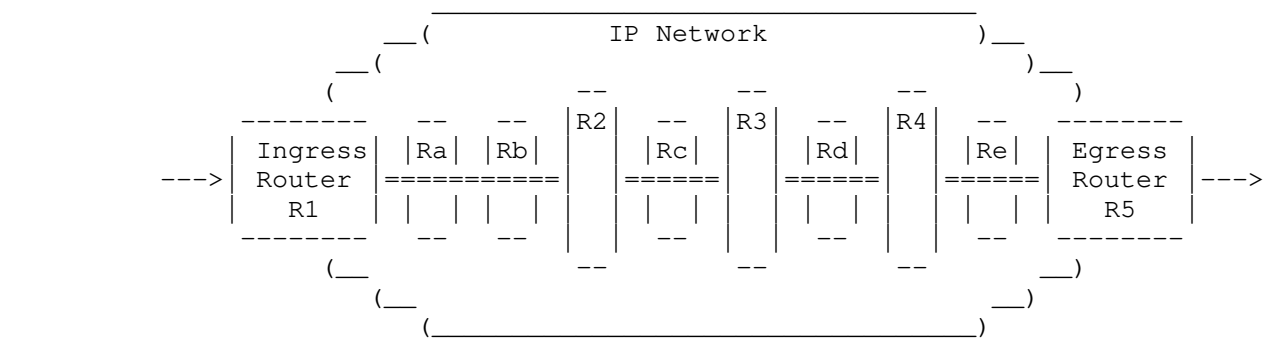


Figure 4: SR Within an IP Network

8. Control Plane

This document is concerned with forwarding plane issues, and a description of applicable control plane mechanisms is out of scope. This section is provided only as a collection of references. No changes to the control plane mechanisms for MPLS-SR are needed or proposed.

A routers that is able to support SR can advertise the fact in the IGP as follows:

- In IS-IS, by using the SR-Capabilities TLV as defined in [I-D.ietf-isis-segment-routing-extensions]
- In OSPF/OSPFv3 by using the Router Information LSA as defined in [I-D.ietf-ospf-segment-routing-extensions] and [I-D.ietf-ospf-ospfv3-segment-routing-extensions].

Nodes can advertise SIDs using the mechanisms defined in [I-D.ietf-isis-segment-routing-extensions], [I-D.ietf-ospf-segment-routing-extensions], or [I-D.ietf-ospf-ospfv3-segment-routing-extensions].

Support for PHP can be indicated in a SID advertisement using flags in the advertisements as follows:

- For IS-IS, the N (no-PHP) flag in the Prefix-SID sub-TLV indicates whether PHP is not to be used.

- For OSPF/OSPFv3, the NP (no-PHP) flag in the Prefix SID Sub-TLV indicates whether PHP is not to be used.

The requirement to use an explicit null SID if PHP is not in use can be indicated in SID advertisement using the Explicit-Null Flag (E-Flag). If set, the penultimate SR transit node replaces the final SID with a SID containing an Explicit-NULL value (0 for IPv4 and 2 for IPv6) before forwarding the packet.

The method of advertising the tunnel encapsulation capability of a router using IS-IS or OSPF are specified in [I-D.ietf-isis-encapsulation-cap] and [I-D.ietf-ospf-encapsulation-cap] respectively. No changes to those procedures are needed in support of this work.

9. OAM

OAM at the payload layer follows the normal OAM procedures for the payload. To the payload the whole SR network looks like a tunnel.

OAM in the IP domain follows the normal IP procedures. This can only be carried out between on the IP hops between pairs of SR nodes.

OAM between instruction processing entities i.e., at the SR layer uses the procedures documented for MPLS.

10. Security Considerations

The security consideration of [I-D.ietf-spring-ipv6-use-cases] and [RFC7510] apply. DTLS [RFC6347] SHOULD be used where security is needed on an MPLS-SR-over-UDP segment.

It is difficult for an attacker to pass a raw MPLS encoded packet into a network and operators have considerable experience at excluding such packets at the network boundaries.

It is easy for an ingress node to detect any attempt to smuggle IP packet into the network since it would see that the UDP destination port was set to MPLS. SR packets not having a destination address terminating in the network would be transparently carried and would pose no security risk to the network under consideration.

11. IANA Considerations

This document makes no IANA requests.

12. Acknowledgements

This draft was partly inspired by [I-D.xu-mpls-unified-source-routing-instruction], and we acknowledge the following authors of version -02 of that draft: Robert Raszuk, Uma Chunduri, Luis M. Contreras, Luay Jalil, Hamid Assarpour, Gunter Van De Velde, Jeff Tantsura, and Shaowen Ma.

Thanks to Joel Halpern, Bruno Decraene, Loa Andersson, Ron Bonica, Eric Rosen, Robert Raszuk, Wim Henderickx, Jim Guichard, and Gunter Van De Velde for their insightful comments on this draft.

13. Contributors

o Mach Chen, Huawei Technologies, mach.chen@huawei.com

14. References

14.1. Normative References

[I-D.ietf-spring-segment-routing]
Filsfils, C., Previdi, S., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", draft-ietf-spring-segment-routing-13 (work in progress), October 2017.

[I-D.ietf-spring-segment-routing-mpls]
Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls-10 (work in progress), June 2017.

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

[RFC3032] Rosen, E., Tappan, D., Fedorkow, G., Rekhter, Y., Farinacci, D., Li, T., and A. Conta, "MPLS Label Stack Encoding", RFC 3032, DOI 10.17487/RFC3032, January 2001, <https://www.rfc-editor.org/info/rfc3032>.

[RFC5462] Andersson, L. and R. Asati, "Multiprotocol Label Switching (MPLS) Label Stack Entry: "EXP" Field Renamed to "Traffic Class" Field", RFC 5462, DOI 10.17487/RFC5462, February 2009, <https://www.rfc-editor.org/info/rfc5462>.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <https://www.rfc-editor.org/info/rfc6347>.

[RFC7510] Xu, X., Sheth, N., Yong, L., Callon, R., and D. Black, "Encapsulating MPLS in UDP", RFC 7510, DOI 10.17487/RFC7510, April 2015, <https://www.rfc-editor.org/info/rfc7510>.

14.2. Informative References

[I-D.ietf-6man-segment-routing-header]
Previdi, S., Filsfils, C., Raza, K., Leddy, J., Field, B., daniel.voyer@bell.ca, d., daniel.bernier@bell.ca, d., Matsushima, S., Leung, I., Linkova, J., Aries, E., Kosugi, T., Vyncke, E., Lebrun, D., Steinberg, D., and R. Raszuk, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-07 (work in progress), July 2017.

[I-D.ietf-isis-encapsulation-cap]
Xu, X., Decraene, B., Raszuk, R., Chunduri, U., Contreras, L., and L. Jalil, "Advertising Tunnelling Capability in IS-IS", draft-ietf-isis-encapsulation-cap-01 (work in progress), April 2017.

[I-D.ietf-isis-segment-routing-extensions]
Previdi, S., Filsfils, C., Bashandy, A., Gredler, H., Litkowski, S., Decraene, B., and j. jefftant@gmail.com, "IS-IS Extensions for Segment Routing", draft-ietf-isis-segment-routing-extensions-13 (work in progress), June 2017.

[I-D.ietf-ospf-encapsulation-cap]
Xu, X., Decraene, B., Raszuk, R., Contreras, L., and L. Jalil, "The Tunnel Encapsulations OSPF Router Information", draft-ietf-ospf-encapsulation-cap-09 (work in progress), October 2017.

[I-D.ietf-ospf-ospfv3-segment-routing-extensions]
Psenak, P., Previdi, S., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPFv3 Extensions for Segment Routing", draft-ietf-ospf-ospfv3-segment-routing-extensions-10 (work in progress), September 2017.

[I-D.ietf-ospf-segment-routing-extensions]
Psenak, P., Previdi, S., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPF Extensions for Segment Routing", draft-ietf-ospf-segment-routing-extensions-21 (work in progress), October 2017.

[I-D.ietf-spring-ipv6-use-cases]
Brzozowski, J., Leddy, J., Filsfils, C., Maglione, R., and M. Townsley, "IPv6 SPRING Use Cases", draft-ietf-spring-ipv6-use-cases-11 (work in progress), June 2017.

[I-D.xu-mpls-unified-source-routing-instruction]
Xu, X., Bashandy, A., Assarpour, H., Ma, S., Henderickx, W., and j. jefftant@gmail.com, "Unified Source Routing Instructions using MPLS Label Stack", draft-xu-mpls-unified-source-routing-instruction-04 (work in progress), September 2017.

[RFC2992] Hopps, C., "Analysis of an Equal-Cost Multi-Path Algorithm", RFC 2992, DOI 10.17487/RFC2992, November 2000, <https://www.rfc-editor.org/info/rfc2992>.

[RFC3985] Bryant, S., Ed. and P. Pate, Ed., "Pseudo Wire Emulation Edge-to-Edge (PWE3) Architecture", RFC 3985, DOI 10.17487/RFC3985, March 2005, <https://www.rfc-editor.org/info/rfc3985>.

[RFC4023] Worster, T., Rekhter, Y., and E. Rosen, Ed., "Encapsulating MPLS in IP or Generic Routing Encapsulation (GRE)", RFC 4023, DOI 10.17487/RFC4023, March 2005, <https://www.rfc-editor.org/info/rfc4023>.

Authors' Addresses

Stewart Bryant (editor)
Huawei

Email: stewart.bryant@gmail.com

Adrian Farrel (editor)
Juniper Networks

Email: afarrel@juniper.net

John Drake
Juniper Networks

Email: jdrake@juniper.net

Jeff Tantsura
Individual

Email: jefftant.ietf@gmail.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 3, 2018

W. Cheng
L. Wang
H. Li
China Mobile
M. Chen
Huawei
R. Zigler
Broadcom
S. Zhan
ZTE

October 30, 2017

Path Segment in MPLS Based Sement Routing Network
draft-cheng-spring-mpls-path-segment-00

Abstract

An SR path is identified by an SR segment list, one or partial segments of the list cannot uniquely identify the SR path.

This document introduces the concept of Path Segment that is used to identify an SR path. When used, it is inserted at the ingress node of the SR path and immediately follows the last segment of the SR path. The Path Segment will not be popped off until it reaches the egress of the SR path, it can be used by the egress node to implement end-2-end SR path protection or performance measurement (PM) of an SR path.

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 https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any

time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

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

Table of Contents

1. Introduction

Segment Routing (SR) [I-D.ietf-spring-segment-routing] is a source routed forwarding method that allows to directly encode forwarding instructions (called segments) in each packet, hence it enables to steer traffic through a network without the per-flow states maintained in the transit nodes. Segment Routing can be instantiated on MPLS data plane or IPv6 data plane. The former is called SR-MPLS

[I-D.ietf-spring-segment-routing-mpls], the latter is called SRv6 [I-D.ietf-6man-segment-routing-header]. SR-MPLS leverages the MPLS label stack to construct SR path, and SRv6 uses the Segment Routing Header to construct SR path.

In an SR-MPLS network, when a packet is transmitted along an SR path, the labels in the MPLS label stack will be swapped or popped. So that no label or only the last label may be left in the MPLS label stack when the packet reaches the egress node. Thus, the egress node cannot determine from which ingress node or SR path the packet comes.

However, to support use cases like end-2-end 1+1 path protection, bidirectional path correlation or performance measurement(PM), the ability to implement path identification is the pre-condition.

Therefore, this document introduces a new segment that is referred to as Path Segment. A Path Segment is defined to unique identify an SR path in a specific context. (e.g., in the context of the egress node or ingress node of an SR path, or within an SR domain). It is normally used by egress nodes for path identification or correlation. Path Segment can only apply to SR-MPLS.

2. Path Segment

This document introduces two options for SR path identification: one label solution and two labels solution.

[Editor notes: it is supposed that the WG will discuss and decide which one is the better solution.]

2.1. One Label Solution

The Path Segment is a single label that is assigned from the Segment Routing Local Block (SRLB) or Segment Routing Global Block (SRGB) of the egress node of an SR path. It means that the Path Segment is unique in the context of the egress node of SR paths. When Path Segment is used, a Path label MUST be inserted at the ingress node and MUST immediately follow the last label of the SR path.

If the Path label is the bottom label, the S bit MUST be set. The value of the TC field MUST be set to the same value as the last segment label of the SR path. The value of the TTL field MUST be set to the same value of the last segment label of the SR path.

Normally, the intermediates node will not see the Path Segment label and do not know how to process it even if they see it. A Path Segment label presenting to an intermediate node is error situation.

The egress node MUST pop the Path label and deliver it to relevant components for further processing.

The label stack with Path Segment is as below (Figure1):

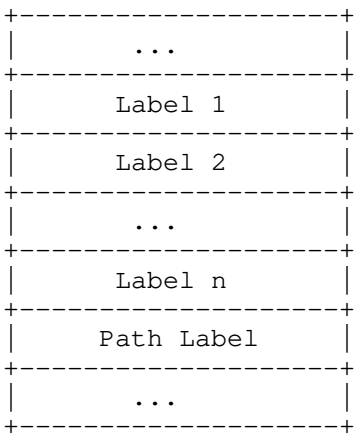


Figure 1: Label Stack with Path Segment

Where:

- The Label 1-n are the segment labels that are used to direct how to steer the packets along the SR path.
- The Path Label identifies the SR path in the context of the egress node of the SR path.

2.1.1. Path Segment Assignment

Several ways can be used to assign the Path Segment. One way is that the Path Segment label is directly assigned by the egress node of an SR path. Where the ingress node of the SR path can directly send a request to the egress node to ask for a Path label. With this way, it needs to set up a communication channel between the ingress node and the egress node. New protocols or extensions to existing protocol may be required.

Another candidate way is to leverage a centralized controller (e.g., PCE) to assign the Path label. The ingress node sends a request to the PCE to compute a SR path and indicate that a Path label is desired. The PCE will compute the path as required. Once the path computed, the PCE will send a request (with computed path and relevant information) to the egress node to ask for a Path label for

the SR path. The egress node will allocate a label to the SR path and build mapping relationship between the label and the path. A reply will be sent back to the PCE, the PCE will send a reply to the ingress node about the path information and the corresponding Path Segment label.

With either way or the variations, the final purpose is to assign a label from the egress node's label space, hence a single label is enough for path identification. Then the ingress node can put the Path Segment label into the label stack when needed, and the egress node can use that Path Segment to implement relevant functionalities.

2.2. Two Labels Solution

Two segments (Source segment and Path segment) are used to identify an SR path. The Source segment is a global node segment, it can uniquely identify a node within an SR domain. It MUST NOT be used for forwarding and indicates that a Path segment immediately follows. The Path segment is a local segment generated at the ingress node to identify an SR path. The combination of Source segment and Path segment can uniquely identify an SR Path with an SR domain.

A node that enables Path segment function will be assigned two node segments. One is for forwarding just as defined in [I-D.ietf-spring-segment-routing], the other is for source identification. The corresponding label of the Source Segment is indexed in the SRGB (or in a of the node to which the Source Segment will be presented.

The Path segment label is a local label that is assigned to an SR path at the ingress node.

The label stack with Source and Path segments is as below (Figure 2):

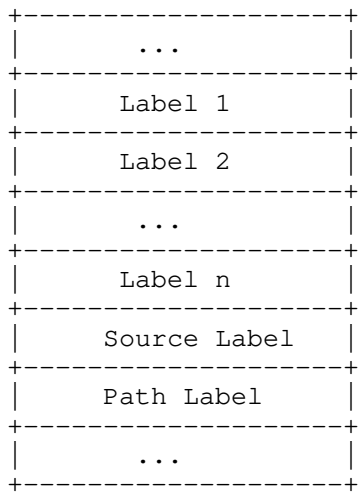


Figure 2: Label Stack with Source and Path Segments

Where:

- The Label 1-n are the segment labels that are used to direct how to steer the packets along an SR path, and the "label n" is the last label of the SR path or the label that directs forwarding packets to the node to which the Source Segment will be presented.

- The Source Label identifies the source of the SR path. The value of the TC and TTL fields of the Source Label MUST be set to the same values as the label (e.g., the Label n) it follows.

- The Path Label identifies the SR path in the context of source node. If the Path label is the bottom label, the S bit MUST be set. The value of the TC and TTL fields SHOULD be set to the same values as the Source label.

The Source and Path label MUST be inserted at the ingress node of an SR path. And they MUST immediately follow the label that directs forwarding packets to the node (e.g., the egress or an intermediate node) to which the Source Segment (as the stack top label) and Path Segment are presented.

If a node receives a packet with an unknown Source Label, the packet MUST be discarded and an error SHOULD be reported.

The Source label and Path label MUST be popped at the node who receives a packet with the Source label as the stack top label.

3. Path Segment Application

3.1. Performance Measurement

To measure the packet loss and delay of the real traffic of an SR path, one fundamental condition is path identification at the measuring points. For an SR path, the ingress node have the complete information of the path, it can use those information for packet counting and/or timestamping. At the egress node, since the Path Segment label (or combination with Source label) can be used to identify the path, path based packet counting and/or timestamping can be implemented as well. Then combined with the mechanisms defined [RFC6374], end-2-end packet loss and/or delay measurement of an SR path can be achieved.

Measuring at intermediate nodes needs more consideration, it will be added in the next version.

3.2. End-2-end Path Protection

For end-2-end 1+1 path protection, the egress node of an path needs to know the set of paths that constitute the primary and the backup(s), in order to select the primary packet for onward transmission, and to discard the packets from the backups.

To do this each path needs a path identifier that is unique at the egress node. Depending on the design, this single unique label chosen by the egress PE or the combination of the source node identifier and a unique path identifier chosen by the source.

There then needs to be a method of binding this path identifiers into equivalence groups such that the egress PE can determine the set of packets that represent a single path and its backup.

It is obvious that this group can be instantiated in the network by an SDN controller.

In a network that is using a distributed control plane the approach will depend on the control protocol used, but the essence of the solution is that which ever PE is responsible for creating the group advertises then as a group of equivalent paths. Whether one of these is advertised as primary and the others as secondary will or all are advertised as of equal status will depend on the details of the underlying protection mechanism.

3.3. Bi-directional SR Tunnel

With the current SR architecture, an SR path is an unidirectional path. In some scenarios, for example, mobile backhaul transport network, there are requirements to support bi-directional path, and the path is normally treated as a single entity and both directions of the path have same fate, for example, failure in one direction will result in switching at both directions.

MPLS supports this by introducing the concepts of co-routed bidirectional LSP and associated bi-directional LSP. With SR, to support bidirectional path, a straightforward way is to bind two unidirectional SR paths to a single bi-directional path. Path segments can be used to correlate the two unidirectional SR paths at both ends of the paths.

4. IANA Considerations

This document makes no request of IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

5. Security Considerations

6. Contributors

The following individuals also contribute to this document.

- Shuangping Zhan, ZTE
- Cheng Li, Huawei

7. Acknowledgements

The authors would like to thank Stewart Bryant for his review, suggestion and comments to this document.

8. References

8.1. Normative References

[I-D.ietf-6man-segment-routing-header]
Previdi, S., Filsfils, C., Raza, K., Leddy, J., Field, B., daniel.voyer@bell.ca, d., daniel.bernier@bell.ca, d., Matsushima, S., Leung, I., Linkova, J., Aries, E., Kosugi, T., Vyncke, E., Lebrun, D., Steinberg, D., and R. Raszuk, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-07 (work in progress), July 2017.

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

8.2. Informative References

[I-D.ietf-spring-segment-routing]
Filsfils, C., Previdi, S., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", draft-ietf-spring-segment-routing-13 (work in progress), October 2017.

[I-D.ietf-spring-segment-routing-mpls]
Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls-10 (work in progress), June 2017.

[RFC6374] Frost, D. and S. Bryant, "Packet Loss and Delay Measurement for MPLS Networks", RFC 6374, DOI 10.17487/RFC6374, September 2011, <https://www.rfc-editor.org/info/rfc6374>.

Authors' Addresses

Weiqiang Cheng
China Mobile

Email: chengweiqiang@chinamobile.com

Lei Wang
China Mobile

Email: wangleiyj@chinamobile.com

Han Li
China Mobile

Email: lihan@chinamobile.com

Mach(Guoyi) Chen
Huawei

Email: mach.chen@huawei.com

Royi Zigler
Broadcom

Email: royi.zigler@broadcom.com

Shuangping Zhan
ZTE

Email: zhan.shuangping@zte.com.cn

SPRING Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 19, 2019

W. Cheng
L. Wang
H. Li
China Mobile
M. Chen
Huawei
R. Gandhi
Cisco Systems, Inc.
R. Zigler
Broadcom
S. Zhan
ZTE

October 16, 2018

Path Segment in MPLS Based Segment Routing Network
draft-cheng-spring-mpls-path-segment-03

Abstract

A Segment Routing (SR) path is identified by an SR segment list, one or partial segments of the list cannot uniquely identify the SR path. Path identification is a pre-requisite for various use-cases such as performance measurement (PM) of an SR path.

This document defines a new type of segment that is referred to as Path Segment, which is used to identify an SR path. When used, it is inserted at the ingress node of the SR path and immediately follows the last segment of the SR path. The Path Segment will not be popped off until it reaches the egress node of the SR path.

Path Segment can be used by the egress node to implement path identification hence to support various use-cases including SR path PM, end-to-end 1+1 SR path protection and bidirectional SR paths correlation.

Status of This Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any

time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 19, 2019.

Table of Contents

1. Introduction

Segment Routing (SR) [RFC8402] is a source routed forwarding method that allows to directly encode forwarding instructions (called segments) in each packet, hence it enables to steer traffic through a network without the per-flow states maintained on the transit nodes. Segment Routing can be instantiated on MPLS data plane or IPv6 data plane. The former is called SR-MPLS, the latter is called SRv6

[RFC8402]. SR-MPLS leverages the MPLS label stack to construct SR path, and SRv6 uses the a new IPv6 Extension Header (EH) called the IPv6 Segment Routing Header (SRH) [I-D.ietf-6man-segment-routing-header] to construct SR path.

In an SR-MPLS network, when a packet is transmitted along an SR path, the labels in the MPLS label stack will be swapped or popped. So that no label or only the last label may be left in the MPLS label stack when the packet reaches the egress node. Thus, the egress node cannot determine from which SR path the packet comes.

However, to support use cases like end-to-end 1+1 path protection (Live-Live case), bidirectional path correlation or performance measurement (PM), the ability to implement path identification is a pre-requisite.

Therefore, this document introduces a new segment that is referred to as Path Segment. A Path Segment is defined to uniquely identify an SR path in the context of the egress node. It is normally used by egress nodes for path identification or correlation.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP14 [RFC2119][RFC8174] when, and only when, they appear in all capitals, as shown here.

1.2. Abbreviations

DM: Delay Measurement.

LM: Loss Measurement.

MPLS: Multiprotocol Label Switching.

PM: Performance Measurement.

PSID: Path Segment ID.

SID: Segment ID.

SL: Segment List.

SR: Segment Routing.

SR-MPLS: Segment Routing instantiated on MPLS data plane.

SRv6: Segment Routing instantiated on IPv6 data plane

2. Path Segment

A Path Segment is a single label that is assigned from the Segment Routing Local Block (SRLB) or Segment Routing Global Block (SRGB) of the egress node of an SR path. It means that the Path Segment is unique in the context of the egress node of the SR path. When Path Segment is used, the Path Segment MUST be inserted at the ingress node and MUST immediately follow the last label of the SR path. The Path Segment may be used to identify an SR-MPLS Policy, its Candidate-Path (CP) or a SID List (SL) [I-D.ietf-spring-segment-routing-policy] terminating on an egress node depending on the use-case.

The value of the TTL field of the Path Segment MUST be set to the same value of the last segment label of the SR path. If the Path Segment is the bottom label, the S bit MUST be set.

Normally, the intermediate nodes will not see the Path Segment label and do not know how to process it. A Path Segment presenting to an intermediate node is an error condition.

The egress node MUST pop the Path Segment. The egress node MAY use the Path Segment for further processing. For example, when performance measurement is enabled on the SR path, it can trigger packet counting or timestamping.

The label stack with Path Segment is as below (Figure1):

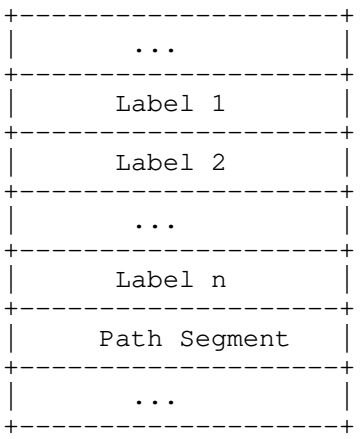


Figure 1: Label Stack with Path Segment

Where:

- The Labels 1 to n are the segment label stack used to direct how to steer the packets along the SR path.

- The Path Segment identifies the SR path in the context of the egress node of the SR path.

3. Nesting of Path Segments

Binding SID (BSID) [RFC8402] can be used for SID list compression. With BSID, an end-to-end SR path can be split into several sub-paths, each sub-path is identified by a BSID. Then an end-to-end SR path can be identified by a list of BSIDs, therefore, it can provide better scalability.

BSID and Path SID (PSID) can be combined to achieve both sub-path and end-to-end path monitoring. A reference model for such a combination in (Figure 2) shows an end-to-end path (A->D) that spans three domains (Access, Aggregation and Core domain) and consists of three sub-paths, one in each sub-domain (sub-path (A->B), sub-path (B->C) and sub-path (C->D)). Each sub-path is allocated a BSID. For nesting the sub-paths, each sub-path is allocated a PSID. Then, the SID list of the end-to-end path can be expressed as <BSID1, BSID2, ..., BSIDn, e-PSID>, where the e-PSID is the PSID of the end-to-end path. The SID list of a sub-path can be expressed as <SID1, SID2, ...SIDn, s-PSID>, where the s-PSID is the PSID of the sub-path.

Figure 2 shows the details of the label stacks when PSID and BSID are used to support both sub-path and end-to-end path monitoring in a multi-domain scenario.

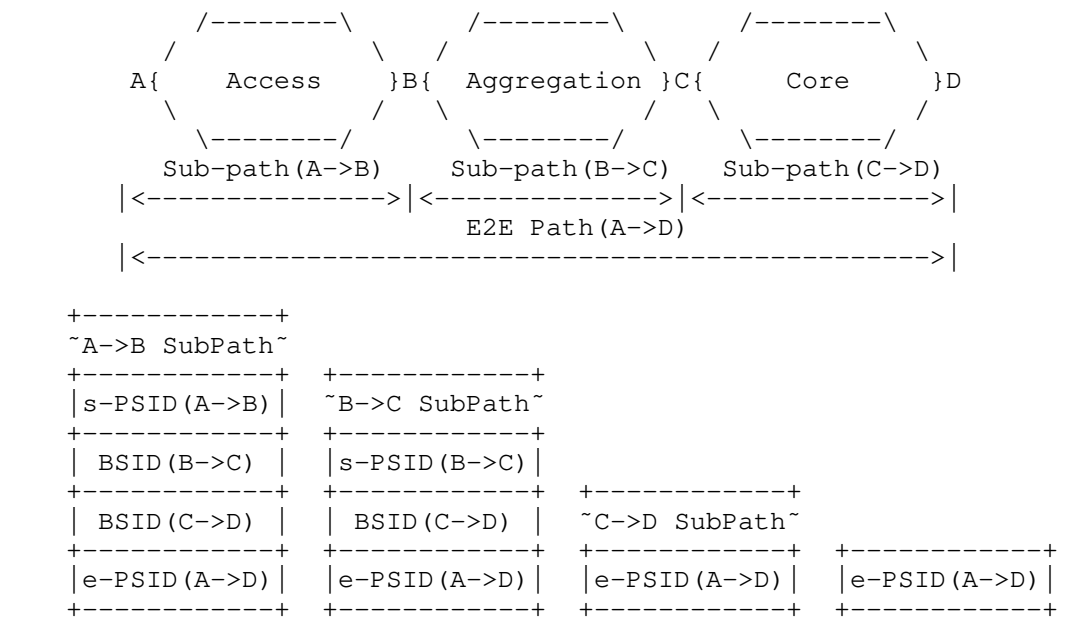


Figure 2: Nesting of Path Segments

4. Path Segment Allocation

Several ways can be used to allocate the Path Segment.

One way is to set up a communication channel (e.g., MPLS Generic Associated Channel (G-ACh)) between the ingress node and the egress node, and the ingress node of the SR path can directly send a request to the egress node to ask for a Path Segment.

Another way is to leverage a centralized controller (e.g., PCE, SDN controller) to assign the Path Segment. PCEP based Path Segment allocation is defined in [I-D.li-pce-sr-path-segment], and SR-policy based path segment allocation is defined in [I-D.li-idr-sr-policy-path-segment-distribution].

5. Path Segment for PM

As defined in [RFC7799], performance measurement can be classified into Active, Passive and Hybrid measurement. For Passive measurement, path identification at the measuring points is the pre-requisite. Path segment can be used by the measuring points (e.g., the ingress/egress nodes of an SR path) or a centralized controller to correlate the packets counts/timestamps that are from the ingress

and egress nodes to a specific SR path, then packet loss/delay can be calculated.

Performance Delay Measurement (DM) and Loss Measurements (LM) in SR networks with MPLS data plane can be found in [I-D.gandhi-spring-sr-mpls-pm] and [I-D.gandhi-spring-udp-pm].

6. Path Segment for Bi-directional SR Path

With the current SR architecture, an SR path is a unidirectional path. In some scenarios, for example, mobile backhaul transport network, there are requirements to support bidirectional path, and the path is normally treated as a single entity and both directions of the path have the same fate, for example, failure in one direction will result in switching at both directions.

MPLS supports this by introducing the concepts of co-routed bidirectional LSP and associated bidirectional LSP. With SR, to support bidirectional path, a straightforward way is to bind two unidirectional SR paths to a single bidirectional path. Path segments can be used to correlate the two unidirectional SR paths at both ends of the paths.

[I-D.li-pce-sr-bidir-path] defines how to use PCEP and Path segment to initiate a bidirectional SR path, and [I-D.li-idr-sr-policy-path-segment-distribution] defines how to use SR policy and Path segment to initiate a bidirectional SR path.

7. Path Segment for End-to-end Path Protection

For end-to-end 1+1 path protection (i.e., Live-Live case), the egress node of an SR path needs to know the set of paths that constitute the primary and the secondary(s), in order to select the primary packet for onward transmission, and to discard the packets from the secondary(s).

To do this, each path needs a path identifier that is unique at the egress node. Depending on the design, this is a single unique path segment label chosen by the egress PE.

There then needs to be a method of binding this path identifiers into equivalence groups such that the egress PE can determine the set of packets that represent a single path and its secondary.

It is obvious that this group can be instantiated in the network by an SDN controller.

8. IANA Considerations

This document does not require any IANA actions.

9. Security Considerations

This document does not introduce additional security requirements and mechanisms other than the ones described in [RFC8402].

10. Contributors

The following individuals also contribute to this document.

- Cheng Li, Huawei

11. Acknowledgements

The authors would like to thank Stewart Bryant, Alexander Vainshtein, Andrew G. Malis and Loa Andersson for their review, suggestions and comments to this document.

The authors would like to acknowledge the contribution from Alexander Vainshtein on "Nesting of Path Segments".

12. References

12.1. Normative References

[I-D.ietf-spring-segment-routing-mpls]
Bashandy, A., Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls-14 (work in progress), June 2018.

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

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <https://www.rfc-editor.org/info/rfc8402>.

12.2. Informative References

[I-D.gandhi-spring-sr-mpls-pm]
Gandhi, R., Filsfils, C., daniel.voyer@bell.ca, d., Salsano, S., Ventre, P., and M. Chen, "Performance Measurement in Segment Routing Networks with MPLS Data Plane", draft-gandhi-spring-sr-mpls-pm-03 (work in progress), September 2018.

[I-D.gandhi-spring-udp-pm]
Gandhi, R., Filsfils, C., daniel.voyer@bell.ca, d., Salsano, S., Ventre, P., and M. Chen, "UDP Path for In-band Performance Measurement for Segment Routing Networks", draft-gandhi-spring-udp-pm-02 (work in progress), September 2018.

[I-D.ietf-6man-segment-routing-header]
Filsfils, C., Previdi, S., Leddy, J., Matsushima, S., and d. daniel.voyer@bell.ca, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-14 (work in progress), June 2018.

[I-D.ietf-spring-segment-routing-policy]
Filsfils, C., Sivabalan, S., daniel.voyer@bell.ca, d., bogdanov@google.com, b., and P. Mattes, "Segment Routing Policy Architecture", draft-ietf-spring-segment-routing-policy-01 (work in progress), June 2018.

[I-D.li-idr-sr-policy-path-segment-distribution]
Li, C., Chen, M., Dong, J., and Z. Li, "Segment Routing Policies for Path Segment and Bi-directional Path", draft-li-idr-sr-policy-path-segment-distribution-00 (work in progress), April 2018.

[I-D.li-pce-sr-bidir-path]
Li, C., Chen, M., Dhody, D., Cheng, W., Li, Z., Dong, J., and R. Gandhi, "PCEP Extension for Segment Routing (SR) Bi-directional Associated Paths", draft-li-pce-sr-bidir-path-01 (work in progress), September 2018.

[I-D.li-pce-sr-path-segment]
Li, C., Chen, M., Dhody, D., Cheng, W., Dong, J., Li, Z., and R. Gandhi, "Path Computation Element Communication Protocol (PCEP) Extension for Path Identification in Segment Routing (SR)", draft-li-pce-sr-path-segment-02 (work in progress), September 2018.

[RFC6374] Frost, D. and S. Bryant, "Packet Loss and Delay Measurement for MPLS Networks", RFC 6374, DOI 10.17487/RFC6374, September 2011, <https://www.rfc-editor.org/info/rfc6374>.

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

[RFC8321] Fioccola, G., Ed., Capello, A., Cociglio, M., Castaldelli, L., Chen, M., Zheng, L., Mirsky, G., and T. Mizrahi, "Alternate-Marking Method for Passive and Hybrid Performance Monitoring", RFC 8321, DOI 10.17487/RFC8321, January 2018, <https://www.rfc-editor.org/info/rfc8321>.

Authors' Addresses

Weiqiang Cheng
China Mobile

Email: chengweiqiang@chinamobile.com

Lei Wang
China Mobile

Email: wangleiyj@chinamobile.com

Han Li
China Mobile

Email: lihan@chinamobile.com

Mach(Guoyi) Chen
Huawei

Email: mach.chen@huawei.com

Rakesh Gandhi
Cisco Systems, Inc.
Canada

Email: rgandhi@cisco.com

Royi Zigler
Broadcom

Email: royi.zigler@broadcom.com

Shuangping Zhan
ZTE

Email: zhan.shuangping@zte.com.cn

SPRING
Internet-Draft
Intended status: Standards Track
Expires: April 9, 2018

F. Clad, Ed.
C. Filsfils
P. Camarillo
Cisco Systems, Inc.
D. Bernier
Bell Canada
B. Decraene
Orange
B. Peirens
Proximus
C. Yadlapalli
AT&T
X. Xu
Huawei
S. Salsano
Universita di Roma "Tor Vergata"
A. AbdelSalam
Gran Sasso Science Institute
G. Dawra
Cisco Systems, Inc.
October 6, 2017

Segment Routing for Service Chaining
draft-clad-spring-segment-routing-service-chaining-00

Abstract

This document defines data plane functionality required to implement service segments and achieve service chaining with MPLS and IPv6, as described in the Segment Routing architecture.

Table of Contents

1. Introduction

Segment Routing (SR) is an architecture based on the source routing paradigm that seeks the right balance between distributed intelligence and centralized programmability. SR can be used with an MPLS or an IPv6 data plane to steer packets through an ordered list of instructions, called segments. These segments may encode simple routing instructions for forwarding packets along a specific network path, or rich behaviors to support use-cases such as service chaining.

In the context of service chaining, each service, running either on a physical appliance or in a virtual environment, is associated with a segment, which can then be used in a segment list to steer packets through the service. Such service segments may be combined together in a segment list to achieve service chaining, but also with other types of segments as defined in [I-D.ietf-spring-segment-routing]. SR thus provides a fully integrated solution for service chaining, overlay and underlay optimization. Furthermore, the IPv6 dataplane natively supports metadata transportation as part of the SR information attached to the packets.

This document describes how SR enables service chaining in a simple and scalable manner, from the segment association to the service up to the traffic classification and steering into the service chain. Several SR proxy behaviors are also defined to support SR service chaining through legacy, SR-unaware, services in various circumstances.

The definition of control plane components, such as segment discovery and SR policy configuration, is outside the scope of this data plane document. These aspects will be defined in a dedicated document.

Familiarity with the following IETF documents is assumed:

- Segment Routing Architecture [I-D.ietf-spring-segment-routing]
- Segment Routing with MPLS data plane [I-D.ietf-spring-segment-routing-mpls]
- Segment Routing Header [I-D.ietf-6man-segment-routing-header]
- SRv6 Network Programming [I-D.filsfils-spring-srv6-network-programming]

2. Terminology

SR-aware service: Service fully capable of processing SR traffic

SR-unaware service: Service unable to process SR traffic or behaving incorrectly for such traffic

SR proxy: Proxy handling the SR processing on behalf of an SR-unaware service

Service Segment: Segment associated with a service, either directly or via an SR proxy

SR SC policy: SR policy, as defined in [I-D.filsfils-spring-segment-routing-policy], that includes at least one Service Segment. An SR SC policy may also contain other types of segments, such as VPN or TE segments.

SR policy head-end: SR node that classifies and steers traffic into an SR policy.

3. Classification and steering

Classification and steering mechanisms are defined in section 12 of [I-D.filsfils-spring-segment-routing-policy] and are independent from the purpose of the SR policy. From a headend perspective, there is no difference whether a policy contains IGP, BGP, peering, VPN and service segments, or any combination of these.

As documented in the above reference, traffic is classified when entering an SR domain. The SR policy head-end may, depending on its capabilities, classify the packets on a per-destination basis, via simple FIB entries, or apply more complex policy routing rules requiring to look deeper into the packet. These rules are expected to support basic policy routing such as 5-tuple matching. In addition, the IPv6 SRH tag field defined in [I-D.ietf-6man-segment-routing-header] can be used to identify and classify packets sharing the same set of properties. Classified traffic is then steered into the appropriate SR policy, which is associated with a weighted set of segment lists.

SR traffic can be re-classified by an SR endpoint along the original SR policy (e.g., DPI service) or a transit node intercepting the traffic. This node is the head-end of a new SR policy that is imposed onto the packet, either as a stack of MPLS labels or as an IPv6 and SRH encapsulation.

4. Services

A service may be a physical appliance running on dedicated hardware, a virtualized service inside an isolated environment such as a VM, container or namespace, or any process running on a compute element. Unless otherwise stated, this document does not make any assumption on the type or execution environment of a service.

SR enables service chaining by assigning a segment identifier, or SID, to each service and sequencing these service SIDs in a segment list. A service SID may be of local significance or directly reachable from anywhere in the routing domain. The latter is realized with SR-MPLS by assigning a SID from the global label block ([I-D.ietf-spring-segment-routing-mpls]), or with SRv6 by advertising the SID locator in the routing protocol ([I-D.filsfils-spring-srv6-network-programming]).

This document categorizes services in two types, depending on whether they are able to behave properly in the presence of SR information or not. These are respectively named SR-aware and SR-unaware services. An SR-aware service can process the SR information in the packets it receives. This means being able to identify the active segment as a local instruction and move forward in the segment list, but also that the service own behavior is not hindered due to the presence of SR information. For example, an SR-aware firewall filtering SRv6 traffic based on its final destination must retrieve that information from the last entry in the SRH rather than the Destination Address field of the IPv6 header. Any service that does not meet these criteria is considered as SR-unaware.

4.1. SR-aware services

An SR-aware service is associated with a locally instantiated service segment, which is used to steer traffic through it.

If the service is configured to intercept all the packets passing through the appliance, the underlying routing system only has to implement a default SR endpoint behavior (SR-MPLS node segment or SRv6 End function), and the corresponding SID will be used to steer traffic through the service.

If the service requires the packets to be directed to a specific virtual interface, networking queue or process, a dedicated SR behavior may be required to steer the packets to the appropriate location. The definition of such service-specific functions is out of the scope of this document.

An SRv6-aware service may also retrieve, store or modify information in the SRH TLVs.

4.2. SR-unaware services

An SR-unaware service is not able to process the SR information in the traffic that it receives. It may either drop the traffic or take erroneous decisions due to the unrecognized routing information. In order to include such services in an SR SC policy, it is thus required to remove the SR information before the service receives the packet, or to alter it in such a way that the service can correctly process the packet.

In this document, we define the concept of an SR proxy as an entity, separate from the service, that performs these modifications and handle the SR processing on behalf of a service. The SR proxy can run as a separate process on the service appliance, on a virtual switch or router on the compute node or on a remote host. In this document, we only assume that the proxy is connected to the service via a layer-2 link.

An SR-unaware service is associated with a service segment instantiated on the SR proxy, which is used to steer traffic through the service. Section 5 describes several SR proxy behaviors to handle the SR information under various circumstances.

5. SR proxy behaviors

This section describes several SR proxy behaviors designed to enable SR service chaining through SR-unaware services. A system implementing one of these functions may handle the SR processing on behalf of an SR-unaware service and allows the service to properly process the traffic that is steered through it.

A service may be located at any hop in an SR policy, including the last segment. However, the SR proxy behaviors defined in this section are dedicated to supporting SR-unaware services at intermediate hops in the segment list. In case an SR-unaware service is at the last segment, it is sufficient to ensure that the SR information is ignored (IPv6 routing extension header with Segments Left equal to 0) or removed before the packet reaches the service (MPLS PHP, SRv6 End.D or PSP).

As illustrated on Figure 1, the generic behavior of an SR proxy has two parts. The first part is in charge of passing traffic from the network to the service. It intercepts the SR traffic destined for the service via a locally instantiated service segment, modifies it in such a way that it appears as non-SR traffic to the service, then

sends it out on a given interface, IFACE-OUT, connected to the service. The second part receives the traffic coming back from the service on IFACE-IN, restores the SR information and forwards it according to the next segment in the list. Unless otherwise stated IFACE-OUT and IFACE-IN can represent the same interface.

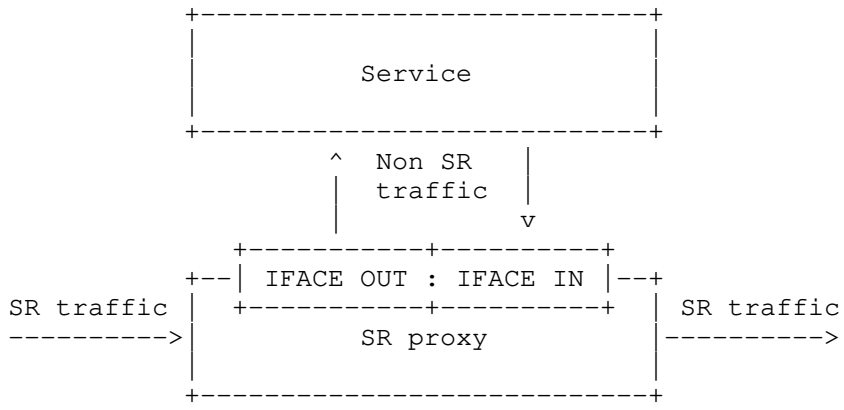


Figure 1: Generic SR proxy

In the next subsections, the following SR proxy mechanisms are defined:

- Static proxy
- Dynamic proxy
- Shared-memory proxy
- Masquerading proxy

Each mechanism has its own characteristics and constraints, which are summarized in the below table. It is up to the operator to select the best one based on the proxy node capabilities, the service behavior and the traffic type. It is also possible to use different proxy mechanisms within the same service chain.

		Static	Dynamic	Shared mem.	Masquerading
SR flavors	SR-MPLS	Y	Y	Y	-
	SRv6 insertion	P	P	P	Y
	SRv6 encapsulation	Y	Y	Y	-
Inner header	Ethernet	Y	Y	Y	-
	IPv4	Y	Y	Y	-
	IPv6	Y	Y	Y	-
Chain agnostic configuration		N	N	Y	Y
Transparent to chain changes		N	Y	Y	Y
Service support	DA modification	Y	Y	Y	NAT
	Payload modification	Y	Y	Y	Y
	Packet generation	Y	Y	cache	cache
	Packet deletion	Y	Y	Y	Y
	Transport endpoint	Y	Y	cache	cache

Figure 2: SR proxy summary

Note: The use of a shared memory proxy requires both the service and the proxy to be running on the same node.

5.1. Static SR proxy

The static proxy is an SR endpoint behavior for processing SR-MPLS or SRv6 encapsulated traffic on behalf of an SR-unaware service. This proxy thus receives SR traffic that is formed of an MPLS label stack or an IPv6 header on top of an inner packet, which can be Ethernet, IPv4 or IPv6.

A static SR proxy segment is associated with the following mandatory parameters:

- INNER-TYPE: Inner packet type
- S-ADDR: Ethernet or IP address of the service (only for inner type IPv4 and IPv6)
- IFACE-OUT: Local interface for sending traffic towards the service
- IFACE-IN: Local interface receiving the traffic coming back from the service
- CACHE: SR information to be attached on the traffic coming back from the service

A static SR proxy segment is thus defined for a specific service, inner packet type and cached SR information. It is also bound to a pair of directed interfaces on the proxy. These may be both directions of a single interface, or opposite directions of two different interfaces. The latter is recommended in case the service is to be used as part of a bi-directional SR SC policy. If the proxy and the service both support 802.1Q, IFACE-OUT and IFACE-IN can also represent sub-interfaces.

The first part of this behavior is triggered when the proxy node receives a packet whose active segment matches a segment associated with the static proxy behavior. It removes the SR information from the packet then sends it on a specific interface towards the associated service. This SR information corresponds to the full label stack for SR-MPLS or to the encapsulation IPv6 header with any attached extension header in the case of SRv6.

The second part is an inbound policy attached to the proxy interface receiving the traffic returning from the service, IFACE-IN. This policy attaches to the incoming traffic the cached SR information associated with the SR proxy segment. If the proxy segment uses the SR-MPLS data plane, CACHE contains a stack of labels to be pushed on top the packets. With the SRv6 data plane, CACHE is defined as a source address, an active segment and an optional SRH (tag, segments

left, segment list and metadata). The proxy encapsulates the packets with an IPv6 header that has the source address, the active segment as destination address and the SRH as a routing extension header. After the SR information has been attached, the packets are forwarded according to the active segment, which is represented by the top MPLS label or the IPv6 Destination Address.

In this scenario, there are no restrictions on the operations that can be performed by the service on the stream of packets. It may operate at all protocol layers, terminate transport layer connections, generate new packets and initiate transport layer connections. This behavior may also be used to integrate an IPv4-only service into an SRv6 policy. However, a static SR proxy segment can be used in only one service chain at a time. As opposed to most other segment types, a static SR proxy segment is bound to a unique list of segments, which represents a directed SR SC policy. This is due to the cached SR information being defined in the segment configuration. This limitation only prevents multiple segment lists from using the same static SR proxy segment at the same time, but a single segment list can be shared by any number of traffic flows. Besides, since the returning traffic from the service is re-classified based on the incoming interface, an interface can be used as receiving interface (IFACE-IN) only for a single SR proxy segment at a time. In the case of a bi-directional SR SC policy, a different SR proxy segment and receiving interface are required for the return direction.

5.1.1. SR-MPLS pseudocode

5.1.1.1. Static proxy for inner type Ethernet - MPLS L2 static proxy segment

Upon receiving an MPLS packet with top label L, where L is an MPLS L2 static proxy segment, a node N does:

```
1.   IF payload type is Ethernet THEN
2.       Pop all labels
3.       Forward the exposed frame on IFACE-OUT
4.   ELSE
5.       Drop the packet
```

Upon receiving on IFACE-IN an Ethernet frame with a destination address different than the interface address, a node N does:

```
1.   Push labels in CACHE on top of the frame Ethernet header
2.   Lookup the top label and proceed accordingly
```

The receiving interface must be configured in promiscuous mode in order to accept those Ethernet frames.

5.1.1.2. Static proxy for inner type IPv4 - MPLS IPv4 static proxy segment

Upon receiving an MPLS packet with top label L, where L is an MPLS IPv4 static proxy segment, a node N does:

```
1.  IF payload type is IPv4 THEN
2.      Pop all labels
3.      Forward the exposed packet on IFACE-OUT towards S-ADDR
4.  ELSE
5.      Drop the packet
```

Upon receiving a non link-local IPv4 packet on IFACE-IN, a node N does:

```
1.  Push labels in CACHE on top of the packet IPv4 header
2.  Decrement inner TTL and update checksum
3.  Lookup the top label and proceed accordingly
```

5.1.1.3. Static proxy for inner type IPv6 - MPLS IPv6 static proxy segment

Upon receiving an MPLS packet with top label L, where L is an MPLS IPv6 static proxy segment, a node N does:

```
1.  IF payload type is IPv6 THEN
2.      Pop all labels
3.      Forward the exposed packet on IFACE-OUT towards S-ADDR
4.  ELSE
5.      Drop the packet
```

Upon receiving a non link-local IPv6 packet on IFACE-IN, a node N does:

```
1.  Push labels in CACHE on top of the packet IPv6 header
2.  Decrement inner Hop Limit
3.  Lookup the top label and proceed accordingly
```

5.1.2. SRv6 pseudocode

5.1.2.1. Static proxy for inner type Ethernet - End.AS2

Upon receiving an IPv6 packet destined for S, where S is an End.AS2 SID, a node N does:

```
1.  IF ENH == 59 THEN                                          ;; Ref1
2.      Remove the (outer) IPv6 header and its extension headers
3.      Forward the exposed frame on IFACE-OUT
4.  ELSE
5.      Drop the packet
```

Ref1: 59 refers to "no next header" as defined by IANA allocation for Internet Protocol Numbers.

Upon receiving on IFACE-IN an Ethernet frame with a destination address different than the interface address, a node N does:

```
1.  IF CACHE.SRH THEN                                          ;; Ref2
2.      Push CACHE.SRH on top of the existing Ethernet header
3.      Set NH value of the pushed SRH to 59
4.  Push outer IPv6 header with SA, DA and traffic class from CACHE
5.  Set outer payload length and flow label
6.  Set NH value to 43 if an SRH was added, or 59 otherwise
7.  Lookup outer DA in appropriate table and proceed accordingly
```

Ref2: CACHE.SRH represents the SRH defined in CACHE, if any, for the static SR proxy segment associated with IFACE-IN.

The receiving interface must be configured in promiscuous mode in order to accept those Ethernet frames.

5.1.2.2. Static proxy for inner type IPv4 - End.AS4

Upon receiving an IPv6 packet destined for S, where S is an End.AS4 SID, a node N does:

```
1.  IF ENH == 4 THEN                                           ;; Ref1
2.      Remove the (outer) IPv6 header and its extension headers
3.      Forward the exposed packet on IFACE-OUT towards S-ADDR
4.  ELSE
5.      Drop the packet
```

Ref1: 4 refers to IPv4 encapsulation as defined by IANA allocation for Internet Protocol Numbers.

Upon receiving a non link-local IPv4 packet on IFACE-IN, a node N does:

```
1.  IF CACHE.SRH THEN                                          ;; Ref2
2.      Push CACHE.SRH on top of the existing IPv4 header
3.      Set NH value of the pushed SRH to 4
4.  Push outer IPv6 header with SA, DA and traffic class from CACHE
5.  Set outer payload length and flow label
6.  Set NH value to 43 if an SRH was added, or 4 otherwise
7.  Decrement inner TTL and update checksum
8.  Lookup outer DA in appropriate table and proceed accordingly
```

Ref2: CACHE.SRH represents the SRH defined in CACHE, if any, for the static SR proxy segment associated with IFACE-IN.

5.1.2.3. Static proxy for inner type IPv6 - End.AS6

Upon receiving an IPv6 packet destined for S, where S is an End.AS6 SID, a node N does:

```
1.  IF ENH == 41 THEN                                          ;; Ref1
2.      Remove the (outer) IPv6 header and its extension headers
3.      Forward the exposed packet on IFACE-OUT towards S-ADDR
4.  ELSE
5.      Drop the packet
```

Ref1: 41 refers to IPv6 encapsulation as defined by IANA allocation for Internet Protocol Numbers.

Upon receiving a non-link-local IPv6 packet on IFACE-IN, a node N does:

```
1.  IF CACHE.SRH THEN                                          ;; Ref2
2.      Push CACHE.SRH on top of the existing IPv6 header
3.      Set NH value of the pushed SRH to 41
4.  Push outer IPv6 header with SA, DA and traffic class from CACHE
5.  Set outer payload length and flow label
6.  Set NH value to 43 if an SRH was added, or 41 otherwise
7.  Decrement inner Hop Limit
8.  Lookup outer DA in appropriate table and proceed accordingly
```

Ref2: CACHE.SRH represents the SRH defined in CACHE, if any, for the static SR proxy segment associated with IFACE-IN.

5.2. Dynamic SR proxy

The dynamic proxy is an improvement over the static proxy that dynamically learns the SR information before removing it from the incoming traffic. The same information can then be re-attached to the traffic returning from the service. As opposed to the static SR proxy, no CACHE information needs to be configured. Instead, the

dynamic SR proxy relies on a local caching mechanism on the node instantiating this segment. Therefore, a dynamic proxy segment cannot be the last segment in an SR SC policy. As mentioned at the beginning of Section 5, a different SR behavior should be used if the service is meant to be the final destination of an SR SC policy.

Upon receiving a packet whose active segment matches a dynamic SR proxy function, the proxy node pops the top MPLS label or applies the SRv6 End behavior, then compares the updated SR information with the cache entry for the current segment. If the cache is empty or different, it is updated with the new SR information. The SR information is then removed and the inner packet is sent towards the service.

The cache entry is not mapped to any particular packet, but instead to an SR SC policy identified by the receiving interface (IFACE-IN). Any non-link-local IP packet or non-local Ethernet frame received on that interface will be re-encapsulated with the cached headers as described in Section 5.1. The service may thus drop, modify or generate new packets without affecting the proxy.

5.2.1. SR-MPLS pseudocode

The static proxy SR-MPLS pseudocode is augmented by inserting the following instructions between lines 1 and 2.

```
1.   IF top label S bit is 0 THEN
2.       Pop top label
3.       IF C(IFACE-IN) different from remaining labels THEN  ;; Ref1
4.           Copy all remaining labels into C(IFACE-IN)       ;; Ref2
5.   ELSE
6.       Drop the packet
```

Ref1: A TTL margin can be configured for the top label stack entry to prevent constant cache updates when multiple equal-cost paths with different hop counts are used towards the SR proxy node. In that case, a TTL difference smaller than the configured margin should not trigger a cache update (provided that the labels are the same).

Ref2: C(IFACE-IN) represents the cache entry associated to the dynamic SR proxy segment. It is identified with IFACE-IN in order to efficiently retrieve the right SR information when a packet arrives on this interface.

In addition, the inbound policy should check that C(IFACE-IN) has been defined before attempting to restore the MPLS label stack, and drop the packet otherwise.

5.2.2. SRv6 pseudocode

The static proxy SRv6 pseudocode is augmented by inserting the following instructions between lines 1 and 2.

```
1.   IF NH=SRH & SL > 0 THEN
2.       Decrement SL and update the IPv6 DA with SRH[SL]
3.       IF C(IFACE-IN) different from IPv6 encaps THEN      ;; Ref1
4.           Copy the IPv6 encaps into C(IFACE-IN)           ;; Ref2
5.   ELSE
6.       Drop the packet
```

Ref1: "IPv6 encaps" represents the IPv6 header and any attached extension header.

Ref2: C(IFACE-IN) represents the cache entry associated to the dynamic SR proxy segment. It is identified with IFACE-IN in order to efficiently retrieve the right SR information when a packet arrives on this interface.

In addition, the inbound policy should check that C(IFACE-IN) has been defined before attempting to restore the IPv6 encapsulation, and drop the packet otherwise.

5.3. Shared memory SR proxy

The shared memory proxy is an SR endpoint behavior for processing SR-MPLS or SRv6 encapsulated traffic on behalf of an SR-unaware service. This proxy behavior leverages a shared-memory interface with the service in order to hide the SR information from an SR-unaware service while keeping it attached to the packet. We assume in this case that the proxy and the service are running on the same compute node. A typical scenario is an SR-capable vrouter running on a container host and forwarding traffic to virtual services isolated within their respective container.

More details will be added in a future revision of this document.

5.4. Masquerading SR proxy

The masquerading proxy is an SR endpoint behavior for processing SRv6 traffic on behalf of an SR-unaware service. This proxy thus receives SR traffic that is formed of an IPv6 header and an SRH on top of an inner payload. The masquerading behavior is independent from the inner payload type. Hence, the inner payload can be of any type but it is usually expected to be a transport layer packet, such as TCP or UDP.

A masquerading SR proxy segment is associated with the following mandatory parameters:

- S-ADDR: Ethernet or IPv6 address of the service
- IFACE-OUT: Local interface for sending traffic towards the service
- IFACE-IN: Local interface receiving the traffic coming back from the service

A masquerading SR proxy segment is thus defined for a specific service and bound to a pair of directed interfaces or sub-interfaces on the proxy. As opposed to the static and dynamic SR proxies, a masquerading segment can be present at the same time in any number of SR SC policies and the same interfaces can be bound to multiple masquerading proxy segments. The only restriction is that a masquerading proxy segment cannot be the last segment in an SR SC policy.

The first part of the masquerading behavior is triggered when the proxy node receives an IPv6 packet whose Destination Address matches a masquerading proxy segment. The proxy inspects the IPv6 extension headers and substitutes the Destination Address with the last segment in the SRH attached to the IPv6 header, which represents the final destination of the IPv6 packet. The packet is then sent out towards the service.

The service receives an IPv6 packet whose source and destination addresses are respectively the original source and final destination. It does not attempt to inspect the SRH, as RFC2460 specifies that routing extension headers are not examined or processed by transit nodes. Instead, the service simply forwards the packet based on its current Destination Address. In this scenario, we assume that the service can only inspect, drop or perform limited changes to the packets. For example, Intrusion Detection Systems, Deep Packet Inspectors and non-NAT Firewalls are among the services that can be supported by a masquerading SR proxy. Variants of the masquerading behavior are defined in Section 5.4.2 and Section 5.4.3 to support a wider range of services.

The second part of the masquerading behavior, also called de-masquerading, is an inbound policy attached to the proxy interface receiving the traffic returning from the service, IFACE-IN. This policy inspects the incoming traffic and triggers a regular SRv6 endpoint processing (End) on any IPv6 packet that contains an SRH. This processing occurs before any lookup on the packet Destination Address is performed and it is sufficient to restore the right active segment as the Destination Address of the IPv6 packet.

5.4.1. SRv6 masquerading proxy pseudocode - End.AM

Masquerading: Upon receiving a packet destined for S, where S is an End.AM SID, a node N processes it as follows.

```
1.  IF NH=SRH & SL > 0 THEN
2.      Update the IPv6 DA with SRH[0]
3.      Forward the packet on IFACE-OUT
4.  ELSE
5.      Drop the packet
```

De-masquerading: Upon receiving a non-link-local IPv6 packet on IFACE-IN, a node N processes it as follows.

```
1.  IF NH=SRH & SL > 0 THEN
2.      Decrement SL
3.      Update the IPv6 DA with SRH[SL]                    ;; Ref1
4.      Lookup DA in appropriate table and proceed accordingly
```

Ref2: This pseudocode can be augmented to support the Penultimate Segment Popping (PSP) endpoint flavor. The exact pseudocode modification are provided in [I-D.filsfils-spring-srv6-network-programming].

5.4.2. Variant 1: NAT

Services modifying the destination address in the packets they process, such as NATs, can be supported by a masquerading proxy with the following modification to the de-masquerading pseudocode.

De-masquerading - NAT: Upon receiving a non-link-local IPv6 packet on IFACE-IN, a node N processes it as follows.

```
1.  IF NH=SRH & SL > 0 THEN
2.      Update SRH[0] with the IPv6 DA
3.      Decrement SL
4.      Update the IPv6 DA with SRH[SL]
5.      Lookup DA in appropriate table and proceed accordingly
```

5.4.3. Variant 2: Caching

Services generating packets or acting as endpoints for transport connections can be supported by adding a dynamic caching mechanism similar to the one described in Section 5.2.

More details will be added in a future revision of this document.

6. Illustrations

We consider the network represented in Figure 3 where:

- A and B are two end hosts using IPv4
- B advertises the prefix 20.0.0.0/8
- 1 to 6 are physical or virtual routers supporting IPv6 and segment routing
- S1 is an SR-aware firewall service
- S2 is an SR-unaware IPS service

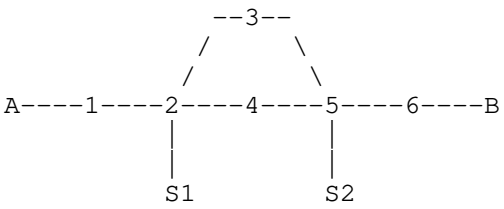


Figure 3: Network with services

All links are configured with an IGP weight of 10 except link 2-3 that is set to 20.

We assume that the path 2-3-5 has a lower latency than 2-4-5.

Nodes 1 to 6 each advertise in the IGP an IPv6 prefix Ck::/64, where k represents the node identifier.

Nodes 1 to 6 are each configured with an SRv6 End segment Ck::/128, where k represents the node identifier.

Node S1 is configured with an SRv6 SID CF1::/128 such that packets arriving at S1 with the Destination Address CF1:: are processed by the service. This SID is either advertised by S1, if it participates in the IGP, or by node 2 on behalf of S1.

Node 5 is also configured with an SRv6 dynamic proxy segments (End.AD) C5::AD:F2 for S2.

Node 6 is also configured with an SRv6 End.DX4 segment C6::D4:B decapsulating the SRv6 and sending the inner IPv4 packets towards D.

Via BGP signaling or an SDN controller, node 1 is programmed with a route 20.0.0.0/8 via C6::D4:B and a color/community requiring low latency and services S1 and S2.

Node 1 either locally computes the path to the egress node or delegates the computation to a PCE. As a result, the SRv6 encapsulation policy < CF1::, C3::, C5::AD:F2, C6::D4:B > is associated with the route 20.0.0.0/8 on node 1.

Upon receiving a packet P from node A and destined to 20.20.20.20, node 1 finds the above table entry and pushes an outer IPv6 header with (SA = C1::, DA = CF1::, NH = SRH) followed by an SRH (C6::D4:B, C5::AD:F2, C3::, CF1::; SL = 3; NH = IPv4). Node 1 then forwards the packet to the first destination address CF1::.

Node 2 forwards P along the shortest path to S1, based on the IPv6 destination address CF1::.

When S1 receives the packet, it identifies a locally instantiated SID and applies the firewall filtering rules. If the packet is not dropped, the SL value is decremented and the DA is updated to the next segment C3::. S1 then sends back to node 2 the packet P with (SA = C1::, DA = C3::, NH = SRH) (C6::D4:B, C5::AD:F2, C3::, CF1::; SL = 2; NH = IPv4).

Node 2 forwards P along the shortest path to node 3, based on the IPv6 destination address C3::.

When 3 receives the packet, 3 matches the DA in its local SID table and finds the bound End function. It thus decrements the SL value and updates the DA to the next segment: C5::AD:F2. Node 3 then forwards packet P with (SA = C1::, DA = C5::AD:F2, NH = SRH) (C6::D4:B, C5::AD:F2, C3::, CF1::; SL = 1; NH = IPv4) towards node 5.

When 5 receives the packet, 5 matches the DA in its local SID table and finds the bound function End.AD(S2). It thus performs the End function (decrement SL and update DA), caches and removes the outer IPv6 header and the SRH, then forwards the inner IPv4 packet towards S2.

S2 receives a regular IPv4 packet headed to 20.20.20.20. It applies the IPS rules and forwards the packet back to node 5.

