

SPRING Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: April 7, 2016

C. Bowers  
P. Sarkar  
H. Gredler  
Juniper Networks  
U. Chunduri  
Ericsson Inc  
October 5, 2015

Advertising Per-Topology and Per-Algorithm Label Blocks  
draft-bowers-spring-adv-per-algorithm-label-blocks-02

Abstract

When segment routing is used in a network that is controlled by a link state IGP (such as ISIS or OSPF), each node in the network can be assigned one or more index numbers, known as "node-SIDs". The node-SIDs are unique within the network, and are known to all the nodes in the network. If an ingress node has a data packet to be sent to an egress node, the ingress node may select a node-SID corresponding to the egress node, and "translate" that node-SID to an MPLS label. The MPLS label represents a particular path to the egress node; the path is determined by applying a routing algorithm to a particular view of the network topology and a particular set of metric assignments to the links of that topology. The packet can then be forwarded by pushing the label on the packet's label stack and transmitting the packet to the next hop on the corresponding path to the egress node. This document compares two different procedures for translating a node-SID to the MPLS label that represents a path chosen by a particular algorithm operating on a particular topology. It also specifies the ISIS extensions needed to support one of the procedures (known as the "per-topology/per-algorithm label block" procedure).

Status of This Memo

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

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

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

This Internet-Draft will expire on April 7, 2016.

#### Copyright Notice

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

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

#### Table of Contents

1. Introduction . . . . .	2
2. Destination-based forwarding using other algorithms . . . . .	3
3. Multi-topology routing . . . . .	5
4. Example: Adding Nodes when Multiple Algorithms are In Use . .	6
5. Proposed configured offset mapping method for assigning per-topology/per-algorithm node-SIDs when using Option 1 . . . .	7
6. Flexibility to create easy-to-interpret label values . . . . .	9
7. Robustness against misconfiguration . . . . .	10
8. ISIS extensions to encode per-topology/per-algorithm label blocks . . . . .	11
9. OSPF extensions to encode per-topology/per-algorithm label blocks . . . . .	12
10. A note on algorithms and topologies . . . . .	12
11. IANA Considerations . . . . .	12
12. Management Considerations . . . . .	12
13. Security Considerations . . . . .	13
14. Acknowledgements . . . . .	13
15. References . . . . .	13
15.1. Normative References . . . . .	13
15.2. Informative References . . . . .	13
Authors' Addresses . . . . .	14

#### 1. Introduction

[I-D.ietf-spring-segment-routing] describes the segment routing architecture. When segment routing is used in a network that is controlled by a link state IGP (such as ISIS or OSPF), each node in the network can be assigned one or more index numbers, known as "node-SIDs". The node-SIDs are unique within the network, and are

known to all the nodes in the network. If an ingress node has a data packet to be sent to an egress node, the ingress node may select a node-SID corresponding to the egress node, and "translate" that node-SID to an MPLS label. The MPLS label represents a particular path to the egress node; the path is determined by applying a routing algorithm to a particular view of the network topology and a particular set of metric assignments to the links of that topology. The packet can then be forwarded by pushing the label on the packet's label stack and transmitting the packet to the next hop on the corresponding path to the egress node.

When a particular network is using a single routing algorithm and a single topology, the procedure for translating a node-SID to an MPLS label is straightforward. Figure 1 shows the formula used to translate a node-SID into an MPLS label when the paths are selected by using the default routing algorithm (Dijkstra's shortest path first algorithm) and the default topology.

$$\text{SPF\_Label}(X,D) = \text{Label\_Block}(X) + \text{Node\_Index}(D)$$

D is the destination node

X is the next-hop along the path to D

Figure 1: Translating Node-SID to Label: The Default Case

As a simple example, when the computing node (Y) needs to forward a packet ultimately destined for node D, Y first determines the shortest path next-hop node to reach D, which in this example is X. Y then adds the Node\_Index value advertised by D to the Label\_Block value advertised by X to determine the label value to apply to the packet before sending it to X.