When 5 receives the packet on the returning interface (IFACE-IN) for S2, 5 retrieves the corresponding cache entry and pushes the updated IPv6 header and SRH. It then forwards P with (SA = C1::, DA = C6:D4:B, NH = SRH) (C6::D4:B, C5::AD:F2, C3::, CF1::; SL = 0; NH = IPv4) to node 6.

When 6 receives the packet, 6 matches the DA in its local SID table and finds the bound function End.DX4. It thus removes the outer IPv6 header and forwards the inner IPv4 packet to node B.

7. Metadata

7.1. MPLS data plane

The MPLS data plane does not provide any native mechanism to attach metadata to a packet.

Workarounds to carry metadata in an SR-MPLS context will be discussed in a future version of this document.

7.2. IPv6 - SRH TLV objects

The IPv6 SRH TLV objects are designed to carry all sorts of metadata. In particular, [I-D.ietf-6man-segment-routing-header] defines the NSH carrier TLV as a container for NSH metadata.

TLV objects can be imposed by the ingress edge router that steers the traffic into the SR SC policy.

An SR-aware service may impose, modify or remove any TLV object attached to the first SRH, either by directly modifying the packet headers or via a control channel between the service and its forwarding plane.

An SR-aware service that re-classifies the traffic and steers it into a new SR SC policy (e.g. DPI) may attach any TLV object to the new SRH.

Metadata imposition and handling will be further discussed in a future version of this document.

7.3. IPv6 - SRH tag

The SRH tag identifies a packet as part of a group or class of packets [I-D.ietf-6man-segment-routing-header].

In a service chaining context, this field can be used as a simple man's metadata to encode additional information in the SRH.

8. Implementation status

The static SR proxy is available for SR-MPLS and SRv6 on various Cisco hardware and software platforms. Furthermore, the following proxies are available on open-source software.

		VPP	Linux
MPLS	Static proxy	Available	In progress
	Dynamic proxy	In progress	In progress
	Shared memory proxy	In progress	In progress
SRv6	Static proxy	Available	In progress
	Dynamic proxy - Inner type Ethernet	In progress	In progress
	Dynamic proxy - Inner type IPv4	Available	Available
	Dynamic proxy - Inner type IPv6	Available	Available
	Shared memory proxy	In progress	In progress
	Masquerading proxy	Available	Available
	Masquerading proxy - NAT variant	In progress	In progress
	Masquerading proxy - Cache variant	In progress	In progress

Open-source implementation status table

9. Relationship with RFC 7665

The Segment Routing solution addresses a wider problem that covers both topological and service chaining policies. The topological and service instructions can be either deployed in isolation or in combination. SR has thus a wider applicability than the architecture defined in [RFC7665]. Furthermore, the inherent property of SR is a stateless network fabric. In SR, there is no state within the fabric to recognize a flow and associate it with a policy. State is only present at the ingress edge of the SR domain, where the policy is encoded into the packets. This is completely different from NSH that relies on state configured at every hop of the service chain.

Hence, there is no linkage between this document and [RFC7665].

10. IANA Considerations

This document has no actions for IANA.

11. Security Considerations

The security requirements and mechanisms described in [I-D.ietf-spring-segment-routing] and [I-D.ietf-6man-segment-routing-header] also apply to this document. Additional considerations will be discussed in future versions of the document.

12. Acknowledgements

TBD.

13. Contributors

Jisu Bhattacharya substantially contributed to the content of this document.

14. References

14.1. Normative References

[I-D.ietf-spring-segment-routing]
Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", draft-ietf-spring-segment-routing-12 (work in progress), June 2017.

14.2. Informative References

[I-D.filsfils-spring-segment-routing-policy]
Filsfils, C., Sivabalan, S., Raza, K., Liste, J., Clad, F., Lin, S., bogdanov@google.com, b., Horneffer, M., Steinberg, D., Decraene, B., and S. Litkowski, "Segment Routing Policy for Traffic Engineering", draft-filsfils-spring-segment-routing-policy-01 (work in progress), July 2017.

[I-D.filsfils-spring-srv6-network-programming]
Filsfils, C., Leddy, J., daniel.voyer@bell.ca, d., daniel.bernier@bell.ca, d., Steinberg, D., Raszuk, R., Matsushima, S., Lebrun, D., Decraene, B., Peirens, B., Salsano, S., Naik, G., Elmalky, H., Jonnalagadda, P., Sharif, M., Ayyangar, A., Mynam, S., Henderickx, W., Bashandy, A., Raza, K., Dukes, D., Clad, F., and P. Camarillo, "SRv6 Network Programming", draft-filsfils-spring-srv6-network-programming-01 (work in progress), June 2017.

[I-D.ietf-6man-segment-routing-header]
Previdi, S., Filsfils, C., Raza, K., Leddy, J., Field, B., daniel.voyer@bell.ca, d., daniel.bernier@bell.ca, d., Matsushima, S., Leung, I., Linkova, J., Aries, E., Kosugi, T., Vyncke, E., Lebrun, D., Steinberg, D., and R. Raszuk, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-07 (work in progress), July 2017.

[I-D.ietf-spring-segment-routing-mpls]
Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls-10 (work in progress), June 2017.

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

Authors' Addresses

Francois Clad (editor)
Cisco Systems, Inc.
France

Email: fclad@cisco.com

Clarence Filsfils
Cisco Systems, Inc.
Belgium

Email: cf@cisco.com

Pablo Camarillo Garvia
Cisco Systems, Inc.
Spain

Email: pcamaril@cisco.com

Daniel Bernier
Bell Canada
Canada

Email: daniel.bernier@bell.ca

Bruno Decraene
Orange
France

Email: bruno.decraene@orange.com

Bart Peirens
Proximus
Belgium

Email: bart.peirens@proximus.com

Chaitanya Yadlapalli
AT&T
USA

Email: cy098d@att.com

Xiaohu Xu
Huawei

Email: xuxiaohu@huawei.com

Stefano Salsano
Universita di Roma "Tor Vergata"
Italy

Email: stefano.salsano@uniroma2.it

Ahmed AbdelSalam
Gran Sasso Science Institute
Italy

Email: ahmed.abdelsalam@gssi.it

Gaurav Dawra
Cisco Systems, Inc.
USA

Email: gdawra@cisco.com

Inter-Domain Routing
Internet-Draft
Intended status: Standards Track
Expires: July 8, 2018

G. Dawra, Ed.
C. Filsfils
Cisco Systems
D. Bernier
Bell Canada
J. Uttaro
AT&T
B. Decraene
Orange
H. Elmalky
Ericsson
X. Xu
Huawei
F. Clad
K. Talaulikar
Cisco Systems

January 4, 2018

BGP Control Plane Extensions for Segment Routing based Service Chaining
draft-dawra-idr-bgp-sr-service-chaining-02

Abstract

The BGP Control Plane for the SR service-chaining solution is consistent with the BGP Control Plane for the topological Segment Routing Traffic Engineering (SR-TE) solution.

- BGP Link-State(BGP-LS) address-family/sub-address-family[RFC7752] is used to discover service and topological characteristics from the network.

- SR-TE policies[I-D.ietf-idr-segment-routing-te-policy] instantiate source-routed policies that may mix service and topological segments.

Table of Contents

1. Introduction

Segments are introduced in the SR architecture [I-D.ietf-spring-segment-routing]. Segment Routing based Service chaining is well described in Section 6 of [I-D.clad-spring-segment-routing-service-chaining] document with an example network and services.

This document extend the example to add a Segment Routing Controller (SR-C) to the network, for the purpose of service discovery and SR policy instantiation.

Consider the network represented in Figure 1 below where:

- A and B are two end hosts using IPv4.
- S1 is an SR-aware firewall Service.
- S2 is an SR-unaware DPI Service.

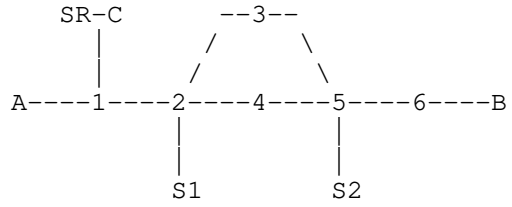


Figure 1: Network with Services

SR Controller (SR-C) is connected to Node 1, but may be attached to any node 1-6 in the network.

SR-C is capable of receiving BGP-LS updates to discover topology, and calculating constrained paths between 1 and 6.

However, if SR-C is configured to computation a constrained path from 1 and 6, including a DPI service (i.e., S2) it is not yet possible due to the lack of service distribution. SR-C does not know where a DPI Service is nor the SID for it. It does not know that S2 is a service it needs.

This document proposes an extension to BGP-LS for Service Chaining to distribute the service information to SR-C. There may be other alternate mechanisms to distribute service information to SR-C and are outside of scope of this document. There are no extensions required in SR-TE Policy SAFI.

2. BGP-LS Extensions for Service Chaining

For an attached service, following data needs to be shared with SR-C:

- Service SID value (e.g. MPLS label or IPv6 address). Service SID MAY only be encoded as LOC:FUNCT, where LOC is the L most significant bits and FUNCT is the 128-L least significant

bits[I-D.filsfils-spring-srv6-network-programming]. ARGs bits, if any, MAY be set to 0 in the advertised service SID.

- Function Identifier (Static Proxy, Dynamic Proxy, Shared Memory Proxy, Masquerading Proxy, SR Aware Service etc).
- Service Type (DPI, Firewall, Classifier, LB etc).
- Traffic Type (IPv4 OR IPv6 OR Ethernet)
- Opaque Data (Such as brand and version, other extra information)

[I-D.clad-spring-segment-routing-service-chaining]defines SR-aware and SR-unaware services. This document will reuse these definitions. Per [RFC7752] Node Attributes are ONLY associated with the Node NLRI. All non-VPN information SHALL be encoded using AFI 16388 / SAFI 71. VPN information SHALL be encoded using AFI 16388 / SAFI 72 with associated RTs.

This document extends SRv6 Node SID TLV [I-D.dawra-idr-bgpls-srv6-ext] and SR-MPLS SID/Label TLV [I-D.ietf-idr-bgp-ls-segment-routing-ext] to associate the Service SID Value with Service-related Information using Service Chaining(SC) Sub-TLV.

Function Sub-TLV [I-D.dawra-idr-bgpls-srv6-ext] of Node SID TLV encodes Identifier(Function ID) along with associated Function Flags.

A Service Chaining (SC) Sub-TLV in Figure 2 is defined as:

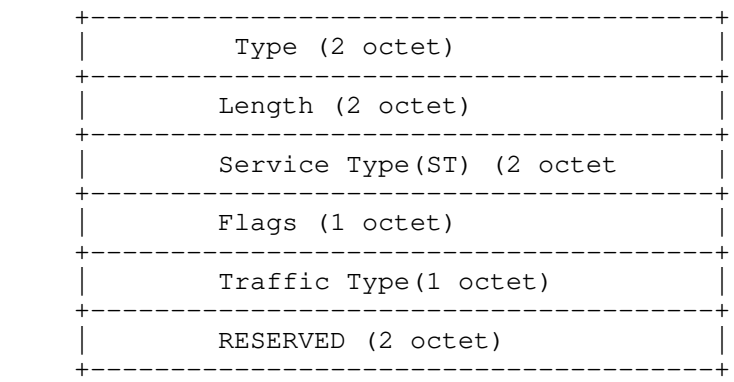


Figure 2: Service Chaining(SC) Sub-TLV

Where:

Type: 16 bit field. TBD

Length: 16 bit field. The total length of the value portion of the TLV.

Service Type(ST): 16bit field. Service Type: categorizes the Service: (such as "Firewall", "Classifier" etc).

Flags: 8 bit field. Bits SHOULD be 0 on transmission and MUST be ignored on reception.

Traffic Type: 8 Bit field. A bit to identify if Service is IPv4 OR IPv6 OR L2 Ethernet Capable. Where:

Bit 0(LSB): Set to 1 if Service is IPv4 Capable

Bit 1: Set to 1 if Service is IPv6 Capable

Bit 2: Set to 1 if Service is Ethernet Capable

RESERVED: 16bit field. SHOULD be 0 on transmission and MUST be ignored on reception.

Service Type(ST) MUST be encoded as part of SC Sub-TLV.

There may be multiple instances of similar Services that needs to be distinguished. For example, firewalls made by different vendors A and B may need to be identified differently because, while they have similar functionality, their behavior is not identical.

In order for SDN Controller to identify the categories of Services and their associated SIDs, this section defines the BGP-LS extensions required to encode these characteristics and other relevant information about these Services.

Another Optional Opaque Metadata(OM) Sub-TLV of Node SID TLV may encode vendor specific information. Multiple of OM Sub-TLVs may be encoded.

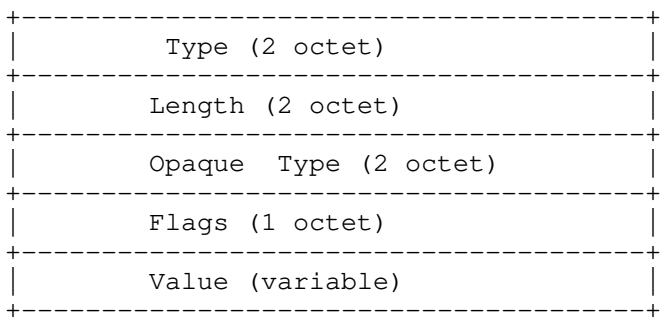


Figure 3: Opaque Metadata(OM) Sub-TLV

- Type: 16 bit field. TBD.
- Length: 16 bit field. The total length of the value portion of the TLV.
- Opaque Type: 8-bit field. Only publishers and consumers of the opaque data are supposed to understand the data.
- Flags: 8 bit field. Bits SHOULD be 0 on transmission and MUST be ignored on reception.
- Value: Variable Length. Based on the data being encoded and length is recorded in length field.

Opaque Metadata(OM) Sub-TLV defined in Figure 3 may encode propriety or Service Opaque information such as:

- Vendor specific Service Information.
- Traffic Limiting Information to particular Service Type.
- Opaque Information unique to the Service
- Propriety Enterprise Service specific Information.

3. Illustration

In our SRv6 example above Figure 1 , Node 5 is configured with an SRv6 dynamic proxy segments (End.AD) C5::AD:F2 for S2.

The BGP-LS advertisement MUST contain and Node SID TLV:

- Service SID: C5::AD:F2 SID

- Function ID: END.AD

The BGP-LS advertisement MUST contain a SC Sub-TLV with:

- Service Type: Deep Packet Inspection(DPI)

- Traffic Type: IPv4 Capable.

The BGP-LS advertisement MAY contain a OM Sub-TLV with:

- Opaque Type: Cisco DPI Version

- Value: 3.5

In our example in Figure 1, using BGP SR-TE SAFI Update [I-D.ietf-idr-segment-routing-te-policy], SR Controller computes the candidate path and pushes the Policy.

SRv6 encapsulation policy < CF1::, C3::, C5::AD:F2, C6::D4:B > is signaled to Node 1 which has mix of service and topological segments.

4. IANA Considerations

This document requests assigning code-points from the registry "BGP-LS Node Descriptor, Link Descriptor, Prefix Descriptor, and Attribute TLVs".

4.1. Service Type Table

IANA is request to create a new top-level registry called "Service Type Table (STT)". Valid values are in the range 0 to 65535. Values 0 and 65535 are to be marked "Reserved, not to be allocated".

Service Value(TBD)	Service	Reference	Date
32	Classifier	ref-to-set	date-to-set
33	Firewall	ref-to-set	date-to-set
34	Load Balancer	ref-to-set	date-to-set
35	DPI	ref-to-set	date-to-set

Figure 4

4.2. Segment routing function Identifier(SFI)

IANA is request to extend a top-level registry called "Segment Routing Function Identifier(SFI)" with new code points. This document extends the SFI values defined in [I-D.dawra-idr-bgpls-srv6-ext]. Details about the Service functions are defined in[I-D.clad-spring-segment-routing-service-chaining].

Function	Function Identifier
Static Proxy	8
Dynamic Proxy	9
Shared Memory Proxy	10
Masquerading Proxy	11
SRv6 Aware Service	12

5. Manageability Considerations

This section is structured as recommended in[RFC5706]

6. Operational Considerations

6.1. Operations

Existing BGP and BGP-LS operational procedures apply. No additional operation procedures are defined in this document.

7. Security Considerations

Procedures and protocol extensions defined in this document do not affect the BGP security model. See the 'Security Considerations' section of [RFC4271]for a discussion of BGP security. Also refer to[RFC4272]and[RFC6952]for analysis of security issues for BGP.

8. Conclusions

This document proposes extensions to the BGP-LS to allow discovery of Services using Segment Routing.

9. Acknowledgements

The authors would like to thank Krishnaswamy Ananthamurthy for his review of this document.

10. References

10.1. Normative References

[I-D.clad-spring-segment-routing-service-chaining]
Clad, F., Filsfils, C., Camarillo, P., daniel.bernier@bell.ca, d., Decraene, B., Peirens, B., Yadlapalli, C., Xu, X., Salsano, S., Abdelsalam, A., and G. Dawra, "Segment Routing for Service Chaining", draft-clad-spring-segment-routing-service-chaining-00 (work in progress), October 2017.

[I-D.dawra-idr-bgpls-srv6-ext]
Dawra, G., Filsfils, C., Talaulikar, K., Sreekantiah, A., and L. Ginsberg, "BGP Link State extensions for IPv6 Segment Routing(SRv6)", draft-dawra-idr-bgpls-srv6-ext-00 (work in progress), October 2017.

[RFC4272] Murphy, S., "BGP Security Vulnerabilities Analysis", RFC 4272, DOI 10.17487/RFC4272, January 2006, <https://www.rfc-editor.org/info/rfc4272>.

[RFC4364] Rosen, E. and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, DOI 10.17487/RFC4364, February 2006, <https://www.rfc-editor.org/info/rfc4364>.

[RFC5706] Harrington, D., "Guidelines for Considering Operations and Management of New Protocols and Protocol Extensions", RFC 5706, DOI 10.17487/RFC5706, November 2009, <https://www.rfc-editor.org/info/rfc5706>.

[RFC6952] Jethanandani, M., Patel, K., and L. Zheng, "Analysis of BGP, LDP, PCEP, and MSDP Issues According to the Keying and Authentication for Routing Protocols (KARP) Design Guide", RFC 6952, DOI 10.17487/RFC6952, May 2013, <https://www.rfc-editor.org/info/rfc6952>.

[RFC7752] Gredler, H., Ed., Medved, J., Previdi, S., Farrel, A., and S. Ray, "North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP", RFC 7752, DOI 10.17487/RFC7752, March 2016, <https://www.rfc-editor.org/info/rfc7752>.

10.2. Informative References

[I-D.dawra-bgp-srv6-vpn]
(Unknown), (., Dawra, G., Filsfils, C., Dukes, D., Brissette, P., Camarillo, P., Leddy, J., daniel.voyer@bell.ca, d., daniel.bernier@bell.ca, d., Steinberg, D., Raszuk, R., Decraene, B., and S. Matsushima, "BGP Signaling of IPv6-Segment-Routing-based VPN Networks", draft-dawra-bgp-srv6-vpn-00 (work in progress), March 2017.

[I-D.filsfils-spring-segment-routing-policy]
Filsfils, C., Sivabalan, S., Raza, K., Liste, J., Clad, F., Talaulikar, K., Hegde, S., daniel.voyer@bell.ca, d., Lin, S., bogdanov@google.com, b., Horneffer, M., Steinberg, D., Decraene, B., Litkowski, S., and P. Mattes, "Segment Routing Policy for Traffic Engineering", draft-filsfils-spring-segment-routing-policy-04 (work in progress), December 2017.

[I-D.filsfils-spring-srv6-network-programming]
Filsfils, C., Leddy, J., daniel.voyer@bell.ca, d., daniel.bernier@bell.ca, d., Steinberg, D., Raszuk, R., Matsushima, S., Lebrun, D., Decraene, B., Peirens, B., Salsano, S., Naik, G., Elmalky, H., Jonnalagadda, P., Sharif, M., Ayyangar, A., Mynam, S., Henderickx, W., Bashandy, A., Raza, K., Dukes, D., Clad, F., and P. Camarillo, "SRv6 Network Programming", draft-filsfils-spring-srv6-network-programming-03 (work in progress), December 2017.

[I-D.ietf-6man-segment-routing-header]
Previdi, S., Filsfils, C., Raza, K., Leddy, J., Field, B., daniel.voyer@bell.ca, d., daniel.bernier@bell.ca, d., Matsushima, S., Leung, I., Linkova, J., Aries, E., Kosugi, T., Vyncke, E., Lebrun, D., Steinberg, D., and R. Raszuk, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-07 (work in progress), July 2017.

[I-D.ietf-bess-evpn-prefix-advertisement]
Rabadan, J., Henderickx, W., Drake, J., Lin, W., and A. Sajassi, "IP Prefix Advertisement in EVPN", draft-ietf-bess-evpn-prefix-advertisement-09 (work in progress), November 2017.

[I-D.ietf-idr-bgp-ls-segment-routing-ext]
Previdi, S., Psenak, P., Filsfils, C., Gredler, H., and M. Chen, "BGP Link-State extensions for Segment Routing", draft-ietf-idr-bgp-ls-segment-routing-ext-03 (work in progress), July 2017.

[I-D.ietf-idr-bgp-prefix-sid]
Previdi, S., Filsfils, C., Lindem, A., Sreekantiah, A., and H. Gredler, "Segment Routing Prefix SID extensions for BGP", draft-ietf-idr-bgp-prefix-sid-08 (work in progress), January 2018.

[I-D.ietf-idr-segment-routing-te-policy]
Previdi, S., Filsfils, C., Mattes, P., Rosen, E., and S. Lin, "Advertising Segment Routing Policies in BGP", draft-ietf-idr-segment-routing-te-policy-01 (work in progress), December 2017.

[I-D.ietf-isis-segment-routing-extensions]
Previdi, S., Ginsberg, L., Filsfils, C., Bashandy, A., Gredler, H., Litkowski, S., Decraene, B., and J. Tantsura, "IS-IS Extensions for Segment Routing", draft-ietf-isis-segment-routing-extensions-15 (work in progress), December 2017.

[I-D.ietf-spring-segment-routing]
Filsfils, C., Previdi, S., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", draft-ietf-spring-segment-routing-14 (work in progress), December 2017.

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

[RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <https://www.rfc-editor.org/info/rfc4271>.

[RFC4659] De Clercq, J., Ooms, D., Carugi, M., and F. Le Faucheur, "BGP-MPLS IP Virtual Private Network (VPN) Extension for IPv6 VPN", RFC 4659, DOI 10.17487/RFC4659, September 2006, <https://www.rfc-editor.org/info/rfc4659>.

[RFC5549] Le Faucheur, F. and E. Rosen, "Advertising IPv4 Network Layer Reachability Information with an IPv6 Next Hop", RFC 5549, DOI 10.17487/RFC5549, May 2009, <https://www.rfc-editor.org/info/rfc5549>.

Authors' Addresses

Gaurav Dawra (editor)
Cisco Systems
USA

Email: gdawra.ietf@gmail.com

Clarence Filsfils
Cisco Systems
Belgium

Email: cfilsfil@cisco.com

Daniel Bernier
Bell Canada
Canada

Email: daniel.bernier@bell.ca

Jim Uttaro
AT&T
USA

Email: ju1738@att.com

Bruno Decraene
Orange
France

Email: bruno.decraene@orange.com

Hani Elmalky
Ericsson
USA

Email: hani.elmalky@gmail.com

Xiaohu Xu
Huawei

Email: xuxiaohu@huawei.com

Francois Clad
Cisco Systems
France

Email: fclad@cisco.com

Ketan Talaulikar
Cisco Systems
India

Email: ketant@cisco.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 29, 2018

D. Dukes, Ed.
C. Filsfils
G. Dawra
P. Camarillo
F. Clad
Cisco Systems
S. Salsano
Univ. of Rome Tor Vergata

October 26, 2017

SR For SDWAN: VPN with Underlay SLA
draft-dukes-sr-for-sdwan-00.txt

Abstract

This document describes how SR enables underlay Service Level Agreements (SLA) to a VPN with scale and security while ensuring service opacity. This solution applies to Over-The-Top VPN (OTT VPN) and Software-Defined WAN (SDWAN).

Status of This Memo

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

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

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

This Internet-Draft will expire on April 29, 2018.

Copyright Notice

Table of Contents

1. Introduction

This document describes how SR enables underlay SLA to a VPN with scale and security while ensuring service opacity. This solution applies to Over-The-Top VPN (OTT VPN) with SLA differentiation, and Software-Defined WAN (SDWAN) with SLA differentiation.

The body of this text uses SRv6 for illustration. A similar solution leveraging SR-MPLS is illustrated in an appendix.

This document assumes familiarity with the following IETF documents:

- Segment Routing Architecture [I-D.ietf-spring-segment-routing]

- Segment Routing with MPLS data plane [I-D.ietf-spring-segment-routing-mpls]
- IPv6 Segment Routing Header [I-D.ietf-6man-segment-routing-header]
- SRv6 Network Programming [I-D.filsfils-spring-srv6-network-programming]
- Segment Routing Policy For Traffic Engineering [I-D.filsfils-spring-segment-routing-policy]
- IS-IS Extensions to Support Segment Routing over IPv6 Dataplane [I-D.bashandy-isis-srv6-extensions]

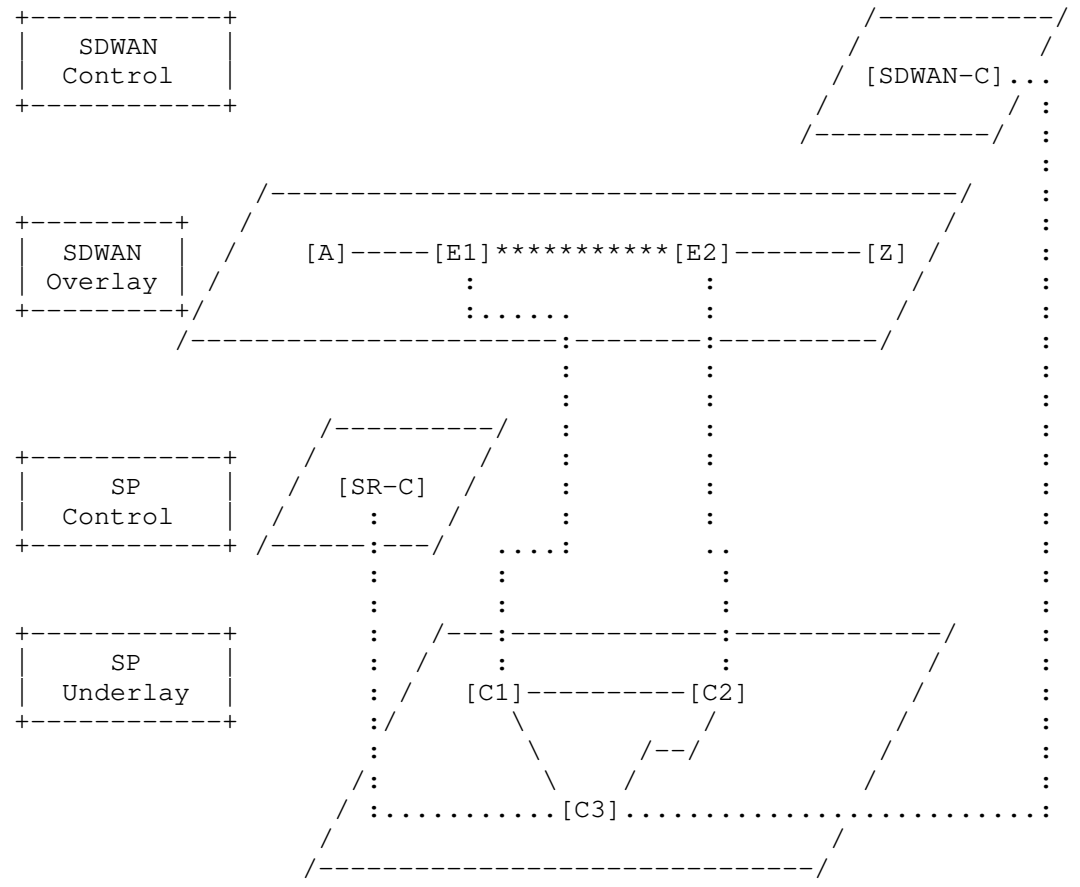
For clarity, this version of the document uses the SDWAN example with SRv6 to illustrate how SR can be used to provide underlay SLA to overlay services. The journey of a packet from the left site to the right site of the SDWAN Overlay is described. The solution applies similarly for the return path.

2. Requirements Notation

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. Single Provider

3.1. Directly Connected CE to PE



**** = logical connection
:... = physical connection, between layers
/--\ = physical connection, within a layer

Figure 1: SDWAN Reference Diagram

An SDWAN overlay is composed of two sites A and Z, connected to the Internet via edge nodes E1 and E2 respectively. E1 and E2 (customer edge nodes) are connected via a Service Provider (SP) underlay to form the VPN between the sites.

C1, C2 and C3 are nodes of the SP underlay, where C1 and C2 are Provider Edge nodes. ISIS is deployed in the SP underlay with the same cost on each link.

E1 and E2 connect to C1 and C2 respectively. The shortest path from C1 to C2 is the best-effort path. The explicit path C1-C3-C2 is the

low-latency path. By default, traffic transported from C1 to C2 follows the best-effort path. By default, an SDWAN cannot benefit from the low-latency path from C1 to C2.

The address of A is 10.10.0.10/32 and the address of Z is 10.26.0.26/32. E1 and E2 respectively advertise 10.10/16 and 10.26/16 to the SDWAN controller SDWAN-C via a secure channel over the Internet. The solution is applicable to any traffic exchanged between the sites, including IPv4, IPv6 or L2. For clarity, a single example with IPv4 in the SDWAN Overlay is used.

The SP operates an SR controller SR-C capable of computing constrained paths from C1 to C2.

3.2. Best-effort Underlay Transport

Let's consider the path taken by traffic from A to Z, across the SDWAN, between nodes E1 and E2 with addresses E1:: and E2:: respectively.

Host A sends a packet P to Z via E1. Packet P has source address 10.10.0.10 and destination address 10.26.0.26, illustrated as P (10.10.0.10,10.26.0.26)(payload). E1, upon receipt of P, determines E2 is the edge node to be used to reach Z. Edge node E1 encrypts, encapsulates and forwards the packet P toward E2 and Z, and it is handled as follow:

- Between A and E1 : P (10.10.0.10,10.26.0.26)(Payload)
- Between E1 and C1 : P (E1::,E2::,NH=ESP)(NH=IPv4,(10.10.0.10,10.26.0.26)(Payload))
  - Note that ESP tunnel mode encapsulation, encryption and authentication is assumed but not required.
- Between C1 and C2 : P (E1::,E2::,NH=ESP)(NH=IPv4,(10.10.0.10,10.26.0.26)(Payload))
- Between C2 and E2 : P (E1::,E2::,NH=ESP)(NH=IPv4,(10.10.0.10,10.26.0.26)(Payload))
- Between E2 and Z : P (10.10.0.10,10.26.0.26)(Payload)

This example illustrates that, classically (i.e., without the SR solution described in this document), the SDWAN cannot leverage the rich infrastructure of the SP to meet its needs. The SP is constrained to offer best-effort transit which does not reflect the capabilities of its infrastructure.

3.3. SR for Underlay SLA Differentiation

SR enables the SDWAN to steer selected flows through selected transport paths of the SP, using the same example in Figure 1.

This small example, with only 3 SP routers, assumes all three support SRv6. As explained in [I-D.filsfils-spring-srv6-network-programming], a typical deployment would only require SRv6 at a few strategic waypoints deployed through the network.

It also assumes ISIS supports the lightweight SRv6 extension described in [I-D.bashandy-isis-srv6-extensions].

The illustration convention from [I-D.filsfils-spring-srv6-network-programming] is used such that:

- SRv6 SID Cj:: is explicitly instantiated at node Cj and bound to the END.PSP function.
- SRv6 SID C1::B21 is a Binding SID (BSID) explicitly instantiated at headend C1 and bound to the SRTE policy <C3::, C2::> towards endpoint C2.
  - Note the return direction would use a BSID C2::B11, bound at headend C2, to the SRTE policy <C3::, C1::> towards endpoint C1.

The Control-Plane (CP) workflow that leads to the instantiation of this Binding SID will be explained in the Control-Plane section.

Let's again consider the path from A to Z for a packet P, but this time E1 has been configured by SDWAN-C to steer packet P into a preferred low-latency path of the SP bound to the binding SID C1:B21.

- Between A and E1
  - P (10.10.0.10,10.26.0.26)(payload)
- Between E1 and C1
  - P (E1::,C1::B21; NH=SRH)(E2::,C1::B21; SL=1; NH=ESP)(NH=IPv4(10.10.0.10,10.26.0.26)(Payload))

When the Binding SID C1::B21 is processed at C1, the SR TE Policy is selected and the SRH for SID list <C3::,C2::> is inserted into P:

- Between C1 and C3

* P (E1::,C3::;NH=SRH)(E2::,C2::,C3::; SL=2;NH=ESP) (NH=IPv4(10.10.0.10,10.26.0.26)(Payload))

At C3, the SegmentsLeft is decremented as the END SID C3:: is processed, and C2:: is placed in the destination address:

- Between C3 and C2
  * P (E1::,C2::;NH=SRH)(E2::,C2::,C3::; SL=1;NH=ESP) (NH=IPv4(10.10.0.10,10.26.0.26)(Payload))

At C2, the SegmentsLeft is decremented to 0, and penultimate segment pop is applied as the END SID C2:: is processed and E2:: is placed in the destination address while the SRH is removed:

- Between C2 and E2
  * P (E1::,E2::,NH=ESP)(NH=IPv4(10.10.0.10,10.26.0.26)(Payload))

Finally, E2 decrypts the packet and strips the outer header to forward the original packet to Z:

- Between E2 and Z
  * P (10.10.0.10,10.26.0.26)(Payload)

The SDWAN edge nodes (E1,E2) maintain their existing behavior of

- Ingress Edge Node: classify ingress traffic, determining the egress edge node, selecting a local output interface, secure the traffic, and forward to the chosen egress edge node.
- Egress Edge Node: decapsulate, decrypt and forward on the internal network.

The only change is that the Ingress node now monitors and selects an SRv6 binding SID then pushes an SRH with two SIDs.

Note as well that the ingress and egress edge nodes never see the actual SID list used by the SP to deliver the preferred path. A variation of this design allows for the BSID to be kept in the packet so that the egress node can detect which packets have been steered on which preferred path (for accounting or monitoring purposes).

This is a fairly simple example of how SRv6 binding SIDs and SR TE policies may be used to provide multiple diverse paths for SDWAN traffic traversing a single provider network.

3.4. Accounting

As per SRv6 network programming [I-D.filsfils-spring-srv6-network-programming], each SRTE policy and its bound BSID is associated with a unique traffic counter. This allows the SP to implement various forms of billing and reporting to the customer of the preferred path.

3.5. Security

The domain of trust security solution documented in [I-D.filsfils-spring-srv6-network-programming] is utilized.

Specifically SEC1, SEC2 and SEC3 guarantee that external traffic to the SP cannot exercise the SID's of the SP.

The following behavior is added: the ACL implementing SEC1 and SEC2 on node C1 is updated to specifically allow traffic from E1:: to C1::B21.

Only the SDWAN edge that has ordered the preferential service can use it.

Any other customer of the SP is unable to use the preferential path bound to BSID C1::B21.

The SDWAN site that has ordered the preferential service is unable to directly program the network of the SP using the internal SID's of the SP. The SDWAN edge node is restricted to the BSID, which opacifies the SP operation.

3.6. Remotely Connected (to PE)

Well known authentication technology with details provided in subsequent revisions will be added, detailing the scenario with SDWAN edge nodes not directly connected to the SP node terminating the binding SID.

4. Multiple Providers

Well known authentication technology with details provided in subsequent revisions will be added, detailing the scenario with SDWAN edge nodes connected to the SP node offering binding SID via an intermediate SP.

5. Control Plane

The SDWAN overlay in Figure 1 is managed by an SDWAN controller, SDWAN-C.

The control protocols used by the SDWAN-C to signal the site routes, the BSID's and the site policies (which traffic on which BSID when) securely over the SP network to E1 and E2 is outside the scope of this document.

The SP underlay operates its internal SR deployment with an SR controller (SR-C). SR-C interacts with the SP's network (Cj) through standardized protocols (PCE[RFC4674] , PCEP [RFC5440]/[RFC4657], BGP RR[RFC4456], BGP-TE [I-D.ietf-idr-segment-routing-te-policy], BGP-LS [RFC7752])

Most likely, the SP would operate its underlay SLA service with a service controller (SERV-C) that is separate from SR-C. To simplify the illustration, this text assumes that SERV-C and SR-C are integrated.

This section describes the high-level interaction between these controllers for the low-latency use-case described in this document, where an enterprise operator installs a policy in the SDWAN-C requiring a low latency service between E1 and E2.

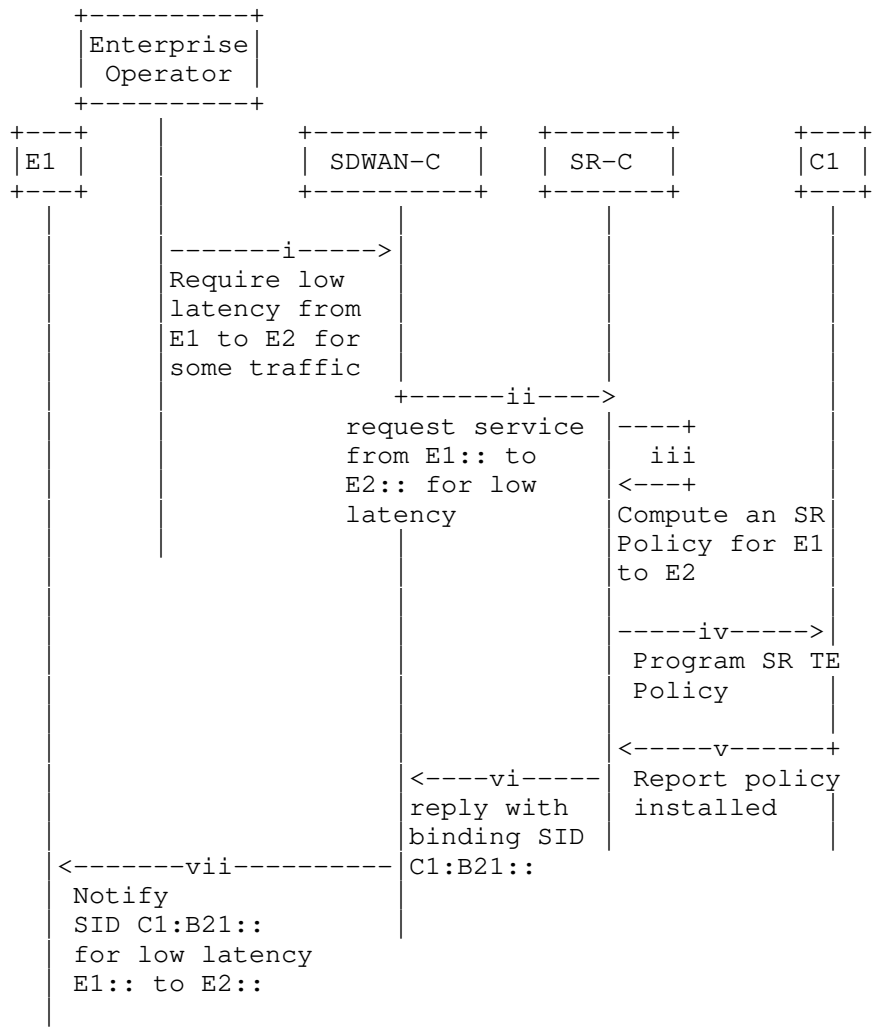


Figure 2: Controlplane Flow

(i) The enterprise operator requests a low-latency path from site E1 to site E2. It defines which traffic needs to be steered on this preferred path.

(ii) SDWAN-C requests a low-latency service from SR-C for the public address of E1 to the public address of E2.

(iii) SR-C computes an SR Policy to satisfy SDWAN-C's request:

A. SR-C maps the E1 and E2 addresses to its managed nodes C1 and C2.

B. SR-C statefully registers the SRTE policy from C1 to C2 for low-latency.

C. SR-C computes the SID list fulfilling the SLA requirement (e.g. <C3::, C2::>). The stateful nature of the SRTE policy ensures that the SID list is updated whenever required due to network state change.

D. SR-C binds a stable Binding SID C1::B21 to the SRTE policy.

(iv) SR-C programs C1 with the computed SRTE policy and the selected BSID. Standardized protocols such as [I-D.ietf-idr-segment-routing-te-policy] or [RFC5440] are used.

(v) C1 installs the policy in its dataplane and reports the status of the SRTE policy to SR-C using standardized protocols [RFC7752] or [RFC5440] and [I-D.negi-pce-segment-routing-ipv6].

(vi) SR-C replies to SDWAN-C with BSID C1::B21

(vii) SDWAN-C programs E1 with the flow classification and steering policy to insert SRv6 SID C1::B21 on the appropriate traffic

6. Benefits

6.1. Scale

The SP network does not hold any per-SDWAN-flow state in the core of its network.

The SP network does not hold any complex L3-L7 flow classification at the edge of its network.

The SP network is unaware of any policy change of the SDWAN instance either in terms of which flow to classify, when to steer it and on which path.

The SP's role only consists in statefully maintaining SRTE policies at the edge of the network and maintaining a few 100's of SID's inside its core network. This is the stateless property of Segment Routing.

6.2. Privacy

The SP network does not share any information of its infrastructure, topology, capacity, internal SID's.

The SDWAN instance does not share any information on its traffic classification, steering policy and business logic.

6.3. Flexible Billing

The traffic destined to a BSID is individually accounted [I-D.filsfils-spring-srv6-network-programming].

The SP and SDWAN instance can agree on various forms of billing for the usage of the preferential path.

6.4. Security

By default, the SP's SR infrastructure is protected by the simple domain of trust solution documented in [I-D.filsfils-spring-srv6-network-programming].

A BSID (and the related preferential path) can only be accessed by the specific SDWAN instance (and site) that ordered the service.

The security solution supports any SDWAN site connection type: directly connected to the SP edge or not.

7. Appendix

7.1. Single Provider Example Using End.BM With an MPLS Core

To be completed in future revisions

7.2. Single Provider Example Using MPLS From CE to PE for BSID

To be completed in future revisions

7.3. Single Provider Example Using SRMPLS Over UDP For CE to PE Not Directly Connected Over Internet

To be completed in future revisions

8. IANA Considerations

No current considerations.

9. Security Considerations

A domain of trust is secured via methods documented in [I-D.filsfils-spring-srv6-network-programming]

10. References

10.1. Informative References

[I-D.bashandy-isis-srv6-extensions]
Ginsberg, L., Bashandy, A., Filsfils, C., and B. Decraene, "IS-IS Extensions to Support Routing over IPv6 Dataplane", draft-bashandy-isis-srv6-extensions-01 (work in progress), September 2017.

[I-D.filsfils-spring-segment-routing-policy]
Filsfils, C., Sivabalan, S., Raza, K., Liste, J., Clad, F., Lin, S., bogdanov@google.com, b., Horneffer, M., Steinberg, D., Decraene, B., and S. Litkowski, "Segment Routing Policy for Traffic Engineering", draft-filsfils-spring-segment-routing-policy-01 (work in progress), July 2017.

[I-D.ietf-6man-segment-routing-header]
Previdi, S., Filsfils, C., Raza, K., Leddy, J., Field, B., daniel.voyer@bell.ca, d., daniel.bernier@bell.ca, d., Matsushima, S., Leung, I., Linkova, J., Aries, E., Kosugi, T., Vyncke, E., Lebrun, D., Steinberg, D., and R. Raszuk, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-07 (work in progress), July 2017.

[I-D.ietf-idr-segment-routing-te-policy]
Previdi, S., Filsfils, C., Mattes, P., Rosen, E., and S. Lin, "Advertising Segment Routing Policies in BGP", draft-ietf-idr-segment-routing-te-policy-00 (work in progress), July 2017.

[I-D.ietf-spring-segment-routing]
Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", draft-ietf-spring-segment-routing-12 (work in progress), June 2017.

[I-D.ietf-spring-segment-routing-mpls]
Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls-10 (work in progress), June 2017.

[I-D.negi-pce-segment-routing-ipv6]
Negi, M., Kaladharan, P., Dhody, D., and S. Sivabalan, "PCEP Extensions for Segment Routing leveraging the IPv6 data plane", draft-negi-pce-segment-routing-ipv6-00 (work in progress), October 2017.

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

[RFC4456] Bates, T., Chen, E., and R. Chandra, "BGP Route Reflection: An Alternative to Full Mesh Internal BGP (IBGP)", RFC 4456, DOI 10.17487/RFC4456, April 2006, <https://www.rfc-editor.org/info/rfc4456>.

[RFC4657] Ash, J., Ed. and J. Le Roux, Ed., "Path Computation Element (PCE) Communication Protocol Generic Requirements", RFC 4657, DOI 10.17487/RFC4657, September 2006, <https://www.rfc-editor.org/info/rfc4657>.

[RFC4674] Le Roux, J., Ed., "Requirements for Path Computation Element (PCE) Discovery", RFC 4674, DOI 10.17487/RFC4674, October 2006, <https://www.rfc-editor.org/info/rfc4674>.

[RFC5440] Vasseur, JP., Ed. and JL. Le Roux, Ed., "Path Computation Element (PCE) Communication Protocol (PCEP)", RFC 5440, DOI 10.17487/RFC5440, March 2009, <https://www.rfc-editor.org/info/rfc5440>.

[RFC7752] Gredler, H., Ed., Medved, J., Previdi, S., Farrel, A., and S. Ray, "North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP", RFC 7752, DOI 10.17487/RFC7752, March 2016, <https://www.rfc-editor.org/info/rfc7752>.

10.2. Normative References'

[I-D.filsfils-spring-srv6-network-programming]
Filsfils, C., Leddy, J., daniel.voyer@bell.ca, d., daniel.bernier@bell.ca, d., Steinberg, D., Raszuk, R., Matsushima, S., Lebrun, D., Decraene, B., Peirens, B., Salsano, S., Naik, G., Elmalky, H., Jonnalagadda, P., Sharif, M., Ayyangar, A., Mynam, S., Henderickx, W., Bashandy, A., Raza, K., Dukes, D., Clad, F., and P. Camarillo, "SRv6 Network Programming", draft-filsfils-spring-srv6-network-programming-01 (work in progress), June 2017.

Authors' Addresses

Darren Dukes (editor)
Cisco Systems
Canada

Email: ddukes@cisco.com

Clarence Filsfils
Cisco Systems
Belgium

Email: cfilsfil@cisco.com

Gaurav Dawra
Cisco Systems
USA

Email: gdawra@cisco.com

Pablo Camarillo Garvia
Cisco Systems
Spain

Email: pcamaril@cisco.com

Francois Clad
Cisco Systems
France

Stefano Salsano
Univ. of Rome Tor Vergata
Italy

Email: stefano.salsano@uniroma2.it

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: October 28, 2018

D. Dukes, Ed.
C. Filsfils
Cisco Systems
G. Dawra
LinkedIn
P. Camarillo
F. Clad
Cisco Systems
S. Salsano
Univ. of Rome Tor Vergata

April 26, 2018

SR For SDWAN: VPN with Underlay SLA
draft-dukes-sr-for-sdwan-01

Abstract

This document describes how SR enables underlay Service Level Agreements (SLA) to a VPN with scale and security while ensuring service opacity. This solution applies to Over-The-Top VPN (OTT VPN) and Software-Defined WAN (SDWAN).

Status of This Memo

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

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

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

This Internet-Draft will expire on October 28, 2018.

Table of Contents

1. Introduction

This document describes how SR enables underlay SLA to a VPN with scale and security while ensuring service opacity. This solution applies to Over-The-Top VPN (OTT VPN) with SLA differentiation, and Software-Defined WAN (SDWAN) with SLA differentiation.

The body of this text uses SRv6 for illustration. A similar solution leveraging SR-MPLS is illustrated in an appendix.

This document assumes familiarity with the following IETF documents:

- Segment Routing Architecture [I-D.ietf-spring-segment-routing]
- Segment Routing with MPLS data plane [I-D.ietf-spring-segment-routing-mpls]
- IPv6 Segment Routing Header [I-D.ietf-6man-segment-routing-header]
- SRv6 Network Programming [I-D.filsfils-spring-srv6-network-programming]
- Segment Routing Policy For Traffic Engineering [I-D.filsfils-spring-segment-routing-policy]
- IS-IS Extensions to Support Segment Routing over IPv6 Dataplane [I-D.bashandy-isis-srv6-extensions]

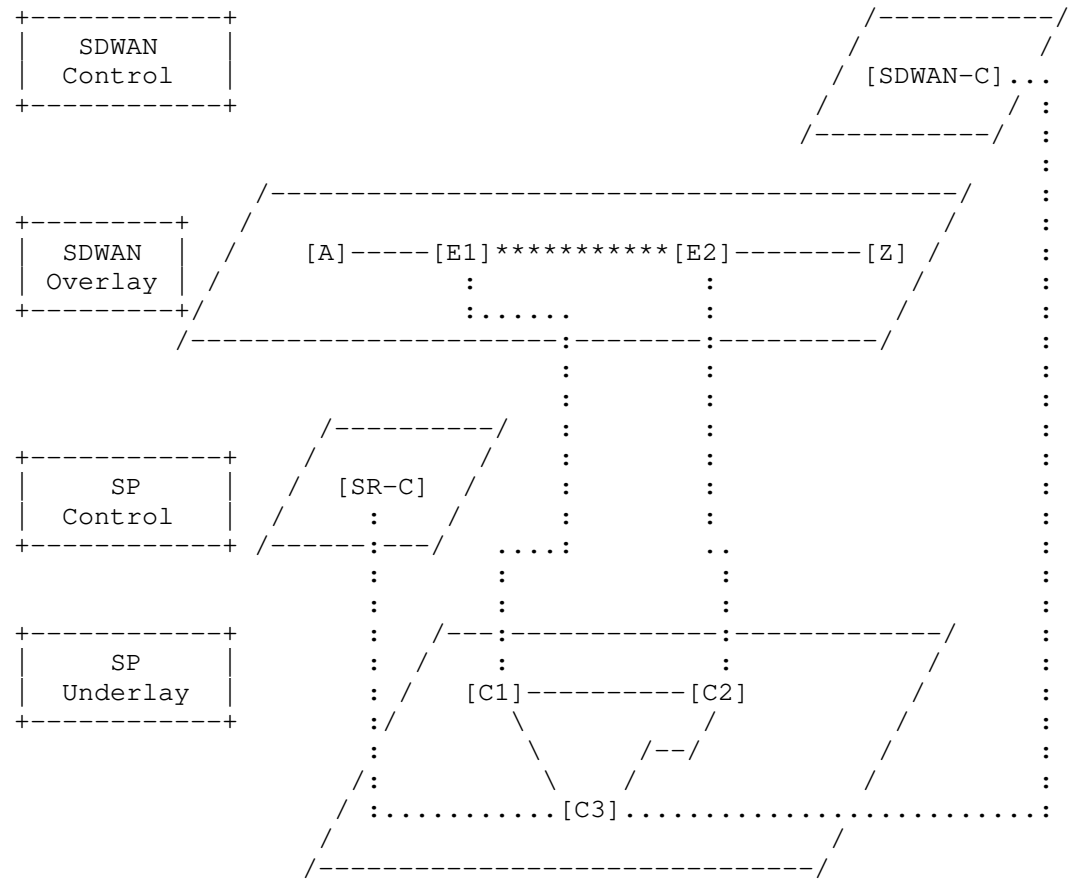
For clarity, this version of the document uses the SDWAN example with SRv6 to illustrate how SR can be used to provide underlay SLA to overlay services. The journey of a packet from the left site to the right site of the SDWAN Overlay is described. The solution applies similarly for the return path.

2. Requirements Notation

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. Single Provider

3.1. Directly Connected CE to PE



**** = logical connection
:... = physical connection, between layers
/--\ = physical connection, within a layer

Figure 1: SDWAN Reference Diagram

An SDWAN overlay is composed of two sites A and Z, connected to the Internet via edge nodes E1 and E2 respectively. E1 and E2 (customer edge nodes) are connected via a Service Provider (SP) underlay to form the VPN between the sites.

C1, C2 and C3 are nodes of the SP underlay, where C1 and C2 are Provider Edge nodes. ISIS is deployed in the SP underlay with the same cost on each link.

E1 and E2 connect to C1 and C2 respectively. The shortest path from C1 to C2 is the best-effort path. The explicit path C1-C3-C2 is the

low-latency path. By default, traffic transported from C1 to C2 follows the best-effort path. By default, an SDWAN cannot benefit from the low-latency path from C1 to C2.

The address of A is 10.10.0.10/32 and the address of Z is 10.26.0.26/32. E1 and E2 respectively advertise 10.10/16 and 10.26/16 to the SDWAN controller SDWAN-C via a secure channel over the Internet. The solution is applicable to any traffic exchanged between the sites, including IPv4, IPv6 or L2. For clarity, a single example with IPv4 in the SDWAN Overlay is used.

The SP operates an SR controller SR-C capable of computing constrained paths from C1 to C2.

3.2. Best-effort Underlay Transport

Let's consider the path taken by traffic from A to Z, across the SDWAN, between nodes E1 and E2 with addresses E1:: and E2:: respectively.

Host A sends a packet P to Z via E1. Packet P has source address 10.10.0.10 and destination address 10.26.0.26, illustrated as P (10.10.0.10,10.26.0.26)(payload). E1, upon receipt of P, determines E2 is the edge node to be used to reach Z. Edge node E1 encrypts, encapsulates and forwards the packet P toward E2 and Z, and it is handled as follow:

- Between A and E1 : P (10.10.0.10,10.26.0.26)(Payload)
- Between E1 and C1 : P (E1::,E2::,NH=ESP)(NH=IPv4,(10.10.0.10,10.26.0.26)(Payload))
  - Note that ESP tunnel mode encapsulation, encryption and authentication is assumed but not required.
- Between C1 and C2 : P (E1::,E2::,NH=ESP)(NH=IPv4,(10.10.0.10,10.26.0.26)(Payload))
- Between C2 and E2 : P (E1::,E2::,NH=ESP)(NH=IPv4,(10.10.0.10,10.26.0.26)(Payload))
- Between E2 and Z : P (10.10.0.10,10.26.0.26)(Payload)

This example illustrates that, classically (i.e., without the SR solution described in this document), the SDWAN cannot leverage the rich infrastructure of the SP to meet its needs. The SP is constrained to offer best-effort transit which does not reflect the capabilities of its infrastructure.

3.3. SR for Underlay SLA Differentiation

SR enables the SDWAN to steer selected flows through selected transport paths of the SP, using the same example in Figure 1.

This small example, with only 3 SP routers, assumes all three support SRv6. As explained in [I-D.filsfils-spring-srv6-network-programming], a typical deployment would only require SRv6 at a few strategic waypoints deployed through the network.

It also assumes ISIS supports the lightweight SRv6 extension described in [I-D.bashandy-isis-srv6-extensions].

The illustration convention from [I-D.filsfils-spring-srv6-network-programming] is used such that:

- SRv6 SID Cj:: is explicitly instantiated at node Cj and bound to the END.PSP function.
- SRv6 SID C1::B21 is a Binding SID (BSID) explicitly instantiated at headend C1 and bound to the SRTE policy <C3::, C2::> towards endpoint C2.
  - Note the return direction would use a BSID C2::B11, bound at headend C2, to the SRTE policy <C3::, C1::> towards endpoint C1.

The Control-Plane (CP) workflow that leads to the instantiation of this Binding SID will be explained in the Control-Plane section.

Let's again consider the path from A to Z for a packet P, but this time E1 has been configured by SDWAN-C to steer packet P into a preferred low-latency path of the SP bound to the binding SID C1:B21.

- Between A and E1
  - P (10.10.0.10,10.26.0.26)(payload)
- Between E1 and C1
  - P (E1::,C1::B21; NH=SRH)(E2::,C1::B21; SL=1; NH=ESP)(NH=IPv4(10.10.0.10,10.26.0.26)(Payload))

When the Binding SID C1::B21 is processed at C1, the SR TE Policy is selected and the SRH for SID list <C3::,C2::> is inserted into P:

- Between C1 and C3

* P (E1::,C3::;NH=SRH)(E2::,C2::,C3::; SL=2;NH=ESP) (NH=IPv4(10.10.0.10,10.26.0.26)(Payload))

At C3, the SegmentsLeft is decremented as the END SID C3:: is processed, and C2:: is placed in the destination address:

- Between C3 and C2
  * P (E1::,C2::;NH=SRH)(E2::,C2::,C3::; SL=1;NH=ESP) (NH=IPv4(10.10.0.10,10.26.0.26)(Payload))

At C2, the SegmentsLeft is decremented to 0, and penultimate segment pop is applied as the END SID C2:: is processed and E2:: is placed in the destination address while the SRH is removed:

- Between C2 and E2
  * P (E1::,E2::,NH=ESP)(NH=IPv4(10.10.0.10,10.26.0.26)(Payload))

Finally, E2 decrypts the packet and strips the outer header to forward the original packet to Z:

- Between E2 and Z
  * P (10.10.0.10,10.26.0.26)(Payload)

The SDWAN edge nodes (E1,E2) maintain their existing behavior of

- Ingress Edge Node: classify ingress traffic, determining the egress edge node, selecting a local output interface, secure the traffic, and forward to the chosen egress edge node.
- Egress Edge Node: decapsulate, decrypt and forward on the internal network.

The only change is that the Ingress node now monitors and selects an SRv6 binding SID then pushes an SRH with two SIDs.

Note as well that the ingress and egress edge nodes never see the actual SID list used by the SP to deliver the preferred path. A variation of this design allows for the BSID to be kept in the packet so that the egress node can detect which packets have been steered on which preferred path (for accounting or monitoring purposes).

This is a fairly simple example of how SRv6 binding SIDs and SR TE policies may be used to provide multiple diverse paths for SDWAN traffic traversing a single provider network.

3.4. Accounting

As per SRv6 network programming [I-D.filsfils-spring-srv6-network-programming], each SRTE policy and its bound BSID is associated with a unique traffic counter. This allows the SP to implement various forms of billing and reporting to the customer of the preferred path.

3.5. Security

The domain of trust security solution documented in [I-D.filsfils-spring-srv6-network-programming] is utilized.

Specifically SEC1, SEC2 and SEC3 guarantee that external traffic to the SP cannot exercise the SID's of the SP.

The following behavior is added: the ACL implementing SEC1 and SEC2 on node C1 is updated to specifically allow traffic from E1:: to C1::B21.

Only the SDWAN edge that has ordered the preferential service can use it.

Any other customer of the SP is unable to use the preferential path bound to BSID C1::B21.

The SDWAN site that has ordered the preferential service is unable to directly program the network of the SP using the internal SID's of the SP. The SDWAN edge node is restricted to the BSID, which opacifies the SP operation.

3.6. Remotely Connected (to PE)

Well known authentication technology with details provided in subsequent revisions will be added, detailing the scenario with SDWAN edge nodes not directly connected to the SP node terminating the binding SID.

4. Multiple Providers

Well known authentication technology with details provided in subsequent revisions will be added, detailing the scenario with SDWAN edge nodes connected to the SP node offering binding SID via an intermediate SP.

5. Control Plane

The SDWAN overlay in Figure 1 is managed by an SDWAN controller, SDWAN-C.

The control protocols used by the SDWAN-C to signal the site routes, the BSID's and the site policies (which traffic on which BSID when) securely over the SP network to E1 and E2 is outside the scope of this document.

The SP underlay operates its internal SR deployment with an SR controller (SR-C). SR-C interacts with the SP's network (Cj) through standardized protocols (PCE[RFC4674] , PCEP [RFC5440]/[RFC4657], BGP RR[RFC4456], BGP-TE [I-D.ietf-idr-segment-routing-te-policy], BGP-LS [RFC7752])

Most likely, the SP would operate its underlay SLA service with a service controller (SERV-C) that is separate from SR-C. To simplify the illustration, this text assumes that SERV-C and SR-C are integrated.

This section describes the high-level interaction between these controllers for the low-latency use-case described in this document, where an enterprise operator installs a policy in the SDWAN-C requiring a low latency service between E1 and E2.

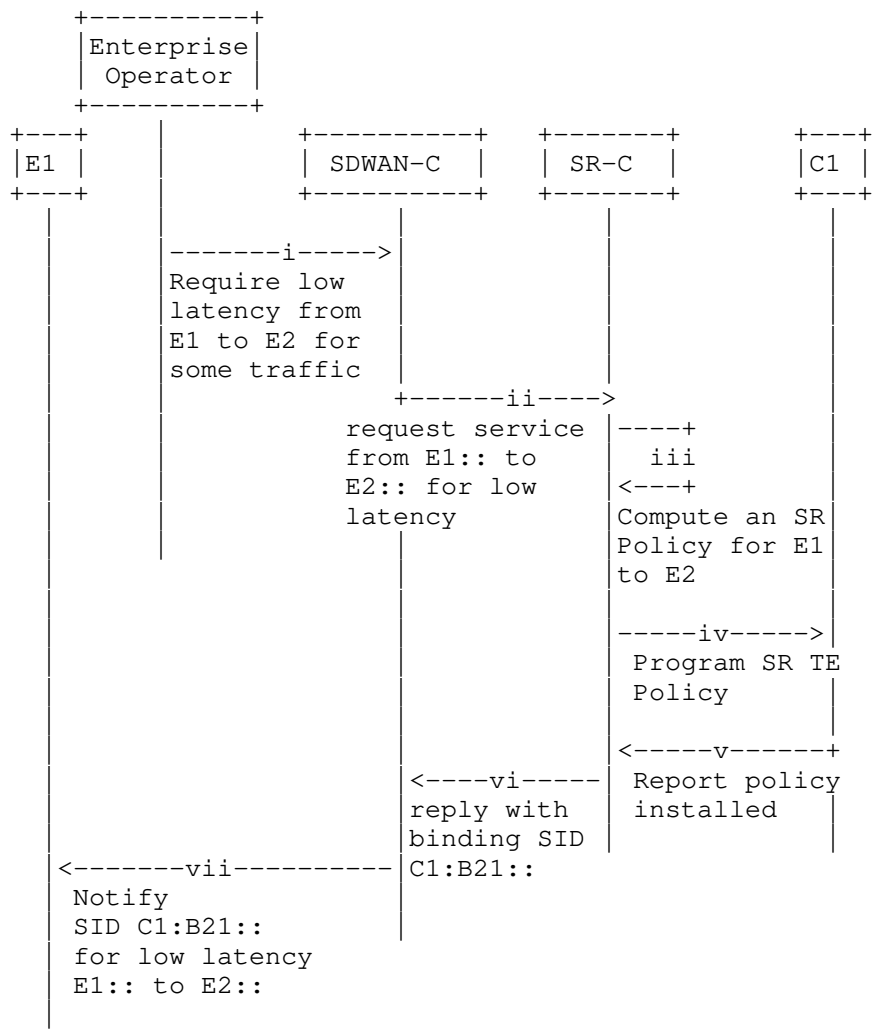


Figure 2: Controlplane Flow

(i) The enterprise operator requests a low-latency path from site E1 to site E2. It defines which traffic needs to be steered on this preferred path.

(ii) SDWAN-C requests a low-latency service from SR-C for the public address of E1 to the public address of E2.

(iii) SR-C computes an SR Policy to satisfy SDWAN-C's request:

A. SR-C maps the E1 and E2 addresses to its managed nodes C1 and C2.

B. SR-C statefully registers the SRTE policy from C1 to C2 for low-latency.

C. SR-C computes the SID list fulfilling the SLA requirement (e.g. <C3::, C2::>). The stateful nature of the SRTE policy ensures that the SID list is updated whenever required due to network state change.

D. SR-C binds a stable Binding SID C1::B21 to the SRTE policy.

(iv) SR-C programs C1 with the computed SRTE policy and the selected BSID. Standardized protocols such as [I-D.ietf-idr-segment-routing-te-policy] or [RFC5440] are used.

(v) C1 installs the policy in its dataplane and reports the status of the SRTE policy to SR-C using standardized protocols [RFC7752] or [RFC5440] and [I-D.negi-pce-segment-routing-ipv6].

(vi) SR-C replies to SDWAN-C with BSID C1::B21

(vii) SDWAN-C programs E1 with the flow classification and steering policy to insert SRv6 SID C1::B21 on the appropriate traffic

6. Benefits

6.1. Scale

The SP network does not hold any per-SDWAN-flow state in the core of its network.

The SP network does not hold any complex L3-L7 flow classification at the edge of its network.

The SP network is unaware of any policy change of the SDWAN instance either in terms of which flow to classify, when to steer it and on which path.

The SP's role only consists in statefully maintaining SRTE policies at the edge of the network and maintaining a few 100's of SID's inside its core network. This is the stateless property of Segment Routing.

6.2. Privacy

The SP network does not share any information of its infrastructure, topology, capacity, internal SID's.

The SDWAN instance does not share any information on its traffic classification, steering policy and business logic.

6.3. Flexible Billing

The traffic destined to a BSID is individually accounted [I-D.filsfils-spring-srv6-network-programming].

The SP and SDWAN instance can agree on various forms of billing for the usage of the preferential path.

6.4. Security

By default, the SP's SR infrastructure is protected by the simple domain of trust solution documented in [I-D.filsfils-spring-srv6-network-programming].

A BSID (and the related preferential path) can only be accessed by the specific SDWAN instance (and site) that ordered the service.

The security solution supports any SDWAN site connection type: directly connected to the SP edge or not.

7. Appendix

7.1. Single Provider Example Using End.BM With an MPLS Core

To be completed in future revisions

7.2. Single Provider Example Using MPLS From CE to PE for BSID

To be completed in future revisions

7.3. Single Provider Example Using SRMPLS Over UDP For CE to PE Not Directly Connected Over Internet

To be completed in future revisions

8. IANA Considerations

No current considerations.

9. Security Considerations

A domain of trust is secured via methods documented in [I-D.filsfils-spring-srv6-network-programming]

10. References

10.1. Informative References

[I-D.bashandy-isis-srv6-extensions]
Ginsberg, L., Bashandy, A., Filsfils, C., Decraene, B., and Z. Hu, "IS-IS Extensions to Support Routing over IPv6 Dataplane", draft-bashandy-isis-srv6-extensions-02 (work in progress), March 2018.

[I-D.filsfils-spring-segment-routing-policy]
Filsfils, C., Sivabalan, S., Raza, K., Liste, J., Clad, F., Talaulikar, K., Ali, Z., Hegde, S., daniel.voyer@bell.ca, d., Lin, S., bogdanov@google.com, b., Krol, P., Horneffer, M., Steinberg, D., Decraene, B., Litkowski, S., and P. Mattes, "Segment Routing Policy for Traffic Engineering", draft-filsfils-spring-segment-routing-policy-05 (work in progress), February 2018.

[I-D.ietf-6man-segment-routing-header]
Previdi, S., Filsfils, C., Leddy, J., Matsushima, S., and d. daniel.voyer@bell.ca, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-12 (work in progress), April 2018.

[I-D.ietf-idr-segment-routing-te-policy]
Previdi, S., Filsfils, C., Jain, D., Mattes, P., Rosen, E., and S. Lin, "Advertising Segment Routing Policies in BGP", draft-ietf-idr-segment-routing-te-policy-02 (work in progress), March 2018.

[I-D.ietf-spring-segment-routing]
Filsfils, C., Previdi, S., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", draft-ietf-spring-segment-routing-15 (work in progress), January 2018.

[I-D.ietf-spring-segment-routing-mpls]
Bashandy, A., Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls-13 (work in progress), April 2018.

[I-D.negi-pce-segment-routing-ipv6]
Negi, M., Kaladharan, P., Dhody, D., and S. Sivabalan, "PCEP Extensions for Segment Routing leveraging the IPv6 data plane", draft-negi-pce-segment-routing-ipv6-01 (work in progress), March 2018.

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

[RFC4456] Bates, T., Chen, E., and R. Chandra, "BGP Route Reflection: An Alternative to Full Mesh Internal BGP (IBGP)", RFC 4456, DOI 10.17487/RFC4456, April 2006, <https://www.rfc-editor.org/info/rfc4456>.

[RFC4657] Ash, J., Ed. and J. Le Roux, Ed., "Path Computation Element (PCE) Communication Protocol Generic Requirements", RFC 4657, DOI 10.17487/RFC4657, September 2006, <https://www.rfc-editor.org/info/rfc4657>.

[RFC4674] Le Roux, J., Ed., "Requirements for Path Computation Element (PCE) Discovery", RFC 4674, DOI 10.17487/RFC4674, October 2006, <https://www.rfc-editor.org/info/rfc4674>.

[RFC5440] Vasseur, JP., Ed. and JL. Le Roux, Ed., "Path Computation Element (PCE) Communication Protocol (PCEP)", RFC 5440, DOI 10.17487/RFC5440, March 2009, <https://www.rfc-editor.org/info/rfc5440>.

[RFC7752] Gredler, H., Ed., Medved, J., Previdi, S., Farrel, A., and S. Ray, "North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP", RFC 7752, DOI 10.17487/RFC7752, March 2016, <https://www.rfc-editor.org/info/rfc7752>.

10.2. Normative References'

[I-D.filsfils-spring-srv6-network-programming]
Filsfils, C., Li, Z., Leddy, J., daniel.voyer@bell.ca, d., daniel.bernier@bell.ca, d., Steinberg, D., Raszuk, R., Matsushima, S., Lebrun, D., Decraene, B., Peirens, B., Salsano, S., Naik, G., Elmalky, H., Jonnalagadda, P., and M. Sharif, "SRv6 Network Programming", draft-filsfils-spring-srv6-network-programming-04 (work in progress), March 2018.

Authors' Addresses

Darren Dukes (editor)
Cisco Systems
Canada

Email: ddukes@cisco.com

Clarence Filsfils
Cisco Systems
Belgium

Email: cfilsfil@cisco.com

Gaurav Dawra
LinkedIn
USA

Email: gdawra@linkedin.com

Pablo Camarillo Garvia
Cisco Systems
Spain

Email: pcamaril@cisco.com

Francois Clad
Cisco Systems
France

Stefano Salsano
Univ. of Rome Tor Vergata
Italy

Email: stefano.salsano@uniroma2.it

SPRING Working Group
Internet-Draft
Intended status: Informational
Expires: May 1, 2018

A. Farrel
J. Drake
Juniper Networks

October 28, 2017

Interconnection of Segment Routing Domains - Problem Statement and Solution Landscape
draft-farrel-spring-sr-domain-interconnect-01

Abstract

Segment Routing (SR) is now a popular forwarding paradigm for use in MPLS and IPv6 networks. It is typically deployed in discrete domains that may be data centers, access networks, or other networks that are under the control of a single operator and that can easily be upgraded to support this new technology.

Traffic originating in one SR domain often terminates in another SR domain, but must transit a backbone network that provides interconnection between those domains.

This document describes a mechanism for providing connectivity between SR domains to enable end-to-end or domain-to-domain traffic engineering.

The approach described: allows connectivity between SR domains, utilizes traffic engineering mechanisms (RSVP-TE or Segment Routing) across the backbone network, makes heavy use of pre-existing technologies requiring the specifications of very few additional mechanisms.

This document some background and a problem statement, explains the solution mechanism, and provides examples. It does not define any new protocol mechanisms.

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 May 1, 2018.

Table of Contents

1. Introduction

Data Centers are a growing market sector. They are being set up by new specialist companies, by enterprises for their own use, by legacy ISPs, and by the new wave of network operators such as Microsoft and Amazon.

The networks inside Data Centers are currently well-planned, but the traffic loads can be unpredictable. There is a need to be able to direct traffic within a Data Center to follow a specific path.

Data Centers are attached to external ("backbone") networks to allow access by users and to facilitate communication among Data Centers. An individual Data Center may be attached to multiple backbone networks, and may have multiple points of attachment to each backbone network. Traffic to or from a Data Center may need to be directed to or from any of these points of attachment.

A variety of networking technologies exist and have been proposed to steer traffic within the Data Center and across the backbone networks. This document proposes an approach that builds on existing technologies to produce mechanisms that provide scalable and flexible interconnection of Data Centers, and that will be easy to operate.

Segment Routing (SR) is a new technology that places forwarding state into each packet as a stack of loose hops as distinct from other pre-existing techniques that require signaling protocols to install state in the network. SR is a popular option for building Data Centers, and is also seeing increasing traction in edge and access networks as well as in backbone networks.

This paper describes mechanisms to provide end-to-end SR connectivity between SR-capable domains across an MPLS backbone network that supports SR and/or MPLS-TE. This is the generalization of the requirement to provide inter-Data Center connectivity.

2. Problem Statement

Consider the network in Figure 1. Without loss of generality, this 'figure can be used to represent the architecture and problem space for steering traffic within and between SR edge domains. The figure shows a single destination for all traffic that we will consider. In this figure we distinguish between the PEs that provide access to the backbone networks and the Gateways that provide access to the SR edge

domains: these may, in fact be the same equipment, and the PEs might be located at the domain edges.

In describing the problem space and the solution we use four terms for network nodes as follows:

SR edge domain : A collection of SR-capable nodes in an edge network attached to the backbone network through one or more gateways. Examples include, access networks, Data Center sites, and blessings of unicorns.

Host : A node within an edge domain. May be an end system or a transit node in the edge domain.

Gateway (GW) : Provides access to or from an edge domain. Examples are CEs, ASBRs, and Data Center gateways.

Provider Edge (PE) : Provides access to or from the backbone network.

Autonomous System Border Router (ASBR) : Provides access to one AS in the backbone network from another AS in the backbone network.

These terms can be seen used in Figure 1 where the various sources and destinations are hosts.

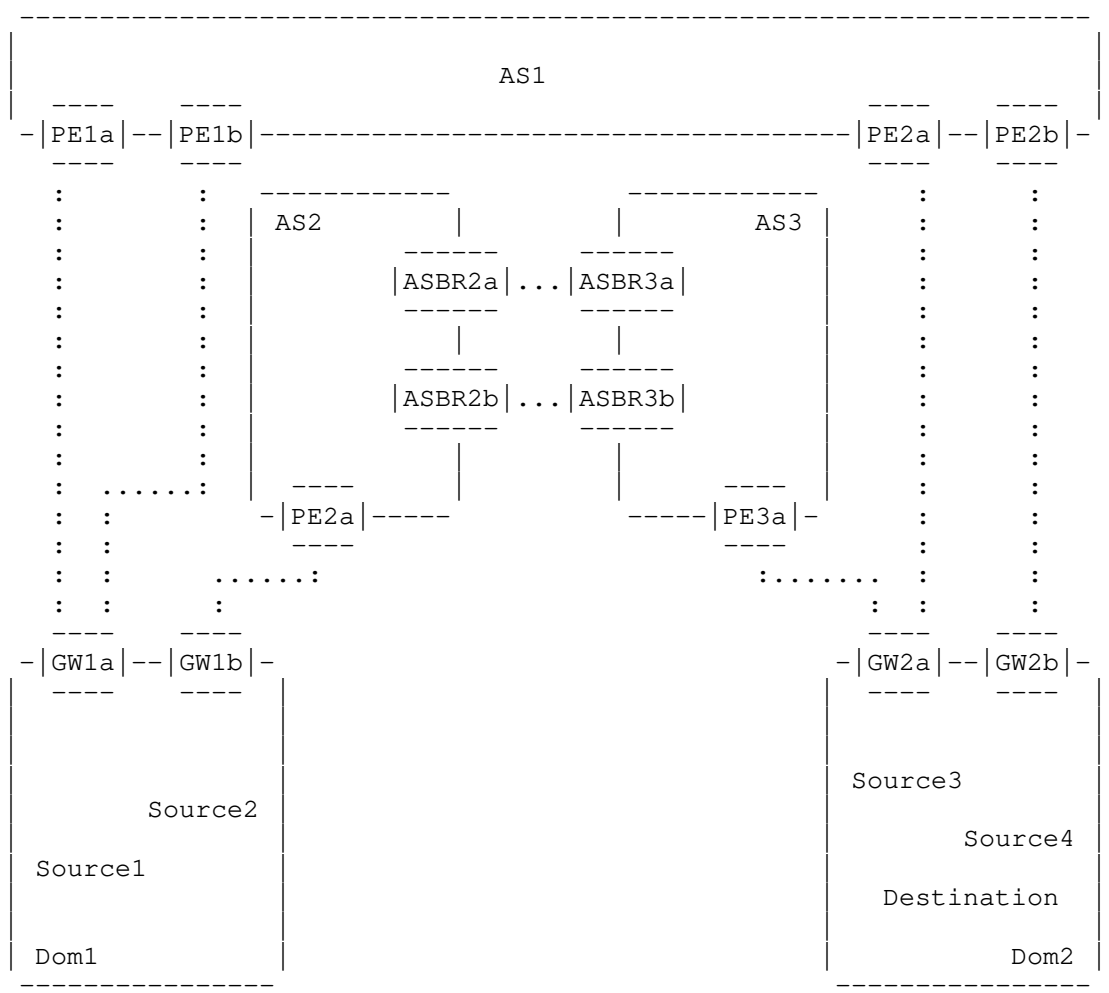


Figure 1: Reference Architecture for SR Domain Interconnect

Traffic to the destination may be sourced from multiple sources within that domain (we show two such sources: Source3 and Source4). Furthermore, traffic intended for the destination may arrive from outside the domain through any of the points of attachment to the backbone networks (we show GW3a and GW3b). This traffic may need to be steered within the domain to achieve load-balancing across network resources, to avoid degraded or out-of-service resources (including planned service outages), and to achieve different qualities of service. Of course, traffic in a remote source domain may also need

to be steered within that domain. We class this problem as "Intra-Domain Traffic Steering".

Traffic across the backbone networks may need to be steered to conform to common Traffic Engineering paradigms. That is, the path across any network (shown in the figure as an AS) or across any collection of networks may need to be chosen. Furthermore, the points of inter-connection between networks may need to be selected and influence the path chosen for the data. We class this problem as "Inter-Domain Traffic Steering".

The composite end-to-end path comprises steering in the source domain, choice of source domain exit point, steering across the backbone networks, choice of network interconnections, choice of destination domain entry point, and steering in the destination domain. These issues may be inter-dependent (for example, the best traffic steering in the source domain may help select the best exit point from that domain, but the connectivity options across the backbone network may drive the selection of a different exit point). We class this combination of problems as "End-to-End Domain Interconnect Traffic Steering".

It should be noted that the solution to the End-to-End Domain Interconnect Traffic Steering problem depends on a number of factors:

- What technology is deployed in the domains.
- What technology is deployed in the backbone networks.
- How much information are the domains willing to share with each other.
- How much information are the backbone network operators and the domain operators are willing to share.

In some cases, the domains and backbone networks are all owned and operated by the same company (with the backbone network often being a private network). In other cases, the domains are operated by one company, with other companies operating the backbone.

3. Solution Technologies

Within the Data Center, Segment Routing (SR from the SPRING working group in the IETF [RFC7855] and [I-D.ietf-spring-segment-routing]) is becoming a dominant solution. SR introduces traffic steering capabilities into an MPLS network [I-D.ietf-spring-segment-routing-mpls] by utilizing existing data plane capabilities (label pop and packet forwarding - "pop and go")

in combination with additions to existing IGPs [I-D.ietf-ospf-segment-routing-extensions], [I-D.ietf-isis-segment-routing-extensions], BGP (as BGP-LU) [I-D.ietf-mpls-rfc3107bis], or a centralized controller to distribute "per-hop" labels. An MPLS label stack can be imposed on a packet to describe a sequence of links/nodes to be transited by the packet; as each hop is transited, the label that represents it is popped from the stack and the packet is forwarded. Thus, on a packet-by-packet basis, traffic can be steered within the Data Center network.

Note that other Data Center data plane technologies also exist. While this document focuses on connecting domains that use MPLS Segment Routing, the techniques are equally applicable to non-MPLS domains (such as those using IP, VXLAN, and NVGRE). See Section 9 for details.

This document broadens the problem space to consider interconnection of any type of edge domain. These may be Data Center sites, but they may equally be access networks, VPN sites, or any other form of domain that includes packet sources and destinations. We particularly focus on "SR edge domains" being source or destination domains that utilize SR, but the domains could use other technologies as described in Section 9.

Backbone networks are commonly based on MPLS hardware. In these networks, a number of different options exist to establish TE paths. Among these options are static LSPs (perhaps set up by an SDN controller), LSP tunnels established using a signaling protocol (such as RSVP-TE), and inter-domain use of SR (as described above for intra-domain steering). Where traffic steering (without resource reservation) is needed, SR may be adequate. Where Traffic Engineering is needed (i.e., traffic steering with resource reservation) RSVP-TE or centralized SDN control are preferred. However, in a network that is fully managed and controlled through a centralized planning tool, resource reservation can be achieved and SR can be used for full Traffic Engineering. These solutions are already used in support of a number of edge-to-edge services such as L3VPN and L2VPN.

3.1. Characteristics of Solution Technologies

Each of the solution technologies mentioned in the previous section has certain characteristics, and the combined solution needs to recognize and address the characteristics in order to make a workable solution.

- When SR is used for traffic steering, the size of the MPLS label stack used in SR scales linearly with the length of the source

route. This can cause issues with MPLS implementations that only support label stacks of a limited size. For example, some MPLS implementations cannot push enough labels on the stack to represent an entire source route. Other implementations may be unable to do the proper "ECMP hashing" if the label stack is too long; they may be unable to read enough of the packet header to find an entropy label or to find the IP header of the payload. Increasing the packet header size also reduces the size of the payload that can be carried in an MPLS packet. There are techniques that can be used to reduce the size of the label stack. For example, a single label (known as a "binding SID") can be used to represent a sequence of nodes; this label can be replaced with a set of labels when the packet reaches the first node in the sequence. It is also possible to combine SR with conventional RSVP-TE by using a binding SID in the label stack to represent an LSP tunnel set up by RSVP-TE.

- Most of the work on using SR for traffic steering assumes that traffic only needs to be steered within a single administrative domain. If the backbone consists of multiple ASes that are part of a common administrative domain, the use of SR across the backbone may prove to be a challenge, and its use in the backbone may be limited to cases where private networks connect the domains, rather than cases where the domains are connected by third-party network operators or by the public Internet.

- RSVP-TE has been used to provide edge-to-edge tunnels through which flows to/from many endpoints can be routed, and this provides a reduction in state while still offering Traffic Engineering across the backbone network. However, this requires O(n2) connections and as the number of edge domains increases this becomes unsustainable.

- A centralized control system, while capable of producing more optimal results than a distributed control system, may present challenges in large and dynamic networks. It relies on all network state being held centrally, and it is difficult to make central control as robust and self-correcting as distributed control.

This paper introduces an approach that blends the best points of each of these solution technologies to achieve a trade-off where RSVP-TE tunnels in the backbone network are stitched together using SR, and end-to-end SR paths can be created under the control of a central controller with routing devolved to the constituent networks where possible.

4. Decomposing the Problem

It is important to decompose the problem to take account of different regions spanned by the end-to-end path. These regions may use different technologies and may be under different administrative control. The separation of administrative control is particularly important because the operator of one region may be unwilling to share information about their networks, and may be resistant to allowing a third party to exert control over their network resources.

Using the reference model in Figure 1, we can consider how to get a packet from Source1 to the Destination. The following decisions must be made:

- In which domain the Destination lies.
- Which exit point from Dom1 to use.
- Which entry point to Dom2 to use.
- How to reach the exit point of Dom1 from Source1.
- How to reach the entry point to Dom2 from the exit point of Dom1.
- How to reach the Destination from the entry point to Dom2.

As already mentioned, these decisions may be inter-related. This enables us to break down the problem into three steps:

1. Get the packet from Source1 to the exit point of Dom1.
2. Get the packet from exit point of Dom1 to entry point of Dom2.
3. Get the packet from entry point of Dom2 to Destination.

The solution needs to achieve this in a way that allows:

- Adequate discovery of preferred elements in the end-to-end path (such as location of destination, destination domain entry point).
- Full control of the end-to-end path if all of the operators are willing.
- Re-use of existing techniques and technologies.

From a technology point of view we must support several functions and mixtures of those functions:

- If the domain uses MPLS Segment Routing, the labels within the domain may be populated by any means including BGP-LU [I-D.ietf-mpls-rfc3107bis], IGP, and central control. Source routes within the domain may be expressed as label stacks pushed by a controller or computed by a source router, or expressed as a single label and programmed into the domain routers by a controller.

- If the domain uses other (non-MPLS) forwarding, the domain processing is specific to that technology. See Section 9 for details.

- If the domains use Segment Routing, the source and destination domains may or may not be in the same Segment Routing domain, so that the prefix-SIDs may be the same or different in the two domains.

- The backbone network may be a single private network under the control of the owner of the domains and comprising one or more ASes, or may be a network operated by one or more third parties.

- The backbone network may utilize MPLS Traffic Engineering tunnels in conjunction with MPLS Segment Routing and the domain-to-domain source route may be provided by stitching TE LSPs.

- A single controller may be used to handle the source and destination domains as well as the backbone network, or there may be a different controller for the backbone network separate from that that controls the two domains, or there may be separate controllers for each network. The controllers may cooperate and share information to different degrees.

All of these different decompositions of the problem reflect different deployment choices and different commercial and operational practices, each with different functional trade-offs. For example, with separate controllers that do not share information and that only cooperate to a limited extent, it will be possible to achieve end-to-end connectivity with optimal routing at each step (domain or backbone AS), but the end-to-end path that is achieved might not be optimal.

5. Solution Space

5.1. Global Optimization of the Paths

Global optimization of the path from one domain to another requires either that the source controller has a complete view of the end-to-

end topology or some form of cooperation between controllers (such as in BRPC in RFC 5441 [RFC5441]).

BGP-LS [RFC7752] can be used to provide the "source" controller with a view of the topology of the backbone. This requires some of the BGP speakers in each AS to have BGP-LS sessions to the controller. Other means of obtaining this view are of course possible.

5.2. Figuring Out the GWs at a Destination Domain for a Given Prefix

Suppose GW1 and GW2 both advertise a route to prefix X, each setting itself as next hop. One might think that the GWs for X could be inferred from the routes' next hop fields, but typically both routes do not get distributed across the backbone, only the "best" route, as selected by BGP. But the best route according to the BGP selection process might not be the route via the GW that we want to use for traffic engineering purposes.

The obvious solution would be to use the ADD-PATH mechanism [RFC7911] to ensure that all routes to X get advertised. However, even if one does this, the identity of the GWs would get lost as soon as the routes got distributed through an ASBR that sets next hop self. And if there are multiple ASes in the backbone, not only will the next hop change several times, but the ADD-PATH mechanism experiences scaling issues. So this "obvious" solution only works within a single AS.

A better solution can be achieved using the Tunnel Encapsulation [I-D.ietf-idr-tunnel-encaps] attribute as follows:

We define a new tunnel type, "SR tunnel" and when the GWs to a given domain advertise a route to a prefix X within the domain, they each include a Tunnel Encapsulation attribute with multiple remote endpoint sub-TLVs each identifying a specific GW to the domain.

In other words, each route advertised by any GW identifies all of the GWs to the same domain (see Section 9 for a discussion of how GWs discover each other). Therefore, only one of the routes needs to be distributed to other ASes, and it doesn't matter how many times the next hop changes, the Tunnel Encapsulation attribute (and its remote endpoint sub-TLVs) remains unchanged.

Further, when a packet destined for prefix X is sent on a TE path to GW1 we want the packet to arrive at GW1 carrying, at the top of its label stack, GW1's label for prefix X. To achieve this we will place the SID/SRGB in a sub-TLV of the Tunnel Encapsulation attribute. We will define the prefix-SID sub-TLV to be essentially identical in syntax to the prefix-SID attribute (see

[I-D.ietf-idr-bgp-prefix-sid]), but the semantics are somewhat different.

It is also possible to define an "MPLS Label Stack" sub-TLV for the Tunnel Encapsulation attribute, and put this in the "SR tunnel" TLV. This allows the destination GW to specify a label stack that it wants packets destined for prefix X to have. This label stack represents a source route through the destination domain.

5.3. Figuring Out the Backbone Egress ASBRs

We need to figure out the backbone egress ASBRs that are attached to a given GW at the destination domain this out in order to properly engineer the path across the backbone.

The "cleanest" way to figure this out is to have the backbone egress ASBRs distribute the information to the source controller using the EPE extensions of BGP-LS [I-D.ietf-idr-bgpls-segment-routing-epe]. The EPE extensions to BGP-LS allow a BGP speaker to say, "Here is a list of my EBGP neighbors, and here is a (locally significant) adjacency-SID for each one."

It may also be possible to consider utilizing cooperating PCEs or a Hierarchical PCE approach in RFC 6805 [RFC6805]. But it should be observed that this question is dependent on the question in Section 5.2. That is, it is not possible to even start the selection of egress ASBRs until it is known which GWs at the destination domain provide access to a given prefix. Once that question has been answered, any number of PCE approaches can be used to select the right egress ASBR and, more generally, the ASBR path across the backbone.

5.4. Making use of RSVP-TE LSPs Across the Backbone

There are a number of ways to carry traffic across the backbone from one domain to another. RSVP-TE is a popular tunneling mechanism in similar scenarios (e.g., L3VPN) because it allows for reservation of resources as well as traffic steering.

A controller can cause an RSVP-TE LSP to be set up by using PCEP to talk to the LSP headend, using PCEP extensions [I-D.ietf-pce-pce-initiated-lsp]. That draft specifies an "LSP-initiate" message that the controller uses to specify the RSVP-TE LSP endpoints, the ERO, a "symbolic pathname", and optionally other attributes (specified in the PCEP specification, RFC 5440 [RFC5440]) such as bandwidth.

When the headend receives an LSP-initiate message, it sets up the RSVP-TE LSP, assigns it a "PLSP-id", and reports the PLSP-id back to the controller in a PCRpt message [I-D.ietf-pce-stateful-pce]. The PCRpt message also contains the symbolic name that the controller assigned to the LSP, as well as containing some information identifying the LSP-initiate message from the controller, and details of exactly how the LSP was set up (RRO, bandwidth, etc.).

The headend can add to the PCRpt message a TE-PATH-BINDING TLV [I-D.sivabalan-pce-binding-label-sid]. This allows the headend to assign a "binding SID" to the LSP, and to report to the controller that a particular binding SID corresponds to a particular LSP. The binding SID is locally scoped to the headend.

The controller can make this label be part of the label stack that it tells the source (or the GW at the source domain) to put on the data packets being sent to prefix X. When the headend receives a packet with this label at the top of the stack it will send the packet onward on the LSP.

5.5. Data Plane

Consolidating all of the above, consider what happens when we want to move a data packet from Source to Destination in Figure 1via the following source route:

Source1---GW1b---PE2a---ASBR2a---ASBR3a---PE3a---GW2a---Destination

Further, assume that there is an RSVP-TE LSP from PE2a to ASBR2a that we want to use, as well as an RSVP-TE LSP from ASBR3a to PE3a that we want to use.

Let’s suppose that the Source pushes a label stack following instructions from the controller (for example, using BGP-LU [I-D.ietf-mpls-rfc3107bis]). We won’t worry for now about source routing through the domains themselves: that is, in practice there may be additional labels in the stack to cover the source route from the Source to GW1b and from GW2a to the Destination, but we will focus only on the labels necessary to leave the source domain, traverse the backbone, and enter the egress domain. So we only care what the stack looks like when the packet gets to GW1b.

When the packet gets to GW1b, the stack should have six labels:

Top Label:

Peer-SID or adjacency-SID identifying link or links to PE2a. These SIDs are distributed from GW1b to the controller via the EPE

extensions of BGP-LS. (This label will get popped by GW1b, which will then send the packet to PE2a.)

Second Label:

Binding SID advertised by PE2a to the controller for the RSVP-TE LSP to ASBR2a. This binding SID is advertised via the PCEP extensions discussed above. (This label will get swapped by PE2a for the label that the LSP's next hop has assigned to the LSP.)

Third Label:

Peer-SID or adjacency-SID identifying link or links to ASBR3a, as advertised to the controller by ASBR2a using the BGP-LS EPE extensions. (This label gets popped by ASBR2a, which then sends the packet to ASBR3a.)

Fourth Label:

Binding SID advertised by ASBR3a for the RSVP-TE LSP to PE3a. This binding SID is advertised via the PCEP extensions discussed above. ASBR3a treats this label just like PE2a treated the second label above.

Fifth label:

Peer-SID or adjacency-SID identifying link or links to GW2a, as advertised to the controller by ASBR3a using the BGP-LS EPE extensions. ASBR3a pops this label and sends the packet to GW2a.

Sixth Label:

Prefix-SID or other label identifying the Destination advertised in a Tunnel Encapsulation attribute by GW2a. (This can be omitted if GW2a is happy to accept IP packets, or prefers a VXLAN tunnel for example. That would be indicated through the Tunnel Encapsulation attribute of course.)

Note that the size of the label stack is proportional to the number of RSVP-TE LSPs that get stitched together by SR.

See Section 7 for some detailed examples that show the concrete use of labels in a sample topology.

In the above example, all labels except the sixth are locally significant labels: peer-SIDs, binding SIDs, or adjacency-SIDs. Only the sixth label, a prefix-SID, has a domain-wide unique value. To impose that label, the source needs to know the SRGB of GW2a. If all

nodes have the same SRGB, this is not a problem. Otherwise, there are a number of different ways GW3a can advertise its SRGB. This can be done via the segment routing extensions of BGP-LS, or it can be done using the prefix-SID attribute or BGP-LU [I-D.ietf-mpls-rfc3107bis], or it can be done using the BGP Tunnel Encapsulation attribute. The exact technique to be used will depend on the details of the deployment scenario.

The reason the above example is primarily based on locally significant labels is that it creates a "strict source route", and it presupposes the EPE extensions of BGP-LS. In some scenarios, the EPE extension to BGP-LS might not be available (or BGP-LS might not be available at all). In other scenarios, it may be desirable to steer a packet through a "loose source route". In such scenarios, the label stack imposed by the source will be based upon a sequence of domain-wide unique "node-SIDs", each representing one of the hops of source route. Each label has to be computed by adding the corresponding node-SID to the SRGB of the node that will act upon the label. One way to learn the node-SIDs and SRGBs is to use the segment routing extensions of BGP-LS. Another way is to use BGP-LU as follows. Each node that may be part of a source route would originate a BGP-LU route with one of its own loopback addresses as the prefix. The BGP prefix-SID attribute would be attached to this route. The prefix-SID attribute would contain a SID, which is the domain-wide unique SID corresponding to the node's loopback address. The attribute would also contain the node's SRGB.

While this technique is useful when BGP-LS is not available, it presupposes that the source controller has some other means of discovering the topology. In this document, we focus primarily on the scenario where BGP-LS, rather than BGP-LU, is used.

5.6. Centralized and Distributed Controllers

A controller or set of controllers are needed to collate topology and TE information from the constituent networks, to apply policies and service requirements to compute paths across those networks, to select an end-to-end path, and to program key nodes in the network to take the right forwarding actions (pushing label stacks, stitching LSPs, forwarding traffic).

- It is commonly understood that a fully optimal end-to-end path can only be computed with full knowledge of the end-to-end topology and available Traffic Engineering resources. Thus, one option is for all information about the domain networks and backbone network to be collected by a central controller that makes all path computations and is responsible for issuing the necessary programming commands. Such a model works best when there is no

commercial or administrative impediment (for example, where the domains and the backbone network are owned and operated by the same organization). There may, however, be some scaling concerns if the component networks are large.

In this mode of operation, each network may use BGP-LS to export Traffic Engineering and topology information to the central controller, and the controller may use PCEP to program the network behavior.

- A similar centralized control mechanism can be used with a scalability improvement that risks a reduction in optimality. In this case, the domain networks can export to the controller just the feasibility of connectivity between data source/sink and gateway, perhaps enhancing this with some information about the Traffic Engineering metrics of the path.

  This approach allows the central controller to understand the end-to-end path that it is selecting, but not to control it fully. The source route from data source to domain egress gateway is left to the source host or a controller in the source domain, while the source route from domain ingress gateway to destination is left as a decision for the domain ingress gateway or to a controller in the destination domain.

  This mode of operation still leaves overall control with a centralized server and that may not be considered suitable when there is separate commercial or administrative control of the networks.

- When there is separate commercial or administrative control of the networks the domain operator will not want the backbone operator to have control of the source routes within the domain and may be reluctant to disclose any information about the topology or resource availability within the domains. Conversely, the backbone operator may be very unwilling to allow the domain operator (a customer) any control over or knowledge about the backbone network.

  This "problem" has already been solved for Traffic Engineering in MPLS networks that span multiple administrative domains and leads to multiple potential solutions:

  * Per-domain path computation in RFC 5152 [RFC5152] can be seen as "best effort optimization". In this mode the controller for each domain is responsible for finding the best path to the next domain, but has no way of knowing which is the best exit

point from the local domain. The resulting path may end up significantly sub-optimal or even blocked.

* Backward recursive path computation (BRPC) in RFC 5441 [RFC5441] is a mechanism that allows controllers to cooperate across a small set of domains (such as ASes) to build a tree of possible paths and so allow the controller for the ingress domain to select the optimal path. The details of the paths within each domain that might reveal confidential information can be hidden using Path Keys in RFC 5520 [RFC5520] BRPC produces optimal paths but scales poorly with an increase in domains and with an increase in connectivity between domains. It can also lead to slow computation times.

* Hierarchical PCE (H-PCE) in RFC 6805 [RFC6805] is a two-level cooperation process between PCEs. The child PCEs remain responsible for computing paths across their domains, and they coordinate with a parent PCE that stitches these paths together to form the end-to-end path. This approach has many similarities with BRPC but can scale better through the maintenance of "domain topology" that shows how the domains are interconnected, and through the ability to pipe-line computation requests to all of the child domains. It has the drawback that some party has to own and operate the parent PCE.

* An alternative approach is documented by the TEAS working group [RFC7926]. In this model each network advertises to controllers for adjacent networks (using BGP-LS) selected information about potential connectivity across the network. It does not have to show full topology and can make its own decisions about which paths it considers optimal for use by its different neighbors and customers. This approach is suitable for the End-to-End Domain Interconnect Traffic Steering problem where the backbone is under different control from the domains because it allows the overlay nature of the use of the backbone network to be treated as a peer network relationship by the controllers of the domains - the domains can be operated using a single controller or a separate controller for each domain.

It is also possible to operate domain interconnection when some or all domains do not have a controller. Segment Routing is capable of routing a packet toward the next hop based on the top label on the stack, and that label does not need to indicate an immediately adjacent node or link. In these cases, the packet may be forwarded untouched, or the forwarding router may impose a locally-determined additional set of labels that define the path to the next hop.

PCE can be used to instruct the source host or a transit node on what label stacks to add to packets. That is, a node that needs to impose labels (either to start routing the packet from the source host, or to advance the packet from a transit router toward the destination) can determine the label stack to use based on local function or can have that stack supplied by a PCE. The PCE Protocol (PCEP) has been extended to allow the PCE to supply a label stack for reaching a specific destination either in response to a request or in an unsolicited manner [I-D.ietf-pce-segment-routing].

6. BGP-LS Considerations

This section gives an overview of the use of BGP-LS to export an abstraction (or summary) of the connectivity across the backbone network by means of two figures that show different views of a sample network.

Figure 2 shows a more complex reference architecture.

Figure 3 represents the minimum set of nodes and links that need to be advertised in BGP-LS with SR in order to perform Domain Interconnect with traffic engineering across the backbone network: the PEs, ASBRs, and gateways (GWs), and the links between them. In particular, EPE [I-D.ietf-idr-bgpls-segment-routing-epe] and TE information with associated segment IDs is advertised in BGP-LS with SR.

Links that are advertised may be physical links, links realized by LSP tunnels, or abstract links. It is assumed that intra-AS links are either real links, RSVP-TE LSPs with allocated bandwidth, or SR TE policies as described in [I-D.previdi-idr-segment-routing-te-policy]. Additional nodes internal to an AS and their links to PEs, ASBRs, and/or GWs may also be advertised (for example to avoid full mesh problems).

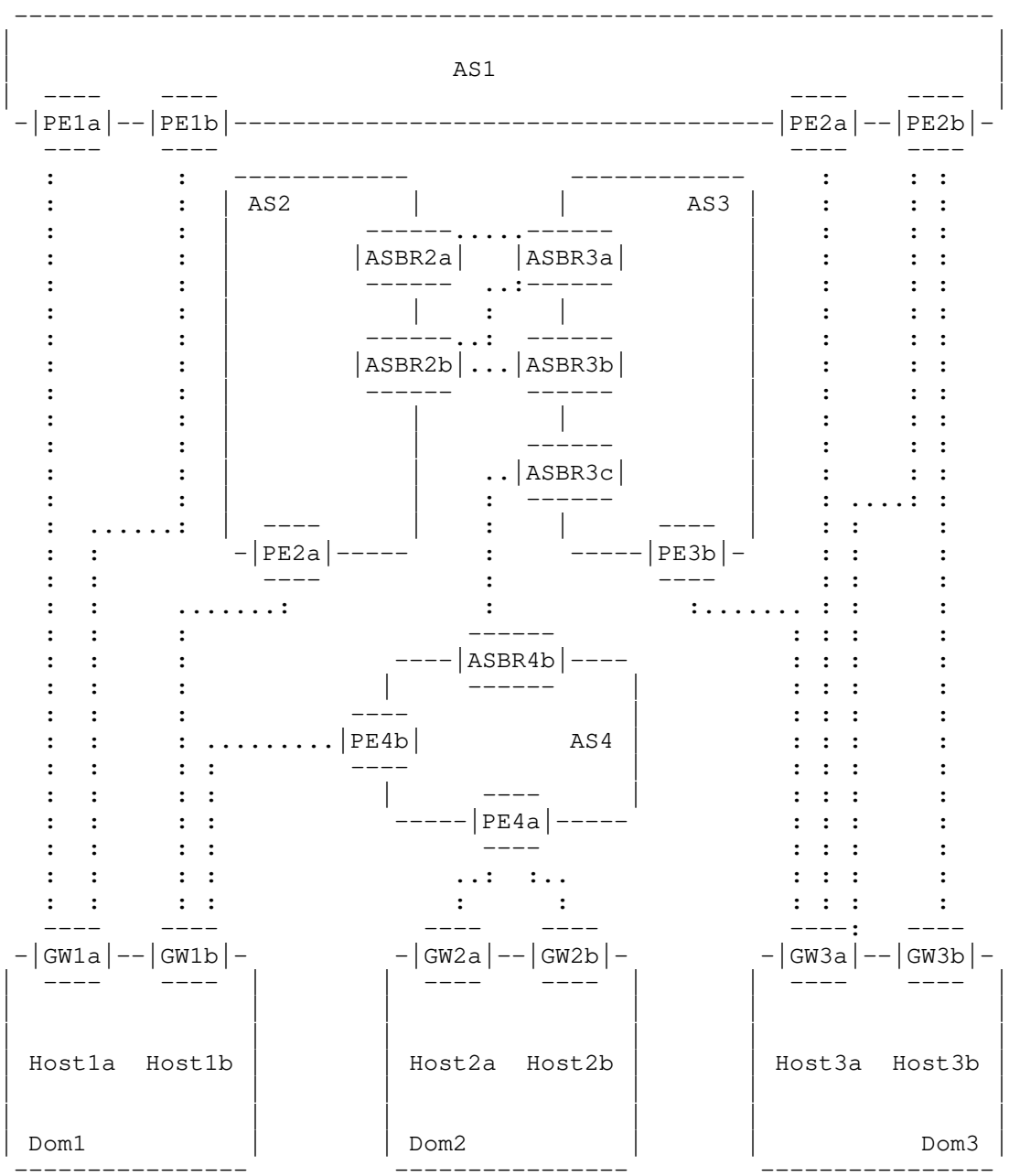


Figure 2: Network View of Example Configuration

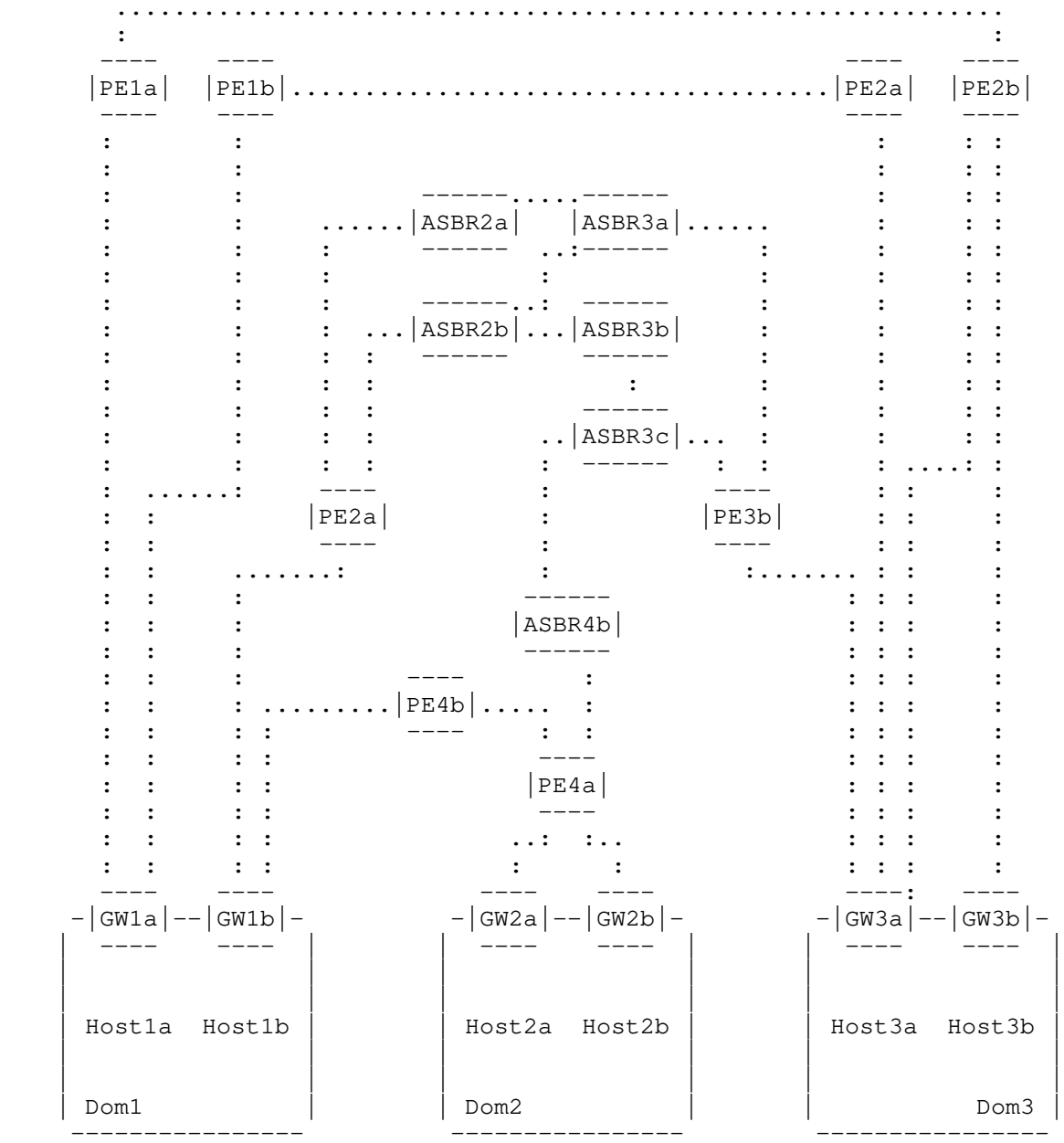


Figure 3: Topology View of Example Configuration

A node (a PCE, router, or host) that is computing a full or partial path correlates the topology information disseminated in BGP-LS with SR with the information advertised with the Tunnel Encapsulation attributes to compute that path and obtain the SIDs for the elements on that path. In order to allow a source host to compute exit points from its domain, some subset of the above information needs to be disseminated within that domain.

What is advertised external to a given AS is controlled by policy at the ASes' PEs, ASBRs, and GWs. Central control of what each node should advertise, based upon analysis of the network as a whole, is an important additional function. This and the amount of policy involved may make the use of a Route Reflector an attractive option.

The configuration of which links to other nodes and the characteristics of those links a given node advertises in BGP-LS with SR is done locally at each node and pairwise coordination between link end-points is required to ensure consistency.

Path Weighted ECMP (PWECMP) is assumed to be used by a GW for a given source domain to send all flows to a given destination domain using all paths in the backbone network to that destination domain in proportion to the minimum bandwidth on each path. PWECMP is also assumed to be used by hosts within a source domain to send flows to that domain's GWs.

7. Worked Examples

Figure 4 shows a view of the links, paths, and labels that can be assigned to part of the sample network shown in Figure 2 and Figure 3. The double-dash lines (===) indicate LSP tunnels across backbone ASes and dotted lines (...) are physical links.

At each node, a label may be assigned to each outgoing link. This is shown in Figure 4. For example, at GW1a the label L201 is assigned to the link connecting GW1a to PE1a. At PE1c, the label L302 is assigned to the link connecting PE1c to GW3b. Labels ("binding SIDs") may also be assigned to RSVP-TE LSPs. For example, at PE1a, label L202 is assigned to the RSVP-TE LSP leading from PE1a to PE1c.

At the destination domain, labels L302 and L305 are "node-SIDs"; they represent GW3b and Host3b respectively, rather than representing particular links.

When a node processes a packet, the label at the top of the label stack indicates the link (or RSVP-TE LSP) on which that node is to transmit the packet. The node pops that label off the label stack before transmitting the packet on the link. However, if the top

label is a node-SID, the node processing the packet is expected to transmit the packet on whatever link it regards as the shortest path to the node represented by the label.

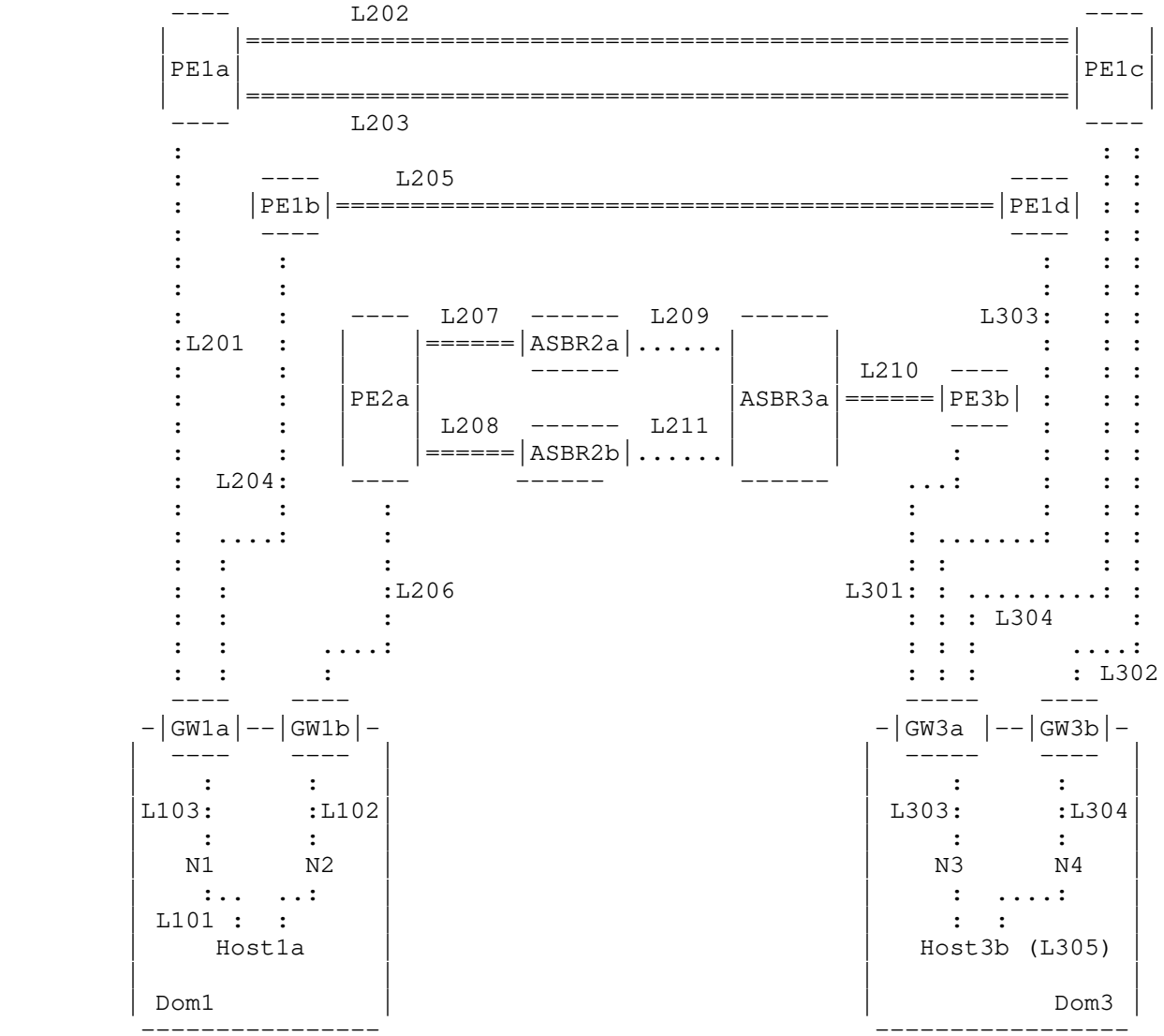


Figure 4: Tunnels and Labels in Example Configuration

Let's consider several different possible ways to direct a packet from Host1a in Dom1 to Host3b in Dom3.

a. Full source route imposed at source

In this case it is assumed that the entity responsible for determining an end-to-end path has access to the topologies of both domains and of the backbone network. This might happen if all of the networks are owned by the same operator in which case the information can be shared into a single database for use by an offline tool, or the information can be distributed using routing protocols such that the source host can see enough to select the path. Alternatively, the end-to-end path could be produced through cooperation between computation entities each responsible for different domains along the path.

If the path is computed externally it is pushed to the source host. Otherwise, it is computed by the source host itself.

Suppose it is desired for a packet from Host1a to travel to Host3b via the following source route:

Host1a->N1->GW1a->PE1a->(RSVP-TE LSP)->PE1c->GW3b->N4->Host3b

Host1a would impose the following label stack would be imposed (with the first label representing the top of stack), and then send the packet to N1:

L103, L201, L202, L302, L304, L305

N1 sees L103 at the top of the stack, so it pops the stack and forwards the packet to GW1a. GW1a sees L201 at the top of the stack, so it pops the stack and forwards the packet to PE1a. PE1a sees L202 at the top of the stack, so it pops the stack and forwards the packet over the RSVP-TE LSP to PE1c. As the packet travels over this LSP, its top label will be an RSVP-TE signaled label representing the LSP. That is, PE1a imposes an additional label stack entry for the tunnel LSP.

At the end of the LSP tunnel, the MPLS tunnel label will be popped, and PE1c will see L302 at the top of the stack. PE1c pops the stack and forwards the packet to GW3b. GW3b will see L304 at the top of the stack, so it pops the stack and forwards the packet to N4. Finally, N4 sees L305 at the top of the stack, so it pops the stack and forwards the packet to Host 3b. No remote visibility into Dom3.

b. It is possible that the source domain does not have visibility into the destination domain.

This occurs if the destination domain does not export its topology, but even in this case, it will export reachability information so that the source host or the path computation entity will know:

* The GWs through which the destination can be reached.

* The SID to use for the destination prefix.

Suppose we want a packet to follow the source route:

Host1a->N1->GW1a->PE1a->(RSVP-TE LSP)->PE1c->GW3b->...->Host3b

(The ellipsis indicates a part of the path that is not explicitly specified.) Thus, the label stack imposed at the source host would be:

L103, L201, L202, L302, L305

Processing is as per case a., but when the packet reaches the GW of the destination domain, it can either simply forward the packet along the shortest path to Host3b, or it can insert additional labels to direct the path to the destination.

c. Dom1 only has reachability information

The source domain (or the path computation entity) may be further restricted in its view of the network. It is possible that it knows the location of the destination in the destination domain, and knows the GWs to the destination domain that provide reachability to the destination, but that it has no view of the backbone network. This leads to the packet being forwarded in a manner similar to 'per-domain path computation' described in Section 5.6.

At the source host a simple label stack is imposed navigating the domain and indicating the destination GW and the destination host.

L101, L103, L302, L305

As the packet leaves the source domain, the source GW determines the PE to use to enter the backbone using nothing more than the BGP preferred route to the destination GW.

When the packet reaches the first PE it has a label stack just identifying the destination GW and host (L302, L305). The PE uses information it has about the backbone network topology and available LSPs to select an LSP tunnel, impose the tunnel label, and forward the packet.

When the packet reaches the end of the LSP tunnel, it is processed as described in case b.

d. Stitched LSPs across the backbone

A variant of all these cases arises when the packet is sent using a path that spans multiple ASes. For example, one that crosses AS2 and AS3 as shown in Figure 2.

In this case, basing the example on case a., the source host would impose the label stack:

L102, L206, L207, L209, L210, L301, L303, L305

and would then send the packet to N2.

When the packet reaches PE2a as previously described and the top label (L207) selects an LSP tunnel that leads to ASBR2a. At the end of that LSP tunnel the next label (L209) routes the packet from ASBR2a to the ASBR3a, where the next label (L210) identifies the next LSP tunnel to use. Thus, SR has been used to stitch together LSPs to make a longer path segment. As the packet emerges from the final LSP tunnel, forwarding continues as previously described.

8. Label Stack Depth Considerations

As described in Section 3.1, one of the issues with a Segment Routing approach is that the label stack can get large, for example when the source route becomes long. A mechanism to mitigate this problem is needed if the solution is to be fully applicable in all environments.

An Internet-Draft called "Segment Routing Traffic Engineering Policy using BGP" [I-D.previdi-idr-segment-routing-te-policy] introduces the concept of hierarchical source routes as a way to compress source route headers. It functions by having the egress node for a set of source routes advertise those source routes along with an explicit request that each node that is an ingress node for one or more of those source routes should advertise a binding SID for the set of source routes for which it is the ingress. (It should be noted that the set of source routes can either be advertised by the egress node as described here, or could be advertised by a controller on behalf

of the egress node.) Such an ingress node advertises its set of source routes and a binding SID as an adjacency in BGP-LS as described in Section 6. These source routes represent the weighted ECMP paths between the ingress node and the egress node. (Note also that the binding SID may be supplied by the node that advertises the source routes - the egress or the controller - or may be chosen by ingress node.)

A remote node that wishes to reach the egress node would then construct a source route consisting of the segment IDs necessary to reach one of the ingress nodes for the path it wishes to use along with the binding SID that the ingress node advertised to identify the set of paths. When the selected ingress node receives a packet with a binding SID it has advertised, it replaces the binding SID with the labels for one of its source routes to the egress node (it will choose one of the source routes in the set according to its own weighting algorithms and policy).

8.1. Worked Example

Consider the topology in Figure 4. Suppose that it is desired to construct full segment routed paths from ingress to egress, but that the resulting label stack (segment route) is too large. In this case the gateways to Dom3 (GW3a and GW3b) can advertise all of the source routes from the gateways to Dom1 (GW1a and GW1b). The gateways to Dom1 then assign binding SIDs to those source routes and advertise those SIDs into BGP-LS.

Thus, GW3b would advertise the two source routes (L201, L202, L302 and L201, L203, L302), and GW1a would advertise into BGP-LS its adjacency to GW3b along with a binding SID. Should Host1a wish to send a packet via GW1a and GW3b, it can include L103 and this binding SID in the source route. GW1a is free to choose which source route to use between itself and GW3b using its weighted ECMP algorithm.

Similarly, GW3a would advertise the following set of source routes:

- L201, L202, L304
- L201, L203, L304
- L204, L205, L303
- L206, L207, L209, L210, L301
- L206, L208, L211, L210, L301

GW1a would advertise a binding SID for the first three, and GW1b would advertise a binding SID for the other two.

9. Gateway Considerations

As described in Section 5, we define a new tunnel type, "SR tunnel", and when the GWs to a given domain advertise a route to a prefix X within the domain, they will each include a Tunnel Encapsulation attribute with multiple tunnel instances each of type "SR tunnel", one for each GW and each containing a Remote Endpoint sub-TLV with that GW's address.

In other words, each route advertised by any GW identifies all of the GWs to the same domain.

Therefore, even if only one of the routes is distributed to other ASes, it will not matter how many times the next hop changes, as the Tunnel Encapsulation attribute (and its remote endpoint sub-TLVs) will remain unchanged.

9.1. Domain Gateway Auto-Discovery

To allow a given domain's GWs to auto-discover each other and to coordinate their operations, the following procedures are implemented [I-D.ietf-bess-datacenter-gateway]:

- Each GW is configured with an identifier for the domain that is common across all GWs to the domain (i.e., all GWs to all domains that are connected) and unique across all domains that are connected.

- A route target [RFC4360] is attached to each GW's auto-discovery route and has its value set to the domain identifier.

- Each GW constructs an import filtering rule to import any route that carries a route target with the same domain identifier that the GW itself uses. This means that only these GWs will import those routes and that all GWs to the same domain will import each other's routes and will learn (auto-discover) the current set of active GWs for the domain.

- The auto-discovery route each GW advertises consists of the following:

  * An IPv4 or IPv6 NLRI containing one of the GW's loopback addresses (that is, with AFI/SAFI that is one of 1/1, 2/1, 1/4, 2/4).

* A Tunnel Encapsulation attribute containing the GW's encapsulation information, which at a minimum consists of an SR tunnel TLV (type to be allocated by IANA) with a Remote Endpoint sub-TLV [I-D.ietf-idr-tunnel-encaps].

To avoid the side effect of applying the Tunnel Encapsulation attribute to any packet that is addressed to the GW, the GW should use a different loopback address.

Each GW will include a Tunnel Encapsulation attribute for each GW that is active for the domain (including itself), and will include these in every route advertised externally to the domain by each GW. As the current set of active GWs changes (due to the addition of a new GW or the failure/removal of an existing GW) each externally advertised route will be re-advertised with the set of SR tunnel instances reflecting the current set of active GWs.

9.2. Relationship to BGP Link State and Egress Peer Engineering

When a remote GW receives a route to a prefix X it can use the SR tunnel instances within the contained Tunnel Encapsulation attribute to identify the GWs through which X can be reached. It uses this information to compute SR TE paths across the backbone network looking at the information advertised to it in SR BGP Link State (BGP-LS) [I-D.gredler-idr-bgp-ls-segment-routing-ext] and correlated using the domain identity. SR Egress Peer Engineering (EPE) [I-D.ietf-idr-bgpls-segment-routing-epe] can be used to supplement the information advertised in the BGP-LS.

9.3. Advertising a Domain Route Externally

When a packet destined for prefix X is sent on an SR TE path to a GW for the domain containing X, it needs to carry the receiving GW's label for X such that this label rises to the top of the stack before the GW complete its processing of the packet. To achieve this we place a prefix-SID sub-TLV for X in each SR tunnel instance in the Tunnel Encapsulation attribute in the externally advertised route for X.

Alternatively, if the GWs for a given domain are configured to allow remote GWs to perform SR TE through that domain for a prefix X, then each GW computes an SR TE path through that domain to X from each of the current active GWs and places each in an MPLS label stack sub-TLV [I-D.ietf-idr-tunnel-encaps] in the SR tunnel instance for that GW.

9.4. Encapsulations

If the GWs for a given domain are configured to allow remote GWs send them a packet in that domain's native encapsulation, then each GW will also include multiple instances of a tunnel TLV for that native encapsulation, one for each GW and each containing a remote endpoint sub-TLV with that GW's address, in externally advertised routes. A remote GW may then encapsulate a packet according to the rules defined via the sub-TLVs included in each of the tunnel TLV instances.

10. Security Considerations

There are several security domains and associated threats in this architecture. SR is itself a data transmission encapsulation that provides no additional security, so security in this architecture relies on higher layer mechanisms (for example, end-to-end encryption of pay-load data), security of protocols used to establish connectivity and distribute network information, and access control so that control plane and data plane packets are not admitted to the network from outside.

This architecture utilizes a number of control plane protocols within domains, within the backbone, and north-south between controllers and domains. Only minor modifications are made to BGP as described in [I-D.ietf-bess-datacenter-gateway], otherwise this achetecture uses existing protocols and extensions so no new security risks are introduced.

Special care should, however, be taken when routing protocols export or import information from or to domains that might have a security model based on secure boundaries and internal mutual trust. This is notable when:

- BGP-LS is used to export topology information from within a domain to a controller that may be sited outside the domain.

- A southbound protocol such as BGP-LU or Netconf is used to install state in the network from a controller that may be sited outside the domain.

In these cases protocol security mechanisms should be used protect the information in transit and to ensure that information entering or leaving the domain and to authenticate the out of domain node (the controller) to ensure that confidential/private information is not lost and that data or configuration is not falsified.

11. Management Considerations

TBD

12. IANA Considerations

This document makes no requests for IANA action.

13. Acknowledgements

TBD

14. Informative References

[I-D.gredler-idr-bgp-ls-segment-routing-ext]
Previdi, S., Psenak, P., Filsfils, C., Gredler, H., Chen, M., and j. jefftant@gmail.com, "BGP Link-State extensions for Segment Routing", draft-gredler-idr-bgp-ls-segment-routing-ext-04 (work in progress), October 2016.

[I-D.ietf-bess-datacenter-gateway]
Drake, J., Farrel, A., Rosen, E., Patel, K., and L. Jalil, "Gateway Auto-Discovery and Route Advertisement for Segment Routing Enabled Domain Interconnection", draft-ietf-bess-datacenter-gateway-00 (work in progress), October 2017.

[I-D.ietf-idr-bgp-prefix-sid]
Previdi, S., Filsfils, C., Lindem, A., Sreekantiah, A., and H. Gredler, "Segment Routing Prefix SID extensions for BGP", draft-ietf-idr-bgp-prefix-sid-07 (work in progress), October 2017.

[I-D.ietf-idr-bgpls-segment-routing-epe]
Previdi, S., Filsfils, C., Patel, K., Ray, S., and J. Dong, "BGP-LS extensions for Segment Routing BGP Egress Peer Engineering", draft-ietf-idr-bgpls-segment-routing-epe-13 (work in progress), June 2017.

[I-D.ietf-idr-tunnel-encaps]
Rosen, E., Patel, K., and G. Velde, "The BGP Tunnel Encapsulation Attribute", draft-ietf-idr-tunnel-encaps-07 (work in progress), July 2017.

[I-D.ietf-isis-segment-routing-extensions]
Previdi, S., Filsfils, C., Bashandy, A., Gredler, H., Litkowski, S., Decraene, B., and j. jefftant@gmail.com, "IS-IS Extensions for Segment Routing", draft-ietf-isis-segment-routing-extensions-13 (work in progress), June 2017.

[I-D.ietf-mpls-rfc3107bis]
Rosen, E., "Using BGP to Bind MPLS Labels to Address Prefixes", draft-ietf-mpls-rfc3107bis-04 (work in progress), August 2017.

[I-D.ietf-ospf-segment-routing-extensions]
Psenak, P., Previdi, S., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPF Extensions for Segment Routing", draft-ietf-ospf-segment-routing-extensions-21 (work in progress), October 2017.

[I-D.ietf-pce-pce-initiated-lsp]
Crabbe, E., Minei, I., Sivabalan, S., and R. Varga, "PCEP Extensions for PCE-initiated LSP Setup in a Stateful PCE Model", draft-ietf-pce-pce-initiated-lsp-11 (work in progress), October 2017.