## 2. Destination-based forwarding using other algorithms

Figure 2 shows two options for generalizing the above formula, to determine locally significant labels corresponding to forwarding next-hops computed using other algorithms.

Option 1a: per-algorithm node index  
 $\text{Label}(X,D,A) = \text{Label\_Block}(X) + \text{Node\_Index}(D,A)$

Option 2a: per-algorithm label block  
 $\text{Label}(X,D,A) = \text{Label\_Block}(X,A) + \text{Node\_Index}(D)$

A is the algorithm for computing destination-based forwarding next-hops

D is the destination node

X is the next hop along the path to D that is determined by algorithm A

Figure 2: Translating Node-SID to Label: Algorithm-Specific Options

Suppose router Y needs to forward a packet to node D along a path computed by algorithm A. Using either option, Y determines the next-hop computed by algorithm A to reach D, which in this example is X. Y then needs to figure out the correct label to apply to the packet so that so that X will also understand that the packet is to be sent to node D along a path computed by algorithm A. The two options shown in Figure 2 differ in how Y determines that label value.

In Option 1a each node advertises a single label block, but advertises a different node index for each algorithm. Y determines the label value of local significance to X to reach D using algorithm A by adding the Node\_Index advertised by node D for algorithm A to the Label\_Block advertised by node X. We refer to this as the per-algorithm node index option.

In Option 2a each node advertises only a single node index, but advertises a different label block for each algorithm. Y determines the label value of local significance to X to reach D using algorithm A by adding the Node\_Index advertised by node D to the Label\_Block for algorithm A advertised by node X. We refer to this as the per-algorithm label block option.

The extensions currently defined in [I-D.ietf-isis-segment-routing-extensions] and [I-D.ietf-ospf-segment-routing-extensions] specify encodings for Option 1a, the per-algorithm node index option. This draft proposes extensions that can be used to support option 2a, the per-algorithm label block option. However, before discussing those extensions, we generalize the formula in Figure 2 further to take into account multiple topologies. This will allow us to define extensions that address the use of both multiple topologies and multiple algorithms.

### 3. Multi-topology routing

The IGP extensions to support multi-topology routing are defined in [RFC4915] for OSPF and [RFC5120] for IS-IS. Figure 3 further generalizes the formulas above to take into account multiple topologies. It shows two options for determining locally significant labels for different topologies and algorithms.

Option 1: per-topology / per-algorithm node index  
 $\text{Label}(X,D,T,A) = \text{Label\_Block}(X) + \text{Node\_Index}(D,T,A)$

Option 2: per-topology / per-algorithm label block  
 $\text{Label}(X,D,T,A) = \text{Label\_Block}(X,T,A) + \text{Node\_Index}(D)$

T is the topology

A is the algorithm for computing destination-based forwarding next-hops

D is the destination node

X is the next hop along the path to D that is determined by algorithm A for topology T

Figure 3: Translating Node-SID to Label: Topology and Algorithm-Specific Options

In Option 1 each node advertises a single label block, but advertises a different node index for each combination of topology and algorithm used. In order for Y to determine the label value that tells X to reach D via the path chosen by algorithm A for topology T, Y adds the Node\_Index advertised by node D for topology T and algorithm A to the Label\_Block advertised by node X. We refer to this as the per-topology/per-algorithm node index option.

In Option 2 each node advertises a single node index and a unique label block along for each combination of topology and algorithm used. In order for Y to determine the label value that tells X to reach D via the path chosen by algorithm A for topology T, Y adds the Node\_Index advertised by node D to the Label\_Block advertised by node X for topology T and algorithm A. We refer to this as the per-topology/per-algorithm label block option.