[I-D.ietf-pce-segment-routing]
Sivabalan, S., Filsfils, C., Tantsura, J., Henderickx, W., and J. Hardwick, "PCEP Extensions for Segment Routing", draft-ietf-pce-segment-routing-10 (work in progress), October 2017.

[I-D.ietf-pce-stateful-pce]
Crabbe, E., Minei, I., Medved, J., and R. Varga, "PCEP Extensions for Stateful PCE", draft-ietf-pce-stateful-pce-21 (work in progress), June 2017.

[I-D.ietf-spring-segment-routing]
Filsfils, C., Previdi, S., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", draft-ietf-spring-segment-routing-13 (work in progress), October 2017.

[I-D.ietf-spring-segment-routing-mpls]
Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls-10 (work in progress), June 2017.

[I-D.previdi-idr-segment-routing-te-policy]
Previdi, S., Filsfils, C., Mattes, P., Rosen, E., and S. Lin, "Advertising Segment Routing Policies in BGP", draft-previdi-idr-segment-routing-te-policy-07 (work in progress), June 2017.

[I-D.sivabalan-pce-binding-label-sid]
Sivabalan, S., Filsfils, C., Previdi, S., Tantsura, J., Hardwick, J., and D. Dhody, "Carrying Binding Label/ Segment-ID in PCE-based Networks.", draft-sivabalan-pce-binding-label-sid-03 (work in progress), July 2017.

[RFC4360] Sangli, S., Tappan, D., and Y. Rekhter, "BGP Extended Communities Attribute", RFC 4360, DOI 10.17487/RFC4360, February 2006, <https://www.rfc-editor.org/info/rfc4360>.

[RFC5152] Vasseur, JP., Ed., Ayyangar, A., Ed., and R. Zhang, "A Per-Domain Path Computation Method for Establishing Inter-Domain Traffic Engineering (TE) Label Switched Paths (LSPs)", RFC 5152, DOI 10.17487/RFC5152, February 2008, <https://www.rfc-editor.org/info/rfc5152>.

[RFC5440] Vasseur, JP., Ed. and JL. Le Roux, Ed., "Path Computation Element (PCE) Communication Protocol (PCEP)", RFC 5440, DOI 10.17487/RFC5440, March 2009, <https://www.rfc-editor.org/info/rfc5440>.

[RFC5441] Vasseur, JP., Ed., Zhang, R., Bitar, N., and JL. Le Roux, "A Backward-Recursive PCE-Based Computation (BRPC) Procedure to Compute Shortest Constrained Inter-Domain Traffic Engineering Label Switched Paths", RFC 5441, DOI 10.17487/RFC5441, April 2009, <https://www.rfc-editor.org/info/rfc5441>.

[RFC5520] Bradford, R., Ed., Vasseur, JP., and A. Farrel, "Preserving Topology Confidentiality in Inter-Domain Path Computation Using a Path-Key-Based Mechanism", RFC 5520, DOI 10.17487/RFC5520, April 2009, <https://www.rfc-editor.org/info/rfc5520>.

[RFC6805] King, D., Ed. and A. Farrel, Ed., "The Application of the Path Computation Element Architecture to the Determination of a Sequence of Domains in MPLS and GMPLS", RFC 6805, DOI 10.17487/RFC6805, November 2012, <https://www.rfc-editor.org/info/rfc6805>.

[RFC7752] Gredler, H., Ed., Medved, J., Previdi, S., Farrel, A., and S. Ray, "North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP", RFC 7752, DOI 10.17487/RFC7752, March 2016, <https://www.rfc-editor.org/info/rfc7752>.

[RFC7855] Previdi, S., Ed., Filsfils, C., Ed., Decraene, B., Litkowski, S., Horneffer, M., and R. Shakir, "Source Packet Routing in Networking (SPRING) Problem Statement and Requirements", RFC 7855, DOI 10.17487/RFC7855, May 2016, <https://www.rfc-editor.org/info/rfc7855>.

[RFC7911] Walton, D., Retana, A., Chen, E., and J. Scudder, "Advertisement of Multiple Paths in BGP", RFC 7911, DOI 10.17487/RFC7911, July 2016, <https://www.rfc-editor.org/info/rfc7911>.

[RFC7926] Farrel, A., Ed., Drake, J., Bitar, N., Swallow, G., Ceccarelli, D., and X. Zhang, "Problem Statement and Architecture for Information Exchange between Interconnected Traffic-Engineered Networks", BCP 206, RFC 7926, DOI 10.17487/RFC7926, July 2016, <https://www.rfc-editor.org/info/rfc7926>.

Authors' Addresses

Adrian Farrel
Juniper Networks

Email: afarrel@juniper.net

John Drake
Juniper Networks

Email: jdrake@juniper.net

SPRING Working Group
Internet-Draft
Intended status: Informational
Expires: April 16, 2019

A. Farrel
J. Drake
Juniper Networks

October 13, 2018

Interconnection of Segment Routing Domains - Problem Statement and Solution Landscape
draft-farrel-spring-sr-domain-interconnect-05

Abstract

Segment Routing (SR) is a forwarding paradigm for use in MPLS and IPv6 networks. It is intended to be deployed in discrete domains that may be data centers, access networks, or other networks that are under the control of a single operator and that can easily be upgraded to support this new technology.

Traffic originating in one SR domain often terminates in another SR domain, but must transit a backbone network that provides interconnection between those domains.

This document describes a mechanism for providing connectivity between SR domains to enable end-to-end or domain-to-domain traffic engineering.

The approach described allows connectivity between SR domains, utilizes traffic engineering mechanisms (RSVP-TE or Segment Routing) across the backbone network, makes heavy use of pre-existing technologies, and requires the specification of very few additional mechanisms.

This document provides some background and a problem statement, explains the solution mechanism, gives references to other documents that define protocol mechanisms, and provides examples. It does not define any new protocol mechanisms.

Copyright Notice

Table of Contents

1. Introduction

Data Centers are a growing market sector. They are being set up by new specialist companies, by enterprises for their own use, by legacy ISPs, and by the new wave of network operators. The networks inside Data Centers are currently well-planned, but the traffic loads can be unpredictable. There is a need to be able to direct traffic within a Data Center to follow a specific path.

Data Centers are attached to external ("backbone") networks to allow access by users and to facilitate communication among Data Centers. An individual Data Center may be attached to multiple backbone networks, and may have multiple points of attachment to each backbone network. Traffic to or from a Data Center may need to be directed to or from any of these points of attachment.

Segment Routing (SR) is a technology that places forwarding state into each packet as a stack of loose hops. SR is an option for building Data Centers, and is also seeing increasing traction in edge and access networks as well as in backbone networks. It is typically deployed in discrete domains that are under the control of a single operator and that can easily be upgraded to support this new technology.

Traffic originating in one SR domain often terminates in another SR domain, but must transit a backbone network that provides interconnection between those domains. This document describes an approach that builds on existing technologies to produce mechanisms that provide scalable and flexible interconnection of SR domains, and that will be easy to operate.

The approach described allows end-to-end connectivity between SR domains across an MPLS backbone network, utilizes traffic engineering mechanisms (RSVP-TE or Segment Routing) across the backbone network, makes heavy use of pre-existing technologies, and requires the specification of very few additional mechanisms.

This document provides some background and a problem statement, explains the solution mechanism, gives references to other documents that define protocol mechanisms, and provides examples. It does not define any new protocol mechanisms.

1.1. Terminology

This document uses Segment Routing terminology from [RFC7855] and [RFC8402]. Particular abbreviations of note are:

- SID: a segment identifier
- SRGB: an SR Global Block

In the context of this document, the terms "optimal" and "optimality" refer to making the best possible use of network resources, and achieving network paths that best meet the objectives of the network operators and customers.

Further terms are defined in Section 2.

2. Problem Statement

Consider the network in Figure 1. Without loss of generality, this figure can be used to represent the architecture and problem space for steering traffic within and between SR edge domains. The figure shows a single destination for all traffic that we will consider.

In describing the problem space and the solution we use five terms for network nodes as follows:

SR domain : This term is defined in [RFC8402]. In this document, an SR domain is a collection of SR-capable nodes under the care of one administrator or protocol. This may mean that each edge network is an SR domain attached to the backbone network through one or more gateways. Examples include, access networks, Data Center sites, backbone networks that run SR, and blessings of unicorns.

Host : A node within an edge domain. It may be an end system or a transit node in the edge domain.

Gateway (GW) : Provides access to or from an edge domain. Examples are Customer Edge nodes (CEs), Autonomous System Border Routers (ASBRs), and Data Center gateways.

Provider Edge (PE) : Provides access to or from the backbone network.

Autonomous System Border Router (ASBR) : Provides access to one Autonomous System (AS) in the backbone network from another AS in the backbone network.

These terms can be seen used in Figure 1 where the various sources and the destination are hosts. In this figure we distinguish between the PEs that provide access to the backbone networks and the Gateways that provide access to the SR edge domains: these may, in fact, be the same equipment and the PEs might be located at the domain edges.

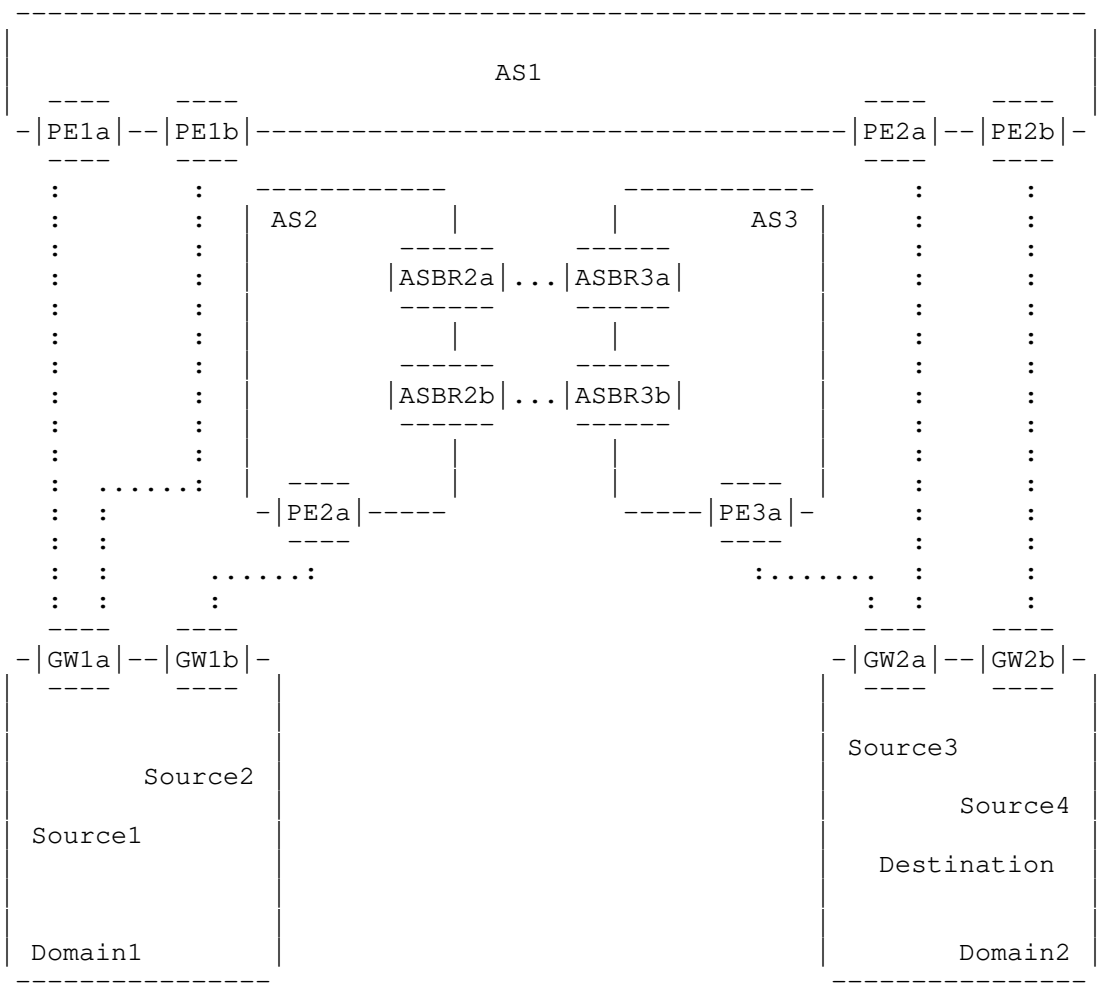


Figure 1: Reference Architecture for SR Domain Interconnect

Traffic to the destination may originate from multiple sources within that domain (we show two such sources: Source3 and Source4). Furthermore, traffic intended for the destination may arrive from

outside the domain through any of the points of attachment to the backbone networks (we show GW2a and GW2b). This traffic may need to be steered within the domain to achieve load-balancing across network resources, to avoid degraded or out-of-service resources (including planned service outages), and to achieve different qualities of service. Of course, traffic in a remote source domain may also need to be steered within that domain. We class this problem as "Intra-Domain Traffic Steering".

Traffic across the backbone networks may need to be steered to conform to common Traffic Engineering (TE) paradigms. That is, the path across any network (shown in the figure as an AS) or across any collection of networks may need to be chosen and may be different from the shortest path first (SPF) routing that would occur without TE. Furthermore, the points of inter-connection between networks may need to be selected and influence the path chosen for the data. We class this problem as "Inter-Domain Traffic Steering".

The composite end-to-end path comprises steering in the source domain, choice of source domain exit point, steering across the backbone networks, choice of network interconnections, choice of destination domain entry point, and steering in the destination domain. These issues may be inter-dependent (for example, the best traffic steering in the source domain may help select the best exit point from that domain, but the connectivity options across the backbone network may drive the selection of a different exit point). We class this combination of problems as "End-to-End Domain Interconnect Traffic Steering".

It should be noted that the solution to the End-to-End Domain Interconnect Traffic Steering problem depends on a number of factors:

- What technology is deployed in the domains.
- What technology is deployed in the backbone networks.
- How much information the domains are willing to share with each other.
- How much information the backbone network operators and the domain operators are willing to share.

In some cases, the domains and backbone networks are all owned and operated by the same company (with the backbone network often being a private network). In other cases, the domains are operated by one company, with other companies operating the backbone.

3. Solution Technologies

Segment Routing (SR from the SPRING working group in the IETF [RFC7855] and [RFC8402]) introduces traffic steering capabilities into an MPLS network [I-D.ietf-spring-segment-routing-mpls] by utilizing existing data plane capabilities (label pop and packet forwarding - "pop and go") in combination with additions to existing IGPs ([I-D.ietf-ospf-segment-routing-extensions] and [I-D.ietf-isis-segment-routing-extensions]), BGP (as BGP-LU) [RFC8277], or a centralized controller to distribute "per-hop" labels. An MPLS label stack can be imposed on a packet to describe a sequence of links/nodes to be transited by the packet; as each hop is transited, the label that represents it is popped from the stack and the packet is forwarded. Thus, on a packet-by-packet basis, traffic can be steered within the SR domain.

This document broadens the problem space to consider interconnection of any type of edge domain. These may be Data Center sites, but they may equally be access networks, VPN sites, or any other form of domain that includes packet sources and destinations. We particularly focus on "SR edge domains" being source or destination domains that utilize MPLS SR, but the domains could use other non-MPLS technologies (such as IP, VXLAN, and NVGRE) as described in Section 9.

Backbone networks are commonly based on MPLS-capable hardware. In these networks, a number of different options exist to establish TE paths. Among these options are static Label Switched Paths (LSPs), perhaps set up by an SDN controller, LSP tunnels established using a signaling protocol (such as RSVP-TE), and inter-domain use of SR (as described above for intra-domain steering). Where traffic steering (without resource reservation) is needed, SR may be adequate; where Traffic Engineering is needed (i.e., traffic steering with resource reservation) RSVP-TE or centralized SDN control are preferred. However, in a network that is fully managed and controlled through a centralized planning tool, resource reservation can be achieved and SR can be used for full Traffic Engineering. These solutions are already used in support of a number of edge-to-edge services such as L3VPN and L2VPN.

3.1. Characteristics of Solution Technologies

Each of the solution technologies mentioned in the previous section has certain characteristics, and the combined solution needs to recognize and address these characteristics in order to make a workable solution.

- When SR is used for traffic steering, the size of the MPLS label stack used in SR scales linearly with the length of the strict source route. This can cause issues with MPLS implementations that only support label stacks of a limited size. For example, some MPLS implementations cannot push enough labels on the stack to represent an entire source route. Other implementations may be unable to do the proper "ECMP hashing" if the label stack is too long; they may be unable to read enough of the packet header to find an entropy label or to find the IP header of the payload. Increasing the packet header size also reduces the size of the payload that can be carried in an MPLS packet. There are techniques that can be used to reduce the size of the label stack. For example, a source route may be made less specific through the use of loose hops requiring fewer labels, or a single label (known as a "binding SID") can be used to represent a sequence of nodes; this label can be replaced with a set of labels when the packet reaches the first node in the sequence. It is also possible to combine SR with conventional RSVP-TE by using a binding SID in the label stack to represent an LSP tunnel set up by RSVP-TE.

- Most of the work on using SR for traffic steering assumes that traffic only needs to be steered within a single administrative domain. If the backbone consists of multiple ASes that are not part of a common administrative domain, the use of SR across the backbone may prove to be a challenge, and its use in the backbone may be limited to cases where private networks connect the domains, rather than cases where the domains are connected by third-party network operators or by the public Internet.

- RSVP-TE has been used to provide edge-to-edge tunnels through which flows to/from many endpoints can be routed, and this provides a reduction in state while still offering Traffic Engineering across the backbone network. However, this requires O(n2) connections and as the number of edge domains increases this becomes unsustainable.

- A centralized control system is capable of producing more efficient use of network resources and of allowing better coordination of network usage and of network diagnostics. However, such a system may present challenges in large and dynamic networks because it relies on all network state being held centrally, and it is difficult to make central control as robust and self-correcting as distributed control.

This document introduces an approach that blends the best points of each of these solution technologies to achieve a trade-off where RSVP-TE tunnels in the backbone network are stitched together using SR, and end-to-end SR paths can be created under the control of a

central controller with routing devolved to the constituent networks where possible.

4. Decomposing the Problem

It is important to decompose the problem to take account of different regions spanned by the end-to-end path. These regions may use different technologies and may be under different administrative control. The separation of administrative control is particularly important because the operator of one region may be unwilling to share information about their networks, and may be resistant to allowing a third party to exert control over their network resources.

Using the reference model in Figure 1, we can consider how to get a packet from Source1 to the Destination. The following decisions must be made:

- In which domain Destination lies.
- Which exit point from Domain1 to use.
- Which entry point to Domain2 to use.
- How to reach the exit point of Domain1 from Source1.
- How to reach the entry point to Domain2 from the exit point of Domain1.
- How to reach Destination from the entry point to Domain2.

As already mentioned, these decisions may be inter-related. This enables us to break down the problem into three steps:

1. Get the packet from Source1 to the exit point of Domain1.
2. Get the packet from exit point of Domain1 to entry point of Domain2.
3. Get the packet from entry point of Domain2 to Destination.

The solution needs to achieve this in a way that allows:

- Adequate discovery of preferred elements in the end-to-end path (such as the location of the destination, and the selection of the destination domain entry point).
- Full control of the end-to-end path if all of the operators are willing.

- Re-use of existing techniques and technologies.

From a technology point of view we must support several functions and mixtures of those functions:

- If a domain uses MPLS Segment Routing, the labels within the domain may be populated by any means including BGP-LU [RFC8277], IGP [I-D.ietf-isis-segment-routing-extensions] [I-D.ietf-ospf-segment-routing-extensions], and central control. Source routes within the domain may be expressed as label stacks pushed by a controller or computed by a source router, or expressed as a single label and programmed into the domain routers by a controller.

- If a domain uses other (non-MPLS) forwarding, the domain processing is specific to that technology. See Section 9 for details.

- If the domains use Segment Routing, the source and destination domains may or may not be in the same 'Segment Routing domain' [RFC8402], so that the prefix-SIDs may be the same or different in the two domains.

- The backbone network may be a single private network under the control of the owner of the domains and comprising one or more ASes, or may be a network operated by one or more third parties.

- The backbone network may utilize MPLS Traffic Engineering tunnels in conjunction with MPLS Segment Routing and the domain-to-domain source route may be provided by stitching TE LSPs.

- A single controller may be used to handle the source and destination domains as well as the backbone network, or there may be a different controller for the backbone network separate from that that controls the two domains, or there may be separate controllers for each network. The controllers may cooperate and share information to different degrees.

All of these different decompositions of the problem reflect different deployment choices and different commercial and operational practices, each with different functional trade-offs. For example, with separate controllers that do not share information and that only cooperate to a limited extent, it will be possible to achieve end-to-end connectivity with optimal routing at each step (domain or backbone AS), but the end-to-end path that is achieved might not be optimal.

5. Solution Space

5.1. Global Optimization of the Paths

Global optimization of the path from one domain to another requires either that the source controller has a complete view of the end-to-end topology or some form of cooperation between controllers (such as in Backward Recursive Path Computation (BRPC) in [RFC5441]).

BGP-LS [RFC7752] can be used to provide the "source" controller with a view of the topology of the backbone: that topology may be abstracted or partial. This requires some of the BGP speakers in each AS to have BGP-LS sessions to the controller. Other means of obtaining this view of the topology are of course possible.

5.2. Figuring Out the GWs at a Destination Domain for a Given Prefix

Suppose GW2a and GW2b both advertise a route to prefix X, each setting itself as next hop. One might think that the GWs for X could be inferred from the routes' next hop fields, but typically only the "best" route (as selected by BGP) gets distributed across the backbone: the other route is discarded. But the best route according to the BGP selection process might not be the route via the GW that we want to use for traffic engineering purposes.

The obvious solution would be to use the ADD-PATH mechanism [RFC7911] to ensure that all routes to X get advertised. However, even if one does this, the identity of the GWs would get lost as soon as the routes got distributed through an ASBR that sets next hop self. And if there are multiple ASes in the backbone, not only will the next hop change several times, but the ADD-PATH mechanism will experience scaling issues. So this "obvious" solution only works within a single AS.

A better solution can be achieved using the Tunnel Encapsulation [I-D.ietf-idr-tunnel-encaps] attribute as follows.

We define a new tunnel type, "SR tunnel", and when the GWs to a given domain advertise a route to a prefix X within the domain, they each include a Tunnel Encapsulation attribute with multiple remote endpoint sub-TLVs each of which identifies a specific GW to the domain.

In other words, each route advertised by any GW identifies all of the GWs to the same domain (see Section 9 for a discussion of how GWs discover each other). Therefore, only one of the routes needs to be distributed to other ASes, and it doesn't matter how many times the next hop changes, the Tunnel Encapsulation attribute (and its remote

endpoint sub-TLVs) remains unchanged and disclose the full list of GWs to the domain.

Further, when a packet destined for prefix X is sent on a TE path to GW2a we want the packet to arrive at GW2a carrying, at the top of its label stack, GW2a's label for prefix X. To achieve this we place the SID/SRGB in a sub-TLV of the Tunnel Encapsulation attribute. We define the prefix-SID sub-TLV to be essentially identical in syntax to the prefix-SID attribute (see [I-D.ietf-idr-bgp-prefix-sid]), but the semantics are somewhat different.

We also define an "MPLS Label Stack" sub-TLV for the Tunnel Encapsulation attribute, and put this in the "SR tunnel" TLV. This allows the destination GW to specify a label stack that it wants packets destined for prefix X to have. This label stack represents a source route through the destination domain.

5.3. Figuring Out the Backbone Egress ASBRs

We need to figure out the backbone egress ASBRs that are attached to a given GW at the destination domain in order to properly engineer the path across the backbone.

The "cleanest" way to do this is to have the backbone egress ASBRs distribute the information to the source controller using the egress peer engineering (EPE) extensions of BGP-LS [I-D.ietf-idr-bgpls-segment-routing-epe]. The EPE extensions to BGP-LS allow a BGP speaker to say, "Here is a list of my EBGP neighbors, and here is a (locally significant) adjacency-SID for each one."

It may also be possible to consider utilizing cooperating PCEs or a Hierarchical PCE approach in [RFC6805]. But it should be observed that this question is dependent on the questions in Section 5.2. That is, it is not possible to even start the selection of egress ASBRs until it is known which GWs at the destination domain provide access to a given prefix. Once that question has been answered, any number of PCE approaches can be used to select the right egress ASBR and, more generally, the ASBR path across the backbone.

5.4. Making use of RSVP-TE LSPs Across the Backbone

There are a number of ways to carry traffic across the backbone from one domain to another. RSVP-TE is a popular mechanism for establishing tunnels across MPLS networks in similar scenarios (e.g., L3VPN) because it allows for reservation of resources as well as traffic steering.

A controller can cause an RSVP-TE LSP to be set up by talking to the LSP head end using PCEP extensions as described in [RFC8281]. That document specifies an "LSP Initiate" message (the PCInitiate message) that the controller uses to specify the RSVP-TE LSP endpoints, the explicit path, a "symbolic pathname", and other optional attributes (specified in the PCEP specification [RFC5440]) such as bandwidth.

When the head end receives a PCInitiate message, it sets up the RSVP-TE LSP, assigns it a "PLSP-id", and reports the PLSP-id back to the controller in a PCRpt message [RFC8231]. The PCRpt message also contains the symbolic name that the controller assigned to the LSP, as well as containing some information identifying the LSP-initiate message from the controller, and details of exactly how the LSP was set up (RRO, bandwidth, etc.).

The head end can add a TE-PATH-BINDING TLV to the PCRpt message [I-D.sivabalan-pce-binding-label-sid]. This allows the head end to assign a "binding SID" to the LSP, and to report to the controller that a particular binding SID corresponds to a particular LSP. The binding SID is locally scoped to the head end.

The controller can make this label be part of the label stack that it tells the source (or the GW at the source domain) to impose on the data packets being sent to prefix X. When the head end receives a packet with this label at the top of the stack it will send the packet onward on the LSP.

5.5. Data Plane

Consolidating all of the above, consider what happens when we want to move a data packet from Source1 to Destination in Figure 1via the following source route:

Source1---GW1b---PE2a---ASBR2a---ASBR3a---PE3a---GW2a---Destination

Further, assume that there is an RSVP-TE LSP from PE2a to ASBR2a and an RSVP-TE LSP from ASBR3a to PE3a both of which we want to use.

Let’s suppose that the Source pushes a label stack as instructed by the controller (for example, using BGP-LU [RFC8277]). We won’t worry for now about source routing through the domains themselves: that is, in practice there may be additional labels in the stack to cover the source route from Source1 to GW1b and from GW2a to the Destination, but we will focus only on the labels necessary to leave the source domain, traverse the backbone, and enter the egress domain. So we only care what the stack looks like when the packet gets to GW1b.

When the packet gets to GW1b, the stack should have six labels:

Top Label:

Peer-SID or adjacency-SID identifying the link or links to PE2a. These SIDs are distributed from GW1b to the controller via the EPE extensions of BGP-LS. This label will get popped by GW1b, which will then send the packet to PE2a.

Second Label:

Binding SID advertised by PE2a to the controller for the RSVP-TE LSP to ASBR2a. This binding SID is advertised via the PCEP extensions discussed above. This label will get swapped by PE2a for the label that the LSP's next hop has assigned to the LSP.

Third Label:

Peer-SID or adjacency-SID identifying the link or links to ASBR3a, as advertised to the controller by ASBR2a using the BGP-LS EPE extensions. This label gets popped by ASBR2a, which then sends the packet to ASBR3a.

Fourth Label:

Binding SID advertised by ASBR3a for the RSVP-TE LSP to PE3a. This binding SID is advertised via the PCEP extensions discussed above. ASBR3a treats this label just like PE2a treated the second label above.

Fifth label:

Peer-SID or adjacency-SID identifying link or links to GW2a, as advertised to the controller by ASBR3a using the BGP-LS EPE extensions. ASBR3a pops this label and sends the packet to GW2a.

Sixth Label:

Prefix-SID or other label identifying the Destination advertised in a Tunnel Encapsulation attribute by GW2a. This can be omitted if GW2a is happy to accept IP packets, or prefers a VXLAN tunnel for example. That would be indicated through the Tunnel Encapsulation attribute of course.

Note that the size of the label stack is proportional to the number of RSVP-TE LSPs that get stitched together by SR.

See Section 7 for some detailed examples that show the concrete use of labels in a sample topology.

In the above example, all labels except the sixth are locally significant labels: peer-SIDs, binding SIDs, or adjacency-SIDs. Only the sixth label, a prefix-SID, has a domain-wide unique value. To impose that label, the source needs to know the SRGB of GW2a. If all nodes have the same SRGB, this is not a problem. Otherwise, there are a number of different ways GW3a can advertise its SRGB. This can be done via the segment routing extensions of BGP-LS, or it can be done using the prefix-SID attribute or BGP-LU [RFC8277], or it can be done using the BGP Tunnel Encapsulation attribute. The technique to be used will depend on the details of the deployment scenario.

The reason the above example is primarily based on locally significant labels is that it creates a "strict source route", and it presupposes the EPE extensions of BGP-LS. In some scenarios, the EPE extension to BGP-LS might not be available (or BGP-LS might not be available at all). In other scenarios, it may be desirable to steer a packet through a "loose source route". In such scenarios, the label stack imposed by the source will be based upon a sequence of domain-wide unique "node-SIDs", each representing one of the hops of source route. Each label has to be computed by adding the corresponding node-SID to the SRGB of the node that will act upon the label. One way to learn the node-SIDs and SRGBs is to use the segment routing extensions of BGP-LS. Another way is to use BGP-LU as follows:

> Each node that may be part of a source route originates a BGP-LU route with one of its own loopback addresses as the prefix. The BGP prefix-SID attribute is attached to this route. The prefix-SID attribute contains a SID that is the domain-wide unique SID corresponding to the node's loopback address. The attribute also contains the node's SRGB.

While this technique is useful when BGP-LS is not available, there needs to be some other means for the source controller to discover the topology. In this document, we focus primarily on the scenario where BGP-LS, rather than BGP-LU, is used.

5.6. Centralized and Distributed Controllers

A controller or set of controllers is needed to collate topology and TE information from the constituent networks, to apply policies and service requirements to compute paths across those networks, to select an end-to-end path, and to program key nodes in the network to take the right forwarding actions (pushing label stacks, stitching LSPs, forwarding traffic).

- It is commonly understood that a fully optimal end-to-end path can only be computed with full knowledge of the end-to-end topology

and available Traffic Engineering resources. Thus, one option is for all information about the domain networks and backbone network to be collected by a central controller that makes all path computations and is responsible for issuing the necessary programming commands. Such a model works best when there is no commercial or administrative impediment (for example, where the domains and the backbone network are owned and operated by the same organization). There may, however, be some scaling concerns if the component networks are large.

In this mode of operation, each network may use BGP-LS to export Traffic Engineering and topology information to the central controller, and the controller may use PCEP to program the network behavior.

- A similar centralized control mechanism can be used with a scalability improvement that risks a reduction in optimality. In this case, the domain networks can export to the controller just the feasibility of connectivity between data source/sink and gateway, perhaps enhancing this with some information about the Traffic Engineering metrics of the potential paths.

  This approach allows the central controller to understand the end-to-end path that it is selecting, but not to control it fully. The source route from data source to domain egress gateway is left to the source host or a controller in the source domain, while the source route from domain ingress gateway to destination is left as a decision for the domain ingress gateway or to a controller in the destination domain and in both cases the traffic may be left to follow the IGP shortest path.

  This mode of operation still leaves overall control with a centralized server and that may not be considered suitable when there is separate commercial or administrative control of the networks.

- When there is separate commercial or administrative control of the networks, the domain operator will not want the backbone operator to have control of the paths within the domains and may be reluctant to disclose any information about the topology or resource availability within the domains. Conversely, the backbone operator may be very unwilling to allow the domain operator (a customer) any control over or knowledge about the backbone network.

  This "problem" has already been solved for Traffic Engineering in MPLS networks that span multiple administrative domains and leads to several potential solutions:

* Per-domain path computation [RFC5152] can be seen as "best effort optimization". In this mode the controller for each domain is responsible for finding the best path to the next domain, but has no way of knowing which is the best exit point from the local domain. The resulting path may end up significantly sub-optimal or even blocked.

* Backward recursive path computation (BRPC) [RFC5441] is a mechanism that allows controllers to cooperate across a small set of domains (such as ASes) to build a tree of possible paths and so allow the controller for the ingress domain to select the optimal path. The details of the paths within each domain that might reveal confidential information can be hidden using Path Keys [RFC5520]. BRPC produces optimal paths, but scales poorly with an increase in domains and with an increase in connectivity between domains. It can also lead to slow computation times.

* Hierarchical PCE (H-PCE) [RFC6805] is a two-level cooperation process between PCEs. The child PCEs remain responsible for computing paths across their domains, and they coordinate with a parent PCE that stitches these paths together to form the end-to-end path. This approach has many similarities with BRPC but can scale better through the maintenance of "domain topology" that shows how the domains are interconnected, and through the ability to pipe-line computation requests to all of the child domains. It has the drawback that some party has to own and operate the parent PCE.

* An alternative approach is documented by the TEAS working group [RFC7926]. In this model each network advertises to controllers for adjacent networks (using BGP-LS) selected information about potential connectivity across the network. It does not have to show full topology and can make its own decisions about which paths it considers optimal for use by its different neighbors and customers. This approach is suitable for the End-to-End Domain Interconnect Traffic Steering problem where the backbone is under different control from the domains because it allows the overlay nature of the use of the backbone network to be treated as a peer network relationship by the controllers of the domains - the domains can be operated using a single controller or a separate controller for each domain.

It is also possible to operate domain interconnection when some or all domains do not have a controller. Segment Routing is capable of routing a packet toward the next hop based on the top label on the stack, and that label does not need to indicate an immediately adjacent node or link. In these cases, the packet may be forwarded

untouched, or the forwarding router may impose a locally-determined additional set of labels that define the path to the next hop.

PCE can be used to instruct the source host or a transit node about what label stacks to add to packets. That is, a node that needs to impose labels (either to start routing the packet from the source host, or to advance the packet from a transit router toward the destination) can determine the label stack to use based on local function or can have that stack supplied by a PCE. The PCE Communication Protocol (PCEP) has been extended to allow the PCE to supply a label stack for reaching a specific destination either in response to a request or in an unsolicited manner [I-D.ietf-pce-segment-routing].

6. BGP-LS Considerations

This section gives an overview of the use of BGP-LS to export an abstraction (or summary) of the connectivity across the backbone network by means of two figures that show different views of a sample network.

Figure 2 shows a more complex reference architecture.

Figure 3 represents the minimum set of nodes and links that need to be advertised in BGP-LS with SR in order to perform Domain Interconnect with traffic engineering across the backbone network: the PEs, ASBRs, and GWs, and the links between them. In particular, EPE [I-D.ietf-idr-bgpls-segment-routing-epe] and TE information with associated segment IDs is advertised in BGP-LS with SR.

Links that are advertised may be physical links, links realized by LSP tunnels or SR paths, or abstract links. It is assumed that intra-AS links are either real links, RSVP-TE LSPs with allocated bandwidth, or SR TE policies as described in [I-D.ietf-idr-segment-routing-te-policy]. Additional nodes internal to an AS and their links to PEs, ASBRs, and/or GWs may also be advertised (for example, to avoid full mesh problems).

Note that Figure 3 does not show full interconnectivity. For example, there is no possibility of connectivity between PE1a and PE1c (because there is no RSVP-TE LSP established across AS1 between these two nodes) and so no link is presented in the topology view. [RFC7926] contains further discussion of topological abstractions that may be useful in understanding this distinction.

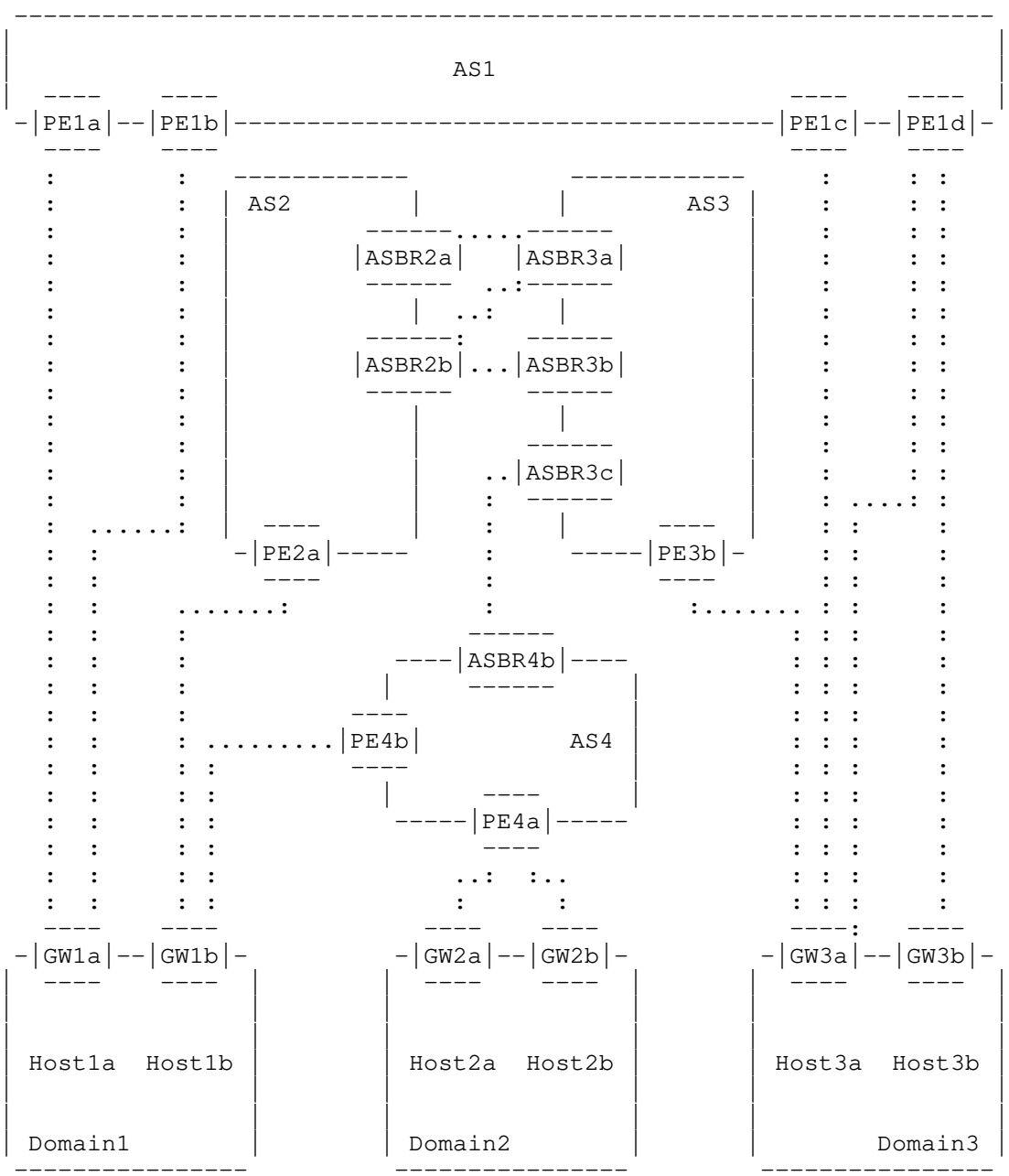


Figure 2: Network View of Example Configuration

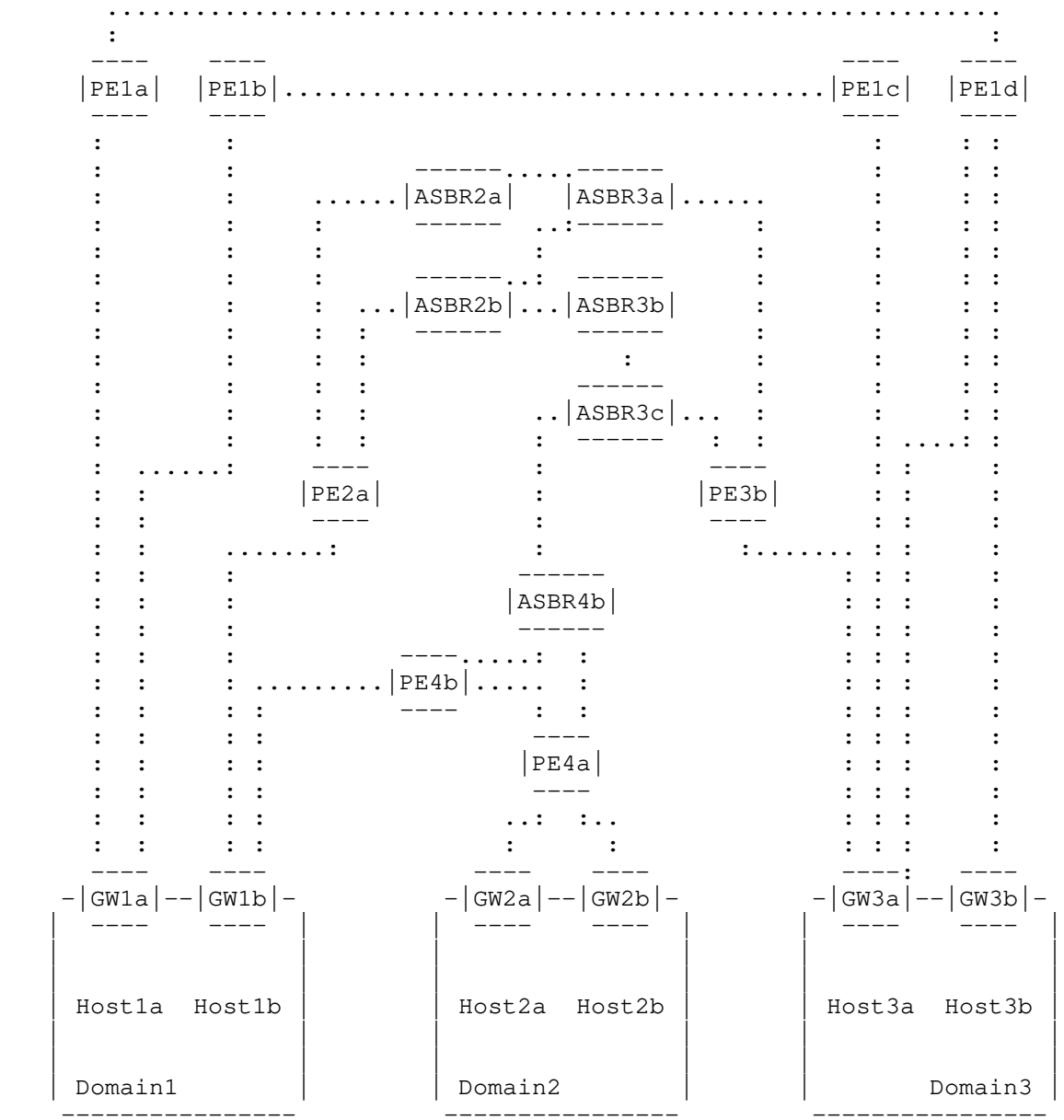


Figure 3: Topology View of Example Configuration

A node (a PCE, router, or host) that is computing a full or partial path correlates the topology information disseminated in BGP-LS with

the information advertised in BGP (with the Tunnel Encapsulation attributes) and uses this to compute that path and obtain the SIDs for the elements on that path. In order to allow a source host to compute exit points from its domain, some subset of the above information needs to be disseminated within that domain.

What is advertised external to a given AS is controlled by policy at the ASes' PEs, ASBRs, and GWs. Central control of what each node should advertise, based upon analysis of the network as a whole, is an important additional function. This and the amount of policy involved may make the use of a Route Reflector an attractive option.

Local configuration at each node determines which links to other nodes are advertised in BGP-LS, and determines which characteristics of those links are advertised. Pairwise coordination between link end-points is required to ensure consistency.

Path Weighted ECMP (PWECMP) is a mechanism to load-balance traffic across parallel equal cost links or paths. In this approach an ingress node distributes the flows from it to a given egress node across the equal cost paths to the egress node in proportion to the lowest bandwidth link on each path. PWECMP can be used by a GW for a given source domain to send all flows to a given destination domain using all paths in the backbone network to that destination domain in proportion to the minimum bandwidth on each path. PWECMP may also be used by hosts within a source domain to send flows to that domain's GWs.

7. Worked Examples

Figure 4 shows a view of the links, paths, and labels that can be assigned to part of the sample network shown in Figure 2 and Figure 3. The double-dash lines (===) indicate LSP tunnels across backbone ASes and dotted lines (...) are physical links.

A label may be assigned to each outgoing link at each node. This is shown in Figure 4. For example, at GW1a the label L201 is assigned to the link connecting GW1a to PE1a. At PE1c, the label L302 is assigned to the link connecting PE1c to GW3b. Labels ("binding SIDs") may also be assigned to RSVP-TE LSPs. For example, at PE1a, label L202 is assigned to the RSVP-TE LSP leading from PE1a to PE1c.

At the destination domain, label L305 is a "node-SID"; it represents Host3b, rather than representing a particular link.

When a node processes a packet, the label at the top of the label stack indicates the link (or RSVP-TE LSP) on which that node is to transmit the packet. The node pops that label off the label stack

before transmitting the packet on the link. However, if the top label is a node-SID, the node processing the packet is expected to transmit the packet on whatever link it regards as the shortest path to the node represented by the label.

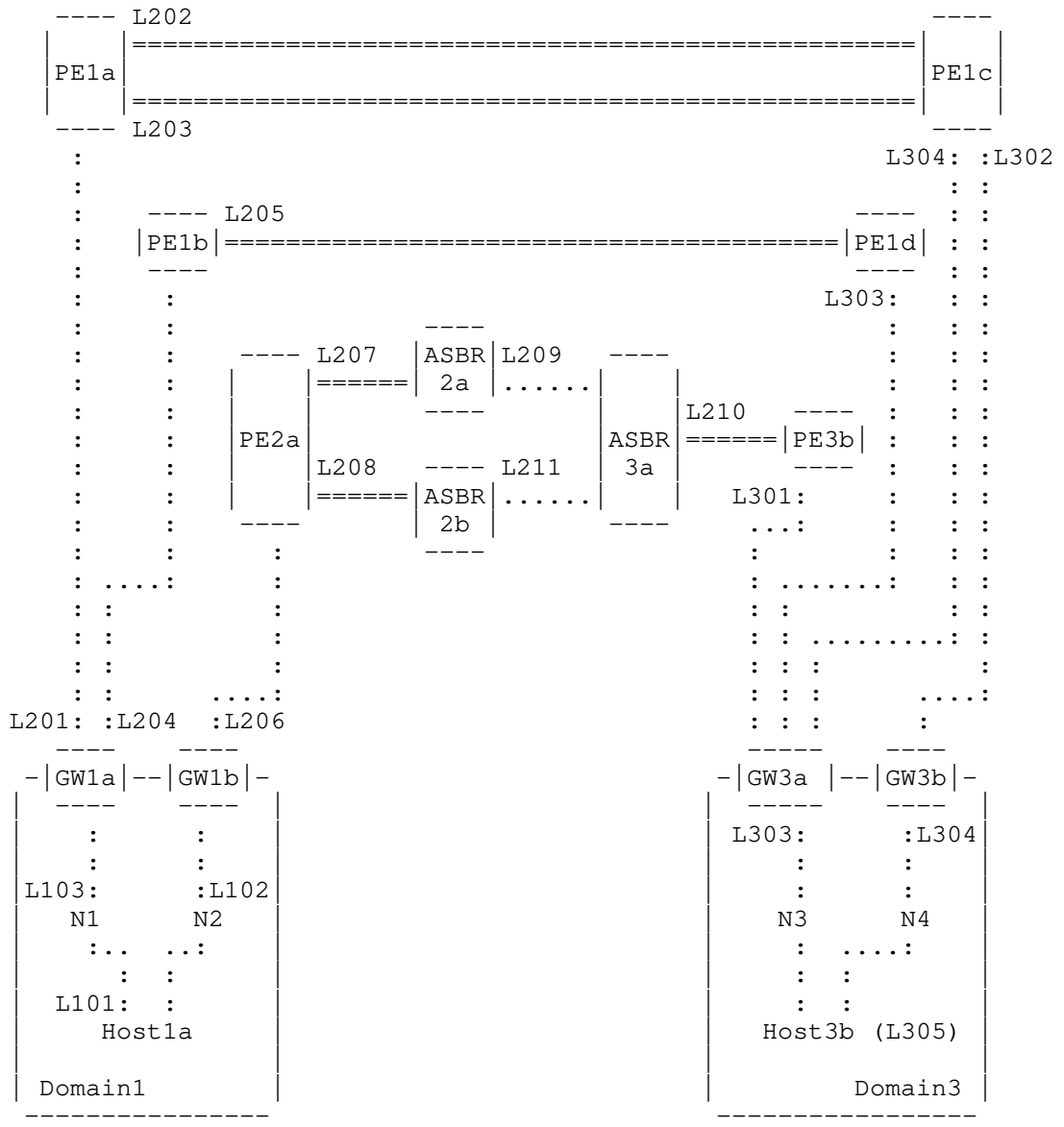


Figure 4: Tunnels and Labels in Example Configuration

Note that label spaces can overlap so that, for example, the figure shows two instances of L303 and L304. This is acceptable because of the separation between the SR domains, and because SIDs applied to outgoing interfaces are locally scoped.

Let's consider several different possible ways to direct a packet from Host1a in Domain1 to Host3b in Domain3.

a. Full source route imposed at source

In this case it is assumed that the entity responsible for determining an end-to-end path has access to the topologies of both the source and destination domains as well as of the backbone network. This might happen if all of the networks are owned by the same operator in which case the information can be shared into a single database for use by an offline tool, or the information can be distributed using routing protocols such that the source host can see enough to select the path. Alternatively, the end-to-end path could be produced through cooperation between computation entities each responsible for different domains along the path.

If the path is computed externally it is pushed to the source host. Otherwise, it is computed by the source host itself.

Suppose it is desired for a packet from Host1a to travel to Host3b via the following source route:

Host1a->N1->GW1a->PE1a->(RSVP-TE LSP)->PE1c->GW3b->N4->Host3b

Host1a imposes the following label stack (with the first label representing the top of stack), and then sends the packet to N1:

L103, L201, L202, L302, L304, L305

N1 sees L103 at the top of the stack, so it pops the stack and forwards the packet to GW1a. GW1a sees L201 at the top of the stack, so it pops the stack and forwards the packet to PE1a. PE1a sees L202 at the top of the stack, so it pops the stack and forwards the packet over the RSVP-TE LSP to PE1c. As the packet travels over this LSP, its top label is an RSVP-TE signaled label representing the LSP. That is, PE1a imposes an additional label stack entry for the tunnel LSP.

At the end of the LSP tunnel, the MPLS tunnel label is popped, and PE1c sees L302 at the top of the stack. PE1c pops the stack and forwards the packet to GW3b. GW3b sees L304 at the top of the stack, so it pops the stack and forwards the packet to N4. Finally, N4 sees L305 at the top of the stack, so it pops the stack and forwards the packet to Host3b.

b. It is possible that the source domain does not have visibility into the destination domain.

This occurs if the destination domain does not export its topology, but does export basic reachability information so that the source host or the path computation entity will know:

+ The GWs through which the destination can be reached.

+ The SID to use for the destination prefix.

Suppose we want a packet to follow the source route:

```
Host1a->N1->GW1a->PE1a->(RSVP-TE
LSP)->PE1c->GW3b->...->Host3b
```

The ellipsis indicates a part of the path that is not explicitly specified. Thus, the label stack imposed at the source host is:

```
L103, L201, L202, L302, L305
```

Processing is as per case a., but when the packet reaches the GW of the destination domain (GW3b) it can either simply forward the packet along the shortest path to Host3b, or it can insert additional labels to direct the path to the destination.

c. Domain1 only has reachability information for the backbone and destination networks

The source domain (or the path computation entity) may be further restricted in its view of the network. It is possible that it knows the location of the destination in the destination domain, and knows the GWs to the destination domain that provide reachability to the destination, but that it has no view of the backbone network. This leads to the packet being forwarded in a manner similar to 'per-domain path computation' described in Section 5.6.

At the source host a simple label stack is imposed navigating the domain and indicating the destination GW and the destination host.

```
L103, L302, L305
```

As the packet leaves the source domain, the source GW (GW1a) determines the PE to use to enter the backbone using nothing

more than the BGP preferred route to the destination GW (it could be PE1a or PE1b).

When the packet reaches the first PE it has a label stack just identifying the destination GW and the host (L302, L305). The PE uses information it has about the backbone network topology and available LSPs to select an LSP tunnel, impose the tunnel label, and forward the packet.

When the packet reaches the end of the LSP tunnel, it is processed as described in case b.

d. Stitched LSPs across the backbone

A variant of all these cases arises when the packet is sent using a path that spans multiple ASes. For example, one that crosses AS2 and AS3 as shown in Figure 2.

In this case, basing the example on case a., the source host imposes the label stack:

L102, L206, L207, L209, L210, L301, L303, L305

It then sends the packet to N2.

When the packet reaches PE2a, as previously described, the top label (L207) indicates an LSP tunnel that leads to ASBR2a. At the end of that LSP tunnel the next label (L209) routes the packet from ASBR2a to ASBR3a, where the next label (L210) identifies the next LSP tunnel to use. Thus, SR has been used to stitch together LSPs to make a longer path segment. As the packet emerges from the final LSP tunnel, forwarding continues as previously described.

8. Label Stack Depth Considerations

As described in Section 3.1, one of the issues with a Segment Routing approach is that the label stack can get large, for example when the source route becomes long. A mechanism to mitigate this problem is needed if the solution is to be fully applicable in all environments.

[I-D.ietf-idr-segment-routing-te-policy] introduces the concept of hierarchical source routes as a way to compress source route headers. It functions by having the egress node for a set of source routes advertise those source routes along with an explicit request that each node that is an ingress node for one or more of those source routes should advertise a binding SID for the set of source routes for which it is the ingress. It should be noted that the set of

source routes can either be advertised by the egress node as described here, or advertised by a controller on behalf of the egress node.

Such an ingress node advertises its set of source routes and a binding SID as an adjacency in BGP-LS as described in Section 6. These source routes represent the weighted ECMP paths between the ingress node and the egress node. Note also that the binding SID may be supplied by the node that advertises the source routes (the egress or the controller), or may be chosen by the ingress.

A remote node that wishes to reach the egress node constructs a source route consisting of the segment IDs necessary to reach one of the ingress nodes for the path it wishes to use along with the binding SID that the ingress node advertised to identify the set of paths. When the selected ingress node receives a packet with a binding SID it has advertised, it replaces the binding SID with the labels for one of its source routes to the egress node (it will choose one of the source routes in the set according to its own weighting algorithms and policy).

8.1. Worked Example

Consider the topology in Figure 4. Suppose that it is desired to construct full segment routed paths from ingress to egress, but that the resulting label stack (segment route) is too large. In this case the gateways to Domain3 (GW3a and GW3b) can advertise all of the source routes from the gateways to Domain1 (GW1a and GW1b). The gateways to Domain1 then assign binding SIDs to those source routes and advertise those SIDs into BGP-LS.

Thus, GW3b advertises the two source routes (L201, L202, L302 and L201, L203, L302), and GW1a advertises into BGP-LS its adjacency to GW3b along with a binding SID. Should Host1a wish to send a packet via GW1a and GW3b, it can include L103 and this binding SID in the source route. GW1a is free to choose which source route to use between itself and GW3b using its weighted ECMP algorithm.

Similarly, GW3a can advertise the following set of source routes:

- L201, L202, L304
- L201, L203, L304
- L204, L205, L303
- L206, L207, L209, L210, L301

- L206, L208, L211, L210, L301

GW1a advertises a binding SID for the first three, and GW1b advertises a binding SID for the other two.

9. Gateway Considerations

As described in Section 5.2, [I-D.ietf-bess-datacenter-gateway] defines a new tunnel type, "SR tunnel", and when the GWs to a given domain advertise a route to a prefix X within the domain, they will each include a Tunnel Encapsulation attribute with multiple tunnel instances each of type "SR tunnel", one for each GW and each containing a Remote Endpoint sub-TLV with that GW's address.

In other words, each route advertised by any GW identifies all of the GWs to the same domain.

Therefore, even if only one of the routes is distributed to other ASes, it will not matter how many times the next hop changes, as the Tunnel Encapsulation attribute (and its remote endpoint sub-TLVs) will remain unchanged.

9.1. Domain Gateway Auto-Discovery

To allow a given domain's GWs to auto-discover each other and to coordinate their operations, the following procedures are implemented as described in [I-D.ietf-bess-datacenter-gateway]:

- Each GW is configured with an identifier of the domain that is common across all GWs to the domain and unique across all domains that are connected.

- A route target [RFC4360] is attached to each GW's auto-discovery route and has its value set to the domain identifier.

- Each GW constructs an import filtering rule to import any route that carries a route target with the same domain identifier that the GW itself uses. This means that only these GWs will import those routes and that all GWs to the same domain will import each other's routes and will learn (auto-discover) the current set of active GWs for the domain.

- The auto-discovery route each GW advertises consists of the following:

  * An IPv4 or IPv6 NLRI containing one of the GW's loopback addresses (that is, with AFI/SAFI that is one of 1/1, 2/1, 1/4, 2/4).

* A Tunnel Encapsulation attribute containing the GW's encapsulation information, which at a minimum consists of an SR tunnel TLV with a Remote Endpoint sub-TLV [I-D.ietf-idr-tunnel-encaps].

To avoid the side effect of applying the Tunnel Encapsulation attribute to any packet that is addressed to the GW, the GW should use a different loopback address in the advertisement from that used to reach the GW itself.

Each GW will include a Tunnel Encapsulation attribute for each GW that is active for the domain (including itself), and will include these in every route advertised by each GW to peers outside the domain. As the current set of active GWs changes (due to the addition of a new GW or the failure/removal of an existing GW) each externally advertised route will be re-advertised with the set of SR tunnel instances reflecting the current set of active GWs.

9.2. Relationship to BGP Link State and Egress Peer Engineering

When a remote GW receives a route to a prefix X it can use the SR tunnel instances within the contained Tunnel Encapsulation attribute to identify the GWs through which X can be reached. It uses this information to compute SR TE paths across the backbone network looking at the information advertised to it in SR BGP Link State (BGP-LS) [I-D.ietf-idr-bgp-ls-segment-routing-ext] and correlated using the domain identity. SR Egress Peer Engineering (EPE) [I-D.ietf-idr-bgpls-segment-routing-epe] can be used to supplement the information advertised in BGP-LS.

9.3. Advertising a Domain Route Externally

When a packet destined for prefix X is sent on an SR TE path to a GW for the domain containing X, it needs to carry the receiving GW's label for X such that this label rises to the top of the stack before the GW completes its processing of the packet. To achieve this we place a prefix-SID sub-TLV for X in each SR tunnel instance in the Tunnel Encapsulation attribute in the externally advertised route for X.

Alternatively, if the GWs for a given domain are configured to allow remote GWs to perform SR TE through that domain for prefix X, then each GW computes an SR TE path through that domain to X from each of the current active GWs and places each in an MPLS label stack sub-TLV [I-D.ietf-idr-tunnel-encaps] in the SR tunnel instance for that GW.

9.4. Encapsulations

If the GWs for a given domain are configured to allow remote GWs to send them packets in that domain's native encapsulation, then each GW will also include multiple instances of a tunnel TLV for that native encapsulation in the externally advertised routes: one for each GW, and each containing a remote endpoint sub-TLV with that GW's address. A remote GW may then encapsulate a packet according to the rules defined via the sub-TLVs included in each of the tunnel TLV instances.

10. Security Considerations

There are several security domains and associated threats in this architecture. SR is itself a data transmission encapsulation that provides no additional security, so security in this architecture relies on higher layer mechanisms (for example, end-to-end encryption of pay-load data), security of protocols used to establish connectivity and distribute network information, and access control so that control plane and data plane packets are not admitted to the network from outside.

This architecture utilizes a number of control plane protocols within domains, within the backbone, and north-south between controllers and domains. Only minor modifications are made to BGP as described in [I-D.ietf-bess-datacenter-gateway], otherwise this architecture uses existing protocols and extensions so no new security risks are introduced.

Special care should, however, be taken when routing protocols export or import information from or to domains that might have a security model based on secure boundaries and internal mutual trust. This is notable when:

- BGP-LS is used to export topology information from within a domain to a controller that is sited outside the domain.
- A southbound protocol such as BGP-LU or Netconf is used to install state in the network from a controller that may be sited outside the domain.

In these cases protocol security mechanisms should be used to protect the information in transit entering or leaving the domain, and to authenticate the out-of-domain nodes (the controller) to ensure that confidential/private information is not lost and that data or configuration is not falsified.

11. Management Considerations

Configuration elements for the approaches described in this document are minor but crucial.

Each GW to a domain is configured with the same identifier of the domain, and that identifier is unique across all domains that are connected. This requires some coordination both within a domain, and between cooperating domains. There are no requirements for how this configuration and coordination is achieved, but it is assumed that management systems are involved.

Policy determines what topology information is shared by a BGP-LS speaker (see Section 6). This applies both to the advertisement of interdomain links and their characteristics, and to the advertisement of summarized domain topology or connectivity. This policy is a local (i.e., domain-scoped) configuration dependent on the objectives and business imperatives of the domain operator.

Domain boundaries are usually configured to limit the control and interaction from other domains (for example, to not allow end-to-end TE paths to be set up across domain boundaries. As noted in Section 9.3, the GWs for a given domain can be configured to allow remote GWs to perform SR TE through that domain for a given prefix, a set of prefixes, or all reachable prefixes.

Similarly, (as described in Section 9.4 the GWs for a given domain can be configured to allow remote GWs to send them packets in that domain's native encapsulation.

12. IANA Considerations

This document makes no requests for IANA action.

13. Acknowledgements

Thanks to Jeffery Zhang for his careful review.

14. Informative References

[I-D.ietf-bess-datacenter-gateway]
Drake, J., Farrel, A., Rosen, E., Patel, K., and L. Jalil, "Gateway Auto-Discovery and Route Advertisement for Segment Routing Enabled Domain Interconnection", draft-ietf-bess-datacenter-gateway-01 (work in progress), May 2018.

[I-D.ietf-idr-bgp-ls-segment-routing-ext]
Previdi, S., Talaulikar, K., Filsfils, C., Gredler, H., and M. Chen, "BGP Link-State extensions for Segment Routing", draft-ietf-idr-bgp-ls-segment-routing-ext-09 (work in progress), October 2018.

[I-D.ietf-idr-bgp-prefix-sid]
Previdi, S., Filsfils, C., Lindem, A., Sreekantiah, A., and H. Gredler, "Segment Routing Prefix SID extensions for BGP", draft-ietf-idr-bgp-prefix-sid-27 (work in progress), June 2018.

[I-D.ietf-idr-bgpls-segment-routing-epe]
Previdi, S., Filsfils, C., Patel, K., Ray, S., and J. Dong, "BGP-LS extensions for Segment Routing BGP Egress Peer Engineering", draft-ietf-idr-bgpls-segment-routing-epe-15 (work in progress), March 2018.

[I-D.ietf-idr-segment-routing-te-policy]
Previdi, S., Filsfils, C., Jain, D., Mattes, P., Rosen, E., and S. Lin, "Advertising Segment Routing Policies in BGP", draft-ietf-idr-segment-routing-te-policy-04 (work in progress), July 2018.

[I-D.ietf-idr-tunnel-encaps]
Rosen, E., Patel, K., and G. Velde, "The BGP Tunnel Encapsulation Attribute", draft-ietf-idr-tunnel-encaps-10 (work in progress), August 2018.

[I-D.ietf-isis-segment-routing-extensions]
Previdi, S., Ginsberg, L., Filsfils, C., Bashandy, A., Gredler, H., Litkowski, S., Decraene, B., and J. Tantsura, "IS-IS Extensions for Segment Routing", draft-ietf-isis-segment-routing-extensions-19 (work in progress), July 2018.

[I-D.ietf-ospf-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-25 (work in progress), April 2018.

[I-D.ietf-pce-segment-routing]
Sivabalan, S., Filsfils, C., Tantsura, J., Henderickx, W., and J. Hardwick, "PCEP Extensions for Segment Routing", draft-ietf-pce-segment-routing-13 (work in progress), October 2018.

[I-D.ietf-spring-segment-routing-mpls]
Bashandy, A., Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls-14 (work in progress), June 2018.

[I-D.sivabalan-pce-binding-label-sid]
Sivabalan, S., Tantsura, J., Filsfils, C., Previdi, S., Hardwick, J., and D. Dhody, "Carrying Binding Label/ Segment-ID in PCE-based Networks.", draft-sivabalan-pce-binding-label-sid-04 (work in progress), March 2018.

[RFC4360] Sangli, S., Tappan, D., and Y. Rekhter, "BGP Extended Communities Attribute", RFC 4360, DOI 10.17487/RFC4360, February 2006, <https://www.rfc-editor.org/info/rfc4360>.

[RFC5152] Vasseur, JP., Ed., Ayyangar, A., Ed., and R. Zhang, "A Per-Domain Path Computation Method for Establishing Inter-Domain Traffic Engineering (TE) Label Switched Paths (LSPs)", RFC 5152, DOI 10.17487/RFC5152, February 2008, <https://www.rfc-editor.org/info/rfc5152>.

[RFC5440] Vasseur, JP., Ed. and JL. Le Roux, Ed., "Path Computation Element (PCE) Communication Protocol (PCEP)", RFC 5440, DOI 10.17487/RFC5440, March 2009, <https://www.rfc-editor.org/info/rfc5440>.

[RFC5441] Vasseur, JP., Ed., Zhang, R., Bitar, N., and JL. Le Roux, "A Backward-Recursive PCE-Based Computation (BRPC) Procedure to Compute Shortest Constrained Inter-Domain Traffic Engineering Label Switched Paths", RFC 5441, DOI 10.17487/RFC5441, April 2009, <https://www.rfc-editor.org/info/rfc5441>.

[RFC5520] Bradford, R., Ed., Vasseur, JP., and A. Farrel, "Preserving Topology Confidentiality in Inter-Domain Path Computation Using a Path-Key-Based Mechanism", RFC 5520, DOI 10.17487/RFC5520, April 2009, <https://www.rfc-editor.org/info/rfc5520>.

[RFC6805] King, D., Ed. and A. Farrel, Ed., "The Application of the Path Computation Element Architecture to the Determination of a Sequence of Domains in MPLS and GMPLS", RFC 6805, DOI 10.17487/RFC6805, November 2012, <https://www.rfc-editor.org/info/rfc6805>.

[RFC7752] Gredler, H., Ed., Medved, J., Previdi, S., Farrel, A., and S. Ray, "North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP", RFC 7752, DOI 10.17487/RFC7752, March 2016, <https://www.rfc-editor.org/info/rfc7752>.

[RFC7855] Previdi, S., Ed., Filsfils, C., Ed., Decraene, B., Litkowski, S., Horneffer, M., and R. Shakir, "Source Packet Routing in Networking (SPRING) Problem Statement and Requirements", RFC 7855, DOI 10.17487/RFC7855, May 2016, <https://www.rfc-editor.org/info/rfc7855>.

[RFC7911] Walton, D., Retana, A., Chen, E., and J. Scudder, "Advertisement of Multiple Paths in BGP", RFC 7911, DOI 10.17487/RFC7911, July 2016, <https://www.rfc-editor.org/info/rfc7911>.

[RFC7926] Farrel, A., Ed., Drake, J., Bitar, N., Swallow, G., Ceccarelli, D., and X. Zhang, "Problem Statement and Architecture for Information Exchange between Interconnected Traffic-Engineered Networks", BCP 206, RFC 7926, DOI 10.17487/RFC7926, July 2016, <https://www.rfc-editor.org/info/rfc7926>.

[RFC8231] Crabbe, E., Minei, I., Medved, J., and R. Varga, "Path Computation Element Communication Protocol (PCEP) Extensions for Stateful PCE", RFC 8231, DOI 10.17487/RFC8231, September 2017, <https://www.rfc-editor.org/info/rfc8231>.

[RFC8277] Rosen, E., "Using BGP to Bind MPLS Labels to Address Prefixes", RFC 8277, DOI 10.17487/RFC8277, October 2017, <https://www.rfc-editor.org/info/rfc8277>.

[RFC8281] Crabbe, E., Minei, I., Sivabalan, S., and R. Varga, "Path Computation Element Communication Protocol (PCEP) Extensions for PCE-Initiated LSP Setup in a Stateful PCE Model", RFC 8281, DOI 10.17487/RFC8281, December 2017, <https://www.rfc-editor.org/info/rfc8281>.

[RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <https://www.rfc-editor.org/info/rfc8402>.

Authors' Addresses

Adrian Farrel
Juniper Networks

Email: adrian@olddog.co.uk

John Drake
Juniper Networks

Email: jdrake@juniper.net

SPRING WG
Internet-Draft
Intended status: Standards Track
Expires: May 3, 2018

S. Hegde
Juniper Networks, Inc.

October 30, 2017

Traffic Accounting for MPLS Segment Routing Paths
draft-hegde-spring-traffic-accounting-for-sr-paths-01

Abstract

Traffic statistics form an important part of operations and maintenance data that are used to create demand matrices and for capacity planning in networks. Segment Routing (SR) is a source routing paradigm that uses stack of labels to represent a path. The SR path specific state is not stored in any other node in the network except the head-end node of the SR path. Traffic statistics specific to each SR path are an important component of the data which helps the controllers to lay out the SR paths in a way that optimizes the use of network resources. SR paths are inherently ECMP aware.

As SR paths do not have state in the core of the network, it is not possible to collect the SR path traffic statistics accurately on each interface. This document describes an MPLS forwarding plane mechanism to identify the SR path to which a packet belongs and so facilitate accounting of traffic for MPLS SR paths.

The mechanisms described in this document may also be applied to other MPLS paths (i.e., Label Switched Paths) and can be used to track traffic statistics in multipoint-to-point environments such as those where LDP is in use.

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 https://datatracker.ietf.org/drafts/current/.

Copyright Notice

Table of Contents

1. Introduction

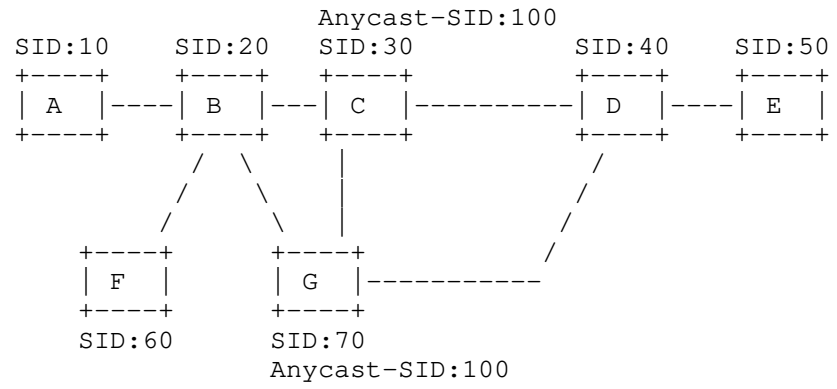
Figure 1 describes an SR enabled network with Node-SIDs and Anycast-SIDs assigned. The SR-Paths with label stacks are as shown in the diagram. The SR-Paths are created (possibly by a central controller) so as to maximize the network resource utilization such as bandwidth. Based on the traffic carried by the SR-Paths, they need to be re-routed occasionally to balance the bandwidth utilization. SR-Paths are inherently ECMP aware.

For example, SR-Path3 in the diagram is balanced across equal cost paths B->C->D and B->G->D. When there is congestion on the link between B and C, the SR path causing the congestion needs to be identified and re-routed. SR paths do not have separate control or forwarding state in any node other than the head-end. Traffic measurement at the head-end node is insufficient to determine the contribution of each SR path to the congestion on the link because of ECMP or Weighted ECMP balancing.

Per-SID traffic measurement on every interface gives some informtion about the traffic carried, but is not sufficient to correctly measure traffic carried by each SR path on the link. If it were possible to identify to which SR path each packet belonged, that information could be used by an external entity to re-route the SR paths to maximize resource utilization.

As SR paths do not have state in the core of the network, it is not possible to collect the SR path traffic statistics accurately on each interface. This document describes an MPLS forwarding plane mechanism to identify the SR path to which a packet belongs and so facilitate accounting of traffic for MPLS SR paths.

The mechanisms described in this document may also be applied to other MPLS paths (i.e., Label Switched Paths) and can be used to track traffic statistics in multipoint-to-point environments such as those where LDP is in use.



SRGB: 1000-2000 on all routers
SR-Path1: A-> 1020,1030
SR-Path2: A-> 1020,1100,1040
SR-Path3: F-> 1020,1040
SR-Path4: A-> 1020,1040,1060

Figure 1: Sample Network

2. Motivation

The motivation of this document is to provide a solution to enable traffic measurement statistics per SR-Path on any node and any link in the network. The objectives listed below help to achieve the requirements in a variety of deployments.

1. The control plane MUST be free of any per SR path state.

2. The forwarding plane MUST be free of any per SR path state.

3. The number of counters created to measure traffic SHOULD be optimized.

4. The additional information carried in each packet SHOULD be minimized.

5. The mechanism SHOULD be applicable to all MPLS environments.

3. Terminology

Source-SID: The (globally unique) Node-SID of the head-end node which places traffic on the SR path. This is a 20 bit number excluding 0-15 and may be encoded in an MPLS label field.

SR-Path-Identifier: An SR-Path-Identifier is an identifier for each SR path in the network. It is unique within the scope of the node that allocated the identifier. If the identifier is allocated by the head-end node (the source) the combination of Source-SID and SR-Path Identifier uniquely identifies an SR path within a network. If the identifier is allocated by a central controller then the SR-Path Identifier is network unique. The SR-Path Identifier is a 19 bit number excluding the values 0-15 and may be encoded in an MPLS label field. See Section 4.

SR-Path-Indicator: The SR-Path-Indicator is an MPLS Special Purpose Label [RFC7274]. This label indicates the presence of an SR-Path Identifier and an Source Node-SID encoded in MPLS label stack entries and situated immediately below this label stack entry in the label stack.

SR-Path-Stats Labels: The SR-Path-Indicator, SR-Path-Identifier, and Source-SID together are termed as the SR-Path-Stats Labels.

4. SR-Path Identifier

4.1. Centrally Managed SR Paths

In controller-based deployments, a controller creates an SR policy, associates a segment list and a Binding SID to the policy, and sends it to the head-end of the SR path as described in [I-D.filsfils-spring-segment-routing-policy]. The controller may also allocate a network-unique SR-Path-Identifier and send it to the head-end along with the policy. When the head-end node receives this policy, if it has not been supplied with an SR-Path-Identifier, it creates a locally-unique identifier for each the SR path network and associates it with SR-TE Policy and advertizes it back to the controller using mechanisms described in [I-D.ietf-idr-te-lsp-distribution].

The SR-Path-Identifier is used for the purpose of traffic accounting as described in Section 5.

4.2. Locally Managed SR Paths

Deployments which do not use a central controller for managing the network configure locally manage SR-Paths on the head-end router. Every SR path in the network is identified using a Source-SID and a source-unique SR-Path-Identifier. The head-end node generates the SR-Path-Identifier for each SR path and associates it with the SR path. An Operator MAY also configure 19-bit globally unique Identifiers on each SR-Path and use it for accounting traffic as described in Section 5

5. Use of the SR-Path-Identifier and Source-SID

The SR-Path-Identifier is a 19 bit number created by the head-end node as described in Section 4. The SR-Path-Identifier and Source-SID are inserted in the packet below a Special Purpose Label called the SR-Path-Indicator. The three values are each carried in a label stack entry as shown in Figure 2.

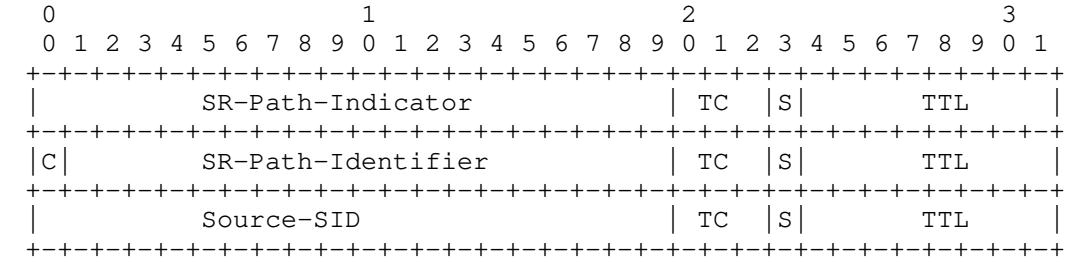


Figure 2: The SR-Path-Stats Labels Encoded in Label Stack Entries

The SR-Path-Indicator label value is TBD-1 to be assigned by IANA.

The SR-Path-Indicator label indicates that the MPLS label stack entries that follow carry an identifier of SR path. These label stack entries MUST NOT be used for forwarding, and if they are encountered at the top of the label stack (for example, at the egress node) they MUST be stripped.

The SR-Path-Identifier label stack entry is inserted immediately below the SR-Path-Indicator. The label field contains two elements:

- The C-flag indicates whether the SR-Path-Identifier is allocated by a central controller or not. If the C-flag is set (one) then this indicates that the SR-Path-Identifier was allocated by a central controller and has global scope, and that a Source-SID is not included. If the C-flag is clear (zero) then the SR-Path-Identifier is scoped by the Source-SID that is included after the SR-Path-Identifier.

- The SR-Path-Identifier identifies the SR path as described in Section 4.

The Source-SID is inserted immediately below the SR-Path-Identifier and is present only if indicated by the setting of the C-flag in the SR-Path-Identifier label stack entry. If present the Source-SID

gives scope to the SR-Path-Identifier. The Source-SID is described in Section 4.

An intermediate node in the network can look into the packet and account the traffic based on the SR-Path-Identifier and Source-SID.

Because it is necessary that the SR-Path-Stats labels are removed when they are found at the top of the label stack, the node imposing the label stack (the ingress) must know which nodes are capable of stripping the labels. This ability is advertised in IGP advertisements defined in TBD and TBD.

6. Inserting the SR-Path-Identifier in Packets

The SR-Path-Identifier and Source-SID are used as a key to account the SR path traffic. The forwarding plane entities should look up the SR-Path-Identifier and Source-SID (if present) values to account the traffic against the right path counters.

The SR-Path-Stats Labels are normally placed at the bottom of the label stack.

Forwarding hardware may have limitations and not support accessing the label stack beyond certain depth. In such cases, the hardware will not be able to find the SR-Path-Stats Labels at the bottom of the label stack if the stack is too deep. To support traffic accounting in such cases it is necessary to insert the SR-Path-Stats Labels within the Readable Label Stack Depth Capability (RLDC) of the nodes in the SR path. The extensions defined in [I-D.ietf-ospf-segment-routing-msd] and [I-D.ietf-isis-segment-routing-msd] describe how the MSD supported by each node is advertised. The head-end node SHOULD insert the SR-Path-Stats Labels at a depth in the label stack such that the nodes in the SR path can access the SR-Path-Identifier for accounting. The SR-Path-Stats Labels may be present multiple times in the label stack of a packet.

In general, if all the nodes in the network support RLDC which is more than the label-stack depth being pushed at the head-end node then the SR-Path-Stats Labels SHOULD be pushed at the bottom of the label-stack. If there are service labels to be inserted, they MUST be pushed at the bottom of the stack. If entropy labels [RFC6790] are to be inserted they SHOULD be pushed next. The SR-Path-Stats Labels SHOULD be pushed next.

It is possible to partially deploy this feature when not all the nodes in the network support the extensions defined in this document. In such scenarios, the special labels MUST NOT get exposed on the top

of the label stack at a node that does not support the extensions defined in this document. This may require multiple blocks of SR-Path-Stats Labels to be inserted in the packet header.

If the egress has not indicated that it is capable of removing the SR-Path-Stats Labels, then they MUST NOT be placed at the bottom of the label stack. In this case the SR-Path-Stats Labels SHOULD be placed at a point in the label stack such that they will be found at the top of stack by the latest node in the SR path that is capable of removing them. In this way, traffic accounting can be performed along as much of the SR path as possible.

7. Traffic-Accounting for Sub SR-Paths in the Network

SR paths may require large label stacks. Some hardware platforms do not support creating such large label stacks (i.e., imposing a large number of labels at once). To overcome this limitation sub-paths are created within the network, and Binding-SIDs are allocated to these sub-paths. When the label representing a Binding-SID is processed it is swapped for a stack of labels. When a head-end node builds the label stack for an SR path, it may use these Binding-SIDs to reduce the depth of the label stack it has to impose and effectively constructs the end-to-end SR path from a series of sub-paths

The sub-paths are not accounted separately. Accounting is performed on the end-to-end SR paths. However, edge routers MAY create Binding-SIDs for BGP-SR-TE Policies as described in [I-D.ietf-idr-segment-routing-te-policy]. Traffic accounting for the traffic carried on the SR paths indicated by these Binding-SIDs can be done separately by allocating separate SR-Path-Identifiers for these sub-paths.

8. Forwarding Plane Procedures

To support per-path traffic accounting, the forwarding plane in a router MUST look through the label stack of a packet for the first instance of the SR-Path-Indicator. The label value in the next label stack entry is the SR-Path-Identifierand the C-flag indicates whether a Source-SID label stack entry is also present. The label values are used as the key for accounting SR path traffic. If the Source-SID label stack entry is absent, an implementation may find it helpful to use a mock Source-SID value of zero for accounting purposes.

The SR-Path-Identifier may be located at different depth in the packet based on the RLDC of nodes in the network as described in Section 6. Finding the SR-Path-Identifier in the packet may be a costly operation and MUST NOT be done unless if SR path accounting is enabled on the device. Implementations MUST include a device-wide

configuration option to enable and disable SR path accounting, and this option MUST default to "off". Implementations SHOULD include more granular configuration (such as per-interface).

A further configuration option is to limit the type of packets to which the procedures described in this section are applied. Thus, the forwarding plane could be configured to inspect only SR packets, or only MPLS packets established using a specific control plane technique (such as LDP). The top label on the incoming packet can be used to determine the nature of the packet and whether to search for the SR-Path-Identifier. The SR labels are predictable and are mostly assigned from SRGB or SRLB. If the top label belongs to any of these label blocks the procedures described in this section may be applied. If the SR label is allocated dynamically as in case of dynamic Adjacency-SIDs, it may be difficult to identify whether the label belongs to SR. It is RECOMMENDED to use configured Adjacency-SIDs when SR path traffic accounting is enabled.

If the top label of the incoming packet is of the right type for accounting and if other appropriate configuration options are enabled, then packet's label stack MUST be examined label by label until an SR-Path-Indicator label is found. The label below SR-Path-Indicator label is the SR-Path-Identifier label and the Source-SID label follows according to the setting of the C-flag. The {incoming interface, SR-Path-Identifier, Source SID} together are the key for traffic accounting. If the Source-SID label stack entry is absent, an implementation may find it helpful to use a mock Source-SID value of zero for accounting purposes.

If a counter does not already exist for that three-tuple, a new counter SHOULD be created. If a counter already exists, it MUST be incremented.

There is no requirement to preemptively create counters for every incoming interface and every SID: the counters need only be created, when a packet is received with the new SR-Path-identifier. This will significantly reduce the number of counters that need to be instantiated as not every interface will receive traffic for any particular SR path.

If the SR-Path-Indicator is the top label in a packet, the SR-Path-Stats labels are popped and further processing is based on the remaining labels in the label stack. Implementations MUST make sure the traffic accounting is carried out before the SR-Path-Stats labels are popped.

9. Consideration of Protection Mechanisms

SR paths typically consist of one or more Node-SIDs, Adjacency-SIDs, Anycast-SIDs, and Binding-SIDs. A variety of protection mechanisms may be in place for these SIDs as described in [I-D.ietf-spring-resiliency-use-cases]. When the head-end node inserts the SR-Path-Stats labels in the label stack, the place in the stack is decided based on whether the node where the special label gets exposed is capable of popping those labels.

When link protection is enabled, the traffic reaches the next-hop node before moving to towards the destination. With link-protection enabled, there is no risk of exposing the special labels at a node that does not support the extensions.

When node-protection is enabled, the traffic skips the next-hop node and reaches the next-next-hop towards the destination. In this case there is a possibility of special labels getting exposed at a node (the Merge Point) that does not support the extensions described in this document. In such cases, the node that receives the packet with special label at the top will discard the packet according to the processing rules of Section 3.18 of [RFC3031]. When using extensions described in this document for traffic accounting and with node-protection enabled in the network, it is RECOMMENDED to make sure all the nodes in the network support the extension.

10. Backward Compatibility

The extensions described in this document are backward compatible. Nodes that do not support the extensions defined in this document will not account the traffic (they will not search for the SR-Path-Indicator), but will forward traffic as normal.

While inserting the SR-Path-Stats labels, the head-end router MUST ensure that the labels are not exposed to the nodes that do not support them. If an error is made such that the SR-Path-Stats labels are exposed at the top of the label stack at a node that does not support this document then that node will discard the packets according to [RFC3031]. While the packets will be black-holed, no further harm will be caused to the network, and since this is a configuration or implementation error, this is an acceptable situation.

If an appropriate point in the label stack cannot be found for the insertion of the SR-Path-Stats labels, the head-end node, head-end MUST NOT insert the SR-Path-Stats labels, but SHOULD continue to label and transmit data. Under such circumstances the head-end node

SHOULD also log the event. A head-end or central controller MAY seek an alternate SR path that allows traffic accounting.

11. Scalability Considerations

The counter space is a limited resource in hardware. As described in Section 8 counters need only be created, when a packet is received with the an SR-Path-Identifier. Furthermore, counters need only be maintained where collection of statistics is configured.

Head-end nodes MUST NOT insert SR-Path-Stats labels by default. Careful configuration of which SR paths have statistics collection enabled will help to minimize the number of counters that need to be maintained at transit nodes.

Transit nodes that are constrained for the number of counters that they can support MAY implement mechanisms that sacrifice some under-used counters to create new counters.

As previously noted, the label stack is a prescious resource itself. That means that under some circumstances it is desirable to only use two labels in the SR-Path-Stats label sequence rather than three. This can be achieved by using a central controller to allocate SR-Path-Identifier values and set the C-flag to indicate that no Source-SID is used.

Conversely, in a large network with a central controller the SR-Path-Identifier may be a prescious resource. That is, there may be more than 2^19 SR paths that need identifiers to be allocated. In this case, a central controller may use knowledge of label stack depth and network node capabilities to allocate SR-Path-Indicators that include a Source-SID (set to indicate the controller, itself) where that would not cause a problem in the network.

12. Security Considerations

As noted in Section 11 the counter space is a limited resource in hardware. This document introduces dynamic creation of counters based on packet headers of the incoming packets. There is the possibility that a DOS attack is mounted by requesting new counter creation on each packet. Implementations SHOULD monitor the counter space and generate appropriate warnings if the counter space is getting exhausted. Implementations SHOULD control the rate at which the counters get created to mitigate DOS attacks.

13. IANA Considerations

IANA maintains a registry called the "Multiprotocol Label Switching Architecture (MPLS) Label Values" registry. IANA is requested to make a new assignment from this registry as follows:

Value	Description	Reference
TBD-1	SR Path Indicator	[This.I-D]

14. Acknowledgements

Thanks to John Drake, Harish Sitaraman, and Ron Bonica for helpful discussions.

15. Contributors

Adrian Farrel
Juniper Networks

Email: afarrel@juinper.net

16. References

16.1. Normative References

[I-D.ietf-idr-te-lsp-distribution]
Previdi, S., Dong, J., Chen, M., Gredler, H., and j. jefftant@gmail.com, "Distribution of Traffic Engineering (TE) Policies and State using BGP-LS", draft-ietf-idr-te-lsp-distribution-07 (work in progress), July 2017.

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

[RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, DOI 10.17487/RFC3031, January 2001, <https://www.rfc-editor.org/info/rfc3031>.

16.2. Informative References

[I-D.filsfils-spring-segment-routing-policy]
Filsfils, C., Sivabalan, S., Raza, K., Liste, J., Clad, F., Lin, S., bogdanov@google.com, b., Horneffer, M., Steinberg, D., Decraene, B., and S. Litkowski, "Segment Routing Policy for Traffic Engineering", draft-filsfils-spring-segment-routing-policy-01 (work in progress), July 2017.

[I-D.ietf-idr-segment-routing-te-policy]
Previdi, S., Filsfils, C., Mattes, P., Rosen, E., and S. Lin, "Advertising Segment Routing Policies in BGP", draft-ietf-idr-segment-routing-te-policy-00 (work in progress), July 2017.

[I-D.ietf-isis-segment-routing-msd]
Tantsura, J., Chunduri, U., Aldrin, S., and L. Ginsberg, "Signaling MSD (Maximum SID Depth) using IS-IS", draft-ietf-isis-segment-routing-msd-04 (work in progress), June 2017.

[I-D.ietf-ospf-segment-routing-msd]
Tantsura, J., Chunduri, U., Aldrin, S., and P. Psenak, "Signaling MSD (Maximum SID Depth) using OSPF", draft-ietf-ospf-segment-routing-msd-05 (work in progress), June 2017.

[I-D.ietf-spring-resiliency-use-cases]
Filsfils, C., Previdi, S., Decraene, B., and R. Shakir, "Resiliency use cases in SPRING networks", draft-ietf-spring-resiliency-use-cases-11 (work in progress), May 2017.

[RFC6790] Kompella, K., Drake, J., Amante, S., Henderickx, W., and L. Yong, "The Use of Entropy Labels in MPLS Forwarding", RFC 6790, DOI 10.17487/RFC6790, November 2012, <https://www.rfc-editor.org/info/rfc6790>.

[RFC7274] Kompella, K., Andersson, L., and A. Farrel, "Allocating and Retiring Special-Purpose MPLS Labels", RFC 7274, DOI 10.17487/RFC7274, June 2014, <https://www.rfc-editor.org/info/rfc7274>.

Author's Address

Shraddha Hegde
Juniper Networks, Inc.
Embassy Business Park
Bangalore, KA 560093
India

Email: shraddha@juniper.net

SPRING WG
Internet-Draft
Intended status: Standards Track
Expires: April 20, 2019

S. Hegde
Juniper Networks, Inc.

October 17, 2018

Traffic Accounting for MPLS Segment Routing Paths
draft-hegde-spring-traffic-accounting-for-sr-paths-02

Abstract

Traffic statistics form an important part of operations and maintenance data that are used to create demand matrices and for capacity planning in networks. Segment Routing (SR) is a source routing paradigm that uses stack of labels to represent a path. The SR path specific state is not stored in any other node in the network except the head-end node of the SR path. Traffic statistics specific to each SR path are an important component of the data which helps the controllers to lay out the SR paths in a way that optimizes the use of network resources. SR paths are inherently ECMP aware.

As SR paths do not have state in the core of the network, it is not possible to collect the SR path traffic statistics accurately on each interface. This document describes an MPLS forwarding plane mechanism to identify the SR path to which a packet belongs and so facilitate accounting of traffic for MPLS SR paths.

The mechanisms described in this document may also be applied to other MPLS paths (i.e., Label Switched Paths) and can be used to track traffic statistics in multipoint-to-point environments such as those where LDP is in use.

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 https://datatracker.ietf.org/drafts/current/.

Copyright Notice

Table of Contents

1. Introduction

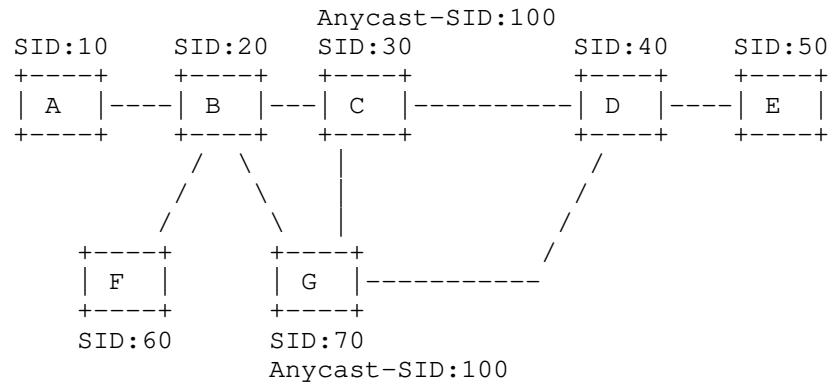
Figure 1 describes an SR enabled network with Node-SIDs and Anycast-SIDs assigned. The SR-Paths with label stacks are as shown in the diagram. The SR-Paths are created (possibly by a central controller) so as to maximize the network resource utilization such as bandwidth. Based on the traffic carried by the SR-Paths, they need to be re-routed occasionally to balance the bandwidth utilization. SR-Paths are inherently ECMP aware.

For example, SR-Path3 in the diagram is balanced across equal cost paths B->C->D and B->G->D. When there is congestion on the link between B and C, the SR path causing the congestion needs to be identified and re-routed. SR paths do not have separate control or forwarding state in any node other than the head-end. Traffic measurement at the head-end node is insufficient to determine the contribution of each SR path to the congestion on the link because of ECMP or Weighted ECMP balancing.

Per-SID traffic measurement on every interface gives some informtion about the traffic carried, but is not sufficient to correctly measure traffic carried by each SR path on the link. If it were possible to identify to which SR path each packet belonged, that information could be used by an external entity to re-route the SR paths to maximize resource utilization.

As SR paths do not have state in the core of the network, it is not possible to collect the SR path traffic statistics accurately on each interface. This document describes an MPLS forwarding plane mechanism to identify the SR path to which a packet belongs and so facilitate accounting of traffic for MPLS SR paths.

The mechanisms described in this document may also be applied to other MPLS paths (i.e., Label Switched Paths) and can be used to track traffic statistics in multipoint-to-point environments such as those where LDP is in use.



SRGB: 1000-2000 on all routers
SR-Path1: A-> 1020,1030
SR-Path2: A-> 1020,1100,1040
SR-Path3: F-> 1020,1040
SR-Path4: A-> 1020,1040,1060

Figure 1: Sample Network

2. Motivation

The motivation of this document is to provide a solution to enable traffic measurement statistics per SR-Path on any node and any link in the network. The objectives listed below help to achieve the requirements in a variety of deployments.

1. The control plane MUST be free of any per SR path state.

2. The forwarding plane MUST be free of any per SR path state.

3. The number of counters created to measure traffic SHOULD be optimized.

4. The additional information carried in each packet SHOULD be minimized.

5. The mechanism SHOULD be applicable to all MPLS environments.

3. Terminology

Source-SID: The (globally unique) Node-SID of the head-end node which places traffic on the SR path. This is a 20 bit number excluding 0-15 and may be encoded in an MPLS label field.

SR-Path-Identifier: An SR-Path-Identifier is an identifier for each SR path in the network. It is unique within the scope of the node that allocated the identifier. If the identifier is allocated by the head-end node (the source) the combination of Source-SID and SR-Path Identifier uniquely identifies an SR path within a network. If the identifier is allocated by a central controller then the SR-Path Identifier is network unique. The SR-Path Identifier is a 19 bit number excluding the values 0-15 and may be encoded in an MPLS label field. See Section 4.

SR-Path-Indicator: The SR-Path-Indicator is an MPLS Special Purpose Label [RFC7274]. This label indicates the presence of an SR-Path Identifier and an Source Node-SID encoded in MPLS label stack entries and situated immediately below this label stack entry in the label stack.

SR-Path-Stats Labels: The SR-Path-Indicator, SR-Path-Identifier, and Source-SID together are termed as the SR-Path-Stats Labels.

4. SR-Path Identifier

4.1. Centrally Managed SR Paths

In controller-based deployments, a controller creates an SR policy, associates a segment list and a Binding SID to the policy, and sends it to the head-end of the SR path as described in [I-D.filsfils-spring-segment-routing-policy]. The controller may also allocate a network-unique SR-Path-Identifier and send it to the head-end along with the policy. When the head-end node receives this policy, if it has not been supplied with an SR-Path-Identifier, it creates a locally-unique identifier for each the SR path network and associates it with SR-TE Policy and advertizes it back to the controller using mechanisms described in [I-D.ietf-idr-te-lsp-distribution].

The SR-Path-Identifier is used for the purpose of traffic accounting as described in Section 5.

4.2. Locally Managed SR Paths

Deployments which do not use a central controller for managing the network configure locally manage SR-Paths on the head-end router. Every SR path in the network is identified using a Source-SID and a source-unique SR-Path-Identifier. The head-end node generates the SR-Path-Identifier for each SR path and associates it with the SR path. An Operator MAY also configure 19-bit globally unique Identifiers on each SR-Path and use it for accounting traffic as described in Section 5

5. Use of the SR-Path-Identifier and Source-SID

The SR-Path-Identifier is a 19 bit number created by the head-end node as described in Section 4. The SR-Path-Identifier and Source-SID are inserted in the packet below a Special Purpose Label called the SR-Path-Indicator. The three values are each carried in a label stack entry as shown in Figure 2.

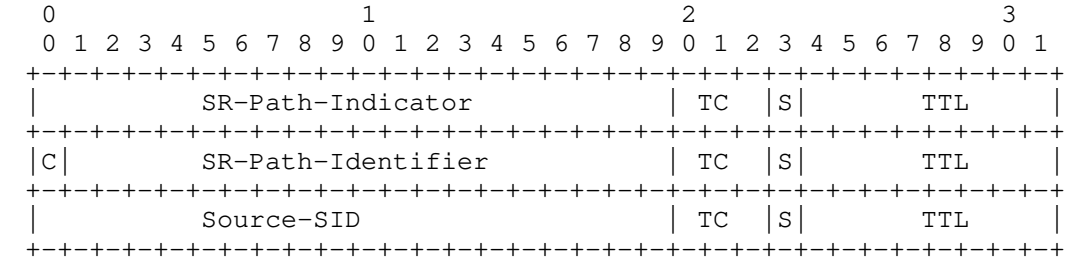


Figure 2: The SR-Path-Stats Labels Encoded in Label Stack Entries

The SR-Path-Indicator label value is TBD-1 to be assigned by IANA.

The SR-Path-Indicator label indicates that the MPLS label stack entries that follow carry an identifier of SR path. These label stack entries MUST NOT be used for forwarding, and if they are encountered at the top of the label stack (for example, at the egress node) they MUST be stripped.

The SR-Path-Identifier label stack entry is inserted immediately below the SR-Path-Indicator. The label field contains two elements:

- The C-flag indicates whether the SR-Path-Identifier is allocated by a central controller or not. If the C-flag is set (one) then this indicates that the SR-Path-Identifier was allocated by a central controller and has global scope, and that a Source-SID is not included. If the C-flag is clear (zero) then the SR-Path-Identifier is scoped by the Source-SID that is included after the SR-Path-Identifier.

- The SR-Path-Identifier identifies the SR path as described in Section 4.

The Source-SID is inserted immediately below the SR-Path-Identifier and is present only if indicated by the setting of the C-flag in the SR-Path-Identifier label stack entry. If present the Source-SID

gives scope to the SR-Path-Identifier. The Source-SID is described in Section 4.

An intermediate node in the network can look into the packet and account the traffic based on the SR-Path-Identifier and Source-SID.

Because it is necessary that the SR-Path-Stats labels are removed when they are found at the top of the label stack, the node imposing the label stack (the ingress) must know which nodes are capable of stripping the labels. This ability is advertised in IGP advertisements defined in TBD and TBD.

6. Inserting the SR-Path-Identifier in Packets

The SR-Path-Identifier and Source-SID are used as a key to account the SR path traffic. The forwarding plane entities should look up the SR-Path-Identifier and Source-SID (if present) values to account the traffic against the right path counters.

The SR-Path-Stats Labels are normally placed at the bottom of the label stack.

Forwarding hardware may have limitations and not support accessing the label stack beyond certain depth. In such cases, the hardware will not be able to find the SR-Path-Stats Labels at the bottom of the label stack if the stack is too deep. To support traffic accounting in such cases it is necessary to insert the SR-Path-Stats Labels within the Readable Label Stack Depth Capability (RLDC) of the nodes in the SR path. The extensions defined in [I-D.ietf-ospf-segment-routing-msd] and [I-D.ietf-isis-segment-routing-msd] describe how the MSD supported by each node is advertised. The head-end node SHOULD insert the SR-Path-Stats Labels at a depth in the label stack such that the nodes in the SR path can access the SR-Path-Identifier for accounting. The SR-Path-Stats Labels may be present multiple times in the label stack of a packet.

In general, if all the nodes in the network support RLDC which is more than the label-stack depth being pushed at the head-end node then the SR-Path-Stats Labels SHOULD be pushed at the bottom of the label-stack. If there are service labels to be inserted, they MUST be pushed at the bottom of the stack. If entropy labels [RFC6790] are to be inserted they SHOULD be pushed next. The SR-Path-Stats Labels SHOULD be pushed next.

It is possible to partially deploy this feature when not all the nodes in the network support the extensions defined in this document. In such scenarios, the special labels MUST NOT get exposed on the top

of the label stack at a node that does not support the extensions defined in this document. This may require multiple blocks of SR-Path-Stats Labels to be inserted in the packet header.

If the egress has not indicated that it is capable of removing the SR-Path-Stats Labels, then they MUST NOT be placed at the bottom of the label stack. In this case the SR-Path-Stats Labels SHOULD be placed at a point in the label stack such that they will be found at the top of stack by the latest node in the SR path that is capable of removing them. In this way, traffic accounting can be performed along as much of the SR path as possible.

7. Traffic-Accounting for Sub SR-Paths in the Network

SR paths may require large label stacks. Some hardware platforms do not support creating such large label stacks (i.e., imposing a large number of labels at once). To overcome this limitation sub-paths are created within the network, and Binding-SIDs are allocated to these sub-paths. When the label representing a Binding-SID is processed it is swapped for a stack of labels. When a head-end node builds the label stack for an SR path, it may use these Binding-SIDs to reduce the depth of the label stack it has to impose and effectively constructs the end-to-end SR path from a series of sub-paths

The sub-paths are not accounted separately. Accounting is performed on the end-to-end SR paths. However, edge routers MAY create Binding-SIDs for BGP-SR-TE Policies as described in [I-D.ietf-idr-segment-routing-te-policy]. Traffic accounting for the traffic carried on the SR paths indicated by these Binding-SIDs can be done separately by allocating separate SR-Path-Identifiers for these sub-paths.

8. Forwarding Plane Procedures

To support per-path traffic accounting, the forwarding plane in a router MUST look through the label stack of a packet for the first instance of the SR-Path-Indicator. The label value in the next label stack entry is the SR-Path-Identifierand the C-flag indicates whether a Source-SID label stack entry is also present. The label values are used as the key for accounting SR path traffic. If the Source-SID label stack entry is absent, an implementation may find it helpful to use a mock Source-SID value of zero for accounting purposes.

The SR-Path-Identifier may be located at different depth in the packet based on the RLDC of nodes in the network as described in Section 6. Finding the SR-Path-Identifier in the packet may be a costly operation and MUST NOT be done unless if SR path accounting is enabled on the device. Implementations MUST include a device-wide

configuration option to enable and disable SR path accounting, and this option MUST default to "off". Implementations SHOULD include more granular configuration (such as per-interface).

A further configuration option is to limit the type of packets to which the procedures described in this section are applied. Thus, the forwarding plane could be configured to inspect only SR packets, or only MPLS packets established using a specific control plane technique (such as LDP). The top label on the incoming packet can be used to determine the nature of the packet and whether to search for the SR-Path-Identifier. The SR labels are predictable and are mostly assigned from SRGB or SRLB. If the top label belongs to any of these label blocks the procedures described in this section may be applied. If the SR label is allocated dynamically as in case of dynamic Adjacency-SIDs, it may be difficult to identify whether the label belongs to SR. It is RECOMMENDED to use configured Adjacency-SIDs when SR path traffic accounting is enabled.

If the top label of the incoming packet is of the right type for accounting and if other appropriate configuration options are enabled, then packet's label stack MUST be examined label by label until an SR-Path-Indicator label is found. The label below SR-Path-Indicator label is the SR-Path-Identifier label and the Source-SID label follows according to the setting of the C-flag. The {incoming interface, SR-Path-Identifier, Source SID} together are the key for traffic accounting. If the Source-SID label stack entry is absent, an implementation may find it helpful to use a mock Source-SID value of zero for accounting purposes.

If a counter does not already exist for that three-tuple, a new counter SHOULD be created. If a counter already exists, it MUST be incremented.

There is no requirement to preemptively create counters for every incoming interface and every SID: the counters need only be created, when a packet is received with the new SR-Path-identifier. This will significantly reduce the number of counters that need to be instantiated as not every interface will receive traffic for any particular SR path.

If the SR-Path-Indicator is the top label in a packet, the SR-Path-Stats labels are popped and further processing is based on the remaining labels in the label stack. Implementations MUST make sure the traffic accounting is carried out before the SR-Path-Stats labels are popped.

9. Consideration of Protection Mechanisms

SR paths typically consist of one or more Node-SIDs, Adjacency-SIDs, Anycast-SIDs, and Binding-SIDs. A variety of protection mechanisms may be in place for these SIDs as described in [I-D.ietf-spring-resiliency-use-cases]. When the head-end node inserts the SR-Path-Stats labels in the label stack, the place in the stack is decided based on whether the node where the special label gets exposed is capable of popping those labels.

When link protection is enabled, the traffic reaches the next-hop node before moving to towards the destination. With link-protection enabled, there is no risk of exposing the special labels at a node that does not support the extensions.

When node-protection is enabled, the traffic skips the next-hop node and reaches the next-next-hop towards the destination. In this case there is a possibility of special labels getting exposed at a node (the Merge Point) that does not support the extensions described in this document. In such cases, the node that receives the packet with special label at the top will discard the packet according to the processing rules of Section 3.18 of [RFC3031]. When using extensions described in this document for traffic accounting and with node-protection enabled in the network, it is RECOMMENDED to make sure all the nodes in the network support the extension.

10. Backward Compatibility

The extensions described in this document are backward compatible. Nodes that do not support the extensions defined in this document will not account the traffic (they will not search for the SR-Path-Indicator), but will forward traffic as normal.

While inserting the SR-Path-Stats labels, the head-end router MUST ensure that the labels are not exposed to the nodes that do not support them. If an error is made such that the SR-Path-Stats labels are exposed at the top of the label stack at a node that does not support this document then that node will discard the packets according to [RFC3031]. While the packets will be black-holed, no further harm will be caused to the network, and since this is a configuration or implementation error, this is an acceptable situation.

If an appropriate point in the label stack cannot be found for the insertion of the SR-Path-Stats labels, the head-end node, head-end MUST NOT insert the SR-Path-Stats labels, but SHOULD continue to label and transmit data. Under such circumstances the head-end node

SHOULD also log the event. A head-end or central controller MAY seek an alternate SR path that allows traffic accounting.

11. Scalability Considerations

The counter space is a limited resource in hardware. As described in Section 8 counters need only be created, when a packet is received with the an SR-Path-Identifier. Furthermore, counters need only be maintained where collection of statistics is configured.

Head-end nodes MUST NOT insert SR-Path-Stats labels by default. Careful configuration of which SR paths have statistics collection enabled will help to minimize the number of counters that need to be maintained at transit nodes.

Transit nodes that are constrained for the number of counters that they can support MAY implement mechanisms that sacrifice some under-used counters to create new counters.

As previously noted, the label stack is a prescious resource itself. That means that under some circumstances it is desirable to only use two labels in the SR-Path-Stats label sequence rather than three. This can be achieved by using a central controller to allocate SR-Path-Identifier values and set the C-flag to indicate that no Source-SID is used.

Conversely, in a large network with a central controller the SR-Path-Identifier may be a prescious resource. That is, there may be more than 2^19 SR paths that need identifiers to be allocated. In this case, a central controller may use knowledge of label stack depth and network node capabilities to allocate SR-Path-Indicators that include a Source-SID (set to indicate the controller, itself) where that would not cause a problem in the network.

12. Security Considerations

As noted in Section 11 the counter space is a limited resource in hardware. This document introduces dynamic creation of counters based on packet headers of the incoming packets. There is the possibility that a DOS attack is mounted by requesting new counter creation on each packet. Implementations SHOULD monitor the counter space and generate appropriate warnings if the counter space is getting exhausted. Implementations SHOULD control the rate at which the counters get created to mitigate DOS attacks.

13. IANA Considerations

IANA maintains a registry called the "Multiprotocol Label Switching Architecture (MPLS) Label Values" registry. IANA is requested to make a new assignment from this registry as follows:

Value	Description	Reference
TBD-1	SR Path Indicator	[This.I-D]

14. Acknowledgements

Thanks to John Drake, Harish Sitaraman, and Ron Bonica for helpful discussions.

15. Contributors

Adrian Farrel
Juniper Networks

Email: afarrel@juinper.net

16. References

16.1. Normative References

[I-D.ietf-idr-te-lsp-distribution] Previdi, S., Talaulikar, K., Dong, J., Chen, M., Gredler, H., and J. Tantsura, "Distribution of Traffic Engineering (TE) Policies and State using BGP-LS", draft-ietf-idr-te-lsp-distribution-09 (work in progress), June 2018.

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

[RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, DOI 10.17487/RFC3031, January 2001, <https://www.rfc-editor.org/info/rfc3031>.

16.2. Informative References

[I-D.filsfils-spring-segment-routing-policy]
Filsfils, C., Sivabalan, S., Hegde, S., daniel.voyer@bell.ca, d., Lin, S., bogdanov@google.com, b., Krol, P., Horneffer, M., Steinberg, D., Decraene, B., Litkowski, S., Mattes, P., Ali, Z., Talaulikar, K., Liste, J., Clad, F., and K. Raza, "Segment Routing Policy Architecture", draft-filsfils-spring-segment-routing-policy-06 (work in progress), May 2018.

[I-D.ietf-idr-segment-routing-te-policy]
Previdi, S., Filsfils, C., Jain, D., Mattes, P., Rosen, E., and S. Lin, "Advertising Segment Routing Policies in BGP", draft-ietf-idr-segment-routing-te-policy-04 (work in progress), July 2018.

[I-D.ietf-isis-segment-routing-msd]
Tantsura, J., Chunduri, U., Aldrin, S., and L. Ginsberg, "Signaling MSD (Maximum SID Depth) using IS-IS", draft-ietf-isis-segment-routing-msd-19 (work in progress), October 2018.

[I-D.ietf-ospf-segment-routing-msd]
Tantsura, J., Chunduri, U., Aldrin, S., and P. Psenak, "Signaling MSD (Maximum SID Depth) using OSPF", draft-ietf-ospf-segment-routing-msd-23 (work in progress), October 2018.

[I-D.ietf-spring-resiliency-use-cases]
Filsfils, C., Previdi, S., Decraene, B., and R. Shakir, "Resiliency use cases in SPRING networks", draft-ietf-spring-resiliency-use-cases-12 (work in progress), December 2017.

[RFC6790] Kompella, K., Drake, J., Amante, S., Henderickx, W., and L. Yong, "The Use of Entropy Labels in MPLS Forwarding", RFC 6790, DOI 10.17487/RFC6790, November 2012, <https://www.rfc-editor.org/info/rfc6790>.

[RFC7274] Kompella, K., Andersson, L., and A. Farrel, "Allocating and Retiring Special-Purpose MPLS Labels", RFC 7274, DOI 10.17487/RFC7274, June 2014, <https://www.rfc-editor.org/info/rfc7274>.

Author's Address

Shraddha Hegde
Juniper Networks, Inc.
Embassy Business Park
Bangalore, KA 560093
India

Email: shraddha@juniper.net

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: July 29, 2018

C. Filsfils, Ed.
S. Previdi, Ed.
Cisco Systems, Inc.
L. Ginsberg
Cisco Systems, Inc
B. Decraene
S. Litkowski
Orange
R. Shakir
Google, Inc.
January 25, 2018

Segment Routing Architecture
draft-ietf-spring-segment-routing-15

Abstract

Segment Routing (SR) leverages the source routing paradigm. A node steers a packet through an ordered list of instructions, called segments. A segment can represent any instruction, topological or service-based. A segment can have a semantic local to an SR node or global within an SR domain. SR allows to enforce a flow through any topological path while maintaining per-flow state only at the ingress nodes to the SR domain.

Segment Routing can be directly applied to the MPLS architecture with no change on the forwarding plane. A segment is encoded as an MPLS label. An ordered list of segments is encoded as a stack of labels. The segment to process is on the top of the stack. Upon completion of a segment, the related label is popped from the stack.

Segment Routing can be applied to the IPv6 architecture, with a new type of routing header. A segment is encoded as an IPv6 address. An ordered list of segments is encoded as an ordered list of IPv6 addresses in the routing header. The active segment is indicated by the Destination Address of the packet. The next active segment is indicated by a pointer in the new routing header.

Requirements Language

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

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Segment Routing (SR) leverages the source routing paradigm. A node steers a packet through an SR Policy instantiated as an ordered list of instructions called segments. A segment can represent any instruction, topological or service-based. A segment can have a semantic local to an SR node or global within an SR domain. SR supports per-flow explicit routing while maintaining per-flow state only at the ingress nodes to the SR domain.

A segment is often referred to by its Segment Identifier (SID).

A segment may be associated with a topological instruction. A topological local segment may instruct a node to forward the packet via a specific outgoing interface. A topological global segment may instruct an SR domain to forward the packet via a specific path to a destination. Different segments may exist for the same destination, each with different path objectives (e.g., which metric is minimized, what constraints are specified).

A segment may be associated with a service instruction (e.g. the packet should be processed by a container or VM associated with the segment). A segment may be associated with a QoS treatment (e.g., shape the packets received with this segment at x Mbps).

The SR architecture supports any type of instruction associated with a segment.

The SR architecture supports any type of control-plane: distributed, centralized or hybrid.

In a distributed scenario, the segments are allocated and signaled by IS-IS or OSPF or BGP. A node individually decides to steer packets on a source-routed policy (e.g., pre-computed local protection [I-D.ietf-spring-resiliency-use-cases]) . A node individually computes the source-routed policy.

In a centralized scenario, the segments are allocated and instantiated by an SR controller. The SR controller decides which nodes need to steer which packets on which source-routed policies. The SR controller computes the source-routed policies. The SR architecture does not restrict how the controller programs the network. Likely options are NETCONF, PCEP and BGP. The SR architecture does not restrict the number of SR controllers. Specifically multiple SR controllers may program the same SR domain. The SR architecture allows these SR controllers to discover which SID's are instantiated at which nodes and which sets of local (SRLB) and global labels (SRGB) are available at which node.

A hybrid scenario complements a base distributed control-plane with a centralized controller. For example, when the destination is outside the IGP domain, the SR controller may compute a source-routed policy on behalf of an IGP node. The SR architecture does not restrict how the nodes which are part of the distributed control-plane interact with the SR controller. Likely options are PCEP and BGP.

Hosts MAY be part of an SR Domain. A centralized controller can inform hosts about policies either by pushing these policies to hosts or responding to requests from hosts.

The SR architecture can be instantiated on various dataplanes. This document introduces two dataplane instantiations of SR: SR over MPLS (SR-MPLS) and SR over IPv6 (SRv6).

Segment Routing can be directly applied to the MPLS architecture with no change on the forwarding plane [I-D.ietf-spring-segment-routing-mpls] A segment is encoded as an MPLS label. An SR Policy is instantiated as a stack of labels. The segment to process (the active segment) is on the top of the stack. Upon completion of a segment, the related label is popped from the stack.

Segment Routing can be applied to the IPv6 architecture with a new type of routing header called the SR header (SRH) [I-D.ietf-6man-segment-routing-header] . An instruction is associated with a segment and encoded as an IPv6 address. An SRv6 segment is

also called an SRv6 SID. An SR Policy is instantiated as an ordered list of SRv6 SID's in the routing header. The active segment is indicated by the Destination Address(DA) of the packet. The next active segment is indicated by the SegmentsLeft (SL) pointer in the SRH. When an SRv6 SID is completed, the SL is decremented and the next segment is copied to the DA. When a packet is steered on an SR policy, the related SRH is added to the packet.

In the context of an IGP-based distributed control-plane, two topological segments are defined: the IGP adjacency segment and the IGP prefix segment.

In the context of a BGP-based distributed control-plane, two topological segments are defined: the BGP peering segment and the BGP prefix segment.

The headend of an SR Policy binds a SID (called Binding segment or BSID) to its policy. When the headend receives a packet with active segment matching the BSID of a local SR Policy, the headend steers the packet into the associated SR Policy.

This document defines the IGP, BGP and Binding segments for the SR-MPLS and SRv6 dataplanes.

Note: This document defines the architecture for Segment Routing, including definitions of basic objects and functions and a description of the overall design. It does NOT define the means of implementing the architecture - that is contained in numerous referencing documents, some of which are mentioned in this document as a convenience to the reader.

2. Terminology

SR-MPLS: the instantiation of SR on the MPLS dataplane

SRv6: the instantiation of SR on the IPv6 dataplane.

Segment: an instruction a node executes on the incoming packet (e.g., forward packet according to shortest path to destination, or, forward packet through a specific interface, or, deliver the packet to a given application/service instance).

SID: a segment identifier. Note that the term SID is commonly used in place of the term Segment, though this is technically imprecise as it overlooks any necessary translation.

SR-MPLS SID: an MPLS label or an index value into an MPLS label space explicitly associated with the segment.

SRv6 SID: an IPv6 address explicitly associated with the segment.

Segment Routing Domain (SR Domain): the set of nodes participating in the source based routing model. These nodes may be connected to the same physical infrastructure (e.g., a Service Provider's network). They may as well be remotely connected to each other (e.g., an enterprise VPN or an overlay). If multiple protocol instances are deployed, the SR domain most commonly includes all of the protocol instances in a network. However, some deployments may wish to sub-divide the network into multiple SR domains, each of which includes one or more protocol instances. It is expected that all nodes in an SR Domain are managed by the same administrative entity.

Active Segment: the segment that is used by the receiving router to process the packet. In the MPLS dataplane it is the top label. In the IPv6 dataplane it is the destination address. [I-D.ietf-6man-segment-routing-header].

PUSH: the instruction consisting of the insertion of a segment at the top of the segment list. In SR-MPLS the top of the segment list is the topmost (outer) label of the label stack. In SRv6, the top of the segment list is represented by the first segment in the Segment Routing Header as defined in [I-D.ietf-6man-segment-routing-header].

NEXT: when the active segment is completed, NEXT is the instruction consisting of the inspection of the next segment. The next segment becomes active. In SR-MPLS, NEXT is implemented as a POP of the top label. In SRv6, NEXT is implemented as the copy of the next segment from the SRH to the Destination Address of the IPv6 header.

CONTINUE: the active segment is not completed and hence remains active. In SR-MPLS, CONTINUE instruction is implemented as a SWAP of the top label. [RFC3031] In SRv6, this is the plain IPv6 forwarding action of a regular IPv6 packet according to its Destination Address.

SR Global Block (SRGB): the set of global segments in the SR Domain. If a node participates in multiple SR domains, there is one SRGB for each SR domain. In SR-MPLS, SRGB is a local property of a node and identifies the set of local labels reserved for global segments. In SR-MPLS, using identical SRGBs on all nodes within the SR Domain is strongly recommended. Doing so eases operations and troubleshooting as the same label represents the same global segment at each node. In SRv6, the SRGB is the set of global SRv6 SIDs in the SR Domain.

SR Local Block (SRLB): local property of an SR node. If a node participates in multiple SR domains, there is one SRLB for each SR domain. In SR-MPLS, SRLB is a set of local labels reserved for local segments. In SRv6, SRLB is a set of local IPv6 addresses reserved

for local SRv6 SID's. In a controller-driven network, some controllers or applications may use the control plane to discover the available set of local segments.

Global Segment: a segment which is part of the SRGB of the domain. The instruction associated to the segment is defined at the SR Domain level. A topological shortest-path segment to a given destination within an SR domain is a typical example of a global segment.

Local Segment: In SR-MPLS, this is a local label outside the SRGB. It may be part of the explicitly advertised SRLB. In SRv6, this can be any IPv6 address i.e., the address may be part of the SRGB but used such that it has local significance. The instruction associated to the segment is defined at the node level.

IGP Segment: the generic name for a segment attached to a piece of information advertised by a link-state IGP, e.g. an IGP prefix or an IGP adjacency.

IGP-Prefix Segment: an IGP-Prefix Segment is an IGP Segment representing an IGP prefix. When an IGP-Prefix Segment is global within the SR IGP instance/topology it identifies an instruction to forward the packet along the path computed using the routing algorithm specified in the algorithm field, in the topology and the IGP instance where it is advertised. Also referred to as Prefix Segment.

Prefix SID: the SID of the IGP-Prefix Segment.

IGP-Anycast Segment: an IGP-Anycast Segment is an IGP-Prefix Segment which identify an anycast prefix advertised by a set of routers.

Anycast-SID: the SID of the IGP-Anycast Segment.

IGP-Adjacency Segment: an IGP-Adjacency Segment is an IGP Segment attached to a unidirectional adjacency or a set of unidirectional adjacencies. By default, an IGP-Adjacency Segment is local (unless explicitly advertised otherwise) to the node that advertises it. Also referred to as Adjacency Segment.

Adj-SID: the SID of the IGP-Adjacency Segment.

IGP-Node Segment: an IGP-Node Segment is an IGP-Prefix Segment which identifies a specific router (e.g., a loopback). Also referred to as Node Segment.

Node-SID: the SID of the IGP-Node Segment.

SR Policy: an ordered list of segments. The headend of an SR Policy steers packets onto the SR policy. The list of segments can be specified explicitly in SR-MPLS as a stack of labels and in SRv6 as an ordered list of SRv6 SID's. Alternatively, the list of segments is computed based on a destination and a set of optimization objective and constraints (e.g., latency, affinity, SRLG, ...). The computation can be local or delegated to a PCE server. An SR policy can be configured by the operator, provisioned via NETCONF [RFC6241] or provisioned via PCEP [RFC5440] . An SR policy can be used for traffic-engineering, OAM or FRR reasons.

Segment List Depth: the number of segments of an SR policy. The entity instantiating an SR Policy at a node N should be able to discover the depth insertion capability of the node N. For example, the PCEP SR capability advertisement described in [I-D.ietf-pce-segment-routing] is one means of discovering this capability.

Forwarding Information Base (FIB): the forwarding table of a node

3. Link-State IGP Segments

Within an SR domain, an SR-capable IGP node advertises segments for its attached prefixes and adjacencies. These segments are called IGP segments or IGP SIDs. They play a key role in Segment Routing and use-cases as they enable the expression of any path throughout the SR domain. Such a path is either expressed as a single IGP segment or a list of multiple IGP segments.

Advertisement of IGP segments requires extensions in link-state IGP protocols. These extensions are defined in
[I-D.ietf-isis-segment-routing-extensions]
[I-D.ietf-ospf-segment-routing-extensions]
[I-D.ietf-ospf-ospfv3-segment-routing-extensions]

3.1. IGP-Prefix Segment, Prefix-SID

An IGP-Prefix segment is an IGP segment attached to an IGP prefix. An IGP-Prefix segment is global (unless explicitly advertised otherwise) within the SR domain. The context for an IGP-Prefix segment includes the prefix, topology, and algorithm. Multiple SIDs MAY be allocated to the same prefix so long as the tuple <prefix, topology, algorithm> is unique.

Multiple instances and topologies are defined in IS-IS and OSPF in: [RFC5120], [RFC8202], [RFC6549] and [RFC4915].

3.1.1. Prefix-SID Algorithm

Segment Routing supports the use of multiple routing algorithms i.e, different constraint based shortest path calculations can be supported. An algorithm identifier is included as part of a Prefix-SID advertisement. Specification of how an algorithm specific path calculation is done is required in the document defining the algorithm.

This document defines two algorithms:

- "Shortest Path": this algorithm is the default behavior. The packet is forwarded along the well known ECMP-aware SPF algorithm employed by the IGPs. However it is explicitly allowed for a midpoint to implement another forwarding based on local policy. The "Shortest Path" algorithm is in fact the default and current behavior of most of the networks where local policies may override the SPF decision.

- "Strict Shortest Path (Strict-SPF)": This algorithm mandates that the packet is forwarded according to ECMP-aware SPF algorithm and instructs any router in the path to ignore any possible local policy overriding the SPF decision. The SID advertised with Strict-SPF algorithm ensures that the path the packet is going to take is the expected, and not altered, SPF path. Note that Fast Reroute (FRR) [RFC5714] mechanisms are still compliant with the Strict Shortest Path. In other words, a packet received with a Strict-SPF SID may be rerouted through a FRR mechanism. Strict-SPF uses the same topology used by "Shortest Path". Obviously, nodes which do not support Strict-SPF will not install forwarding entries for this algorithm. Restricting the topology only to those nodes which support this algorithm will not produce the desired forwarding paths since the desired behavior is to follow the path calculated by "Shortest Path". Therefore, a source SR node MUST NOT use a source-routing policy containing a strict SPF segment if the path crosses a node not supporting the strict-SPF algorithm.

An IGP-Prefix Segment identifies the path, to the related prefix, computed as per the associated algorithm. A packet injected anywhere within the SR domain with an active Prefix-SID is expected to be forwarded along a path computed using the specified algorithm. For this to be possible, a fully connected topology of routers supporting the specified algorithm is required.

3.1.2. SR-MPLS

When SR is used over the MPLS dataplane SIDs are an MPLS label or an index into an MPLS label space (either SRGB or SRLB).

Where possible, it is recommended that identical SRGBs be configured on all nodes in an SR Domain. This simplifies troubleshooting as the same label will be associated with the same prefix on all nodes. In addition, it simplifies support for anycast as detailed in Section 3.3.

The following behaviors are associated with SR operating over the MPLS dataplane:

- the IGP signaling extension for IGP-Prefix segment includes a flag to indicate whether directly connected neighbors of the node on which the prefix is attached should perform the NEXT operation or the CONTINUE operation when processing the SID. This behavior is equivalent to Penultimate Hop Popping (NEXT) or Ultimate Hop Popping (CONTINUE) in MPLS.

- A Prefix-SID is allocated in the form of an MPLS label (or an index in the SRGB) according to a process similar to IP address allocation. Typically, the Prefix-SID is allocated by policy by the operator (or NMS) and the SID very rarely changes.

- While SR allows to attach a local segment to an IGP prefix, it is specifically assumed that when the terms "IGP-Prefix Segment" and "Prefix-SID" are used, the segment is global (the SID is allocated from the SRGB or as an index into the advertised SRGB). This is consistent with all the described use-cases that require global segments attached to IGP prefixes.

- The allocation process MUST NOT allocate the same Prefix-SID to different IP prefixes.

- If a node learns a Prefix-SID having a value that falls outside the locally configured SRGB range, then the node MUST NOT use the Prefix-SID and SHOULD issue an error log reporting a misconfiguration.

- If a node N advertises Prefix-SID SID-R for a prefix R that is attached to N, if N specifies CONTINUE as the operation to be performed by directly connected neighbors, N MUST maintain the following FIB entry:

```
   Incoming Active Segment: SID-R
   Ingress Operation: NEXT
   Egress interface: NULL
```

- A remote node M MUST maintain the following FIB entry for any learned Prefix-SID SID-R attached to IP prefix R:

```
  Incoming Active Segment: SID-R
  Ingress Operation:
     If the next-hop of R is the originator of R
     and instructed to remove the active segment: NEXT
     Else: CONTINUE
  Egress interface: the interface towards the next-hop along the
                    path computed using the algorithm advertised with
                    the SID toward prefix R.
```

As Prefix-SIDs are specific to a given algorithm, if traffic associated with an algorithm arrives at a node which does not support that algorithm the traffic will be dropped as there will be no forwarding entry matching the incoming label.

3.1.3. SRv6

When SR is used over the IPv6 dataplane:

- A Prefix-SID is an IPv6 address.

- An operator MUST explicitly instantiate an SRv6 SID. IPv6 node addresses are not SRv6 SIDs by default.

A node N advertising an IPv6 address R usable as a segment identifier MUST maintain the following FIB entry:

```
   Incoming Active Segment: R
   Ingress Operation: NEXT
   Egress interface: NULL
```

Note that forwarding to R does not require an entry in the FIBs of all other routers for R. Forwarding can be and most often will be achieved by a shorter mask prefix which covers R.

Independent of Segment Routing support, any remote IPv6 node will maintain a plain IPv6 FIB entry for any prefix, no matter if the prefix represents a segment or not. This allows forwarding of packets to the node which owns the SID even by nodes which do not support Segment Routing.

Support of multiple algorithms applies to SRv6. Since algorithm specific SIDs are simply IPv6 addresses, algorithm specific forwarding entries can be achieved by assigning algorithm specific subnets to the (set of) algorithm specific SIDs which a node allocates.

Nodes which do not support a given algorithm may still have a FIB entry covering an algorithm specific address even though an algorithm specific path has not been calculated by that node. This is mitigated by the fact that nodes which do not support a given algorithm will not be included in the topology associated with that algorithm specific SPF and so traffic using the algorithm specific destination will normally not flow via the excluded node. If such traffic were to arrive and be forwarded by such a node, it will still progress towards the destination node. The nexthop will either be a node which supports the algorithm - in which case the packet will be forwarded along algorithm specific paths (or be dropped if none are available) - or the nexthop will be a node which does NOT support the algorithm - in which case the packet will continue to be forwarded along Algorithm 0 paths towards the destination node.

3.2. IGP-Node Segment, Node-SID

An IGP Node-SID MUST NOT be associated with a prefix that is owned by more than one router within the same routing domain.

3.3. IGP-Anycast Segment, Anycast SID

An "Anycast Segment" or "Anycast SID" enforces the ECMP-aware shortest-path forwarding towards the closest node of the anycast set. This is useful to express macro-engineering policies or protection mechanisms.

An IGP-Anycast segment MUST NOT reference a particular node.

Within an anycast group, all routers in an SR domain MUST advertise the same prefix with the same SID value.

3.3.1. Anycast SID in SR-MPLS

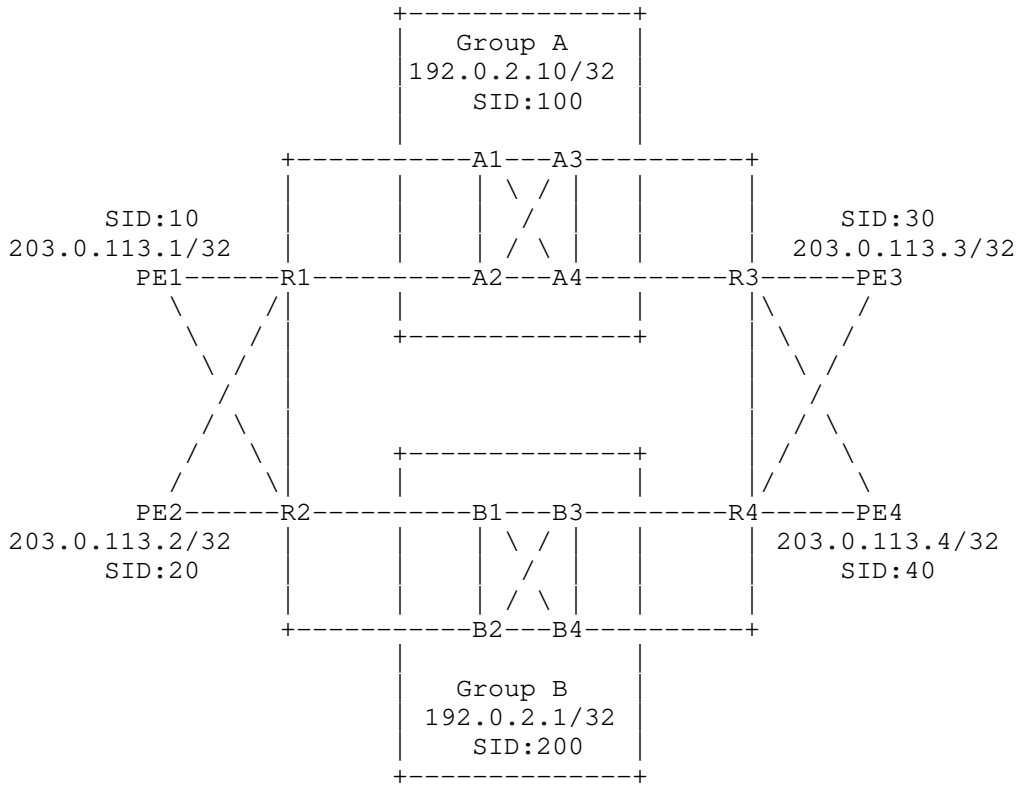


Figure 1: Transit device groups

The figure above describes a network example with two groups of transit devices. Group A consists of devices {A1, A2, A3 and A4}. They are all provisioned with the anycast address 192.0.2.10/32 and the anycast SID 100.

Similarly, group B consists of devices {B1, B2, B3 and B4} and are all provisioned with the anycast address 192.0.2.1/32, anycast SID 200. In the above network topology, each PE device has a path to each of the groups A and B.

PE1 can choose a particular transit device group when sending traffic to PE3 or PE4. This will be done by pushing the anycast SID of the group in the stack.

Processing the anycast, and subsequent segments, requires special care.

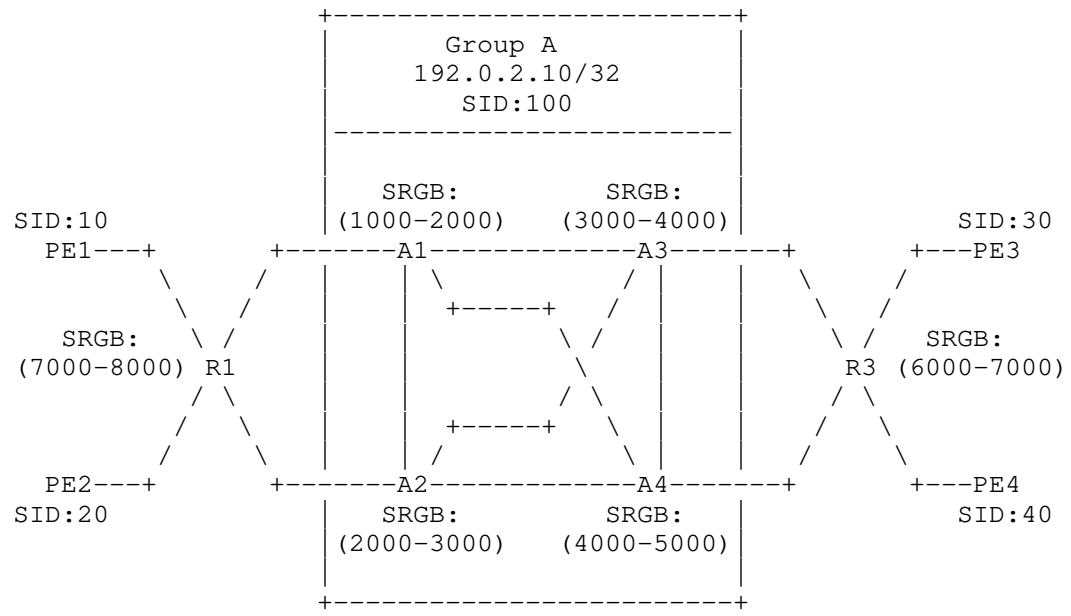


Figure 2: Transit paths via anycast group A

Considering an MPLS deployment, in the above topology, if device PE1 (or PE2) requires to send a packet to the device PE3 (or PE4) it needs to encapsulate the packet in an MPLS payload with the following stack of labels.

- Label allocated by R1 for anycast SID 100 (outer label).
- Label allocated by the nearest router in group A for SID 30 (for destination PE3).

While the first label is easy to compute, in this case since there are more than one topologically nearest devices (A1 and A2), unless A1 and A2 allocated the same label value to the same prefix, determining the second label is impossible. Devices A1 and A2 may be devices from different hardware vendors. If both don’t allocate the same label value for SID 30, it is impossible to use the anycast group "A" as a transit anycast group towards PE3. Hence, PE1 (or PE2) cannot compute an appropriate label stack to steer the packet exclusively through the group A devices. Same holds true for devices PE3 and PE4 when trying to send a packet to PE1 or PE2.

To ease the use of anycast segment, it is recommended to configure identical SRGBs on all nodes of a particular anycast group. Using

this method, as mentioned above, computation of the label following the anycast segment is straightforward.

Using anycast segment without configuring identical SRGBs on all nodes belonging to the same device group may lead to misrouting (in an MPLS VPN deployment, some traffic may leak between VPNs).

3.4. IGP-Adjacency Segment, Adj-SID

The adjacency is formed by the local node (i.e., the node advertising the adjacency in the IGP) and the remote node (i.e., the other end of the adjacency). The local node MUST be an IGP node. The remote node may be an adjacent IGP neighbor or a non-adjacent neighbor (e.g., a Forwarding Adjacency, [RFC4206]).

A packet injected anywhere within the SR domain with a segment list {SN, SNL}, where SN is the Node-SID of node N and SNL is an Adj-SID attached by node N to its adjacency over link L, will be forwarded along the shortest-path to N and then be switched by N, without any IP shortest-path consideration, towards link L. If the Adj-SID identifies a set of adjacencies, then the node N load-balances the traffic among the various members of the set.

Similarly, when using a global Adj-SID, a packet injected anywhere within the SR domain with a segment list {SNL}, where SNL is a global Adj-SID attached by node N to its adjacency over link L, will be forwarded along the shortest-path to N and then be switched by N, without any IP shortest-path consideration, towards link L. If the Adj-SID identifies a set of adjacencies, then the node N does load-balance the traffic among the various members of the set. The use of global Adj-SID allows to reduce the size of the segment list when expressing a path at the cost of additional state (i.e.: the global Adj-SID will be inserted by all routers within the area in their forwarding table).

An "IGP Adjacency Segment" or "Adj-SID" enforces the switching of the packet from a node towards a defined interface or set of interfaces. This is key to theoretically prove that any path can be expressed as a list of segments.

The encodings of the Adj-SID include a set of flags supporting the following functionalities:

- Eligible for Protection (e.g., using IPFRR or MPLS-FRR). Protection allows that in the event the interface(s) associated with the Adj-SID are down, that the packet can still be forwarded via an alternate path. The use of protection is clearly a policy

based decision i.e., for a given policy protection may or may not be desirable.

- Indication whether the Adj-SID has local or global scope. Default scope SHOULD be Local.

- Indication whether the Adj-SID is persistent across control plane restarts. Persistence is a key attribute in ensuring that an SR Policy does not temporarily result in misforwarding due to reassignment of an Adj-SID.

A weight (as described below) is also associated with the Adj-SID advertisement.

A node SHOULD allocate one Adj-SID for each of its adjacencies.

A node MAY allocate multiple Adj-SIDs for the same adjacency. An example is to support an Adj-SID which is eligible for protection and an Adj-SID which is NOT eligible for protection.

A node MAY associate the same Adj-SID to multiple adjacencies.

In order to be able to advertise in the IGP all the Adj-SIDs representing the IGP adjacencies between two nodes, parallel adjacency suppression MUST NOT be performed by the IGP.

When a node binds an Adj-SID to a local data-link L, the node MUST install the following FIB entry:

```
Incoming Active Segment: V
Ingress Operation: NEXT
Egress Interface: L
```

The Adj-SID implies, from the router advertising it, the forwarding of the packet through the adjacency(ies) identified by the Adj-SID, regardless of its IGP/SPF cost. In other words, the use of adjacency segments overrides the routing decision made by the SPF algorithm.

3.4.1. Parallel Adjacencies

Adj-SIDs can be used in order to represent a set of parallel interfaces between two adjacent routers.

A node MUST install a FIB entry for any locally originated adjacency segment (Adj-SID) of value W attached to a set of links B with:

```
   Incoming Active Segment: W
   Ingress Operation: NEXT
   Egress interface: load-balance between any data-link within set B
```

When parallel adjacencies are used and associated to the same Adj-SID, and in order to optimize the load balancing function, a "weight" factor can be associated to the Adj-SID advertised with each adjacency. The weight tells the ingress (or an SDN/orchestration system) about the load-balancing factor over the parallel adjacencies. As shown in Figure 3, A and B are connected through two parallel adjacencies

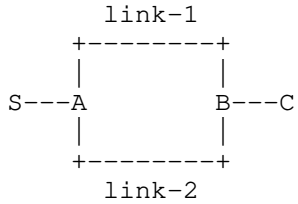


Figure 3: Parallel Links and Adj-SIDs

Node A advertises following Adj-SIDs and weights:

- Link-1: Adj-SID 1000, weight: 1
- Link-2: Adj-SID 1000, weight: 2

Node S receives the advertisements of the parallel adjacencies and understands that by using Adj-SID 1000 node A will load-balance the traffic across the parallel links (link-1 and link-2) according to a 1:2 ratio i.e., twice as many packets will flow over Link-2 as compared to Link-1.

3.4.2. LAN Adjacency Segments

In LAN subnetworks, link-state protocols define the concept of Designated Router (DR, in OSPF) or Designated Intermediate System (DIS, in IS-IS) that conduct flooding in broadcast subnetworks and that describe the LAN topology in a special routing update (OSPF Type2 LSA or IS-IS Pseudonode LSP).

The difficulty with LANs is that each router only advertises its connectivity to the DR/DIS and not to each of the individual nodes in the LAN. Therefore, additional protocol mechanisms (IS-IS and OSPF) are necessary in order for each router in the LAN to advertise an Adj-SID associated to each neighbor in the LAN.

3.5. Inter-Area Considerations

In the following example diagram it is assumed that the all areas are part of a single SR Domain.

The example here below assumes the IPv6 control plane with the MPLS dataplane.

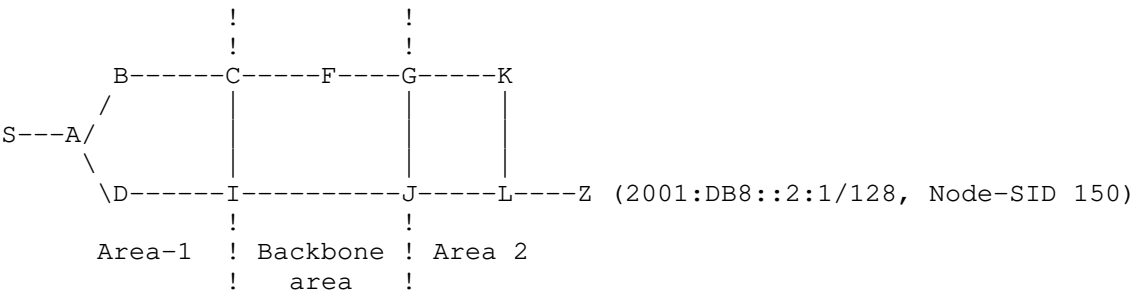


Figure 4: Inter-Area Topology Example

In area 2, node Z allocates Node-SID 150 to his local IPv6 prefix 2001:DB8::2:1/128.

Area Border Routers (ABR) G and J will propagate the prefix and its SIDs into the backbone area by creating a new instance of the prefix according to normal inter-area/level IGP propagation rules.

Nodes C and I will apply the same behavior when leaking prefixes from the backbone area down to area 1. Therefore, node S will see prefix 2001:DB8::2:1/128 with Prefix-SID 150 and advertised by nodes C and I.

It therefore results that a Prefix-SID remains attached to its related IGP Prefix through the inter-area process, which is the expected behavior in a single SR Domain.

When node S sends traffic to 2001:DB8::2:1/128, it pushes Node-SID(150) as active segment and forward it to A.

When packet arrives at ABR I (or C), the ABR forwards the packet according to the active segment (Node-SID(150)). Forwarding continues across area borders, using the same Node-SID(150), until the packet reaches its destination.

4. BGP Peering Segments

BGP segments may be allocated and distributed by BGP.

4.1. BGP Prefix Segment

A BGP-Prefix segment is a BGP segment attached to a BGP prefix.

A BGP-Prefix segment is global (unless explicitly advertised otherwise) within the SR domain.

The BGP Prefix SID is the BGP equivalent to the IGP Prefix Segment.

A likely use-case for the BGP Prefix Segment is an IGP-free hyper-scale spine-leaf topology where connectivity is learned solely via BGP [RFC7938]

4.2. BGP Peering Segments

In the context of BGP Egress Peer Engineering (EPE), as described in [I-D.ietf-spring-segment-routing-central-epe], an EPE enabled Egress PE node MAY advertise segments corresponding to its attached peers. These segments are called BGP peering segments or BGP peering SIDs. They enable the expression of source-routed inter-domain paths.

An ingress border router of an AS may compose a list of segments to steer a flow along a selected path within the AS, towards a selected egress border router C of the AS and through a specific peer. At minimum, a BGP peering Engineering policy applied at an ingress PE involves two segments: the Node SID of the chosen egress PE and then the BGP peering segment for the chosen egress PE peer or peering interface.

Three types of BGP peering segments/SIDs are defined: PeerNode SID, PeerAdj SID and PeerSet SID.

- PeerNode SID: a BGP PeerNode segment/SID is a local segment. At the BGP node advertising it, its semantics is:
  - SR header operation: NEXT.
  - Next-Hop: the connected peering node to which the segment is related.
- PeerAdj SID: a BGP PeerAdj segment/SID is a local segment. At the BGP node advertising it, the semantic is:
  - SR header operation: NEXT.

  * Next-Hop: the peer connected through the interface to which the segment is related.

- PeerSet SID. a BGP PeerSet segment/SID is a local segment. At the BGP node advertising it, the semantic is:

  * SR header operation: NEXT.

  * Next-Hop: load-balance across any connected interface to any peer in the related group.

  A peer set could be all the connected peers from the same AS or a subset of these. A group could also span across AS. The group definition is a policy set by the operator.

The BGP extensions necessary in order to signal these BGP peering segments are defined in [I-D.ietf-idr-bgpls-segment-routing-epe]

5. Binding Segment

In order to provide greater scalability, network opacity, and service independence, SR utilizes a Binding SID (BSID). The BSID is bound to an SR policy, instantiation of which may involve a list of SIDs. Any packets received with active segment = BSID are steered onto the bound SR Policy.

A BSID may either be a local or a global SID. If local, a BSID SHOULD be allocated from the SRLB. If global, a BSID MUST be allocated from the SRGB.

Use of a BSID allows the instantiation of the policy (the SID list) to be stored only on the node(s) which need to impose the policy. Direction of traffic to a node supporting the policy then only requires imposition of the BSID. If the policy changes, this also means that only the nodes imposing the policy need to be updated. Users of the policy are not impacted.

5.1. IGP Mirroring Context Segment

One use case for a Binding Segment is to provide support for an IGP node to advertise its ability to process traffic originally destined to another IGP node, called the Mirrored node and identified by an IP address or a Node-SID, provided that a "Mirroring Context" segment be inserted in the segment list prior to any service segment local to the mirrored node.

When a given node B wants to provide egress node A protection, it advertises a segment identifying node's A context. Such segment is called "Mirror Context Segment" and identified by the Mirror SID.

The Mirror SID is advertised using the binding segment defined in SR IGP protocol extensions [I-D.ietf-isis-segment-routing-extensions] .

In the event of a failure, a point of local repair (PLR) diverting traffic from A to B does a PUSH of the Mirror SID on the protected traffic. B, when receiving the traffic with the Mirror SID as the active segment, uses that segment and processes underlying segments in the context of A.

6. Multicast

Segment Routing is defined for unicast. The application of the source-route concept to Multicast is not in the scope of this document.

7. IANA Considerations

This document does not require any action from IANA.

8. Security Considerations

Segment Routing is applicable to both MPLS and IPv6 data planes.

Segment Routing adds some meta-data (instructions) to the packet, with the list of forwarding path elements (e.g., nodes, links, services, etc.) that the packet must traverse. It has to be noted that the complete source routed path may be represented by a single segment. This is the case of the Binding SID.

SR by default operates within a trusted domain. Traffic MUST be filtered at the domain boundaries.

The use of best practices to reduce the risk of tampering within the trusted domain is important. Such practices are discussed in [RFC4381] and are applicable to both SR-MPLS and SRv6.

8.1. SR-MPLS

When applied to the MPLS data plane, Segment Routing does not introduce any new behavior or any change in the way MPLS data plane works. Therefore, from a security standpoint, this document does not define any additional mechanism in the MPLS data plane.

SR allows the expression of a source routed path using a single segment (the Binding SID). Compared to RSVP-TE which also provides explicit routing capability, there are no fundamental differences in term of information provided. Both RSVP-TE and Segment Routing may express a source routed path using a single segment.

When a path is expressed using a single label, the syntax of the meta-data is equivalent between RSVP-TE [RFC3209] and SR.

When a source routed path is expressed with a list of segments additional meta-data is added to the packet consisting of the source routed path the packet must follow expressed as a segment list.

When a path is expressed using a label stack, if one has access to the meaning (i.e.: the Forwarding Equivalence Class) of the labels, one has the knowledge of the explicit path. For the MPLS data plane, as no data plane modification is required, there is no fundamental change of capability. Yet, the occurrence of label stacking will increase.

SR domain boundary routers MUST filter any external traffic destined to a label associated with a segment within the trusted domain. This includes labels within the SRGB of the trusted domain, labels within the SRLB of the specific boundary router, and labels outside either of these blocks. External traffic is any traffic received from an interface connected to a node outside the domain of trust.

From a network protection standpoint, there is an assumed trust model such that any node imposing a label stack on a packet is assumed to be allowed to do so. This is a significant change compared to plain IP offering shortest path routing but not fundamentally different compared to existing techniques providing explicit routing capability such as RSVP-TE. By default, the explicit routing information MUST NOT be leaked through the boundaries of the administered domain. Segment Routing extensions that have been defined in various protocols, leverage the security mechanisms of these protocols such as encryption, authentication, filtering, etc.

In the general case, a segment routing capable router accepts and install labels only if these labels have been previously advertised by a trusted source. The received information is validated using existing control plane protocols providing authentication and security mechanisms. Segment Routing does not define any additional security mechanism in existing control plane protocols.

Segment Routing does not introduce signaling between the source and the mid points of a source routed path. With SR, the source routed path is computed using SIDs previously advertised in the IP control

plane. Therefore, in addition to filtering and controlled advertisement of SIDs at the boundaries of the SR domain, filtering in the data plane is also required. Filtering MUST be performed on the forwarding plane at the boundaries of the SR domain and may require looking at multiple labels/instruction.

For the MPLS data plane, there are no new requirements as the existing MPLS architecture already allows such source routing by stacking multiple labels. And for security protection, [RFC4381] and [RFC5920] already call for the filtering of MPLS packets on trust boundaries.

8.2. SRv6

When applied to the IPv6 data plane, Segment Routing does introduce the Segment Routing Header (SRH, [I-D.ietf-6man-segment-routing-header]) which is a type of Routing Extension header as defined in [RFC8200].

The SRH adds some meta-data to the IPv6 packet, with the list of forwarding path elements (e.g., nodes, links, services, etc.) that the packet must traverse and that are represented by IPv6 addresses. A complete source routed path may be encoded in the packet using a single segment (single IPv6 address).

SR domain boundary routers MUST filter any external traffic destined to an address within the SRGB of the trusted domain or the SRLB of the specific boundary router. External traffic is any traffic received from an interface connected to a node outside the domain of trust.

From a network protection standpoint, there is an assumed trust model such that any node adding an SRH to the packet is assumed to be allowed to do so. Therefore, by default, the explicit routing information MUST NOT be leaked through the boundaries of the administered domain. Segment Routing extensions that have been defined in various protocols, leverage the security mechanisms of these protocols such as encryption, authentication, filtering, etc.

In the general case, an SR IPv6 router accepts and install segments identifiers (in the form of IPv6 addresses), only if these SIDs are advertised by a trusted source. The received information is validated using existing control plane protocols providing authentication and security mechanisms. Segment Routing does not define any additional security mechanism in existing control plane protocols.

Problems which may arise when the above behaviors are not implemented or when the assumed trust model is violated (e.g., through a security breach) include:

- Malicious looping
- Evasion of access controls
- Hiding the source of DOS attacks

Security concerns with source routing at the IPv6 data plane are more completely discussed in [RFC5095]. The new IPv6-based segment routing header is defined in [I-D.ietf-6man-segment-routing-header]. This document also discusses the above security concerns.

8.3. Congestion Control

SR does not introduce new requirements for congestion control. By default, traffic delivery is assumed to be best effort. Congestion control may be implemented at endpoints. Where SR policies are in use bandwidth allocation may be managed by monitoring incoming traffic associated with the binding SID identifying the SR policy. Other solutions such as [RFC8084] may be applicable.

9. Manageability Considerations

In SR enabled networks, the path the packet takes is encoded in the header. As the path is not signaled through a protocol, OAM mechanisms are necessary in order for the network operator to validate the effectiveness of a path as well as to check and monitor its liveness and performance. However, it has to be noted that SR allows to reduce substantially the number of states in transit nodes and hence the number of elements that a transit node has to manage is smaller.

SR OAM use cases for the MPLS data plane are defined in [I-D.ietf-spring-oam-usecase]. SR OAM procedures for the MPLS data plane are defined in [RFC8287].

SR routers receive advertisements of SIDs (index, label or IPv6 address) from the different routing protocols being extended for SR. Each of these protocols have monitoring and troubleshooting mechanisms to provide operation and management functions for IP addresses that must be extended in order to include troubleshooting and monitoring functions of the SID.

SR architecture introduces the usage of global segments. Each global segment MUST be bound to a unique index or address within an SR

domain. The management of the allocation of such index or address by the operator is critical for the network behavior to avoid situations like mis-routing. In addition to the allocation policy/tooling that the operator will have in place, an implementation SHOULD protect the network in case of conflict detection by providing a deterministic resolution approach.

When a path is expressed using a label stack, the occurrence of label stacking will increase. A node may want to signal in the control plane its ability in terms of size of the label stack it can support.

A YANG data model [RFC6020] for segment routing configuration and operations has been defined in [I-D.ietf-spring-sr-yang].

When Segment Routing is applied to the IPv6 data plane, segments are identified through IPv6 addresses. The allocation, management and troubleshooting of segment identifiers is no different than the existing mechanisms applied to the allocation and management of IPv6 addresses.

The DA of the packet gives the active segment address. The segment list in the SRH gives the entire path of the packet. The validation of the source routed path is done through inspection of DA and SRH present in the packet header matched to the equivalent routing table entries.

In the context of SR over the IPv6 data plane, the source routed path is encoded in the SRH as described in [I-D.ietf-6man-segment-routing-header]. The SR IPv6 source routed path is instantiated into the SRH as a list of IPv6 address where the active segment is in the Destination Address (DA) field of the IPv6 packet header. Typically, by inspecting in any node the packet header, it is possible to derive the source routed path it belongs to. Similar to the context of SR over MPLS data plane, an implementation may originate path control and monitoring packets where the source routed path is inserted in the SRH and where each segment of the path inserts in the packet the relevant data in order to measure the end to end path and performance.

10. Contributors

The following people have substantially contributed to the definition of the Segment Routing architecture and to the editing of this document:

Ahmed Bashandy
Cisco Systems, Inc.
Email: bashandy@cisco.com

Martin Horneffer
Deutsche Telekom
Email: Martin.Horneffer@telekom.de

Wim Henderickx
Nokia
Email: wim.henderickx@nokia.com

Jeff Tantsura
Email: jefftant@gmail.com

Edward Crabbe
Email: edward.crabbe@gmail.com

Igor Milojevic
Email: milojevicigor@gmail.com

Saku Ytti
TDC
Email: saku@ytti.fi

11. Acknowledgements

We would like to thank Dave Ward, Peter Psenak, Dan Frost, Stewart Bryant, Pierre Francois, Thomas Telkamp, Ruediger Geib, Hannes Gredler, Pushpasis Sarkar, Eric Rosen, Chris Bowers and Alvaro Retana for their comments and review of this document.

12. References

12.1. Normative References

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

[RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, DOI 10.17487/RFC3031, January 2001, <https://www.rfc-editor.org/info/rfc3031>.

[RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <https://www.rfc-editor.org/info/rfc8200>.

12.2. Informative References

[I-D.ietf-6man-segment-routing-header]
Previdi, S., Filsfils, C., Raza, K., Dukes, D., Leddy, J., Field, B., daniel.voyer@bell.ca, d., daniel.bernier@bell.ca, d., Matsushima, S., Leung, I., Linkova, J., Aries, E., Kosugi, T., Vyncke, E., Lebrun, D., Steinberg, D., and R. Raszuk, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-08 (work in progress), January 2018.

[I-D.ietf-idr-bgpls-segment-routing-epe]
Previdi, S., Filsfils, C., Patel, K., Ray, S., and J. Dong, "BGP-LS extensions for Segment Routing BGP Egress Peer Engineering", draft-ietf-idr-bgpls-segment-routing-epe-14 (work in progress), December 2017.

[I-D.ietf-isis-segment-routing-extensions]
Previdi, S., Ginsberg, L., Filsfils, C., Bashandy, A., Gredler, H., Litkowski, S., Decraene, B., and J. Tantsura, "IS-IS Extensions for Segment Routing", draft-ietf-isis-segment-routing-extensions-15 (work in progress), December 2017.

[I-D.ietf-ospf-ospfv3-segment-routing-extensions]
Psenak, P., Previdi, S., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPFv3 Extensions for Segment Routing", draft-ietf-ospf-ospfv3-segment-routing-extensions-10 (work in progress), September 2017.

[I-D.ietf-ospf-segment-routing-extensions]
Psenak, P., Previdi, S., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPF Extensions for Segment Routing", draft-ietf-ospf-segment-routing-extensions-24 (work in progress), December 2017.

[I-D.ietf-pce-segment-routing]
Sivabalan, S., Filsfils, C., Tantsura, J., Henderickx, W., and J. Hardwick, "PCEP Extensions for Segment Routing", draft-ietf-pce-segment-routing-11 (work in progress), November 2017.

[I-D.ietf-spring-oam-usecase]
Geib, R., Filsfils, C., Pignataro, C., and N. Kumar, "A Scalable and Topology-Aware MPLS Dataplane Monitoring System", draft-ietf-spring-oam-usecase-10 (work in progress), December 2017.

[I-D.ietf-spring-resiliency-use-cases]
Filsfils, C., Previdi, S., Decraene, B., and R. Shakir, "Resiliency use cases in SPRING networks", draft-ietf-spring-resiliency-use-cases-12 (work in progress), December 2017.

[I-D.ietf-spring-segment-routing-central-epe]
Filsfils, C., Previdi, S., Dawra, G., Aries, E., and D. Afanasiev, "Segment Routing Centralized BGP Egress Peer Engineering", draft-ietf-spring-segment-routing-central-epe-10 (work in progress), December 2017.

[I-D.ietf-spring-segment-routing-mpls]
Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls-11 (work in progress), October 2017.

[I-D.ietf-spring-sr-yang]
Litkowski, S., Qu, Y., Sarkar, P., and J. Tantsura, "YANG Data Model for Segment Routing", draft-ietf-spring-sr-yang-08 (work in progress), December 2017.

[RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, DOI 10.17487/RFC3209, December 2001, <https://www.rfc-editor.org/info/rfc3209>.

[RFC4206] Kompella, K. and Y. Rekhter, "Label Switched Paths (LSP) Hierarchy with Generalized Multi-Protocol Label Switching (GMPLS) Traffic Engineering (TE)", RFC 4206, DOI 10.17487/RFC4206, October 2005, <https://www.rfc-editor.org/info/rfc4206>.

[RFC4381] Behringer, M., "Analysis of the Security of BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4381, DOI 10.17487/RFC4381, February 2006, <https://www.rfc-editor.org/info/rfc4381>.

[RFC4915] Psenak, P., Mirtorabi, S., Roy, A., Nguyen, L., and P. Pillay-Esnault, "Multi-Topology (MT) Routing in OSPF", RFC 4915, DOI 10.17487/RFC4915, June 2007, <https://www.rfc-editor.org/info/rfc4915>.

[RFC5095] Abley, J., Savola, P., and G. Neville-Neil, "Deprecation of Type 0 Routing Headers in IPv6", RFC 5095, DOI 10.17487/RFC5095, December 2007, <https://www.rfc-editor.org/info/rfc5095>.

[RFC5120] Przygienda, T., Shen, N., and N. Sheth, "M-ISIS: Multi Topology (MT) Routing in Intermediate System to Intermediate Systems (IS-ISs)", RFC 5120, DOI 10.17487/RFC5120, February 2008, <https://www.rfc-editor.org/info/rfc5120>.

[RFC5440] Vasseur, JP., Ed. and JL. Le Roux, Ed., "Path Computation Element (PCE) Communication Protocol (PCEP)", RFC 5440, DOI 10.17487/RFC5440, March 2009, <https://www.rfc-editor.org/info/rfc5440>.

[RFC5714] Shand, M. and S. Bryant, "IP Fast Reroute Framework", RFC 5714, DOI 10.17487/RFC5714, January 2010, <https://www.rfc-editor.org/info/rfc5714>.

[RFC5920] Fang, L., Ed., "Security Framework for MPLS and GMPLS Networks", RFC 5920, DOI 10.17487/RFC5920, July 2010, <https://www.rfc-editor.org/info/rfc5920>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/info/rfc6020>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC6549] Lindem, A., Roy, A., and S. Mirtorabi, "OSPFv2 Multi-Instance Extensions", RFC 6549, DOI 10.17487/RFC6549, March 2012, <https://www.rfc-editor.org/info/rfc6549>.

[RFC7938] Lapukhov, P., Premji, A., and J. Mitchell, Ed., "Use of BGP for Routing in Large-Scale Data Centers", RFC 7938, DOI 10.17487/RFC7938, August 2016, <https://www.rfc-editor.org/info/rfc7938>.

[RFC8084] Fairhurst, G., "Network Transport Circuit Breakers", BCP 208, RFC 8084, DOI 10.17487/RFC8084, March 2017, <https://www.rfc-editor.org/info/rfc8084>.

[RFC8202] Ginsberg, L., Previdi, S., and W. Henderickx, "IS-IS Multi-Instance", RFC 8202, DOI 10.17487/RFC8202, June 2017, <https://www.rfc-editor.org/info/rfc8202>.

[RFC8287] Kumar, N., Ed., Pignataro, C., Ed., Swallow, G., Akiya, N., Kini, S., and M. Chen, "Label Switched Path (LSP) Ping/Traceroute for Segment Routing (SR) IGP-Prefix and IGP-Adjacency Segment Identifiers (SIDs) with MPLS Data Planes", RFC 8287, DOI 10.17487/RFC8287, December 2017, <https://www.rfc-editor.org/info/rfc8287>.

Authors' Addresses

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

Email: cfilsfil@cisco.com

Stefano Previdi (editor)
Cisco Systems, Inc.
Italy

Email: stefano@previdi.net

Les Ginsberg
Cisco Systems, Inc

Email: ginsberg@cisco.com

Bruno Decraene
Orange
FR

Email: bruno.decraene@orange.com

Stephane Litkowski
Orange
FR

Email: stephane.litkowski@orange.com

Rob Shakir
Google, Inc.
1600 Amphitheatre Parkway
Mountain View, CA 94043
US

Email: robjs@google.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: June 24, 2018

C. Filsfils, Ed.
S. Previdi
G. Dawra, Ed.
Cisco Systems, Inc.
E. Aries
Juniper Networks
D. Afanasiev
Yandex

December 21, 2017

Segment Routing Centralized BGP Egress Peer Engineering
draft-ietf-spring-segment-routing-central-epe-10

Abstract

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

The Segment Routing architecture can be directly applied to the MPLS dataplane with no change on the forwarding plane. It requires a minor extension to the existing link-state routing protocols.

This document illustrates the application of Segment Routing to solve the BGP Egress Peer Engineering (BGP-EPE) requirement. The SR-based BGP-EPE solution allows a centralized (Software Defined Network, SDN) controller to program any egress peer policy at ingress border routers or at hosts within the domain.

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 https://datatracker.ietf.org/drafts/current/.

Table of Contents

1. Introduction

The document is structured as follows:

- Section 1 states the BGP-EPE problem statement and provides the key references.

- Section 2 defines the different BGP Peering Segments and the semantic associated to them.

- Section 3 describes the automated allocation of BGP Peering Segment-IDs (SIDs) by the BGP-EPE enabled egress border router and the automated signaling of the external peering topology and the related BGP Peering SID's to the collector [I-D.ietf-idr-bgpls-segment-routing-epe].

- Section 4 overviews the components of a centralized BGP-EPE controller. The definition of the BGP-EPE controller is outside the scope of this document.

- Section 5 overviews the methods that could be used by the centralized BGP-EPE controller to implement a BGP-EPE policy at an ingress border router or at a source host within the domain. The exhaustive definition of all the means to program an BGP-EPE input policy is outside the scope of this document.

For editorial reasons, the solution is described with IPv6 addresses and MPLS SIDs. This solution is equally applicable to IPv4 with MPLS SIDs and also to IPv6 with native IPv6 SIDs.

1.1. Problem Statement

The BGP-EPE problem statement is defined in [RFC7855].

A centralized controller should be able to instruct an ingress Provider Edge router (PE) or a content source within the domain to

use a specific egress PE and a specific external interface/neighbor to reach a particular destination.

Let's call this solution "BGP-EPE" for "BGP Egress Peer Engineering". The centralized controller is called the "BGP-EPE Controller". The egress border router where the BGP-EPE traffic steering functionality is implemented is called a BGP-EPE enabled border router. The input policy programmed at an ingress border router or at a source host is called a BGP-EPE policy.

The requirements that have motivated the solution described in this document are listed here below:

- The solution MUST apply to the Internet use-case where the Internet routes are assumed to use IPv4 unlabeled or IPv6 unlabeled. It is not required to place the Internet routes in a VRF and allocate labels on a per route, or on a per-path basis.

- The solution MUST support any deployed iBGP schemes (RRs, confederations or iBGP full meshes).

- The solution MUST be applicable to both routers with external and internal peers.

- The solution should minimize the need for new BGP capabilities at the ingress PEs.

- The solution MUST accommodate an ingress BGP-EPE policy at an ingress PE or directly at a source within the domain.

- The solution MAY support automated Fast Reroute (FRR) and fast convergence mechanisms.

The following reference diagram is used throughout this document.

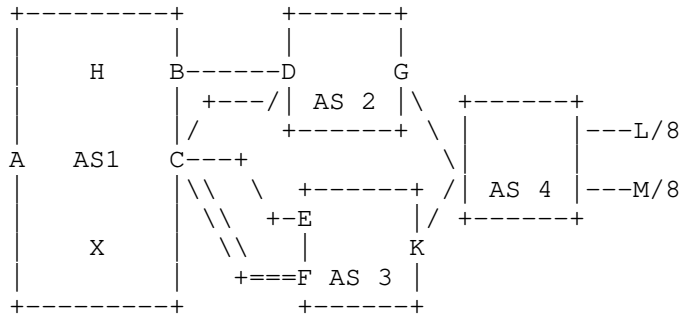


Figure 1: Reference Diagram

IP addressing:

- C's interface to D: 2001:db8:cd::c/64, D's interface: 2001:db8:cd::d/64
- C's interface to E: 2001:db8:ce::c/64, E's interface: 2001:db8:ce::e/64
- C's upper interface to F: 2001:db8:cf1::c/64, F's interface: 2001:db8:cf1::f/64
- C's lower interface to F: 2001:db8:cf2::c/64, F's interface: 2001:db8:cf2::f/64
- BGP router-ID of C: 192.0.2.3
- BGP router-ID of D: 192.0.2.4
- BGP router-ID of E: 192.0.2.5
- BGP router-ID of F: 192.0.2.6
- Loopback of F used for eBGP multi-hop peering to C: 2001:db8:f::f/128
- C's loopback is 2001:db8:c::c/128 with SID 64

C's BGP peering:

- Single-hop eBGP peering with neighbor 2001:db8:cd::d (D)
- Single-hop eBGP peering with neighbor 2001:db8:ce::e (E)
- Multi-hop eBGP peering with F on IP address 2001:db8:f::f (F)

C's resolution of the multi-hop eBGP session to F:

- Static route to 2001:db8:f::f/128 via 2001:db8:cf1::f
- Static route to 2001:db8:f::f/128 via 2001:db8:cf2::f

C is configured with local policy that defines a BGP PeerSet as the set of peers (2001:db8:ce::e for E and 2001:db8:f::f for F)

X is the BGP-EPE controller within AS1 domain.

H is a content source within AS1 domain.

2. BGP Peering Segments

As defined in [I-D.ietf-spring-segment-routing], certain segments are defined by a BGP-EPE capable node and corresponding to its attached peers. These segments are called BGP peering segments or BGP Peering SIDs. They enable the expression of source-routed inter-domain paths.

An ingress border router of an AS may compose a list of segments to steer a flow along a selected path within the AS, towards a selected egress border router C of the AS and through a specific peer. At minimum, a BGP Egress Peering Engineering policy applied at an ingress EPE involves two segments: the Node SID of the chosen egress EPE and then the BGP Peering Segment for the chosen egress EPE peer or peering interface.

[I-D.ietf-spring-segment-routing] defines three types of BGP peering segments/SIDs: PeerNode SID, PeerAdj SID and PeerSet SID.

A Peer Node Segment is a segment describing a peer, including the SID (PeerNode SID) allocated to it.

A Peer Adjacency Segment is a segment describing a link, including the SID (PeerAdj SID) allocated to it.

A Peer Set Segment is a segment describing a link or a node that is part of the set, including the SID (PeerSet SID) allocated to the set.

3. Distribution of Topology and TE Information using BGP-LS

In ships-in-the-night mode with respect to the pre-existing iBGP design, a BGP-LS [RFC7752] session is established between the BGP-EPE enabled border router and the BGP-EPE controller.

As a result of its local configuration and according to the behavior described in [I-D.ietf-idr-bgpls-segment-routing-epe], node C allocates the following BGP Peering Segments ([I-D.ietf-spring-segment-routing]):

- A PeerNode segment for each of its defined peer (D: 1012, E: 1022 and F: 1052).
- A PeerAdj segment for each recursing interface to a multi-hop peer (e.g.: the upper and lower interfaces from C to F in figure 1).

- A PeerSet segment to the set of peers (E and F). In this case the PeerSet represents a set of peers (E, F) belonging to the same AS (AS 3).

C programs its forwarding table accordingly:

Incoming Label	Operation	Outgoing Interface
1012	POP	link to D
1022	POP	link to E
1032	POP	upper link to F
1042	POP	lower link to F
1052	POP	load balance on any link to F
1060	POP	load balance on any link to E or to F

C signals the related BGP-LS NLRI's to the BGP-EPE controller. Each such BGP-LS route is described in the following subsections according to the encoding details defined in [I-D.ietf-idr-bgpls-segment-routing-epe].

3.1. PeerNode SID to D

Descriptors:

- Local Node Descriptors (BGP router-ID, ASN, BGP-LS Identifier): 192.0.2.3, AS1, 1000
- Remote Node Descriptors (BGP router-ID, ASN): 192.0.2.4, AS2
- Link Descriptors (IPv6 Interface Address, IPv6 Neighbor Address): 2001:db8:cd::c, 2001:db8:cd::d

Attributes:

- PeerNode SID: 1012

3.2. PeerNode SID to E

Descriptors:

- Local Node Descriptors (BGP router-ID, ASN, BGP-LS Identifier)): 192.0.2.3, AS1, 1000
- Remote Node Descriptors (BGP router-ID, ASN): 192.0.2.5, AS3
- Link Descriptors (IPv6 Interface Address, IPv6 Neighbor Address): 2001:db8:ce::c, 2001:db8:ce::e

Attributes:

- PeerNode SID: 1022
- PeerSetSID: 1060
- Link Attributes: see section 3.3.2 of [RFC7752]

3.3. PeerNode SID to F

Descriptors:

- Local Node Descriptors (BGP router-ID, ASN, BGP-LS Identifier)): 192.0.2.3, AS1, 1000
- Remote Node Descriptors (BGP router-ID, ASN): 192.0.2.6, AS3
- Link Descriptors (IPv6 Interface Address, IPv6 Neighbor Address): 2001:db8:c::c, 2001:db8:f::f

Attributes:

- PeerNode SID: 1052
- PeerSetSID: 1060

3.4. First PeerAdj to F

Descriptors:

- Local Node Descriptors (BGP router-ID, ASN, BGP-LS Identifier)): 192.0.2.3, AS1, 1000
- Remote Node Descriptors (BGP router-ID, ASN): 192.0.2.6, AS3
- Link Descriptors (IPv6 Interface Address, IPv6 Neighbor Address): 2001:db8:cf1::c, 2001:db8:cf1::f

Attributes:

- PeerAdj-SID: 1032
- LinkAttributes: see section 3.3.2 of [RFC7752]

3.5. Second PeerAdj to F

Descriptors:

- Local Node Descriptors (BGP router-ID, ASN, BGP-LS Identifier)): 192.0.2.3 , AS1
- Remote Node Descriptors (peer router-ID, peer ASN): 192.0.2.6, AS3
- Link Descriptors (IPv6 Interface Address, IPv6 Neighbor Address): 2001:db8:cf2::c, 2001:db8:cf2::f

Attributes:

- PeerAdj-SID: 1042
- LinkAttributes: see section 3.3.2 of [RFC7752]

3.6. Fast Reroute (FRR)

A BGP-EPE enabled border router MAY allocate a FRR backup entry on a per BGP Peering SID basis. One example is as follows:

- PeerNode SID
  1. If multi-hop, backup via the remaining PeerADJ SIDs (if available) to the same peer.
  2. Else backup via another PeerNode SID to the same AS.
  3. Else pop the PeerNode SID and perform an IP lookup.
- PeerAdj SID
  1. If to a multi-hop peer, backup via the remaining PeerADJ SIDs (if available) to the same peer.
  2. Else backup via a PeerNode SID to the same AS.
  3. Else pop the PeerNode SID and perform an IP lookup.
- PeerSet SID
  1. Backup via remaining PeerNode SIDs in the same PeerSet.
  2. Else pop the PeerNode SID and IP lookup.

Let's illustrate different types of possible backups using the reference diagram and considering the Peering SIDs allocated by C.

PeerNode SID 1052, allocated by C for peer F:

- Upon the failure of the upper connected link CF, C can reroute all the traffic onto the lower CF link to the same peer (F).

PeerNode SID 1022, allocated by C for peer E:

- Upon the failure of the connected link CE, C can reroute all the traffic onto the link to PeerNode SID 1052 (F).

PeerNode SID 1012, allocated by C for peer D:

- Upon the failure of the connected link CD, C can pop the PeerNode SID and lookup the IP destination address in its FIB and route accordingly.

PeerSet SID 1060, allocated by C for the set of peers E and F:

- Upon the failure of a connected link in the group, the traffic to PeerSet SID 1060 is rerouted on any other member of the group.

For specific business reasons, the operator might not want the default FRR behavior applied to a PeerNode SID or any of its dependent PeerADJ SID.

The operator should be able to associate a specific backup PeerNode SID for a PeerNode SID: e.g., 1022 (E) must be backed up by 1012 (D) which overrules the default behavior which would have preferred F as a backup for E.

4. BGP-EPE Controller

In this section, Let's provide a non-exhaustive set of inputs that a BGP-EPE controller would likely collect such as to perform the BGP-EPE policy decision.

The exhaustive definition is outside the scope of this document.

4.1. Valid Paths From Peers

The BGP-EPE controller should collect all the BGP paths (i.e.: IP destination prefixes) advertised by all the BGP-EPE enabled border router.

This could be realized by setting an iBGP session with the BGP-EPE enabled border router, with the router configured to advertise all paths using BGP add-path [RFC7911] and the original next-hop preserved.

In this case, C would advertise the following Internet routes to the BGP-EPE controller:

- NLRI <2001:db8:abcd::/48>, next-hop 2001:db8:cd::d, AS Path {AS 2, 4}
  - X (i.e.: the BGP-EPE controller) knows that C receives a path to 2001:db8:abcd::/48 via neighbor 2001:db8:cd::d of AS2.
- NLRI <2001:db8:abcd::/48>, next-hop 2001:db8:ce::e, AS Path {AS 3, 4}
  - X knows that C receives a path to 2001:db8:abcd::/48 via neighbor 2001:db8:ce::e of AS2.
- NLRI <2001:db8:abcd::/48>, next-hop 2001:db8:f::f, AS Path {AS 3, 4}
  - X knows that C has an eBGP path to 2001:db8:abcd::/48 via AS3 via neighbor 2001:db8:f::f

An alternative option would be for a BGP-EPE collector to use BGP Monitoring Protocol (BMP) [RFC7854] to track the Adj-RIB-In of BGP-EPE enabled border routers.

4.2. Intra-Domain Topology

The BGP-EPE controller should collect the internal topology and the related IGP SIDs.

This could be realized by collecting the IGP LSDB of each area or running a BGP-LS session with a node in each IGP area.

4.3. External Topology

Thanks to the collected BGP-LS routes described in section 2, the BGP-EPE controller is able to maintain an accurate description of the egress topology of node C. Furthermore, the BGP-EPE controller is able to associate BGP Peering SIDs to the various components of the external topology.

4.4. SLA characteristics of each peer

The BGP-EPE controller might collect SLA characteristics across peers. This requires an BGP-EPE solution as the SLA probes need to be steered via non-best-path peers.

Unidirectional SLA monitoring of the desired path is likely required. This might be possible when the application is controlled at the source and the receiver side. Unidirectional monitoring dissociates the SLA characteristic of the return path (which cannot usually be controlled) from the forward path (the one of interest for pushing content from a source to a consumer and the one which can be controlled).

Alternatively, Extended Metrics, as defined in [RFC7810] could also be advertised using BGP-LS ([I-D.ietf-idr-te-pm-bgp]).

4.5. Traffic Matrix

The BGP-EPE controller might collect the traffic matrix to its peers or the final destinations. IPFIX [RFC7011] is a likely option.

An alternative option consists in collecting the link utilization statistics of each of the internal and external links, also available in the current definition of [RFC7752].

4.6. Business Policies

The BGP-EPE controller should be configured or collect business policies through any desired mechanisms. These mechanisms by which these policies are configured or collected are outside the scope of this document.

4.7. BGP-EPE Policy

On the basis of all these inputs (and likely others), the BGP-EPE Controller decides to steer some demands away from their best BGP path.

The BGP-EPE policy is likely expressed as a two-entry segment list where the first element is the IGP prefix SID of the selected egress border router and the second element is a BGP Peering SID at the selected egress border router.

A few examples are provided hereafter:

- Prefer egress PE C and peer AS AS2: {64, 1012}. "64" being the SID of PE C as defined in Section 1.1.

- Prefer egress PE C and peer AS AS3 via eBGP peer 2001:db8:ce::e, {64, 1022}.

- Prefer egress PE C and peer AS AS3 via eBGP peer 2001:db8:f::f, {64, 1052}.

- Prefer egress PE C and peer AS AS3 via interface 2001:db8:cf2::f of multi-hop eBGP peer 2001:db8:f::f, {64, 1042}.

- Prefer egress PE C and any interface to any peer in the group 1060: {64, 1060}.

Note that the first SID could be replaced by a list of segments. This is useful when an explicit path within the domain is required for traffic engineering purposes. For example, if the Prefix SID of node B is 60 and the BGP-EPE controller would like to steer the traffic from A to C via B then through the external link to peer D then the segment list would be {60, 64, 1012}.

5. Programming an input policy

The detailed/exhaustive description of all the means to implement an BGP-EPE policy are outside the scope of this document. A few examples are provided in this section.

5.1. At a Host

A static IP/MPLS route can be programmed at the host H. The static route would define a destination prefix, a next-hop and a label stack to push. Assuming a global SRGB, at least on all access routers connecting the hosts, the same policy can be programmed across all hosts, which is convenient.

5.2. At a router - SR Traffic Engineering tunnel

The BGP-EPE controller can configure the ingress border router with an SR traffic engineering tunnel T1 and a steering-policy S1 which causes a certain class of traffic to be mapped on the tunnel T1.

The tunnel T1 would be configured to push the required segment list.

The tunnel and the steering policy could be configured via multiple means. A few examples are given below:

- PCEP according to [I-D.ietf-pce-segment-routing] and [I-D.ietf-pce-pce-initiated-lsp].

- Netconf ([RFC6241]).

- Other static or ephemeral APIs

Example: at router A (Figure 1).

```
Tunnel T1: push {64, 1042}
IP route L/8 set next-hop T1
```

5.3. At a Router - BGP Labeled Unicast route (RFC8277)

The BGP-EPE Controller could build a BGP Labeled Unicast route [RFC8277]) route (from scratch) and send it to the ingress router:

- NLRI: the destination prefix to engineer: e.g., L/8.
- Next-Hop: the selected egress border router: C.
- Label: the selected egress peer: 1042.
- AS path: reflecting the selected valid AS path.
- Some BGP policy to ensure it will be selected as best by the ingress router. Note that as discussed in RFC 8277 section 5, the comparison of labeled and unlabeled unicast BGP route is implementation dependent and hence may require an implementation specific policy on each ingress router.

This BGP Labeled unicast route (RFC8277) "overwrites" an equivalent or less-specific "best path". As the best-path is changed, this BGP-EPE input policy option may influence the path propagated to the upstream peer/customers. Indeed, implementations treating the SAFI-1 and SAFI-4 routes for a given prefix as comparable would trigger a BGP WITHDRAW of the SAFI-1 route to their BGP upstream peers.

5.4. At a Router - VPN policy route

The BGP-EPE Controller could build a VPNv4 route (from scratch) and send it to the ingress router:

- NLRI: the destination prefix to engineer: e.g., L/8.
- Next-Hop: the selected egress border router: C.
- Label: the selected egress peer: 1042.
- Route-Target: selecting the appropriate VRF at the ingress router.
- AS path: reflecting the selected valid AS path.

- Some BGP policy to ensure it will be selected as best by the ingress router in the related VRF.

The related VRF must be preconfigured. A VRF fallback to the main FIB might be beneficial to avoid replicating all the "normal" Internet paths in each VRF.

6. IPv6 Dataplane

The described solution is applicable to IPv6, either with MPLS-based or IPv6-Native segments. In both cases, the same three steps of the solution are applicable:

- BGP-LS-based signaling of the external topology and BGP Peering Segments to the BGP-EPE controller.

- Collection of various inputs by the BGP-EPE controller to come up with a policy decision.

- Programming at an ingress router or source host of the desired BGP-EPE policy which consists in a list of segments to push on a defined traffic class.

7. Benefits

The BGP-EPE solutions described in this document have the following benefits:

- No assumption on the iBGP design within AS1.

- Next-Hop-Self on the Internet routes propagated to the ingress border routers is possible. This is a common design rule to minimize the number of IGP routes and to avoid importing external churn into the internal routing domain.

- Consistent support for traffic engineering within the domain and at the external edge of the domain.

- Support both host and ingress border router BGP-EPE policy programming.

- BGP-EPE functionality is only required on the BGP-EPE enabled egress border router and the BGP-EPE controller: an ingress policy can be programmed at the ingress border router without any new functionality.

- Ability to deploy the same input policy across hosts connected to different routers (assuming the global property of IGP prefix SIDs).

8. IANA Considerations

This document does not request any IANA allocations.

9. Manageability Considerations

The BGP-EPE use-case described in this document requires BGP-LS ([RFC7752]) extensions that are described in [I-D.ietf-idr-bgpls-segment-routing-epe]. The required extensions consists of additional BGP-LS descriptors and TLVs that will follow the same. Manageability functions of BGP-LS, described in [RFC7752] also apply to the extensions required by the EPE use-case.

Additional Manageability considerations are described in [I-D.ietf-idr-bgpls-segment-routing-epe].

10. Security Considerations

[RFC7752] defines BGP-LS NLRIs and their associated security aspects.

[I-D.ietf-idr-bgpls-segment-routing-epe] defines the BGP-LS extensions required by the BGP-EPE mechanisms described in this document. BGP-EPE BGP-LS extensions also include the related security.

11. Contributors

Daniel Ginsburg substantially contributed to the content of this document.

12. Acknowledgements

The authors would like to thank Acee Lindem for his comments and contribution.

13. References

13.1. Normative References

[I-D.ietf-idr-bgpls-segment-routing-epe]
Previdi, S., Filsfils, C., Patel, K., Ray, S., and J. Dong, "BGP-LS extensions for Segment Routing BGP Egress Peer Engineering", draft-ietf-idr-bgpls-segment-routing-epe-14 (work in progress), December 2017.

[I-D.ietf-spring-segment-routing]
Filsfils, C., Previdi, S., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", draft-ietf-spring-segment-routing-14 (work in progress), December 2017.

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

[RFC7752] Gredler, H., Ed., Medved, J., Previdi, S., Farrel, A., and S. Ray, "North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP", RFC 7752, DOI 10.17487/RFC7752, March 2016, <https://www.rfc-editor.org/info/rfc7752>.

13.2. Informative References

[I-D.ietf-idr-te-pm-bgp]
Ginsberg, L., Previdi, S., Wu, Q., Gredler, H., Ray, S., Tantsura, J., and C. Filsfils, "BGP-LS Advertisement of IGP Traffic Engineering Performance Metric Extensions", draft-ietf-idr-te-pm-bgp-08 (work in progress), August 2017.

[I-D.ietf-pce-pce-initiated-lsp]
Crabbe, E., Minei, I., Sivabalan, S., and R. Varga, "PCEP Extensions for PCE-initiated LSP Setup in a Stateful PCE Model", draft-ietf-pce-pce-initiated-lsp-11 (work in progress), October 2017.

[I-D.ietf-pce-segment-routing]
Sivabalan, S., Filsfils, C., Tantsura, J., Henderickx, W., and J. Hardwick, "PCEP Extensions for Segment Routing", draft-ietf-pce-segment-routing-11 (work in progress), November 2017.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC7011] Claise, B., Ed., Trammell, B., Ed., and P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information", STD 77, RFC 7011, DOI 10.17487/RFC7011, September 2013, <https://www.rfc-editor.org/info/rfc7011>.

[RFC7810] Previdi, S., Ed., Giacalone, S., Ward, D., Drake, J., and Q. Wu, "IS-IS Traffic Engineering (TE) Metric Extensions", RFC 7810, DOI 10.17487/RFC7810, May 2016, <https://www.rfc-editor.org/info/rfc7810>.

[RFC7854] Scudder, J., Ed., Fernando, R., and S. Stuart, "BGP Monitoring Protocol (BMP)", RFC 7854, DOI 10.17487/RFC7854, June 2016, <https://www.rfc-editor.org/info/rfc7854>.

[RFC7855] Previdi, S., Ed., Filsfils, C., Ed., Decraene, B., Litkowski, S., Horneffer, M., and R. Shakir, "Source Packet Routing in Networking (SPRING) Problem Statement and Requirements", RFC 7855, DOI 10.17487/RFC7855, May 2016, <https://www.rfc-editor.org/info/rfc7855>.

[RFC7911] Walton, D., Retana, A., Chen, E., and J. Scudder, "Advertisement of Multiple Paths in BGP", RFC 7911, DOI 10.17487/RFC7911, July 2016, <https://www.rfc-editor.org/info/rfc7911>.

[RFC8277] Rosen, E., "Using BGP to Bind MPLS Labels to Address Prefixes", RFC 8277, DOI 10.17487/RFC8277, October 2017, <https://www.rfc-editor.org/info/rfc8277>.

Authors' Addresses

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

Email: cfilsfil@cisco.com

Stefano Previdi
Cisco Systems, Inc.
Italy

Email: stefano@previdi.net

Gaurav Dawra (editor)
Cisco Systems, Inc.
USA

Email: gdawra.ietf@gmail.com

Ebben Aries
Juniper Networks
1133 Innovation Way
Sunnyvale CA 94089
US

Email: exa@juniper.net

Dmitry Afanasiev
Yandex
RU

Email: fl0w@yandex-team.ru

Network Working Group
Internet Draft
Intended status: Standards Track
Expires: June 2019

A. Bashandy, Ed.
Arrcus
C. Filsfils, Ed.
S. Previdi,
Cisco Systems, Inc.
B. Decraene
S. Litkowski
Orange
R. Shakir
Google

December 9, 2018

Segment Routing with MPLS data plane
draft-ietf-spring-segment-routing-mpls-18

Abstract

Segment Routing (SR) leverages the source routing paradigm. A node steers a packet through a controlled set of instructions, called segments, by prepending the packet with an SR header. In the MPLS dataplane, the SR header is instantiated through a label stack. This document specifies the forwarding behavior to allow instantiating SR over the MPLS dataplane.

Table of Contents

1. Introduction

The Segment Routing architecture RFC8402 can be directly applied to the MPLS architecture with no change in the MPLS forwarding plane. This document specifies the forwarding plane behavior to allow Segment Routing to operate on top of the MPLS data plane. This document does not address the control plane behavior. Control plane behavior is specified in other documents such as [I-D.ietf-isis-segment-routing-extensions], [I-D.ietf-ospf-segment-routing-extensions], and [I-D.ietf-ospf-ospfv3-segment-routing-extensions].

The Segment Routing problem statement is described in [RFC7855].

Co-existence of SR over MPLS forwarding plane with LDP [RFC5036] is specified in [I-D.ietf-spring-segment-routing-ldp-interop].

Policy routing and traffic engineering using segment routing can be found in [I-D.ietf-spring-segment-routing-policy]

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. MPLS Instantiation of Segment Routing

MPLS instantiation of Segment Routing fits in the MPLS architecture as defined in [RFC3031] both from a control plane and forwarding plane perspective:

- From a control plane perspective, [RFC3031] does not mandate a single signaling protocol. Segment Routing makes use of various control plane protocols such as link state IGPs [I-D.ietf-isis-segment-routing-extensions], [I-D.ietf-ospf-segment-routing-extensions] and [I-D.ietf-ospf-ospfv3-segment-routing-extensions]. The flooding mechanisms of link state IGPs fits very well with label stacking on ingress. Future control layer protocol and/or policy/configuration can be used to specify the label stack.

- From a forwarding plane perspective, Segment Routing does not require any change to the forwarding plane because Segment IDs (SIDs) are instantiated as MPLS labels and the Segment routing header instantiated as a stack of MPLS labels.

We call "MPLS Control Plane Client (MCC)" any control plane entity installing forwarding entries in the MPLS data plane. IGPs with SR extensions [I-D.ietf-isis-segment-routing-extensions], [I-D.ietf-ospf-segment-routing-extensions], [I-D.ietf-ospf-ospfv3-segment-routing-extensions] and LDP [RFC5036] are examples of MCCs. Local configuration and policies applied on a router are also examples of MCCs.

In order to have a node segment to reach the node, a network operator SHOULD configure at least one node segment per routing instance, topology, algorithm. Otherwise, the node is not reachable within the routing instance, topology or along the routing algorithm, which restrict its ability to be used by a SR policy, including for TI-LFA. An implementation MAY check that an IGP node-SID is not associated with a prefix that is owned by more than one router within the same

routing domain. If so, it SHOULD NOT use this Node-SID, MAY use another one if available, and SHOULD log an error.

2.1. Multiple Forwarding Behaviors for the Same Prefix

The SR architecture does not prohibit having more than one SID for the same prefix. In fact, by allowing multiple SIDs for the same prefix, it is possible to have different forwarding behaviors (such as different paths, different ECMP/UCMP behaviors,...,etc) for the same destination.

Instantiating Segment routing over the MPLS forwarding plane fits seamlessly with this principle. An operator may assign multiple MPLS labels or indices to the same prefix and assign different forwarding behaviors to each label/SID. The MCC in the network downloads different MPLS labels/SIDs to the FIB for different forwarding behaviors. The MCC at the entry of an SR domain or at any point in the domain can choose to apply a particular forwarding behavior to a particular packet by applying the PUSH action to that packet using the corresponding SID.

2.2. SID Representation in the MPLS Forwarding Plane

When instantiating SR over the MPLS forwarding plane, a SID is represented by an MPLS label or an index [RFC8402].

A global segment MUST be a label, or an index which may be mapped to an MPLS label within the Segment Routing Global Block (SRGB) of the node installing the global segment in its FIB/receiving the labeled packet. Section 2.4 specifies the procedure to map a global segment represented by an index to an MPLS label within the SRGB.

The MCC MUST ensure that any label value corresponding to any SID it installs in the forwarding plane follows the following rules:

- The label value MUST be unique within the router on which the MCC is running. i.e. the label MUST only be used to represent the SID and MUST NOT be used to represent more than one SID or for any other forwarding purpose on the router.

- The label value MUST NOT come from the range of special purpose labels [RFC7274].

Labels allocated in this document are considered per platform downstream allocated labels [RFC3031].

2.3. Segment Routing Global Block and Local Block

The concepts of Segment Routing Global Block (SRGB) and global SID are explained in [RFC8402]. In general, the SRGB need not be a contiguous range of labels.

For the rest of this document, the SRGB is specified by the list of MPLS Label ranges [Ll(1),Lh(1)], [Ll(2),Lh(2)],..., [Ll(k),Lh(k)] where Ll(i) =< Lh(i).

The following rules apply to the list of MPLS ranges representing the SRGB

- The list of ranges comprising the SRGB MUST NOT overlap.
- Every range in the list of ranges specifying the SRGB MUST NOT cover or overlap with a reserved label value or range [RFC7274], respectively.
- If the SRGB of a node does not conform to the structure specified in this section or to the previous two rules, then this SRGB MUST be completely ignored by all routers in the routing domain and the node MUST be treated as if it does not have an SRGB.
- The list of label ranges MUST only be used to instantiate global SIDs into the MPLS forwarding plane

A Local segment MAY be allocated from the Segment Routing Local Block (SRLB) [RFC8402] or from any unused label as long as it does not use a special purpose label. The SRLB consists of the range of local labels reserved by the node for certain local segments. In a controller-driven network, some controllers or applications MAY use the control plane to discover the available set of local SIDs on a particular router [I-D.ietf-spring-segment-routing-policy]. The rules applicable to the SRGB are also applicable to the SRLB, except rule that says that the SRGB MUST only be used to instantiate global SIDs into the MPLS forwarding plane. The recommended, minimum, or maximum size of the SRGB and/or SRLB is a matter of future study

2.4. Mapping a SID Index to an MPLS label

This sub-section specifies how the MPLS label value is calculated given the index of a SID. The value of the index is determined by an MCC such as IS-IS [I-D.ietf-isis-segment-routing-extensions] or OSPF [I-D.ietf-ospf-segment-routing-extensions]. This section only specifies how to map the index to an MPLS label. The calculated MPLS

label is downloaded to the FIB, sent out with a forwarded packet, or both.

Consider a SID represented by the index "I". Consider an SRGB as specified in Section 2.3. The total size of the SRGB, represented by the variable "Size", is calculated according to the formula:

size = Lh(1)- Ll(1) + 1 + Lh(2)- Ll(2) + 1 + ... + Lh(k)- Ll(k) + 1

The following rules MUST be applied by the MCC when calculating the MPLS label value corresponding the SID index value "I".

- 0 =< I < size. If the index "I" does not satisfy the previous inequality, then the label cannot be calculated.
- The label value corresponding to the SID index "I" is calculated as follows
  - j = 1 , temp = 0
  - While temp + Lh(j)- Ll(j) < I
    - temp = temp + Lh(j)- Ll(j) + 1
    - j = j+1
  - label = I - temp + Ll(j)

An example for how a router calculates labels and forwards traffic based on the procedure described in this section can be found in Appendix A.1.

2.5. Incoming Label Collision

MPLS Architecture [RFC3031] defines Forwarding Equivalence Class (FEC) term as the set of packets with similar and / or identical characteristics which are forwarded the same way and are bound to the same MPLS incoming (local) label. In Segment-Routing MPLS, local label serves as the SID for given FEC.

We define Segment Routing (SR) FEC as one of the following [RFC8402]:

- (Prefix, Routing Instance, Topology, Algorithm [RFC8402]), where a topology identifies a set of links with metrics. For the purpose of incoming label collision resolution, the same Topology numerical value SHOULD be used on all routers to identify the same set of links with metrics. For MCCs where the "Topology" and/or "Algorithm" fields are not defined, the numerical value of zero MUST be used for these two fields. For the purpose of incoming label collision resolution, a routing instance is identified by a single incoming label downloader to FIB. Two MCCs running on the same router are considered different routing instances if the only way the two instances can know about the other's incoming labels is through redistribution. The numerical value used to identify a routing instance MAY be derived from other configuration or MAY be explicitly configured. If it is derived from other configuration, then the same numerical value SHOULD be derived from the same configuration as long as the configuration survives router reload. If the derived numerical value varies for the same configuration, then an implementation SHOULD make numerical value used to identify a routing instance configurable.

- (next-hop, outgoing interface), where the outgoing interface is physical or virtual.

- (number of adjacencies, list of next-hops, list of outgoing interfaces IDs in ascending numerical order). This FEC represents parallel adjacencies [RFC8402]

- (Endpoint, Color) representing an SR policy [RFC8042]

- (Mirrored SID) The Mirrored SID [RFC8042, Section 5.1] is the IP address advertised by the advertising node to identify the mirror-SID. The IP address is encoded as specified in Section 2.5.1.

This section covers the RECOMMENDED procedure to handle the scenario where, because of an error/misconfiguration, more than one SR FEC as defined in this section, map to the same incoming MPLS label. Examples illustrating the behavior specified in this section can be found in Appendix A.2.

An incoming label collision occurs if the SIDs of the set of FECs {FEC1, FEC2,..., FECk} maps to the same incoming SR MPLS label "L1".

Suppose an anycast prefix is advertised with a prefix-SID by some, but not all, of the nodes that advertise that prefix. If the prefix-SID subTLVs result in mapping that anycast prefix to the same incoming label, then the advertisement of the prefix-SID by some, but

not all, of advertising nodes SHOULD NOT be treated as a label collision.

An implementation MUST NOT allow the MCCs belonging to the same router to assign the same incoming label to more than one SR FEC. An implementation that allows such behavior is considered as faulty. Procedures defined in this document equally applies to this case, both for incoming label collision (Section 2.5) and the effect on outgoing label programming (Section 2.6).

The objective of the following steps is to deterministically install in the MPLS Incoming Label Map, also known as label FIB, a single FEC with the incoming label "L1". Remaining FECs may be installed in the IP FIB without incoming label.

The procedure in this section relies completely on the local FEC and label database within a given router.

The collision resolution procedure is as follows

1. Given the SIDs of the set of FECs, {FEC1, FEC2,..., FECk} map to the same MPLS label "L1".

2. Within an MCC, apply tie-breaking rules to select one FEC only and assign the label to it. The losing FECs are handled as if no labels are attached to them. The losing FECs with a non-zero algorithm are not installed in FIB.

    a. If the same set of FECs are attached to the same label "L1", then the tie-breaking rules MUST always select the same FEC irrespective of the order in which the FECs and the label "L1" are received. In other words, the tie-breaking rule MUST be deterministic. For example, a first-come-first-serve tie-breaking is not allowed.

3. If there is still collision between the FECs belonging to different MCCs, then re-apply the tie-breaking rules to the remaining FECs to select one FEC only and assign the label to that FEC

4. Install into the IP FIB the selected FEC and its incoming label in the label FIB.

5. The remaining FECs with the default algorithm (see the specification of prefix-SID algorithm [RFC8402]) are installed in the FIB natively, such as pure IP entries in case of Prefix FEC, without any incoming labels corresponding to their SIDs. The remaining FECs with a non-zero algorithm are not installed in the FIB.

2.5.1. Tie-breaking Rules

The default tie-breaking rules SHOULD be as follows:

1. if FECi has the lowest FEC administrative distance among the competing FECs as defined in this section below, filter away all the competing FECs with higher administrative distance.

2. if more than one competing FEC remains after step 1, select the smallest numerical FEC value

These rules deterministically select the FEC to install in the MPLS forwarding plane for the given incoming label.

This document defines the default tie breaking rules that SHOULD be implemented. An implementation MAY choose to implement additional tie-breaking rules. All routers in a routing domain SHOULD use the same tie-breaking rules to maximize forwarding consistency.

Each FEC is assigned an administrative distance. The FEC administrative distance is encoded as an 8-bit value. The lower the value, the better the administrative distance.

The default FEC administrative distance order starting from the lowest value SHOULD be

- Explicit SID assignment to a FEC that maps to a label outside the SRGB irrespective of the owner MCC. An explicit SID assignment is a static assignment of a label to a FEC such that the assignment survives router reboot.
    - An example of explicit SID allocation is static assignment of a specific label to an adj-SID.
    - An implementation of explicit SID assignment MUST guarantee collision freeness on the same router
- Dynamic SID assignment:

- For all FEC types except for SR policy, the FEC types are ordered using the default administrative distance ordering defined by the implementation.

- Binding SID [RFC8402] assigned to SR Policy always has a higher default administrative distance than the default administrative distance of any other FEC type

A user SHOULD ensure that the same administrative distance preference is used on all routers to maximize forwarding consistency.

The numerical sort across FECs SHOULD be performed as follows:

- Each FEC is assigned a FEC type encoded in 8 bits. The following are the type code point for each SR FEC defined at the beginning of this Section:

  - 120: (Prefix, Routing Instance, Topology, Algorithm)

  - 130: (next-hop, outgoing interface)

  - 140: Parallel Adjacency [RFC8402]

  - 150: an SR policy [RFC8402].

  - 160: Mirror SID [RFC8402]

  - The numerical values above are mentioned to guide implementation. If other numerical values are used, then the numerical values must maintain the same greater-than ordering of the numbers mentioned here.

- The fields of each FEC are encoded as follows

  - Routing Instance ID represented by 16 bits. For routing instances that are identified by less than 16 bits, encode the Instance ID in the least significant bits while the most significant bits are set to zero

  - Address Family represented by 8 bits, where IPv4 encoded as 100 and IPv6 is encoded as 110. These numerical values are mentioned to guide implementations. If other numerical values are used, then the numerical value of IPv4 MUST be less than the numerical value for IPv6

  - All addresses are represented in 128 bits as follows

. IPv6 address is encoded natively

. IPv4 address is encoded in the most significant bits and the remaining bits are set to zero

- o All prefixes are represented by (128 + 8) bits.

  . A prefix is encoded in the most significant bits and the remaining bits are set to zero.

  . The prefix length is encoded before the prefix in a field of size 8 bits.

- o Topology ID is represented by 16 bits. For routing instances that identify topologies using less than 16 bits, encode the topology ID in the least significant bits while the most significant bits are set to zero

- o Algorithm is encoded in a 16 bits field.

- o The Color ID is encoded using 32 bits

- o Choose the set of FECs of the smallest FEC type code point

- o Out of these FECs, choose the FECs with the smallest address family code point

- o Encode the remaining set of FECs as follows

  - o Prefix, Routing Instance, Topology, Algorithm: (Prefix Length, Prefix, routing_instance_id, Topology, SR Algorithm,)

  - o (next-hop, outgoing interface): (next-hop, outgoing_interface_id)

  - o (number of adjacencies, list of next-hops in ascending numerical order, list of outgoing interface IDs in ascending numerical order). This encoding is used to encode a parallel adjacency [RFC8402]

  - o (Endpoint, Color): (Endpoint_address, Color_id)

  - o (IP address): This is the encoding for a mirror SID FEC. The IP address is encoded as described above in this section

- o Select the FEC with the smallest numerical value

The numerical values mentioned in this section are for guidance only. If other numerical values are used then the other numerical values MUST maintain the same numerical ordering among different

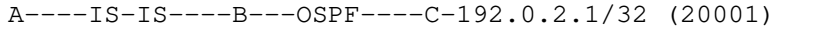
2.5.2. Redistribution between Routing Protocol Instances

The following rule SHOULD be applied when redistributing SIDs with prefixes between routing protocol instances:

- If the receiving instance's SRGB is the same as the SRGB of origin instance, then
  - the index is redistributed with the route
- Else
  - the index is not redistributed and if needed it is the duty of the receiving instance to allocate a fresh index relative to its own SRGB. Note that in that case, the receiving instance MUST compute its local label according to section 2.4 and install it in FIB.

It is outside the scope of this document to define local node behaviors that would allow to map the original index into a new index in the receiving instance via the addition of an offset or other policy means.

2.5.2.1. Illustration



Consider the simple topology above.

- A and B are in the IS-IS domain with SRGB [16000-17000]
- B and C are in OSPF domain with SRGB [20000-21000]
- B redistributes 192.0.2.1/32 into IS-IS domain
- In that case A learns 192.0.2.1/32 as an IP leaf connected to B as usual for IP prefix redistribution
- However, according to the redistribution rule above rule, B decides not to advertise any index with 192.0.2.1/32 into IS-IS because the SRGB is not the same.

2.5.2.2. Illustration 2

Consider the example in the illustration described in Section 2.5.2.1.

When router B redistributes the prefix 192.0.2.1/32, router B decides to allocate and advertise the same index 1 with the prefix 192.0.2.1/32

Within the SRGB of the IS-IS domain, index 1 corresponds to the local label 16001

- Hence according to the redistribution rule above, router B programs the incoming label 16001 in its FIB to match traffic arriving from the IS-IS domain destined to the prefix 192.0.2.1/32.

2.6. Effect of Incoming Label Collision on Outgoing Label Programming

For the determination of the outgoing label to use, the ingress node pushing new segments, and hence a stack of MPLS labels, MUST use, for a given FEC, the same label that has been selected by the node receiving the packet with that label exposed as top label. So in case of incoming label collision on this receiving node, the ingress node MUST resolve this collision using this same "Incoming Label Collision resolution procedure", using the data of the receiving node.

In the general case, the ingress node may not have exactly the same data of the receiving node, so the result may be different. This is under the responsibility of the network operator. But in typical case, e.g. where a centralized node or a distributed link state IGP is used, all nodes would have the same database. However to minimize the chance of misforwarding, a FEC that loses its incoming label to the tie-breaking rules specified in Section 2.5 MUST NOT be installed in FIB with an outgoing segment routing label based on the SID corresponding to the lost incoming label.

Examples for the behavior specified in this section can be found in Appendix A.3.

2.7. PUSH, CONTINUE, and NEXT

PUSH, NEXT, and CONTINUE are operations applied by the forwarding plane. The specifications of these operations can be found in [RFC8402]. This sub-section specifies how to implement each of these operations in the MPLS forwarding plane.

2.7.1. PUSH

PUSH corresponds to pushing one or more labels on top of an incoming packet then sending it out of a particular physical interface or virtual interface, such as UDP tunnel [RFC7510] or L2TPv3 tunnel [RFC4817], towards a particular next-hop. When pushing labels onto a packet's label stack, the Time-to-Live (TTL) field ([RFC3032], [RFC3443]) and the Traffic Class (TC) field ([RFC3032], [RFC5462]) of each label stack entry must, of course, be set. This document does not specify any set of rules for setting these fields; that is a matter of local policy. Sections 2.10 and 2.11 specify additional details about forwarding behavior.

2.7.2. CONTINUE

In the MPLS forwarding plane, the CONTINUE operation corresponds to swapping the incoming label with an outgoing label. The value of the outgoing label is calculated as specified in Sections 2.10 and 2.11.

2.7.3. NEXT

In the MPLS forwarding plane, NEXT corresponds to popping the topmost label. The action before and/or after the popping depends on the instruction associated with the active SID on the received packet prior to the popping. For example suppose the active SID in the received packet was an Adj-SID [RFC8402], then on receiving the packet, the node applies NEXT operation, which corresponds to popping the top most label, and then sends the packet out of the physical or virtual interface (e.g. UDP tunnel [RFC7510] or L2TPv3 tunnel [RFC4817]) towards the next-hop corresponding to the adj-SID.

2.7.3.1. Mirror SID

If the active SID in the received packet was a Mirror SID [RFC8402, Section 5.1] allocated by the receiving router, then the receiving router applies NEXT operation, which corresponds to popping the top most label, then performs a lookup using the contents of the packet after popping the outer most label in the mirrored forwarding table. The method by which the lookup is made, and/or the actions applied to the packet after the lookup in the mirror table depends on the contents of the packet and the mirror table. Note that the packet exposed after popping the top most label may or may not be an MPLS packet. A mirror SID can be viewed as a generalization of the context label in [RFC5331] because a mirror SID does not make any assumptions about the packet underneath the top label.

2.8. MPLS Label Downloaded to FIB for Global and Local SIDs

The label corresponding to the global SID "Si" represented by the global index "I" downloaded to FIB is used to match packets whose active segment (and hence topmost label) is "Si". The value of this label is calculated as specified in Section 2.4.

For Local SIDs, the MCC is responsible for downloading the correct label value to FIB. For example, an IGP with SR extensions [I-D.ietf-isis-segment-routing-extensions, I-D.ietf-ospf-segment-routing-extensions] allocates and downloads the MPLS label corresponding to an Adj-SID [RFC8402].

2.9. Active Segment

When instantiated in the MPLS domain, the active segment on a packet corresponds to the topmost label on the packet that is calculated according to the procedure specified in Sections 2.10 and 2.11. When arriving at a node, the topmost label corresponding to the active SID matches the MPLS label downloaded to FIB as specified in Section 2.4.

2.10. Forwarding behavior for Global SIDs

This section specifies forwarding behavior, including the calculation of outgoing labels, that corresponds to a global SID when applying PUSH, CONTINUE, and NEXT operations in the MPLS forwarding plane.

This document covers the calculation of the outgoing label for the top label only. The case where the outgoing label is not the top label and is part of a stack of labels that instantiates a routing policy or a traffic engineering tunnel is outside the scope of this document and may be covered in other documents such as [I-D.ietf-spring-segment-routing-policy].

2.10.1. Forwarding for PUSH and CONTINUE of Global SIDs

Suppose an MCC on a router "R0" determines that PUSH or CONTINUE operation is to be applied to an incoming packet related to the global SID "Si" represented by the global index "I" and owned by the router Ri before sending the packet towards a neighbor "N" directly connected to "R0" through a physical or virtual interface such as UDP tunnel [RFC7510] or L2TPv3 tunnel [RFC4817].

The method by which the MCC on router "R0" determines that PUSH or CONTINUE operation must be applied using the SID "Si" is beyond the scope of this document. An example of a method to determine the SID "Si" for PUSH operation is the case where IS-IS [I-D.ietf-isis-

segment-routing-extensions] receives the prefix-SID "Si" sub-TLV advertised with prefix "P/m" in TLV 135 and the destination address of the incoming IPv4 packet is covered by the prefix "P/m".

For CONTINUE operation, an example of a method to determine the SID "Si" is the case where IS-IS [I-D.ietf-isis-segment-routing-extensions] receives the prefix-SID "Si" sub-TLV advertised with prefix "P" in TLV 135 and the top label of the incoming packet matches the MPLS label in FIB corresponding to the SID "Si" on the router "R0".

The forwarding behavior for PUSH and CONTINUE corresponding to the SID "Si"

- If the neighbor "N" does not support SR or advertises an invalid SRGB or a SRGB that is too small for the SID "Si"
  - If it is possible to send the packet towards the neighbor "N" using standard MPLS forwarding behavior as specified in [RFC3031] and [RFC3032], then forward the packet. The method by which a router decides whether it is possible to send the packet to "N" or not is beyond the scope of this document. For example, the router "R0" can use the downstream label determined by another MCC, such as LDP [RFC5036], to send the packet.
  - Else if there are other useable next-hops, then use other next-hops to forward the incoming packet. The method by which the router "R0" decides on the possibility of using other next-hops is beyond the scope of this document. For example, the MCC on "R0" may chose the send an IPv4 packet without pushing any label to another next-hop.
  - Otherwise drop the packet.
- Else
  - Calculate the outgoing label as specified in Section 2.4 using the SRGB of the neighbor "N"
  - If the operation is PUSH
    - Push the calculated label according the MPLS label pushing rules specified in [RFC3032]
  - Else

. swap the incoming label with the calculated label according to the label swapping rules in [RFC3032]

o Send the packet towards the neighbor "N"

2.10.2. Forwarding for NEXT Operation for Global SIDs

As specified in Section 2.7.3 NEXT operation corresponds to popping the top most label. The forwarding behavior is as follows

- Pop the topmost label
- Apply the instruction associated with the incoming label that has been popped

The action on the packet after popping the topmost label depends on the instruction associated with the incoming label as well as the contents of the packet right underneath the top label that got popped. Examples of NEXT operation are described in Appendix A.1.

2.11. Forwarding Behavior for Local SIDs

This section specifies the forwarding behavior for local SIDs when SR is instantiated over the MPLS forwarding plane.

2.11.1. Forwarding for PUSH Operation on Local SIDs

Suppose an MCC on a router "R0" determines that PUSH operation is to be applied to an incoming packet using the local SID "Si" before sending the packet towards a neighbor "N" directly connected to R0 through a physical or virtual interface such as UDP tunnel [RFC7510] or L2TPv3 tunnel [RFC4817].

An example of such local SID is an Adj-SID allocated and advertised by IS-IS [I-D.ietf-isis-segment-routing-extensions]. The method by which the MCC on "R0" determines that PUSH operation is to be applied to the incoming packet is beyond the scope of this document. An example of such method is backup path used to protect against a failure using TI-LFA [I-D.bashandy-rtgwg-segment-routing-ti-lfa].

As mentioned in [RFC8402], a local SID is specified by an MPLS label. Hence the PUSH operation for a local SID is identical to label push operation [RFC3032] using any MPLS label. The forwarding action after pushing the MPLS label corresponding to the local SID is also determined by the MCC. For example, if the PUSH operation was done to

forward a packet over a backup path calculated using TI-LFA, then the forwarding action may be sending the packet to a certain neighbor that will in turn continue to forward the packet along the backup path

2.11.2. Forwarding for CONTINUE Operation for Local SIDs

A local SID on a router "R0" corresponds to a local label. In such scenario, the outgoing label towards a next-hop "N" is determined by the MCC running on the router "R0"and the forwarding behavior for CONTINUE operation is identical to swap operation [RFC3032] on an MPLS label.

2.11.3. Outgoing label for NEXT Operation for Local SIDs

NEXT operation for Local SIDs is identical to NEXT operation for global SIDs specified in Section 2.10.2.

3. IANA Considerations

This document does not make any request to IANA.

4. Manageability Considerations

This document describes the applicability of Segment Routing over the MPLS data plane. Segment Routing does not introduce any change in the MPLS data plane. Manageability considerations described in [RFC8402] applies to the MPLS data plane when used with Segment Routing. SR OAM use cases for the MPLS data plane are defined in [RFC8403]. SR OAM procedures for the MPLS data plane are defined in [RFC8287].

5. Security Considerations

This document does not introduce additional security requirements and mechanisms other than the ones described in [RFC8402].

6. Contributors

The following contributors have substantially helped the definition and editing of the content of this document:

Martin Horneffer
Deutsche Telekom
Email: Martin.Horneffer@telekom.de

Wim Henderickx
Nokia
Email: wim.henderickx@nokia.com

Jeff Tantsura
Email: jefftant@gmail.com
Edward Crabbe
Email: edward.crabbe@gmail.com

Igor Milojevic
Email: milojevicigor@gmail.com

Saku Ytti
Email: saku@ytti.fi

7. Acknowledgements

The authors would like to thank Les Ginsberg, Chris Bowers, Himanshu Shah, Adrian Farrel, Alexander Vainshtein, Przemyslaw Krol, Darren Dukes, and Zafar Ali for their valuable comments on this document.

This document was prepared using 2-Word-v2.0.template.dot.

8. References

8.1. Normative References

[RFC8402] Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402 July 2018, <http://www.rfc-editor.org/info/rfc8402>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 0.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, DOI 10.17487/RFC3031, January 2001, <http://www.rfc-editor.org/info/rfc3031>.

[RFC3032] Rosen, E., Tappan, D., Fedorkow, G., Rekhter, Y., Farinacci, D., Li, T., and A. Conta, "MPLS Label Stack Encoding", RFC 3032, DOI 10.17487/RFC3032, January 2001, <http://www.rfc-editor.org/info/rfc3032>.

[RFC3443] P. Agarwal, P. and Akyol, B. "Time To Live (TTL) Processing in Multi-Protocol Label Switching (MPLS) Networks", RFC 3443, DOI 10.17487/RFC3443, January 2003, <http://www.rfc-editor.org/info/rfc3443>.

[RFC5462] Andersson, L., and Asati, R., " Multiprotocol Label Switching (MPLS) Label Stack Entry: "EXP" Field Renamed to "Traffic Class" Field", RFC 5462, DOI 10.17487/RFC5462, February 2009, <http://www.rfc-editor.org/info/rfc5462>.

[RFC7274] K. Kompella, L. Andersson, and A. Farrel, "Allocating and Retiring Special-Purpose MPLS Labels", RFC7274 DOI 10.17487/RFC7274, May 2014 <http://www.rfc-editor.org/info/rfc7274>

[RFC8174] B. Leiba, " Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", RFC7274 DOI 10.17487/RFC8174, May 2017 <http://www.rfc-editor.org/info/rfc8174>

8.2. Informative References

[I-D.ietf-isis-segment-routing-extensions] Previdi, S., Filsfils, C., Bashandy, A., Gredler, H., Litkowski, S., Decraene, B., and j. jefftant@gmail.com, "IS-IS Extensions for Segment Routing", draft-ietf-isis-segment-routing-extensions-13 (work in progress), June 2017.

[I-D.ietf-ospf-ospfv3-segment-routing-extensions] Psenak, P., Previdi, S., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPFv3 Extensions for Segment Routing", draft-ietf-ospf-ospfv3-segment-routing-extensions-09 (work in progress), March 2017.

[I-D.ietf-ospf-segment-routing-extensions] Psenak, P., Previdi, S., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPF Extensions for Segment Routing", draft-ietf-ospf-segment-routing-extensions-16 (work in progress), May 2017.

[I-D.ietf-spring-segment-routing-ldp-interop] Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., and S. Litkowski, "Segment Routing interworking with LDP", draft-ietf-spring-segment-routing-ldp-interop-08 (work in progress), June 2017.

[I-D.bashandy-rtgwg-segment-routing-ti-lfa], Bashandy, A., Filsfils, C., Decraene, B., Litkowski, S., Francois, P., Voyer, P. Clad, F., and Camarillo, P., "Topology Independent Fast Reroute using Segment Routing", draft-bashandy-rtgwg-segment-routing-ti-lfa-05 (work in progress), October 2018,

[RFC7855] Previdi, S., Ed., Filsfils, C., Ed., Decraene, B., Litkowski, S., Horneffer, M., and R. Shakir, "Source Packet Routing in Networking (SPRING) Problem Statement and Requirements", RFC 7855, DOI 10.17487/RFC7855, May 2016, <http://www.rfc-editor.org/info/rfc7855>.

[RFC5036] Andersson, L., Acreo, AB, Minei, I., Thomas, B., " LDP Specification", RFC5036, DOI 10.17487/RFC5036, October 2007, <https://www.rfc-editor.org/info/rfc5036>

[RFC5331] Aggarwal, R., Rekhter, Y., Rosen, E., " MPLS Upstream Label Assignment and Context-Specific Label Space", RFC5331 DOI 10.17487/RFC5331, August 2008, <http://www.rfc-editor.org/info/rfc5331>.

[RFC7510] Xu, X., Sheth, N., Yong, L., Callon, R., and D. Black, "Encapsulating MPLS in UDP", RFC 7510, DOI 10.17487/RFC7510, April 2015, <https://www.rfc-editor.org/info/rfc7510>.

[RFC4817] Townsley, M., Pignataro, C., Wainner, S., Seely, T., Young, T., "Encapsulation of MPLS over Layer 2 Tunneling Protocol Version 3", RFC4817, DOI 10.17487/RFC4817, March 2007, <https://www.rfc-editor.org/info/rfc4817>

[RFC8287] N. Kumar, C. Pignataro, G. Swallow, N. Akiya, S. Kini, and M. Chen " Label Switched Path (LSP) Ping/Traceroute for Segment Routing (SR) IGP-Prefix and IGP-Adjacency Segment Identifiers (SIDs) with MPLS Data Planes" RFC8287, DOI 10.17487/RFC8287, December 2017, https://www.rfc-editor.org/info/rfc8287

[RFC8403] R. Geib, C. Filsfils, C. Pignataro, N. Kumar, "A Scalable and Topology-Aware MPLS Data-Plane Monitoring System", RFC8403, DOI 10.17487/RFC8403, July 2018, <https://www.rfc-editor.org/info/rfc8403>

[I-D.ietf-spring-segment-routing-policy] Filsfils, C., Sivabalan, S., Raza, K., Liste, J. , Clad, F., Voyer, D., Bogdanov, A., Mattes, P., " Segment Routing Policy for Traffic Engineering", draft-ietf-spring-segment-routing-policy-01 (work in progress), June 2018

9. Authors' Addresses

Ahmed Bashandy (editor)
Arrcus

Email: abashandy.ietf@gmail.com

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

Email: cfilsfil@cisco.com

Stefano Previdi
Cisco Systems, Inc.
Italy

Email: stefano@previdi.net

Bruno Decraene
Orange
FR

Email: bruno.decraene@orange.com

Stephane Litkowski
Orange
FR

Email: stephane.litkowski@orange.com

Rob Shakir
Google
US

Email: robjs@google.com

Appendix A. Examples

A.1. IGP Segments Example

Consider the network diagram of Figure 1 and the IP address and IGP Segment allocation of Figure 2. Assume that the network is running IS-IS with SR extensions [I-D.ietf-isis-segment-routing-extensions] and all links have the same metric. The following examples can be constructed.

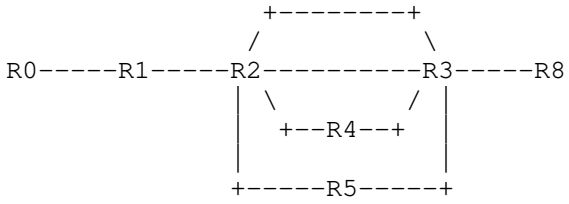


Figure 1: IGP Segments - Illustration

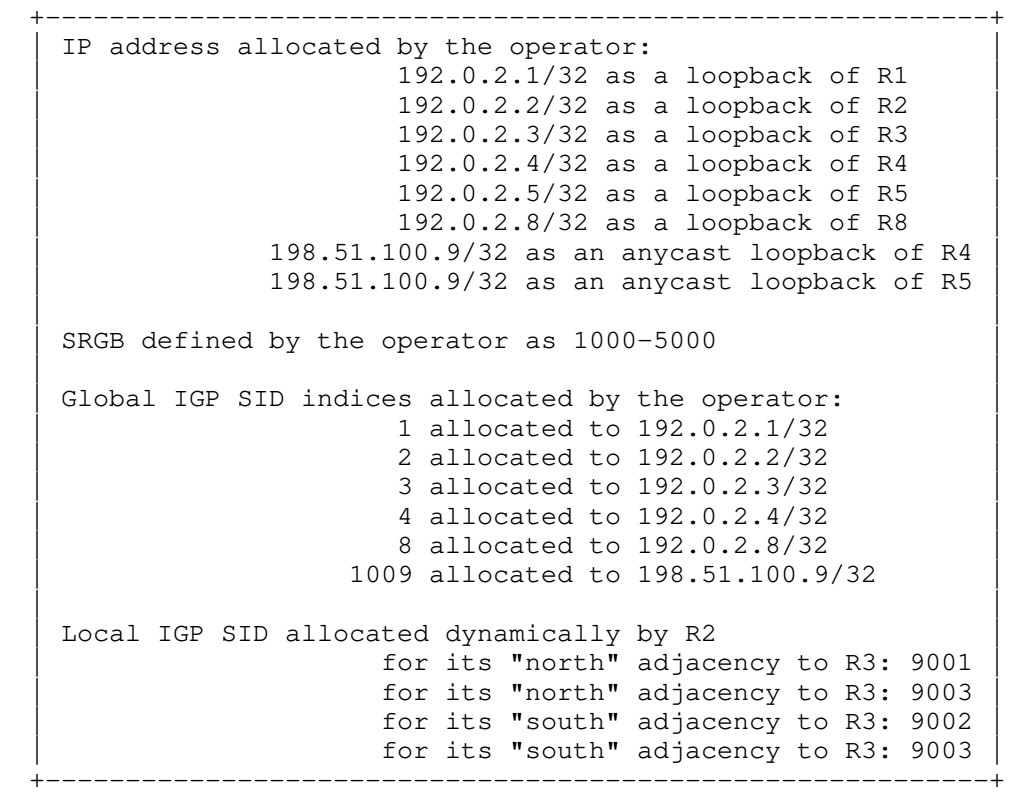


Figure 2: IGP Address and Segment Allocation - Illustration

Suppose R1 wants to send an IPv4 packet P1 to R8. In this case, R1 needs to apply PUSH operation to the IPv4 packet.

Remember that the SID index "8" is a global IGP segment attached to the IP prefix 192.0.2.8/32. Its semantic is global within the IGP domain: any router forwards a packet received with active segment 8 to the next-hop along the ECMP-aware shortest-path to the related prefix.

R2 is the next-hop along the shortest path towards R8. By applying the steps in Section 2.8 the outgoing label downloaded to R1's FIB corresponding to the global SID index 8 is 1008 because the SRGB of R2 is [1000,5000] as shown in Figure 2.

Because the packet is IPv4, R1 applies the PUSH operation using the label value 1008 as specified in Section 2.10.1. The resulting MPLS

header will have the "S" bit [RFC3032] set because it is followed directly by an IPv4 packet.

The packet arrives at router R2. Because the top label 1008 corresponds to the IGP SID "8", which is the prefix-SID attached to the prefix 192.0.2.8/32 owned by the node R8, then the instruction associated with the SID is "forward the packet using all ECMP/UCMP interfaces and all ECMP/UCMP next-hop(s) along the shortest/useable path(s) towards R8". Because R2 is not the penultimate hop, R2 applies the CONTINUE operation to the packet and sends it to R3 using one of the two links connected to R3 with top label 1008 as specified in Section 2.10.1.

R3 receives the packet with top label 1008. Because the top label 1008 corresponds to the IGP SID "8", which is the prefix-SID attached to the prefix 192.0.2.8/32 owned by the node R8, then the instruction associated with the SID is "send the packet using all ECMP interfaces and all next-hop(s) along the shortest path towards R8". Because R3 is the penultimate hop, we assume that R3 performs penumtimate hop popping, which corresponds to the NEXT operation, then sends the packet to R8. The NEXT operation results in popping the outer label and sending the packet as a pure IPv4 packet to R8.

In conclusion, the path followed by P1 is R1-R2--R3-R8. The ECMP-awareness ensures that the traffic be load-shared between any ECMP path, in this case the two links between R2 and R3.

A.2. Incoming Label Collision Examples

This section describes few examples to illustrate the handling of label collision described in Section 2.5.

For the examples in this section, we assume that Node A has the following:

- OSPF default admin distance for implementation=50
- ISIS default admin distance for implementation=60

A.2.1. Example 1

Illustration of incoming label collision resolution for the same FEC type using MCC administrative distance.

FEC1:

- OSPF prefix SID advertisement from node B for 198.51.100.5/32 with index=5
- OSPF SRGB on node A = [1000,1999]
- Incoming label=1005

FEC2:

- ISIS prefix SID advertisement from node C for 203.0.113.105/32 with index=5
- ISIS SRGB on node A = [1000,1999]
- Incoming label=1005

FEC1 and FEC2 both use dynamic SID assignment. Since neither ofthe FEC types is SR Policy, we use the default admin distances of 50 and 60 to break the tie. So FEC1 wins.

A.2.2. Example 2

Illustration of incoming label collision resolution for different FEC types using the MCC administrative distance.

FEC1:

- Node A receives an OSPF prefix sid advertisement from node B for 198.51.100.6/32 with index=6
- OSPF SRGB on node A = [1000,1999]
- Hence the incoming label on node A corresponding to 198.51.100.6/32 is 1006

FEC2:

ISIS on node A assigns the label 1006 to the globally significant adj-SID (I.e. when advertised the "L" flag is clear in the adj-SID sub-TLV as described in [I-D.ietf-isis-segment-routing-extensions]) towards one of its neighbors. Hence the incoming label corresponding to this adj-SID 1006. Assume Node A allocates this adj-SID dynamically, and it may differ across router reboots.

FEC1 and FEC2 both use dynamic SID assignment. Since neither of the FEC types is SR Policy, we use the default admin distances of 50 and 60 to break the tie. So FEC1 wins.

A.2.3. Example 3

Illustration of incoming label collision resolution based on preferring static over dynamic SID assignment

FEC1:
OSPF on node A receives a prefix SID advertisement from node B for 198.51.100.7/32 with index=7. Assuming that the OSPF SRGB on node A is [1000,1999], then incoming label corresponding to 198.51.100.7/32 is 1007

FEC2:
The operator on node A configures ISIS on node A to assign the label 1007 to the globally significant adj-SID (I.e. when advertised the "L" flag is clear in the adj-SID sub-TLV as described in [I-D.ietf-isis-segment-routing-extensions]) towards one of its neighbor advertisement from node A with label=1007

Node A assigns this adj-SID explicitly via configuration, so the adj-SID survives router reboots.

FEC1 uses dynamic SID assignment, while FEC2 uses explicit SID assignment. So FEC2 wins.

A.2.4. Example 4

Illustration of incoming label collision resolution using FEC type default administrative distance

FEC1:
OSPF on node A receives a prefix SID advertisement from node B for 198.51.100.8/32 with index=8. Assuming that OSPF SRGB on node A = [1000,1999], the incoming label corresponding to 198.51.100.8/32 is 1008.

FEC2:
Suppose the SR Policy advertisement from controller to node A for the policy identified by (Endpoint = 192.0.2.208, color = 100) and

consisting of SID-List = <S1, S2> assigns the globally significant Binding-SID label 1008

From the point of view of node A, FEC1 and FEC2 both use dynamic SID assignment. Based on the default administrative distance outlined in Section 2.5.1, the binding SID has a higher administrative distance than the prefix-SID and hence FEC1 wins.

A.2.5. Example 5

Illustration of incoming label collision resolution based on FEC type preference

FEC1:
ISIS on node A receives a prefix SID advertisement from node B for 203.0.113.110/32 with index=10. Assuming that the ISIS SRGB on node A is [1000,1999], then incoming label corresponding to 203.0.113.110/32 is 1010.

FEC2:
ISIS on node A assigns the label 1010 to the globally significant adj-SID (I.e. when advertised the "L" flag is clear in the adj-SID sub-TLV as described in [I-D.ietf-isis-segment-routing-extensions]) towards one of its neighbors).

Node A allocates this adj-SID dynamically, and it may differ across router reboots. Hence both FEC1 and FEC2 both use dynamic SID assignment.

Since both FECs are from the same MCC, they have the same default admin distance. So we compare FEC type code-point. FEC1 has FEC type code-point=120, while FEC2 has FEC type code-point=130. Therefore, FEC1 wins.

A.2.6. Example 6

Illustration of incoming label collision resolution based on address family preference.

FEC1:
ISIS on node A receives prefix SID advertisement from node B for 203.0.113.111/32 with index 11. Assuming that the ISIS SRGB on node A is [1000,1999], the incoming label on node A for 203.0.113.111/32 is 1011.

FEC2:
ISIS on node A prefix SID advertisement from node C for 2001:DB8:1000::11/128 with index=11. Assuming that the ISIS SRGB on node A is [1000,1999], the incoming label on node A for 2001:DB8:1000::11/128 is 1011

FEC1 and FEC2 both use dynamic SID assignment. Since both FECs are from the same MCC, they have the same default admin distance. So we compare FEC type code-point. Both FECs have FEC type code-point=120. So we compare address family. Since IPv4 is preferred over IPv6, FEC1 wins.

A.2.7. Example 7

Illustration incoming label collision resolution based on prefix length.

FEC1:
ISIS on node A receives a prefix SID advertisement from node B for 203.0.113.112/32 with index 12. Assuming that ISIS SRGB on node A is [1000,1999], the incoming label for 203.0.113.112/32 on node A is 1012.

FEC2:
ISIS on node A receives a prefix SID advertisement from node C for 203.0.113.128/30 with index 12. Assuming that the ISIS SRGB on node A is [1000,1999], then incoming label for 203.0.113.128/30 on node A is 1012

FEC1 and FEC2 both use dynamic SID assignment. Since both FECs are from the same MCC, they have the same default admin distance. So we compare FEC type code-point. Both FECs have FEC type code-point=120. So we compare address family. Both are IPv4 address family, so we compare prefix length. FEC1 has prefix length=32, and FEC2 has prefix length=30, so FEC2 wins.

A.2.8. Example 8

Illustration of incoming label collision resolution based on the numerical value of the FECs.

FEC1:
ISIS on node A receives a prefix SID advertisement from node B for 203.0.113.113/32 with index 13. Assuming that ISIS SRGB on node A is

[1000,1999], then the incoming label for 203.0.113.113/32 on node A is 1013

FEC2:
ISIS on node A receives a prefix SID advertisement from node C for 203.0.113.213/32 with index 13. Assuming that ISIS SRGB on node A is [1000,1999], then the incoming label for 203.0.113.213/32 on node A is 1013

FEC1 and FEC2 both use dynamic SID assignment. Since both FECs are from the same MCC, they have the same default admin distance. So we compare FEC type code-point. Both FECs have FEC type code-point=120. So we compare address family. Both are IPv4 address family, so we compare prefix length. Prefix lengths are the same, so we compare prefix. FEC1 has the lower prefix, so FEC1 wins.

A.2.9. Example 9

Illustration of incoming label collision resolution based on routing instance ID.

FEC1:
ISIS on node A receives a prefix SID advertisement from node B for 203.0.113.114/32 with index 14. Assume that this ISIS instance on node A has the Routing Instance ID 1000 and SRGB [1000,1999]. Hence the incoming label for 203.0.113.114/32 on node A is 1014

FEC2:
ISIS on node A receives a prefix SID advertisement from node C for 203.0.113.114/32 with index=14. Assume that this is another instance of ISIS on node A with a different routing Instance ID 2000 but the same SRGB [1000,1999]. Hence incoming label for 203.0.113.114/32 on node A 1014

These two FECs match all the way through the prefix length and prefix. So Routing Instance ID breaks the tie, with FEC1 winning.

A.2.10. Example 10

Illustration of incoming label collision resolution based on topology ID.

FEC1:
ISIS on node A receives a prefix SID advertisement from node B for 203.0.113.115/32 with index=15. Assume that this ISIS instance on

node A has Routing Instance ID 1000. Assume that the prefix advertisement of 203.0.113.115/32 was received in ISIS Multi-topology advertisement with ID = 50. If the ISIS SRGB for this routing instance on node A is [1000,1999], then incoming label of 203.0.113.115/32 for topology 50 on node A is 1015

FEC2:
ISIS on node A receives a prefix SID advertisement from node C for 203.0.113.115/32 with index 15. Assume that it is the same routing Instance ID = 1000 but 203.0.113.115/32 was advertised with a different ISIS Multi-topology ID = 40. If the ISIS SRGB on node A is [1000,1999], then incoming label of 203.0.113.115/32 for topology 40 on node A is also 1015

These two FECs match all the way through the prefix length, prefix, and Routing Instance ID. We compare ISIS Multi-topology ID, so FEC2 wins.

A.2.11. Example 11

Illustration of incoming label collision for resolution based on algorithm ID.

FEC1:
ISIS on node A receives a prefix SID advertisement from node B for 203.0.113.116/32 with index=16 Assume that ISIS on node A has Routing Instance ID = 1000. Assume that node B advertised 203.0.113.116/32 with ISIS Multi-topology ID = 50 and SR algorithm = 0. Assume that the ISIS SRGB on node A = [1000,1999]. Hence the incoming label corresponding to this advertisement of 203.0.113.116/32 is 1016.

FEC2:
ISIS on node A receives a prefix SID advertisement from node C for 203.0.113.116/32 with index=16. Assume that it is the same ISIS instance on node A with Routing Instance ID = 1000. Also assume that node C advertised 203.0.113.116/32 with ISIS Multi-topology ID = 50 but with SR algorithm = 22. Since it is the same routing instance, the SRGB on node A = [1000,1999]. Hence the incoming label corresponding to this advertisement of 203.0.113.116/32 by node C is also 1016.

These two FECs match all the way through the prefix length, prefix, and Routing Instance ID, and Multi-topology ID. We compare SR algorithm ID, so FEC1 wins.

A.2.12. Example 12

Illustration of incoming label collision resolution based on FEC numerical value and independent of how the SID assigned to the colliding FECs.

FEC1:
ISIS on node A receives a prefix SID advertisement from node B for 203.0.113.117/32 with index 17. Assume that the ISIS SRGB on node A is [1000,1999], then the incoming label is 1017

FEC2:
Suppose there is an ISIS mapping server advertisement (SID/Label Binding TLV) from node D has Range 100 and Prefix = 203.0.113.1/32. Suppose this mapping server advertisement generates 100 mappings, one of which maps 203.0.113.17/32 to index 17. Assuming that it is the same ISIS instance, then the SRGB is [1000,1999] and hence the incoming label for 1017.

The fact that FEC1 comes from a normal prefix SID advertisement and FEC2 is generated from a mapping server advertisement is not used as a tie-breaking parameter. Both FECs use dynamic SID assignment, are from the same MCC, have the same FEC type code-point=120. Their prefix lengths are the same as well. FEC2 wins based on lower numerical prefix value, since 203.0.113.17 is less than 203.0.113.117.

A.2.13. Example 13

Illustration of incoming label collision resolution based on address family preference

FEC1:
SR Policy advertisement from controller to node A. Endpoint address=2001:DB8:3000::100, color=100, SID-List=<S1, S2> and the Binding-SID label=1020

FEC2:
SR Policy advertisement from controller to node A. Endpoint address=192.0.2.60, color=100, SID-List=<S3, S4> and the Binding-SID label=1020

The FECs match through the tie-breaks up to and including having the same FEC type code-point=140. FEC2 wins based on IPv4 address family being preferred over IPv6.

A.2.14. Example 14

Illustration of incoming label resolution based on numerical value of the policy endpoint.

FEC1:
SR Policy advertisement from controller to node A. Endpoint address=192.0.2.70, color=100, SID-List=<S1, S2> and Binding-SID label=1021

FEC2:
SR Policy advertisement from controller to node A Endpoint address=192.0.2.71, color=100, SID-List=<S3, S4> and Binding-SID label=1021

The FECs match through the tie-breaks up to and including having the same address family. FEC1 wins by having the lower numerical endpoint address value.

A.3. Examples for the Effect of Incoming Label Collision on Outgoing Label

This section presents examples to illustrate the effect of incoming label collision on the selection of the outgoing label described in Section 2.6.

A.3.1. Example 1

Illustration of the effect of incoming label resolution on the outgoing label

FEC1:
ISIS on node A receives a prefix SID advertisement from node B for 203.0.113.122/32 with index 22. Assuming that the ISIS SRGB on node A is [1000,1999] the corresponding incoming label is 1022.

FEC2:
ISIS on node A receives a prefix SID advertisement from node C for 203.0.113.222/32 with index=22 Assuming that the ISIS SRGB on node A is [1000,1999] the corresponding incoming label is 1022.

FEC1 wins based on lowest numerical prefix value. This means that node A installs a transit MPLS forwarding entry to SWAP incoming label 1022, with outgoing label N and use outgoing interface I. N is determined by the index associated with FEC1 (index 22) and the SRGB advertised by the next-hop node on the shortest path to reach 203.0.113.122/32.

Node A will generally also install an imposition MPLS forwarding entry corresponding to FEC1 for incoming prefix=203.0.113.122/32 pushing outgoing label N, and using outgoing interface I.

The rule in Section 2.6 means node A MUST NOT install an ingress MPLS forwarding entry corresponding to FEC2 (the losing FEC, which would be for prefix 203.0.113.222/32).

A.3.2. Example 2

Illustration of the effect of incoming label collision resolution on outgoing label programming on node A

FEC1:
- SR Policy advertisement from controller to node A
- Endpoint address=192.0.2.80, color=100, SID-List=<S1, S2>
- Binding-SID label=1023

FEC2:
- SR Policy advertisement from controller to node A
- Endpoint address=192.0.2.81, color=100, SID-List=<S3, S4>
- Binding-SID label=1023

FEC1 wins by having the lower numerical endpoint address value. This means that node A installs a transit MPLS forwarding entry to SWAP incoming label=1023, with outgoing labels and outgoing interface determined by the SID-List for FEC1.

In this example, we assume that node A receives two BGP/VPN routes:

- R1 with VPN label=V1, BGP next-hop = 192.0.2.80, and color=100,
- R2 with VPN label=V2, BGP next-hop = 192.0.2.81, and color=100,

We also assume that A has a BGP policy which matches on color=100 that allows that its usage as SLA steering information. In this case, node A will install a VPN route with label stack = <S1,S2,V1> (corresponding to FEC1).

The rule described in section 2.6 means that node A MUST NOT install a VPN route with label stack = <S3,S4,V1> (corresponding to FEC2.)

Network Working Group
Internet-Draft
Intended status: Informational
Expires: June 2, 2019

C. Filsfils, Ed.
S. Previdi
Cisco Systems, Inc.
G. Dawra
LinkedIn
E. Aries
Juniper Networks
P. Lapukhov
Facebook
November 29, 2018

BGP-Prefix Segment in large-scale data centers
draft-ietf-spring-segment-routing-msdc-11

Abstract

This document describes the motivation and benefits for applying segment routing in BGP-based large-scale data-centers. It describes the design to deploy segment routing in those data-centers, for both the MPLS and IPv6 dataplanes.

Table of Contents

1. Introduction

Segment Routing (SR), as described in [I-D.ietf-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 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-cases described in this document should be considered in the context of the BGP-based large-scale data-center (DC) design described in [RFC7938]. This document extends 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 [RFC7938] that outlines a practical network design suitable for data-centers of various scales:

- 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 (as described in [RFC7938]). In a Clos topology, the minimum number of parallel paths between two elements is determined by the "width" of the "Tier-1" stage. See Figure 1 below for an illustration of the concept.

- Large-scale data-centers commonly use a routing protocol, such as BGP-4 [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.

- 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 "Tier-1" 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 only the topologies that have a single path from Tier-1 to Tier-3 are considered below. The rest could be treated similarly, with a few modifications to the logic.

2.1. Reference design

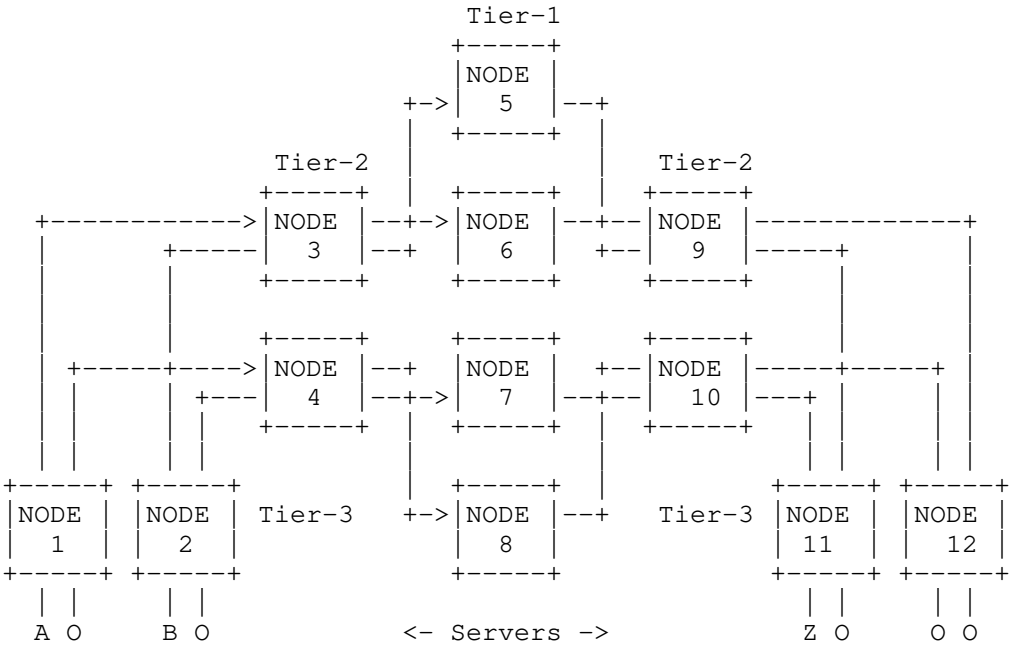


Figure 1: 5-stage Clos topology

In the reference topology illustrated in Figure 1, It is assumed:

- Each node is its own AS (Node X has AS X). 4-byte AS numbers are recommended ([RFC6793]).
  - For simple and efficient route propagation filtering, Node5, Node6, Node7 and Node8 use the same AS, Node3 and Node4 use the same AS, Node9 and Node10 use the same AS.
  - In case of 2-byte autonomous system numbers are used and for efficient usage of the scarce 2-byte Private Use AS pool, different Tier-3 nodes might use the same AS.
  - Without loss of generality, these details will be simplified in this document and assume that each node has its own AS.
- Each node peers with its neighbors with a 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.

- Each node originates the IPv4 address of its loopback interface into BGP and announces it to its neighbors.
  - The loopback of Node X is 192.0.2.x/32.

In this document, the Tier-1, Tier-2 and Tier-3 nodes are referred to respectively as Spine, Leaf and ToR (top of rack) nodes. When a ToR node acts as a gateway to the "outside world", it is referred to as a border node.

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:

- 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 does not perform efficiently when flow lifetime 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.

- Shortest-path routing with ECMP implements an 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.

- 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.

First, it will be explained 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 (BGP-Prefix-SID)

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.

The BGP Prefix Segment is defined as the BGP-Prefix-SID Attribute in [I-D.ietf-idr-bgp-prefix-sid] which contains an index. Throughout this document the BGP Prefix Segment Attribute is referred as the BGP-Prefix-SID and the encoded index as the label-index.

In this document, the network design decision has been made to assume that all the nodes are allocated the same SRGB (Segment Routing Global Block), e.g. [16000, 23999]. This provides operational simplification as explained in Section 8, but this is not a requirement.

For illustration purpose, when considering an MPLS data-plane, it is assumed that the label-index allocated to prefix 192.0.2.x/32 is X. As a result, a local label (16000+x) is allocated for prefix 192.0.2.x/32 by each node throughout the DC fabric.

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

4.2. eBGP Labeled Unicast (RFC8277)

Referring to Figure 1 and [RFC7938], the following design modifications are introduced:

- Each node peers with its neighbors via a eBGP session with extensions defined in [RFC8277] (named "eBGP8277" throughout this document) and with the BGP-Prefix-SID attribute extension as defined in [I-D.ietf-idr-bgp-prefix-sid].

- The forwarding plane at Tier-2 and Tier-1 is MPLS.

- 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 into a path from server A to server Z within the topology of Figure 1.

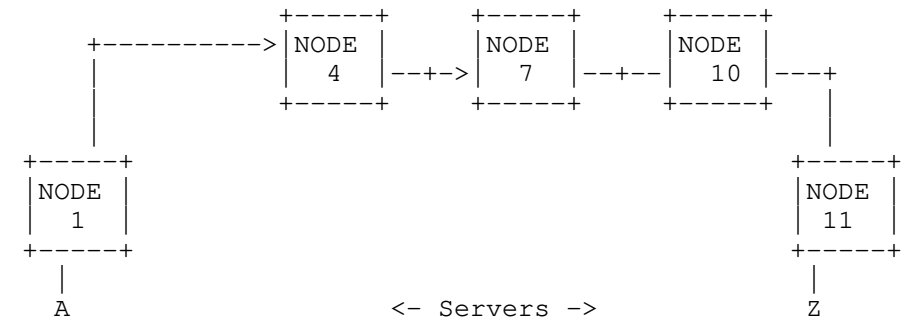


Figure 2: Path from A to Z via nodes 1, 4, 7, 10 and 11

Referring to Figure 1 and Figure 2 and assuming the IP address with the AS and label-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 a BGP-Prefix-SID with label-index: index11 [I-D.ietf-idr-bgp-prefix-sid].

Node11 sends the following eBGP8277 update to Node10:

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

Node10 receives the above update. As it is SR capable, Node10 is able to interpret the BGP-Prefix-SID and hence understands that it should allocate the label from its own SRGB block, offset by the Label-Index received in the BGP-Prefix-SID (16000+11 hence 16011) to the NLRI instead of allocating a non-deterministic 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 eBGP8277 update to Node7:

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

Node7 receives the above update. As it is SR capable, Node7 is able to interpret the BGP-Prefix-SID 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 eBGP8277 NLRI as the outgoing label (the index is only used to derive the local/incoming label).

Node7 sends the following eBGP8277 update to Node4:

. IP Prefix: 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-SID: Label-Index 11

Node4 receives the above update. As it is SR capable, Node4 is able to interpret the BGP-Prefix-SID 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 eBGP8277 NLRI as outgoing label (the index is only used to derive the local/incoming label).

Node4 sends the following eBGP8277 update to Node1:

. IP Prefix: 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-SID: Label-Index 11

Node1 receives the above update. As it is SR capable, Node1 is able to interpret the BGP-Prefix-SID 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 eBGP8277 NLRI as outgoing label (the index is only used to derive the local/incoming label).

4.2.2. Data Plane

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 Node1, Node4, Node7 and Node10.

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

Node10 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 Node4, Node7 and Node10. 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 the Egress Peer Engineering (EPE) use-case as defined in [I-D.ietf-spring-segment-routing-central-epe]. For example, applications at HostA would send their Z-destined traffic to Node1 with an MPLS label stack where the top label is 16011 and the next label is an EPE peer segment ([I-D.ietf-spring-segment-routing-central-epe]) at Node11 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-SID and label allocation throughout the DC fabric.

A few examples follow:

- Normal forwarding to Node11: a packet with top label 16011 received by any node in the fabric will be forwarded along the ECMP-aware BGP best-path towards Node11 and the label 16011 is penultimate-popped at Node10 (or at Node 9).

- Traffic-engineered path to Node11: an application on a host behind Node1 might want to restrict its traffic to paths via the Spine node Node5. The application achieves this by sending its packets with a label stack of {16005, 16011}. BGP Prefix SID 16005 directs the packet up to Node5 along the path (Node1, Node3, Node5). BGP-Prefix-SID 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-SID and let us show how the fabric connectivity is preserved.

From a signaling viewpoint, nothing would change: even though Node7 does not support the BGP-Prefix-SID, 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-SID. The neighbors of Node7 adapt automatically as they always use the label in the BGP8277 NLRI as outgoing label.

Node4 does understand the BGP-Prefix-SID 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 Label Switched Path (LSP) is preserved because the outgoing label is always derived from the received label within the BGP8277 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-SID 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 (RFC8277)

The same exact design as eBGP8277 is used with the following modifications:

All nodes use the same AS number.

Each node peers with its neighbors via an internal BGP session (iBGP) with extensions defined in [RFC8277] (named "iBGP8277" throughout this document).

Each node acts as a route-reflector for each of its neighbors and with the next-hop-self option. Next-hop-self is a well known operational feature which consists of rewriting the next-hop of a BGP update prior to send it to the neighbor. Usually, it's a common practice to apply next-hop-self behavior towards iBGP peers for eBGP learned routes. In the case outlined in this section it is proposed to use the next-hop-self mechanism also to iBGP learned routes.

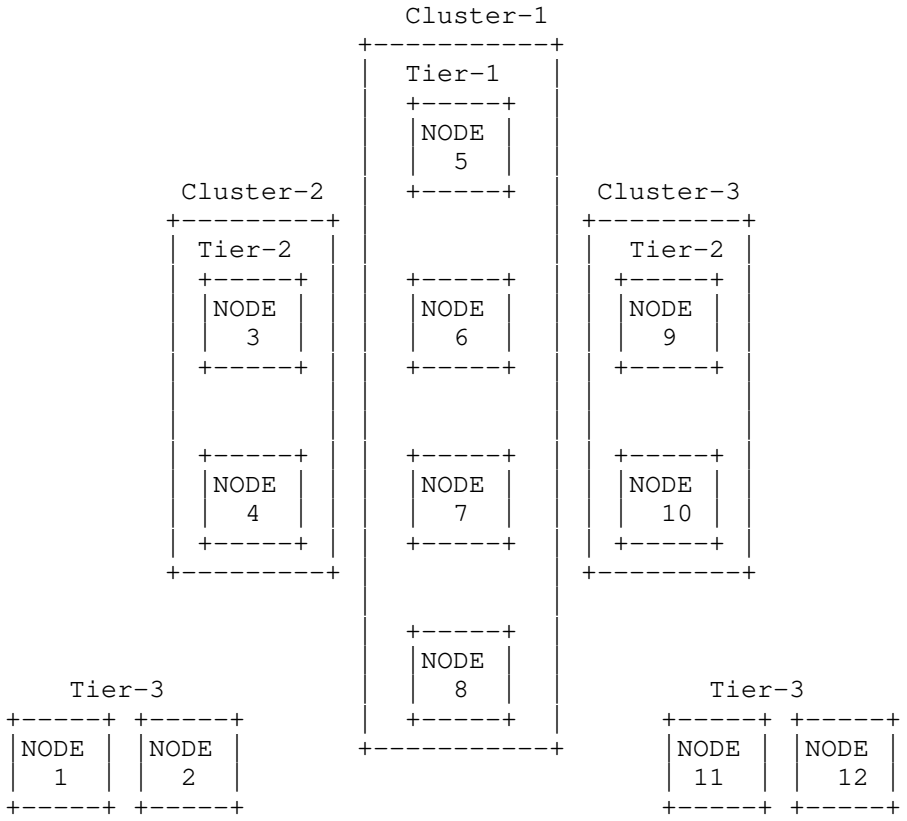


Figure 9: iBGP Sessions with Reflection and Next-Hop-Self

For simple and efficient route propagation filtering and as illustrated in Figure 9:

Node5, Node6, Node7 and Node8 use the same Cluster ID (Cluster-1)

Node3 and Node4 use the same Cluster ID (Cluster-2)

Node9 and Node10 use the same Cluster ID (Cluster-3)

The control-plane behavior is mostly the same as described in the previous section: the only difference is that the eBGP8277 path propagation is simply replaced by an iBGP8277 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 [RFC7938] is reused with one single modification. It is highlighted using the example of the reachability to Node11 via spine node Node5.

Node5 originates 2001:DB8::5/128 with the attached BGP-Prefix-SID for IPv6 packets destined to segment 2001:DB8::5 ([I-D.ietf-idr-bgp-prefix-sid]).

Node11 originates 2001:DB8::11/128 with the attached BGP-Prefix-SID advertising the support of the SRH 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 Segment Routing header (SRH, [I-D.ietf-6man-segment-routing-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, it is noted 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.

It is also noted, 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. Additional Benefits

7.1. MPLS Dataplane with operational simplicity

As required by [RFC7938], no new signaling protocol is introduced. The BGP-Prefix-SID is a lightweight extension to BGP Labeled Unicast [RFC8277]. 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 [RFC7938].

Provided the same SRGB is configured on all nodes, all nodes use the same MPLS label for a given IP prefix. This is simpler from an operation standpoint, as discussed in Section 8

7.2. 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 the source ToR node by pushing the BGP-Prefix-SID of the destination ToR for intra-DC traffic, or the BGP-Prefix-SID for the the border node for inter-DC or DC-to-outside-world traffic.

7.3. 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.ietf-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 node (say Node12 in Figure 1 instead of another border node, 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 node Node12. Once the flow gets to border node Node12, the active segment is 9999 (because of PHP on the upstream neighbor of Node12). This EPE PeerNode segment forces border node Node12 to forward the packet to peer AS 9999, without any IP lookup at the border node. 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 node Node12 is not required if all FIB lookups are avoided there by using EPE.

7.4. Anycast

The design presented in this document preserves the availability and load-balancing properties of the base design presented in [I-D.ietf-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 Node11 and Node12 (in addition to their node-specific loopbacks). Doing so, the EPE controller could express a default "go-to-the-Internet via any border node" policy as segment list {16020}. Indeed, from any host in the DC fabric or from any ToR node, 16020 steers the packet towards the border Node11 or Node12 leveraging ECMP where available along the best paths to these nodes.

8. Preferred SRGB Allocation

In the MPLS case, it is recommend to use same SRGBs at each node.

Different SRGBs in each node likely increase the complexity of the solution both from an operational 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 us to construct simple policies applicable across the fabric.

Let us consider two applications A and B respectively connected to Node1 and Node2 (ToR nodes). 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 Node5 and Node8.

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 and where the SRGB of each node is advertised using the Originator SRGB TLV of the BGP-Prefix-SID as defined in [I-D.ietf-idr-bgp-prefix-sid]: SRGB of Node K starts at value K*1000 and the SRGB length is 1000 (e.g. Node1's SRGB is [1000, 1999], Node2's SRGB is [2000, 2999], ...).

In this case, not only the controller would need to collect and store all of these different SRGB's (e.g., through the Originator SRGB TLV of the BGP-Prefix-SID), 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.

9. IANA Considerations

This document does not make any IANA request.

10. Manageability Considerations

The design and deployment guidelines described in this document are based on the network design described in [RFC7938].

The deployment model assumed in this document is based on a single domain where the interconnected DCs are part of the same administrative domain (which, of course, is split into different autonomous systems). The operator has full control of the whole domain and the usual operational and management mechanisms and procedures are used in order to prevent any information related to internal prefixes and topology to be leaked outside the domain.

As recommended in [I-D.ietf-spring-segment-routing], the same SRGB should be allocated in all nodes in order to facilitate the design, deployment and operations of the domain.

When EPE ([I-D.ietf-spring-segment-routing-central-epe]) is used (as explained in Section 7.3, the same operational model is assumed. EPE information is originated and propagated throughout the domain towards an internal server and unless explicitly configured by the operator, no EPE information is leaked outside the domain boundaries.

11. Security Considerations

This document proposes to apply Segment Routing to a well known scalability requirement expressed in [RFC7938] using the BGP-Prefix-SID as defined in [I-D.ietf-idr-bgp-prefix-sid].

It has to be noted, as described in Section 10 that the design illustrated in [RFC7938] and in this document, refer to a deployment model where all nodes are under the same administration. In this context, it is assumed that the operator doesn't want to leak outside of the domain any information related to internal prefixes and topology. The internal information includes prefix-sid and EPE information. In order to prevent such leaking, the standard BGP mechanisms (filters) are applied on the boundary of the domain.

Therefore, the solution proposed in this document does not introduce any additional security concerns from what expressed in [RFC7938] and [I-D.ietf-idr-bgp-prefix-sid]. It is assumed that the security and confidentiality of the prefix and topology information is preserved by outbound filters at each peering point of the domain as described in Section 10.

12. Acknowledgements

The authors would like to thank Benjamin Black, Arjun Sreekantiah, Keyur Patel, Acee Lindem and Anoop Ghanwani for their comments and review of this document.

13. Contributors

Gaya Nagarajan
Facebook
US

Email: gaya@fb.com

Gaurav Dawra
Cisco Systems
US

Email: gdawra.ietf@gmail.com

Dmitry Afanasiev
Yandex
RU

Email: fl0w@yandex-team.ru

Tim Laberge
Cisco
US

Email: tlaberge@cisco.com

Edet Nkposong
Salesforce.com Inc.
US

Email: enkposong@salesforce.com

Mohan Nanduri
Microsoft
US

Email: mnanduri@microsoft.com

James Uttaro
ATT
US

Email: ju1738@att.com

Saikat Ray
Unaffiliated
US

Email: raysaikat@gmail.com

Jon Mitchell
Unaffiliated
US

Email: jrmitche@puck.nether.net

14. References

14.1. Normative References

[I-D.ietf-idr-bgp-prefix-sid]
Previdi, S., Filsfils, C., Lindem, A., Sreekantiah, A., and H. Gredler, "Segment Routing Prefix SID extensions for BGP", draft-ietf-idr-bgp-prefix-sid-27 (work in progress), June 2018.

[I-D.ietf-spring-segment-routing]
Filsfils, C., Previdi, S., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", draft-ietf-spring-segment-routing-15 (work in progress), January 2018.

[I-D.ietf-spring-segment-routing-central-epe]
Filsfils, C., Previdi, S., Dawra, G., Aries, E., and D. Afanasiev, "Segment Routing Centralized BGP Egress Peer Engineering", draft-ietf-spring-segment-routing-central-epe-10 (work in progress), December 2017.

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

[RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <https://www.rfc-editor.org/info/rfc4271>.

[RFC7938] Lapukhov, P., Premji, A., and J. Mitchell, Ed., "Use of BGP for Routing in Large-Scale Data Centers", RFC 7938, DOI 10.17487/RFC7938, August 2016, <https://www.rfc-editor.org/info/rfc7938>.

[RFC8277] Rosen, E., "Using BGP to Bind MPLS Labels to Address Prefixes", RFC 8277, DOI 10.17487/RFC8277, October 2017, <https://www.rfc-editor.org/info/rfc8277>.

14.2. Informative References

[I-D.ietf-6man-segment-routing-header] Filsfils, C., Previdi, S., Leddy, J., Matsushima, S., and d. daniel.voyer@bell.ca, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-15 (work in progress), October 2018.

[RFC6793] Vohra, Q. and E. Chen, "BGP Support for Four-Octet Autonomous System (AS) Number Space", RFC 6793, DOI 10.17487/RFC6793, December 2012, <https://www.rfc-editor.org/info/rfc6793>.

Authors' Addresses

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

Email: cfilsfil@cisco.com

Stefano Previdi
Cisco Systems, Inc.
Italy

Email: stefano@previdi.net

Gaurav Dawra
LinkedIn
USA

Email: gdawra.ietf@gmail.com

Ebben Aries
Juniper Networks
1133 Innovation Way
Sunnyvale CA 94089
US

Email: exa@juniper.net

Petr Lapukhov
Facebook
US

Email: petr@fb.com

SPRING Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 19, 2019

S. Litkowski
Orange Business Service
Y. Qu
Huawei
P. Sarkar
Individual
J. Tantsura
Apstra
A. Lindem
Cisco

February 15, 2019

YANG Data Model for Segment Routing
draft-ietf-spring-sr-yang-11

Abstract

This document defines a YANG data model ([RFC6020], [RFC7950]) for segment routing ([RFC8402]) configuration and operation. This YANG model is intended to be used on network elements to configure or operate segment routing. This document defines also generic containers that SHOULD be reused by IGP protocol modules to support segment routing.

Status of This Memo

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

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

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

This Internet-Draft will expire on August 19, 2019.

Table of Contents

1. Introduction

This document defines a YANG data model for segment routing configuration and operation. This document does not define the IGP extensions to support segment routing but defines generic groupings that SHOULD be reused by IGP extension modules. The reason of this design choice is to not require implementations to support all IGP extensions. For example, an implementation may support IS-IS extension but not OSPF.

The YANG modules in this document conform to the Network Management Datastore Architecture (NMDA) [RFC8342].

2. Terminology and Notation

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].

2.1. Tree diagram

Tree diagrams used in this document follow the notation defined in [RFC8340].

2.2. Prefixes in Data Node Names

In this document, names of data nodes, actions, and other data model objects are often used without a prefix, as long as it is clear from the context in which YANG module each name is defined. Otherwise, names are prefixed using the standard prefix associated with the corresponding YANG module, as shown in Table 1.

Prefix	YANG module	Reference
if	ietf-interfaces	[RFC8343]
rt	ietf-routing	[RFC8349]
rt-types	ietf-routing-types	[RFC8294]
yang	ietf-yang-types	[RFC6991]
inet	ietf-inet-types	[RFC6991]

Table 1: Prefixes and Corresponding YANG Modules

3. Design of the Data Model

As the module definition is just starting, it is expected that there will be changes as the module matures.

```
module: ietf-segment-routing
  augment /rt:routing:
    +--rw segment-routing
       +--rw transport-type?      identityref
       +--ro node-capabilities
       |  +--ro transport-planes* [transport-plane]
       |  |  +--ro transport-plane    identityref
       |  +--ro entropy-readable-label-depth?   uint8
       +--rw msd {max-sid-depth}?
       |  +--rw node-msd?   uint8
       |  +--rw link-msd
       |     +--rw link-msds* [interface]
```

```
|        +--rw interface    if:interface-ref
|        +--rw msd?         uint8
+--rw bindings
|  +--rw mapping-server {mapping-server}?
|  |  +--rw policy* [name]
|  |     +--rw name       string
|  |     +--rw entries
|  |        +--rw mapping-entry* [prefix algorithm]
|  |           +--rw prefix        inet:ip-prefix
|  |           +--rw value-type?   enumeration
|  |           +--rw start-sid     uint32
|  |           +--rw range?        uint32
|  |           +--rw algorithm     identityref
|  +--rw connected-prefix-sid-map
|  |  +--rw connected-prefix-sid* [prefix algorithm]
|  |     +--rw prefix               inet:ip-prefix
|  |     +--rw value-type?          enumeration
|  |     +--rw start-sid            uint32
|  |     +--rw range?               uint32
|  |     +--rw algorithm            identityref
|  |     +--rw last-hop-behavior?   enumeration
|  |                                {sid-last-hop-behavior}?
|  +--rw local-prefix-sid
|     +--rw local-prefix-sid* [prefix algorithm]
|        +--rw prefix        inet:ip-prefix
|        +--rw value-type?   enumeration
|        +--rw start-sid     uint32
|        +--rw range?        uint32
|        +--rw algorithm     identityref
+--rw global-srgb
|  +--rw srgb* [lower-bound upper-bound]
|     +--rw lower-bound    uint32
|     +--rw upper-bound    uint32
+--rw srlb
|  +--rw srlb* [lower-bound upper-bound]
|     +--rw lower-bound    uint32
|     +--rw upper-bound    uint32
+--ro label-blocks*
|  +--ro lower-bound?   uint32
|  +--ro upper-bound?   uint32
|  +--ro size?          uint32
|  +--ro free?          uint32
|  +--ro used?          uint32
|  +--ro scope?         enumeration
+--ro sid-list
   +--ro sid* [target sid source source-protocol binding-type]
      +--ro target             string
      +--ro sid                uint32
```

```
               +--ro algorithm?         uint8
               +--ro source             inet:ip-address
               +--ro used?              boolean
               +--ro source-protocol    -> /rt:routing
                                           /control-plane-protocols
                                           /control-plane-protocol/name
               +--ro binding-type       enumeration
               +--ro scope?             enumeration

     notifications:
       +---n segment-routing-global-srgb-collision
       |  +--ro srgb-collisions*
       |     +--ro lower-bound?          uint32
       |     +--ro upper-bound?          uint32
       |     +--ro routing-protocol?     -> /rt:routing
       |                                    /control-plane-protocols
       |                                    /control-plane-protocol/name
       |     +--ro originating-rtr-id?   router-id
       +---n segment-routing-global-sid-collision
       |  +--ro received-target?       string
       |  +--ro new-sid-rtr-id?        router-id
       |  +--ro original-target?       string
       |  +--ro original-sid-rtr-id?   router-id
       |  +--ro index?                 uint32
       |  +--ro routing-protocol?      -> /rt:routing
       |                                  /control-plane-protocols
       |                                  /control-plane-protocol/name
       +---n segment-routing-index-out-of-range
          +--ro received-target?    string
          +--ro received-index?     uint32
          +--ro routing-protocol?   -> /rt:routing
                                       /control-plane-protocols
                                       /control-plane-protocol/name
```

4. Configuration

This module augments the "/rt:routing:" with a segment-routing container. This container defines all the configuration parameters related to segment-routing.

The segment-routing configuration is split in global configuration and interface configuration.

The global configuration includes :

- segment-routing transport type : The underlying transport type for segment routing. The version of the model limits the transport

type to an MPLS dataplane. The transport-type is only defined once for a particular routing-instance and is agnostic to the control plane used. Only a single transport-type is supported in this version of the model.

- bindings : Defines prefix to SID mappings. The operator can control advertisement of Prefix-SID independently for IPv4 and IPv6. Two types of mappings are available :
  - Mapping-server : maps non local prefixes to a segment ID. Configuration of bindings does not automatically allow advertisement of those bindings. Advertisement must be controlled by each routing-protocol instance (see Section 5). Multiple mapping policies may be defined.
  - Connected prefixes : maps connected prefixes to a segment ID. Advertisement of the mapping will be done by IGP when enabled for segment routing (see Section 5). The SID value can be expressed as an index (default), or an absolute value. The "last-hop-behavior" configuration dictates the PHP behavior: "explicit-null", "php", or "non-php".
- SRGB (Segment Routing Global Block): Defines a list of label blocks represented by a pair of lower-bound/upper-bound labels. The SRGB is also agnostic to the control plane used. So all routing-protocol instance will have to advertise the same SRGB.
- SRLB (Segment Routing Local Block): Defines a list of label blocks represented by a pair of lower-bound/upper-bound labels, reserved for lcoal SIDs.

5. IGP Control plane configuration

Support of segment-routing extensions for a particular IGP control plane is done by augmenting routing-protocol configuration with segment-routing extensions. This augmentation SHOULD be part of separate YANG modules in order to not create any dependency for implementations to support all protocol extensions.

This module defines groupings that SHOULD be used by IGP segment routing modules.

The "controlplane-cfg" grouping defines the generic global configuration for the IGP.

The "enabled" leaf enables segment-routing extensions for the routing-protocol instance.

The "bindings" container controls the routing-protocol instance's advertisement of local bindings and the processing of received bindings.

5.1. IGP interface configuration

The interface configuration is part of the "igp-interface-cfg" grouping and includes Adjacency SID properties.

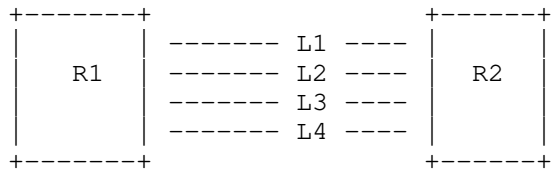
5.1.1. Adjacency SID properties

5.1.1.1. Bundling

This section is a first proposal on how to use S-bit in Adj-SID to create bundles. Authors would like to trigger discussion based on this first proposal.

In case of parallel IP links between routers, an additional Adjacency SID may be advertised representing more than one adjacency (i.e., a bundle of adjacencies). The "advertise-adj-group-sid" configuration controls whether or not an additional adjacency SID is advertised.

The "advertise-adj-group-sid" would be a list of "group-id". The "group-id" will permit to identify interfaces that must be bundled together.



In the figure above, R1 and R2 are interconnected by four links. A routing protocol adjacency is established on each link. Operator would like to create segment-routing Adj-SID that represent some bundles of links. We can imagine two different bundles : L1/L2 and L2/L3. To achieve this behavior, the service provider will configure a "group-id" X for both interfaces L1 and L2 and a "group-id" Y for both interfaces L3 and L3. This will result in R1 advertising an additional Adj-SID for each adjacency, for example a Adj-SID with S flag set and value of 400 will be added to L1 and L2. A Adj-SID with S flag set and value of 500 will be added to L3 and L4. As L1/L2 and L3/L4 does not share the same "group-id", a different SID value will be allocated.

5.1.1.2. Protection

The "advertise-protection" defines how protection for an interface is advertised. It does not control the activation or deactivation of protection. If the "single" option is used, a single Adj-SID will be advertised for the interface. If the interface is protected, the B-Flag for the Adj-SID advertisement will be set. If the "dual" option is used and if the interface is protected, two Adj-SIDs will be advertised for the interface adjacencies. One Adj-SID will always have the B-Flag set and the other will have the B-Flag clear. This option is intended to be used in the case of traffic engineering where a path must use either protected segments or non-protected segments.

6. States

The operational states contains information reflecting the usage of allocated SRGB labels.

It also includes a list of all global SIDs, their associated bindings, and other information such as the source protocol and algorithm.

7. Notifications

The model defines the following notifications for segment-routing.

- segment-routing-global-srgb-collision: Rasied when a control plan advertised SRGB blocks have conflicts.

- segment-routing-global-sid-collision: Raised when a control plane advertised index is already associated with another target (in this version, the only defined targets are IPv4 and IPv6 prefixes).

- segment-routing-index-out-of-range: Raised when a control plane advertised index fall outside the range of SRGBs configured for the network device.

8. YANG Module

The following RFCs and drafts are not referenced in the document text but are referenced in the ietf-segment-rouuting-common.yang and/or ietf-segment-routing.yang module: [RFC6991], [RFC8294], and [RFC8476].

```
<CODE BEGINS> file "ietf-segment-routing-common@2019-02-15.yang"
module ietf-segment-routing-common {
```

```
    namespace "urn:ietf:params:xml:ns:yang:ietf-segment-routing-common";
    prefix sr-cmn;

    import ietf-inet-types {
      prefix inet;
    }

    organization
      "IETF SPRING - SPRING Working Group";

    contact
      "WG Web:   <http://tools.ietf.org/wg/spring/>
       WG List:  <mailto:spring@ietf.org>

       Editor:    Stephane Litkowski
                  <mailto:stephane.litkowski@orange.com>
       Editor:    Yingzhen Qu
                  <mailto:yingzhen.qu@huawei.com>

       Author:    Acee Lindem
                  <mailto:acee@cisco.com>
       Author:    Pushpasis Sarkar
                  <mailto:pushpasis.ietf@gmail.com>
       Author:    Jeff Tantsura
                  <jefftant.ietf@gmail.com>

      ";
    description
      "The YANG module defines a collection of types and groupings for
       Segment routing.

       Copyright (c) 2017 IETF Trust and the persons identified as
       authors of the code.  All rights reserved.

       Redistribution and use in source and binary forms, with or
       without modification, is permitted pursuant to, and subject
       to the license terms contained in, the Simplified BSD License
       set forth in Section 4.c of the IETF Trust's Legal Provisions
       Relating to IETF Documents
       (http://trustee.ietf.org/license-info).

       This version of this YANG module is part of RFC XXXX;
       see the RFC itself for full legal notices.";

    reference "RFC XXXX";

    revision 2019-02-15 {
      description
```

```
        "
        * Addressed YANG Doctor's review comments;
        ";
      reference "RFC XXXX: YANG Data Model for Segment Routing.";
    }

    revision 2018-06-25 {
      description
        "
        * Renamed readable-label-stack-depth to
        * entropy-readable-label-depth;
        ";
      reference "RFC XXXX: YANG Data Model for Segment Routing.";
    }
    revision 2017-07-01 {
      description
        "
         *Conform to RFC6087BIS Appendix C
        ";
      reference "RFC XXXX: YANG Data Model for Segment Routing.";
    }
    revision 2017-03-10 {
      description
        "
         * Add support of SRLB
        ";
      reference "RFC XXXX: YANG Data Model for Segment Routing.";
    }
    revision 2016-10-28 {
      description
        "
         * Add support of MSD (Maximum SID Depth)
         * Update contact info
        ";
      reference "RFC XXXX: YANG Data Model for Segment Routing.";
    }
    revision 2016-10-24 {
      description
        "Initial";
      reference "RFC XXXX: YANG Data Model for Segment Routing.";
    }

    feature sid-last-hop-behavior {
      description
        "Configurable last hop behavior.";
    }

    identity segment-routing-transport {
```

```
      description
        "Base identity for segment routing transport.";
    }

    identity segment-routing-transport-mpls {
      base segment-routing-transport;
      description
        "This identity represents MPLS transport for segment
         routing.";
    }

    identity segment-routing-transport-ipv6 {
      base segment-routing-transport;
      description
        "This identity represents IPv6 transport for segment
         routing.";
    }

    identity prefix-sid-algorithm {
      description
        "Base identity for prefix-sid algorithm.";
    }

    identity prefix-sid-algorithm-shortest-path {
      base prefix-sid-algorithm;
      description
        "The default behavior of prefix-sid algorithm.";
    }

    identity prefix-sid-algorithm-strict-spf {
      base prefix-sid-algorithm;
      description
        "This algorithm mandates that the packet is forwared
         according to ECMP-aware SPF algorithm.";
    }

    grouping srlr {
      description
        "Grouping for SR Label Range configuration.";
      leaf lower-bound {
          type uint32;
          description
            "Lower value in the block.";
        }
      leaf upper-bound {
          type uint32;
          description
            "Upper value in the block.";
```

```
      }
    }

    grouping srgb {
      description
        "Grouping for SR Global Label range.";
      list srgb {
        key "lower-bound upper-bound";
        ordered-by user;
        description
          "List of global blocks to be
           advertised.";
        uses srlr;
      }
    }

    grouping srlb {
      description
        "Grouping for SR Local Block range.";
      list srlb {
        key "lower-bound upper-bound";
        ordered-by user;
        description
          "List of SRLBs.";
        uses srlr;
      }
    }

    grouping sid-value-type {
      description
        "Defines how the SID value is expressed.";
      leaf value-type {
        type enumeration {
          enum "index" {
            description
              "The value will be
               interpreted as an index.";
          }
          enum "absolute" {
            description
              "The value will become
               interpreted as an absolute
               value.";
          }
        }
        default "index";
        description
          "This leaf defines how value
```

```
            must be interpreted.";
        }
      }

      grouping prefix-sid {
        description
          "This grouping defines cfg of prefix SID.";
        leaf prefix {
          type inet:ip-prefix;
          description
            "connected prefix sid.";
        }
        uses prefix-sid-attributes;
      }

      grouping ipv4-sid {
        description
          "This grouping defines ipv4 prefix SID.";
        leaf prefix {
          type inet:ipv4-prefix;
          description
            "connected prefix sid.";
        }
        uses prefix-sid-attributes;
      }
      grouping ipv6-sid {
        description
          "This grouping defines ipv6 prefix SID.";
        leaf prefix {
          type inet:ipv6-prefix;
          description
            "connected prefix sid.";
        }
        uses prefix-sid-attributes;
      }

      grouping last-hop-behavior {
        description
          "Defines last hop behavior";
        leaf last-hop-behavior {
          if-feature "sid-last-hop-behavior";
          type enumeration {
            enum "explicit-null" {
              description
                "Use explicit-null for the SID.";
            }
            enum "no-php" {
              description
```

```
            "Do no use PHP for the SID.";
        }
        enum "php" {
          description
            "Use PHP for the SID.";
        }
      }
      description
        "Configure last hop behavior.";
    }
  }

  grouping node-capabilities {
    description
      "Containing SR node capabilities.";
    container node-capabilities {
      config false;
      description
        "Shows the SR capability of the node.";
      list transport-planes {
        key "transport-plane";
        description
          "List of supported transport planes.";
        leaf transport-plane {
          type identityref {
            base segment-routing-transport;
          }
          description
            "Transport plane supported";
        }
      }
      leaf entropy-readable-label-depth {
        type uint8;
        description
          "Maximum label statck depth that
           the router can read. ";
      }
    }
  }

  grouping prefix-sid-attributes {
    description
      "Containing SR attributes for a prefix.";
    uses sid-value-type;
    leaf start-sid {
      type uint32;
      mandatory true;
      description
```

```
          "Value associated with
           prefix. The value must
           be interpreted in the
           context of value-type.";
      }
      leaf range {
        type uint32;
        description
          "Describes how many SIDs could be
           allocated.";
      }
      leaf algorithm {
        type identityref {
          base prefix-sid-algorithm;
        }
        description
          "Prefix-sid algorithm.";
      }
    }
  }
  <CODE ENDS>
  <CODE BEGINS> file "ietf-segment-routing@2019-02-15.yang"
  module ietf-segment-routing {
    namespace "urn:ietf:params:xml:ns:yang:ietf-segment-routing";
    prefix sr;

    import ietf-inet-types {
      prefix inet;
    }
    import ietf-yang-types {
      prefix yang;
    }
    import ietf-routing {
      prefix rt;
    }
    import ietf-interfaces {
      prefix if;
    }
    import ietf-routing-types {
      prefix rt-types;
    }
    import ietf-segment-routing-common {
      prefix sr-cmn;
    }

    organization
      "IETF SPRING - SPRING Working Group";
    contact
```

```
      "WG Web:   <http://tools.ietf.org/wg/spring/>
       WG List:  <mailto:spring@ietf.org>

       Editor:    Stephane Litkowski
                 <mailto:stephane.litkowski@orange.com>
       Editor:    Yingzhen Qu
                 <mailto:yingzhen.qu@huawei.com>

       Author:    Acee Lindem
                 <mailto:acee@cisco.com>
       Author:    Pushpasis Sarkar
                 <mailto:pushpasis.ietf@gmail.com>
       Author:    Jeff Tantsura
                 <jefftant.ietf@gmail.com>

      ";
    description
      "The YANG module defines a generic configuration model for
       Segment routing common across all of the vendor
       implementations.

       Copyright (c) 2018 IETF Trust and the persons identified as
       authors of the code.  All rights reserved.

       Redistribution and use in source and binary forms, with or
       without modification, is permitted pursuant to, and subject
       to the license terms contained in, the Simplified BSD License
       set forth in Section 4.c of the IETF Trust's Legal Provisions
       Relating to IETF Documents
       (http://trustee.ietf.org/license-info).

       This version of this YANG module is part of RFC XXXX;
       see the RFC itself for full legal notices.";

    reference "RFC XXXX";

    revision 2019-02-15 {
      description
        "
        * Addressed YANG Doctor's review comments;
        ";
      reference "RFC XXXX: YANG Data Model for Segment Routing.";
    }

    revision 2018-06-25 {
      description
        "";
      reference "RFC XXXX: YANG Data Model for Segment Routing.";
```

```
    }
    revision 2017-07-01 {
      description
        "
         * Implement NMDA model
         *Conform to RFC6087BIS Appendix C
        ";
      reference "RFC XXXX: YANG Data Model for Segment Routing.";
    }

    revision 2017-03-10 {
      description
        "
         * Change global-sid-list to sid-list and add a leaf scope
         * Added support of SRLB
         * Added support of local sids
         * fixed indentations
        ";
      reference "RFC XXXX: YANG Data Model for Segment Routing.";
    }
    revision 2016-10-28 {
      description
        "
         * Add support of MSD (Maximum SID Depth)
         * Update contact info
        ";
      reference "RFC XXXX: YANG Data Model for Segment Routing.";
    }
    revision 2016-10-24 {
      description
        "
         * Moved common SR types and groupings to a seperate module
        ";
      reference "RFC XXXX: YANG Data Model for Segment Routing.";
    }
    revision 2016-07-07 {
      description
        "
         * Add support of prefix-sid algorithm configuration
         * change routing-protocols to control-plane-protocols
        ";
      reference "RFC XXXX: YANG Data Model for Segment Routing.";
    }
    revision 2016-03-17 {
      description
        "
         * Add notification segment-routing-global-srgb-collision
         * Add router-id to segment-routing-global-sid-collision
```

```
         * Remove routing-instance
         * Add typedef router-id
        ";
      reference "RFC XXXX: YANG Data Model for Segment Routing.";
    }
    revision 2015-10-17 {
      description
        "
         * Add per-protocol SRGB config feature
         * Move SRBG config to a grouping
        ";
      reference "RFC XXXX: YANG Data Model for Segment Routing.";
    }
    revision 2015-06-22 {
      description
        "
         * Prefix SID config moved to
         connected-prefix-sid-map in global SR cfg
         rather than IGP.
        ";
      reference "draft-litkowski-spring-sr-yang-01";
    }
    revision 2015-04-23 {
      description
        "
         * Node flag deprecated from prefixSID
         * SR interface cfg moved to protocol
         * Adding multiple binding policies for SRMS
        ";
      reference "";
    }
    revision 2015-02-27 {
      description
        "Initial";
      reference "draft-litkowski-spring-sr-yang-00";
    }

    feature mapping-server {
      description
        "Support of Segment Routing Mapping Server (SRMS).";
    }

    feature protocol-srgb {
      description
        "Support per-protocol Segment Routing Global Block (SRGB)
         configuration.";
    }
```

```
    feature max-sid-depth {
      description
        "Support of signaling MSD (Maximum SID Depth) in IGP.";
    }

    typedef system-id {
      type string {
        pattern "[0-9A-Fa-f]{4}\\.[0-9A-Fa-f]{4}\\.[0-9A-Fa-f]{4}\\.00";
      }
      description
        "This type defines ISIS system id using pattern,
         system id looks like : 0143.0438.AeF0.00";
    }

    typedef router-id {
      type union {
        type system-id;
        type rt-types:router-id;
      }
      description
        "OSPF/BGP router id or ISIS system ID.";
    }

    grouping sr-controlplane {
      description
        "Defines protocol configuration.";
      container segment-routing {
        description
          "segment routing global config.";
        leaf enabled {
          type boolean;
          default "false";
          description
            "Enables segment-routing
             protocol extensions.";
        }
        container bindings {
          description
            "Control of binding advertisement
             and reception.";
          container advertise {
            description
              "Authorize the advertise
               of local mappings in binding TLV.";
            leaf-list policies {
              type string;
              description
                "List of policies to be advertised.";
```

```
            }
          }
          leaf receive {
            type boolean;
            default "true";
            description
              "Authorize the reception and usage
               of binding TLV.";
          }
        }
      }
    }

    grouping igp-interface {
      description
        "Grouping for IGP interface cfg.";
      container segment-routing {
        description
          "container for SR interface cfg.";
        container adjacency-sid {
          description
            "Defines the adjacency SID properties.";
          list advertise-adj-group-sid {
            key "group-id";
            description
              "Control advertisement of S flag.
               Enable to advertise a common Adj-SID
               for parallel links.";
            leaf group-id {
              type uint32;
              description
                "The value is an internal value to identify
                 a group-ID. Interfaces with the same
                 group-ID will be bundled together.";
            }
          }
          leaf advertise-protection {
            type enumeration {
              enum "single" {
                description
                  "A single Adj-SID is associated
                   with the adjacency and reflects
                   the protection configuration.";
              }
              enum "dual" {
                description
                  "Two Adj-SIDs will be associated
                   with the adjacency if interface
```

```
                    is protected. In this case
                    one will be enforced with
                    backup flag set, the other
                    will be enforced to backup flag unset.
                    In case, protection is not configured,
                    a single Adj-SID will be advertised
                    with backup flag unset.";
                }
              }
              description
                "If set, the Adj-SID refers to an
                 adjacency being protected.";
            }
          }
        }
      }

      grouping max-sid-depth {
        description
          "MSD configuration grouping.";
        leaf node-msd {
          type uint8;
          description
            "Node MSD is the lowest MSD supported by the node.";
        }
        container link-msd {
          description
            "Link MSD is a number representing the particular link
             MSD value.";
          list link-msds {
            key "interface";
            description
              "List of link MSDs.";
            leaf interface {
              type if:interface-ref;
              description
                "Name of the interface.";
            }
            leaf msd {
              type uint8;
              description
                "SID depth of the interface associated with the link.";
            }
          }
        }
      }

      augment "/rt:routing" {
```

```
      description
        "This augments routing data model (RFC 8349)
         with segment-routing.";
      container segment-routing {
        description
          "segment routing global config.";
        leaf transport-type {
          type identityref {
            base sr-cmn:segment-routing-transport;
          }
          default "sr-cmn:segment-routing-transport-mpls";
          description
            "Dataplane to be used.";
        }
        uses sr-cmn:node-capabilities;
        container msd {
          if-feature "max-sid-depth";
          description
            "MSD configuration.";
          uses max-sid-depth;
        }
        container bindings {
          description
            "List of bindings.";
          container mapping-server {
            if-feature "mapping-server";
            description
              "Configuration of mapping-server
               local entries.";
            list policy {
              key "name";
              description
                "Definition of mapping policy.";
              leaf name {
                type string;
                description
                  "Name of the mapping policy.";
              }
              container entries {
                description
                  "IPv4/IPv6 mapping entries.";
                list mapping-entry {
                  key "prefix algorithm";
                  description
                    "Mapping entries.";
                  uses sr-cmn:prefix-sid;
                }
              }
```

```
          }
        }
        container connected-prefix-sid-map {
          description
            "Prefix SID configuration.";
          list connected-prefix-sid {
            key "prefix algorithm";
            description
              "List of prefix SID mapped to
               ipv4/ipv6 local prefixes.";
            uses sr-cmn:prefix-sid;
            uses sr-cmn:last-hop-behavior;
          }
        }
        container local-prefix-sid {
          description
            "Local sid configuration.";
          list local-prefix-sid {
            key "prefix algorithm";
            description
              "List of local ipv4/ipv6 prefix-sid.";
            uses sr-cmn:prefix-sid;
          }
        }
      }
      container global-srgb {
        description
          "Global SRGB configuration.";
        uses sr-cmn:srgb;
      }
      container srlb {
        description
          "SR Local Block configuration.";
        uses sr-cmn:srlb;
      }

      list label-blocks {
        config false;
        description
          "List of labels blocks currently
           in use.";
        leaf lower-bound {
          type uint32;
          description
            "Lower bound of the label block.";
        }
        leaf upper-bound {
          type uint32;
```

```
            description
              "Upper bound of the label block.";
          }
          leaf size {
            type uint32;
            description
              "Number of indexes in the block.";
          }
          leaf free {
            type uint32;
            description
              "Number of indexes free in the block.";
          }
          leaf used {
            type uint32;
            description
              "Number of indexes used in the block.";
          }
          leaf scope {
            type enumeration {
              enum "global" {
                description
                  "Global sid.";
              }
              enum "local" {
                description
                  "Local sid.";
              }
            }
            description
              "Scope of this label block.";
          }
        }
        container sid-list {
          config false;
          description
            "List of prefix and SID associations.";
          list sid {
            key "target sid source source-protocol binding-type";
            ordered-by system;
            description
              "Binding.";
            leaf target {
              type string;
              description
                "Defines the target of the binding.
                 It can be a prefix or something else.";
            }
```

```
            leaf sid {
              type uint32;
              description
                "Index associated with the prefix.";
            }
            leaf algorithm {
              type uint8;
              description
                "Algorithm to be used for the prefix
                 SID.";
            }
            leaf source {
              type inet:ip-address;
              description
                "IP address of the router than own
                 the binding.";
            }
            leaf used {
              type boolean;
              description
                "Defines if the binding is used
                 in forwarding plane.";
            }
            leaf source-protocol {
              type leafref {
                path "/rt:routing/rt:control-plane-protocols/"
                   + "rt:control-plane-protocol/rt:name";
              }
              description
                "Rtg protocol that owns the binding";
            }
            leaf binding-type {
              type enumeration {
                enum "prefix-sid" {
                  description
                    "Binding is learned from
                     a prefix SID.";
                }
                enum "binding-tlv" {
                  description
                    "Binding is learned from
                     a binding TLV.";
                }
              }
              description
                "Type of binding.";
            }
            leaf scope {
```

```
              type enumeration {
                enum "global" {
                  description
                    "Global sid.";
                }
                enum "local" {
                  description
                    "Local sid.";
                }
              }
              description
                "The sid is local or global.";
            }
          }
        }
      }
    }

    notification segment-routing-global-srgb-collision {
      description
        "This notification is sent when received SRGB blocks from
         a router conflict.";
      list srgb-collisions {
        description
          "List of SRGB blocks that conflict.";
        leaf lower-bound {
          type uint32;
          description
            "Lower value in the block.";
        }
        leaf upper-bound {
          type uint32;
          description
            "Upper value in the block.";
        }
        leaf routing-protocol {
          type leafref {
            path "/rt:routing/rt:control-plane-protocols/"
               + "rt:control-plane-protocol/rt:name";
          }
          description
            "Routing protocol reference that received the event.";
        }
        leaf originating-rtr-id {
          type router-id;
          description
            "Originating router id of this SRGB block.";
        }
```

```
      }
    }
    notification segment-routing-global-sid-collision {
      description
        "This notification is sent when a new mapping is learned
         , containing mapping
         where the SID is already used.
         The notification generation must be throttled with at least
         a 5 second gap. ";
      leaf received-target {
        type string;
        description
          "Target received in the controlplane that
                              caused SID collision.";
      }
      leaf new-sid-rtr-id {
        type router-id;
        description
          "Router Id that advertising the conflicting SID.";
      }
      leaf original-target {
        type string;
        description
          "Target already available in database that have the same SID
           as the received target.";
      }
      leaf original-sid-rtr-id {
        type router-id;
        description
          "Original router ID that advertised the conflicting SID.";
      }
      leaf index {
        type uint32;
        description
          "Value of the index used by two different prefixes.";
      }
      leaf routing-protocol {
        type leafref {
          path "/rt:routing/rt:control-plane-protocols/"
             + "rt:control-plane-protocol/rt:name";
        }
        description
          "Routing protocol reference that received the event.";
      }
    }
    notification segment-routing-index-out-of-range {
      description
        "This notification is sent when a binding
```

```
         is received, containing a segment index
         which is out of the local configured ranges.
         The notification generation must be throttled with at least
         a 5 second gap. ";
      leaf received-target {
        type string;
        description
          "Target received in the controlplane
                           that caused SID collision.";
      }
      leaf received-index {
        type uint32;
        description
          "Value of the index received.";
      }
      leaf routing-protocol {
        type leafref {
          path "/rt:routing/rt:control-plane-protocols/"
             + "rt:control-plane-protocol/rt:name";
        }
        description
          "Routing protocol reference that received the event.";
      }
    }
  }
  <CODE ENDS>
```

9. Security Considerations

TBD.

10. Acknowledgements

Authors would like to thank Derek Yeung, Acee Lindem, Greg Hankins, Hannes Gredler, Uma Chunduri, Jeffrey Zhang, Shradda Hedge, Les Ginsberg for their contributions.

11. IANA Considerations

TBD.

12. References

12.1. Normative References

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

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/info/rfc6020>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/info/rfc6991>.

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/info/rfc7950>.

[RFC8294] Liu, X., Qu, Y., Lindem, A., Hopps, C., and L. Berger, "Common YANG Data Types for the Routing Area", RFC 8294, DOI 10.17487/RFC8294, December 2017, <https://www.rfc-editor.org/info/rfc8294>.

[RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <https://www.rfc-editor.org/info/rfc8342>.

[RFC8343] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 8343, DOI 10.17487/RFC8343, March 2018, <https://www.rfc-editor.org/info/rfc8343>.

[RFC8349] Lhotka, L., Lindem, A., and Y. Qu, "A YANG Data Model for Routing Management (NMDA Version)", RFC 8349, DOI 10.17487/RFC8349, March 2018, <https://www.rfc-editor.org/info/rfc8349>.

[RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <https://www.rfc-editor.org/info/rfc8402>.

[RFC8476] Tantsura, J., Chunduri, U., Aldrin, S., and P. Psenak, "Signaling Maximum SID Depth (MSD) Using OSPF", RFC 8476, DOI 10.17487/RFC8476, December 2018, <https://www.rfc-editor.org/info/rfc8476>.

[RFC8491] Tantsura, J., Chunduri, U., Aldrin, S., and L. Ginsberg, "Signaling Maximum SID Depth (MSD) Using IS-IS", RFC 8491, DOI 10.17487/RFC8491, November 2018, <https://www.rfc-editor.org/info/rfc8491>.

12.2. Informative References

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <https://www.rfc-editor.org/info/rfc8340>.

Authors' Addresses

Stephane Litkowski
Orange Business Service

Email: stephane.litkowski@orange.com

Yingzhen Qu
Huawei

Email: yingzhen.qu@huawei.com

Pushpasis Sarkar
Individual

Email: pushpasis.ietf@gmail.com

Jeff Tantsura
Apstra

Email: jefftant.ietf@gmail.com

Acee Lindem
Cisco
301 Mindenhall Way
Cary, NC 27513
US

Email: acee@cisco.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 1, 2018

X. Xu
Huawei
A. Bashandy
Cisco
H. Assarpour
Broadcom
S. Ma
Juniper
W. Henderickx
Nokia
J. Tantsura
Individual

September 28, 2017

Unified Source Routing Instructions using MPLS Label Stack
draft-xu-mpls-unified-source-routing-instruction-04

Abstract

MPLS Segment Routing (SR-MPLS in short) is an MPLS data plane-based source routing paradigm in which a sender of a packet is allowed to partially or completely specify the route the packet takes through the network by imposing stacked MPLS labels to the packet. SR-MPLS could be leveraged to realize a unified source routing mechanism across MPLS, IPv4 and IPv6 data planes by using an MPLS label stack as a unified source routing instruction set while preserving backward compatibility with SR-MPLS.

Status of This Memo

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

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

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

This Internet-Draft will expire on April 1, 2018.

Table of Contents

1. Introduction

MPLS Segment Routing (SR-MPLS in short) [I-D.ietf-spring-segment-routing-mpls] is an MPLS data plane-based source routing paradigm in which a sender of a packet is allowed to partially or completely specify the route the packet takes through the network by imposing stacked MPLS labels to the packet. SR-MPLS could be leveraged to realize a unified source routing mechanism across MPLS, IPv4 and IPv6 data planes by using an MPLS label stack as a unified source routing instruction set while preserving backward compatibility with SR-MPLS. More specifically, the source routing instruction set information contained in a source routed packet could be uniformly encoded as an MPLS label stack no matter the underlay is IPv4, IPv6 or MPLS.

Although the source routing instructions are encoded as MPLS labels, this is a hardware convenience rather than an indication that the whole MPLS protocol stack and in particular the MPLS control protocols need to be deployed. Note that the complexity associated with the whole MPLS protocol stack is largely due to the complex control plane protocols.

Section 3 describes various use cases for the unified source routing instruction mechanism and Section 4 describes a typical application scenario and how the packet forwarding happens.

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].

2. Terminology

This memo makes use of the terms defined in [RFC3031] and [I-D.ietf-spring-segment-routing-mpls].

3. Use Cases

The unified source routing mechanism across IPv4, IPv6 and MPLS is useful at least in the following use cases:

- Incremental deployment of the SR-MPLS technology [I-D.xu-mpls-spring-islands-connection-over-ip]. Since there is no need to run any other label distribution protocol (e.g., LDP, see [I-D.ietf-spring-segment-routing-ldp-interop] for more details.) on those non-SR-MPLS routers for incremental deployment purposes, the network provisioning is greatly simplified, which is one of the major claimed benefits of the SR-MPLS technology (i.e., running a single protocol).

- Overcome the load-balancing dilemma encountered by SR-MPLS. In fact, this unified source routing mechanism is even useful in a fully upgraded SR-MPLS network since the load-balancing dilemma encountered by SR-MPLS [I-D.ietf-mpls-spring-entropy-label] due to the maximum Readable Label-stack Depth (RLD) hardware limitation [I-D.ietf-ospf-mpls-elc] [I-D.ietf-isis-mpls-elc] [I-D.ietf-idr-bgp-ls-segment-routing-rld] and the Maximum SID Depth (MSD) hardware limitation [I-D.ietf-ospf-segment-routing-msd] [I-D.ietf-isis-segment-routing-msd] [I-D.ietf-idr-bgp-ls-segment-routing-msd] by using the MPLS-in-UDP

encapsulation [RFC7510] where the source port of the UDP tunnel header is used as an entropy field.

- A poor man's light-weight alternative to SRv6 [I-D.ietf-6man-segment-routing-header]. At least, it could be deployed as an interim until full featured SRv6 is available on more platforms. Since the Source Routing Header (SRH) [I-D.ietf-6man-segment-routing-header] consisting of an ordered list of 128-bit long IPv6 addresses is now replaced by an ordered list of 32-bit long label entries (i.e., label stack), the encapsulation overhead and forwarding performance issues associated with SRv6 are eliminated.

- A new IPv4 source routing mechanism which has overcome the security vulnerability issues associated with the traditional IPv4 source routing mechanism.

- Traffic Engineering scenarios where only a few routers (e.g., the entry and exit nodes of each plane in the dual-plane network case or the egress node in the Egress Peer Engineering (EPE) case) are specified as segments of explicit paths. In this way, only a few routers are required to support the SR-MPLS capability while all the other routers just need to support IP forwarding capability, which would significantly reduce the deployment cost of the SR-MPLS technology.

- MPLS-based Service Function Chaining (SFC) [I-D.xu-mpls-service-chaining]. Based on the unified source routing mechanism as described in this document, only SFC-related nodes including Service Function Forwarders (SFF), Service Functions (SF) and classifiers are required to recognize the SFC encapsulation header in the MPLS label stack form, while the intermediate routers just need to support vanilla IP forwarding (either IPv4 or IPv6). In other words, it undoubtedly complies with the transport-independence requirement for the SFC encapsulation header as listed in the SFC architecture document [RFC7665].

4. Packet Forwarding Procedures

The primary objective of this document is to describe how SR-MPLS capable routers and IP-only routers can seamlessly co-exist and interoperate. This section describes the forwarding information base (FIB) entry and the forwarding behavior that allow the deployment of SR-MPLS when some routers are IPv4 only or IPv6 only. Note that OSPF or ISIS is assumed to be enabled in the following examples as described in Section 4.1 and 4.2, in fact, it's no doubt that BGP could be used as a replacement.

4.1. Forwarding Entry Construction

This sub-section describes the how to construct the forwarding information base (FIB) entry on an SR-MPLS-capable router when some or all of the next-hops along the shortest path towards a prefix-SID are IPv4-only or IPv6-only routers. Consider the router "A" receiving a labeled packet whose top label L(E) corresponds to the prefix-SID is "SID(E)" of prefix "P(E)" advertised by the router "E". Suppose the ith next-hop router "NHi" along the shortest path from the router "A" towards the prefix-SID "SID(E)" is not SR-MPLS capable. That is both routers "A" and "E" are SR-MPLS capable but the next hop "NHi" along the shortest path from "A" to "E". The following applies:

- It is assumed that the router "E" advertises the SR-Capabilities sub-TLV as described in and [I-D.ietf-ospf-segment-routing-extensions], which includes the SRGB because router "E" is SR-MPLS capabile.

- The owning router "E" MUST advertise the encapsulation endpoint and the tunnel type using [I-D.ietf-isis-encapsulation-cap] and/or [I-D.ietf-ospf-encapsulation-cap] .

- If "A" and "E" are in different areas/levels, then

  * The OSPF Tunnel Encapsulation TLV [I-D.ietf-ospf-encapsulation-cap] and/or the ISIS Tunnel Encapsulation sub-TLV [I-D.ietf-isis-encapsulation-cap] are flooded domain-wide.

  * The OSPF SID/label range TLV [I-D.ietf-ospf-segment-routing-extensions] and the ISIS SR-Capabilities Sub-TLV [I-D.ietf-isis-segment-routing-extensions] are advertised domain-wide. This way router "A" knows the characteristics of the owning router "E".

  * When the owning router "E" is running ISIS and advertises the prefix "P(E) ", the router "E" uses the extended reachability TLV (TLVs 135, 235, 236, 237) and associates the IPv4/IPv6 and/or IPv4/IPv6 source router ID sub-TLV(s) [RFC7794].

  * When the owning router "E" is running OSPF and advertises the prefix "P(E)", the router "E" uses the OSPFv2 Extended Prefix Opaque LSA [RFC7684] and sets the flooding scope to AS-wide.

  * When the owning router "E" is running ISIS and advertises the ISIS capabilities TLV (TLV 242) [RFC7981], it must set the "router-ID" field to a valid value or include IPV6 TE router-

ID sub-TLV (TLV 12), or do both. The "S" bit (flooding scope) of the ISIS capabilities TLV (TLV 242) MUST be set to "1" .

- Router "A" programs the FIB entry corresponding to the "SID(E)" as follows:

  * If NP (OSPF) or P (ISIS) flag is clear,

  *

    + pop the outer label.

  * If NP (OSPF) or P (ISIS) is set,

  *

    + the outer label is SID(E) plus the lower bound of the SRGB of "E".

  * Encapsulate the packet according to the encapsulation advertised in [I-D.ietf-isis-encapsulation-cap] or [I-D.ietf-ospf-encapsulation-cap].

  * Send the packet towards the next hop "NHi".

4.2. Packet Forwarding Procedures

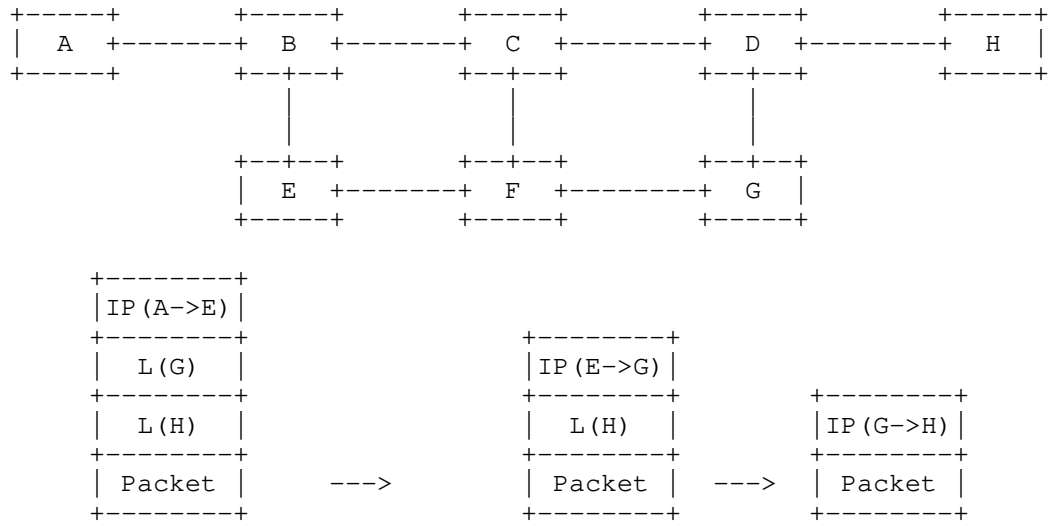


Figure 1

As shown in Figure 1, Assume Router A, E, G and H are SR-MPLS-capable routers while the remaining are only capable of forwarding IP packets. Router A, E, G and H advertise their Segment Routing related information via IS-IS or OSPF. Now assume router A wants to send a given IP or MPLS packet via an explicit path of {E->G->H}, router A would impose an MPLS label stack corresponding to that explicit path on the received IP packet. Since there is no Label Switching Path (LSP) towards router E, router A would replace the top label indicating router E with an IP-based tunnel for MPLS (e.g., MPLS-over-UDP [RFC7510]) towards router E and then send it out. In other words, router A would pop the top label and then encapsulate the MPLS packet with an IP-based tunnel towards router E. When the IP-encapsulated MPLS packet arrives at router E, router E would strip the IP-based tunnel header and then process the decapsulated MPLS packet accordingly. Since there is no LSP towards router G which is indicated by the current top label of the decapsulated MPLS packet, router E would replace the current top label with an IP-based tunnel towards router G and send it out. When the packet arrives at router G, router G would strip the IP-based tunnel header and then process the decapsulated MPLS packet. Since there is no LSP towards router H, router G would replace the current top label with an IP-based tunnel towards router H. Now the packet encapsulated with the IP-based tunnel towards router H is exactly the original packet that router A had intended to send towards router H. If the packet is an MPLS packet, router G could use any IP-based tunnel for MPLS (e.g., MPLS-over-UDP [RFC7510]). If the packet is an IP packet, router G could use any IP tunnel for IP (e.g., IP-in-UDP [I-D.xu-intarea-ip-in-udp]). That original IP or MPLS packet would be forwarded towards router H via an IP-based tunnel. When the encapsulated packet arrives at router H, router H would decapsulate it into the original packet and then process it accordingly.

Note that in the above description, it’s assumed that the label associated with each prefix-SID advertised by the owner of the prefix-SID is a Penultimate Hop Popping (PHP) label (e.g., the NP-flag [I-D.ietf-ospf-segment-routing-extensions] associated with the corresponding prefix SID is not set).

Figure 2 demostrates the packet walk in the case where the label associated with each prefix-SID advertised by the owner of the prefix-SID is not a Penultimate Hop Popping (PHP) label (e.g., the NP-flag [I-D.ietf-ospf-segment-routing-extensions] associated with the corresponding prefix SID is set).

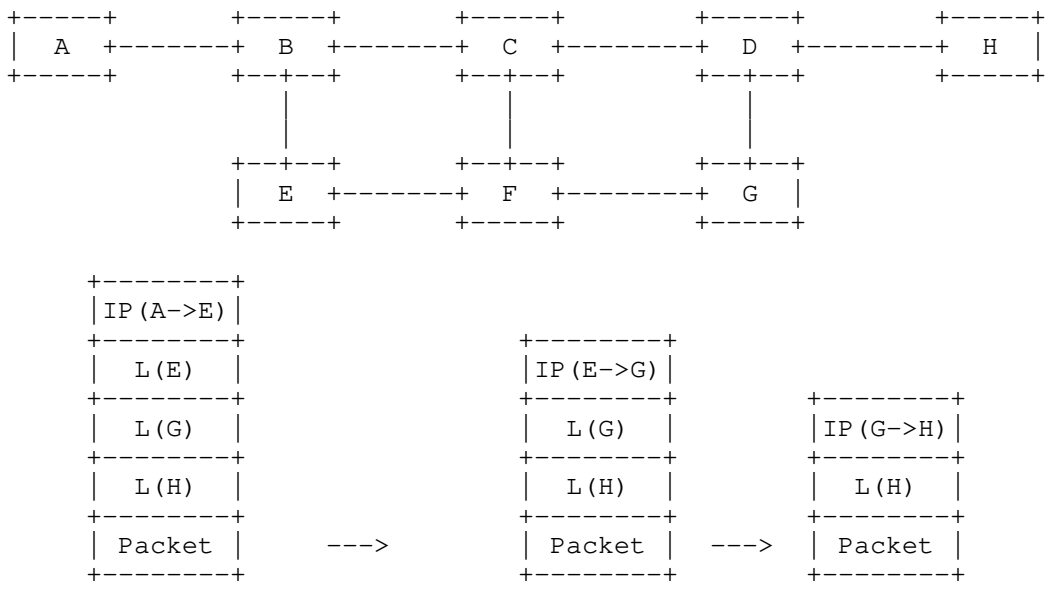


Figure 2

Although the above description is based on the use of prefix-SIDs, the unified source routing instruction approach is actually applicable to the use of adj-SIDs as well. For instance, when the top label of a received MPLS packet indicates an given adj-SID and the corresponding adjacent node to that adj-SID is not MPLS-capable, the top label would be replaced by an IP-based tunnel towards that adjacent node and then forwarded over the correponding link indicated by that adj-SID.

When encapsulating an MPLS packet with an IP-based tunnel header (e.g., a UDP header as per [RFC7510]), the corresponding entropy field (i.e., the source port in the MPLS-in-UDP case) should be filled with an entropy value that is generated by the encapsulator to uniquely identify a flow. However, what constitutes a flow is locally determined by the encapsulator. For instance, if the MPLS label stack contains at least one entropy label and the encapsulator is capable of reading that entropy label, the entropy label value could be directly copied to the entropy field (e.g., the source port of the UDP header). Otherwise, the encapsulator may have to perform a hash on the whole label stack or the five-tuple of the MPLS payload if the payload is determined as an IP packet. To avoid re-performing hash on the whole packet when re-encapsulating the packet with an IP-based tunnel header (e.g., a UDP tunnel header), especially when the encapsulator could not obtain at least one entropy label due to some reasons (e.g., 1) there is no EL at all in the label stack; 2) the encapsulator couldn't recognize the ELI; 3) the encapsulator could

not read the EL due to the RLD limit), it's RECOMMENDED that the entropy value contained in the packet (e.g., the UDP source port value) is kept when stripping the IP-based tunnel header (e.g., the UDP tunnel header). As such, the entropy value could be directly copied to the entropy field (e.g., the source port of the UDP tunnel header) when re-encapsulating the packet with an IP-based tunnel header (e.g., a UDP tunnel header). As such, the load-balancing dilemma encountered by SR-MPLS as described in [I-D.ietf-mpls-spring-entropy-label] due to the maximum Readable Label-stack Depth (RLD) hardware limitation [I-D.ietf-ospf-mpls-elc] [I-D.ietf-isis-mpls-elc] and the Maximum SID Depth (MSD) hardware limitation [I-D.ietf-ospf-segment-routing-msd] [I-D.ietf-isis-segment-routing-msd] is gone. That's the reason why this unified source routing mechanism is even useful in a fully upgraded SR-MPLS network environment.

5. Contributors

Clarence Filsfils
Cisco
Email: cfilsfil@cisco.com

Robert Raszuk
Bloomberg LP
Email: robert@raszuk.net

Uma Chunduri
Huawei
Email: uma.chunduri@gmail.com

Luis M. Contreras
Telefonica I+D
Email: luismiguel.contrerasmurillo@telefonica.com

Luay Jalil
Verizon
Email: luay.jalil@verizon.com

Gunter Van De Velde
Nokia
Email: gunter.van_de_velde@nokia.com

Tal Mizrahi
Marvell
Email: talmi@marvell.com

6. Acknowledgements

Thanks Joel Halpern, Bruno Decraene, Loa Andersson and Stewart Bryant for their insightful comments on this document.

7. IANA Considerations

No IANA action is required.

8. Security Considerations

TBD.

9. References

9.1. Normative References

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

9.2. Informative References

[I-D.ietf-6man-segment-routing-header]
Previdi, S., Filsfils, C., Raza, K., Leddy, J., Field, B., daniel.voyer@bell.ca, d., daniel.bernier@bell.ca, d., Matsushima, S., Leung, I., Linkova, J., Aries, E., Kosugi, T., Vyncke, E., Lebrun, D., Steinberg, D., and R. Raszuk, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-07 (work in progress), July 2017.

[I-D.ietf-idr-bgp-ls-segment-routing-msd]
Tantsura, J., Chunduri, U., Mirsky, G., and S. Sivabalan, "Signaling Maximum SID Depth using Border Gateway Protocol Link-State", draft-ietf-idr-bgp-ls-segment-routing-msd-00 (work in progress), July 2017.

[I-D.ietf-idr-bgp-ls-segment-routing-rld]
Velde, G., Henderickx, W., Bocci, M., and K. Patel, "Signalling ERLD using BGP-LS", draft-ietf-idr-bgp-ls-segment-routing-rld-00 (work in progress), July 2017.

[I-D.ietf-isis-encapsulation-cap]
Xu, X., Decraene, B., Raszuk, R., Chunduri, U., Contreras, L., and L. Jalil, "Advertising Tunnelling Capability in IS-IS", draft-ietf-isis-encapsulation-cap-01 (work in progress), April 2017.

[I-D.ietf-isis-mpls-elc]
Xu, X., Kini, S., Sivabalan, S., Filsfils, C., and S. Litkowski, "Signaling Entropy Label Capability Using IS-IS", draft-ietf-isis-mpls-elc-02 (work in progress), October 2016.

[I-D.ietf-isis-segment-routing-extensions]
Previdi, S., Filsfils, C., Bashandy, A., Gredler, H., Litkowski, S., Decraene, B., and j. jefftant@gmail.com, "IS-IS Extensions for Segment Routing", draft-ietf-isis-segment-routing-extensions-13 (work in progress), June 2017.

[I-D.ietf-isis-segment-routing-msd]
Tantsura, J., Chunduri, U., Aldrin, S., and L. Ginsberg, "Signaling MSD (Maximum SID Depth) using IS-IS", draft-ietf-isis-segment-routing-msd-04 (work in progress), June 2017.

[I-D.ietf-mpls-spring-entropy-label]
Kini, S., Kompella, K., Sivabalan, S., Litkowski, S., Shakir, R., and j. jefftant@gmail.com, "Entropy label for SPRING tunnels", draft-ietf-mpls-spring-entropy-label-06 (work in progress), May 2017.

[I-D.ietf-ospf-encapsulation-cap]
Xu, X., Decraene, B., Raszuk, R., Contreras, L., and L. Jalil, "The Tunnel Encapsulations OSPF Router Information", draft-ietf-ospf-encapsulation-cap-08 (work in progress), September 2017.

[I-D.ietf-ospf-mpls-elc]
Xu, X., Kini, S., Sivabalan, S., Filsfils, C., and S. Litkowski, "Signaling Entropy Label Capability Using OSPF", draft-ietf-ospf-mpls-elc-04 (work in progress), November 2016.

[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-19 (work in progress), August 2017.

[I-D.ietf-ospf-segment-routing-msd]
Tantsura, J., Chunduri, U., Aldrin, S., and P. Psenak, "Signaling MSD (Maximum SID Depth) using OSPF", draft-ietf-ospf-segment-routing-msd-05 (work in progress), June 2017.

[I-D.ietf-spring-segment-routing-ldp-interop]
Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., and S. Litkowski, "Segment Routing interworking with LDP", draft-ietf-spring-segment-routing-ldp-interop-08 (work in progress), June 2017.

[I-D.ietf-spring-segment-routing-mpls]
Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls-10 (work in progress), June 2017.

[I-D.xu-intarea-ip-in-udp]
Xu, X., Lee, Y., and F. Yongbing, "Encapsulating IP in UDP", draft-xu-intarea-ip-in-udp-04 (work in progress), December 2016.

[I-D.xu-mpls-service-chaining]
Xu, X., Bryant, S., Assarpour, H., Shah, H., Contreras, L., daniel.bernier@bell.ca, d., jefftant@gmail.com, j., Ma, S., and M. Vigoureux, "Service Chaining using Unified Source Routing Instructions", draft-xu-mpls-service-chaining-03 (work in progress), June 2017.

[I-D.xu-mpls-spring-islands-connection-over-ip]
Xu, X., Raszuk, R., Chunduri, U., Contreras, L., and L. Jalil, "Connecting MPLS-SPRING Islands over IP Networks", draft-xu-mpls-spring-islands-connection-over-ip-00 (work in progress), October 2016.

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

[RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, DOI 10.17487/RFC3031, January 2001, <https://www.rfc-editor.org/info/rfc3031>.

[RFC4817] Townsley, M., Pignataro, C., Wainner, S., Seely, T., and J. Young, "Encapsulation of MPLS over Layer 2 Tunneling Protocol Version 3", RFC 4817, DOI 10.17487/RFC4817, March 2007, <https://www.rfc-editor.org/info/rfc4817>.

[RFC7510] Xu, X., Sheth, N., Yong, L., Callon, R., and D. Black, "Encapsulating MPLS in UDP", RFC 7510, DOI 10.17487/RFC7510, April 2015, <https://www.rfc-editor.org/info/rfc7510>.

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

[RFC7684] Psenak, P., Gredler, H., Shakir, R., Henderickx, W., Tantsura, J., and A. Lindem, "OSPFv2 Prefix/Link Attribute Advertisement", RFC 7684, DOI 10.17487/RFC7684, November 2015, <https://www.rfc-editor.org/info/rfc7684>.

[RFC7794] Ginsberg, L., Ed., Decraene, B., Previdi, S., Xu, X., and U. Chunduri, "IS-IS Prefix Attributes for Extended IPv4 and IPv6 Reachability", RFC 7794, DOI 10.17487/RFC7794, March 2016, <https://www.rfc-editor.org/info/rfc7794>.

[RFC7981] Ginsberg, L., Previdi, S., and M. Chen, "IS-IS Extensions for Advertising Router Information", RFC 7981, DOI 10.17487/RFC7981, October 2016, <https://www.rfc-editor.org/info/rfc7981>.

Authors' Addresses

Xiaohu Xu
Huawei

Email: xuxiaohu@huawei.com

Ahmed Bashandy
Cisco

Email: bashandy@cisco.com

Hamid Assarpour
Broadcom

Email: hamid.assarpour@broadcom.com

Shaowen Ma
Juniper

Email: mashao@juniper.net

Wim Henderickx
Nokia

Email: wim.henderickx@nokia.com

Jeff Tantsura
Individual

Email: jefftant@gmail.com