Note that the formulas in Figure 3 can of course be applied even if there is only one algorithm and/or only one topology. For example, if the use case uses multiple topologies but only uses the default shortest path algorithm (algorithm=0), then option 2 can be written as:  $\text{Label}(X,D,T,0) = \text{Label\_Block}(X,T,0) + \text{Node\_Index}(D)$ , which is

independent of algorithm. Similarly, if the use case only uses the default topology (topology=0) but uses different algorithms, then option 2 can be written as `Label(X,D,0,A) = Label_Block(X,0,A) + Node_Index(D)`.

#### 4. Example: Adding Nodes when Multiple Algorithms are In Use

The following example illustrates the practical difficulties associated with using the per-topology/per-algorithm node index option alone (option 1 in Figure 3 ). This example is intentionally simplified to illustrate the need for some kind of convention to manage the assignment of the unique node index values required by option 1, even in a simple scenario. The sections below discuss a more complex example, as well as a specific proposal to manage the assignment of unique node index values. This simplified example assumes that the operator does not use multi-topology routing, i.e. that the default topology is used.

Suppose an operator has a network with 100 nodes, which we will refer to as R0-R99. The operator assigns the unique node index values 0-99 to those nodes for algorithm=0, in order to accomplish shortest path routing based on IGP metrics with SR labels. Each node will need to advertise a label block of size=100.

Assume that at some future point in time, the IETF defines algorithm=2 to mean shortest path routing based on latency, and vendors implement this. (See section Section 10 for more discussion of this example.) Suppose that the operator wants to use latency-based SPF routes for some traffic and metric-based SPF routes for other traffic. The operator will need to define a new set of unique node index values for algorithm=2. A reasonable choice would be to assign node index values of 100-199 to R0-R99 for algorithm=2. Each node will now need to advertise a label block of size=200. So far the need for per-algorithm node index values is an annoyance, but not too difficult to deal with.

Now assume that the operator needs to add 10 new nodes to the SR domain, specifically nodes R100-R109. Each node will now need to advertise a label block of size=220. The main issue is deciding how to assign per-algorithm node index for the 10 new nodes. One option is to redo the node index numbering scheme so that R0-R109 have node index values 0-109 for algorithm=0 and node index values 110-229 for algorithm=2. However, this requires renumbering existing nodes. The other option is to avoid renumbering of nodes by assigning nodes R100-R109 node index values 200-209 for algorithm=0 and node index values 210-219 for algorithm=1. Each of these approaches has drawbacks. The first requires renumbering existing nodes, while the

second is difficult to maintain since there is no obvious relationship between the node index values for different algorithms.

In order to reduce the complexity associated with option 1 in this simple example, a certain amount of pre-planning together with some convention for assigning node index values to algorithms or topologies would be useful. Specific proposals for managing unique node index values when using option 1 are discussed below. First however, we illustrate the advantages of option 2 for this simple example.

The use of per-algorithm label blocks avoids the problems associated with assigning and maintaining unique node index values for each forwarding algorithm.

When the SR domain is initially deployed, R0-R99 can be assigned node index values 0-99, as one would expect. When support for algorithm=1 gets added, the operator does not need to assign and configure any new node index values. Instead, the routers automate the process by advertising different label blocks for each forwarding algorithm.

When another 10 nodes are added to the SR domain, R100-R109 get assigned node index values 100-109 as one would expect. And the router advertises a label block of size=110 for each algorithm, as one would expect. Adding new nodes in the presence of multiple forwarding algorithms is simplified significantly with the use of per-algorithm label blocks.

5. Proposed configured offset mapping method for assigning per-topology/per-algorithm node-SIDs when using Option 1

If a network operator uses option 1, which requires the assignment of unique per-topology/per-algorithm node-SIDs, then it is clear that a common convention or methodology would be useful to help assign and maintain those unique node-SIDs. The methodology described in this section represents the authors' understanding of a proposal to manage assignment of node-SIDs when using option 1, as discussed on the SPRING mailing list.

The proposed method for managing the assignment of unique node index values for each topology/algorithm pair involves configuring a mapping from each topology/algorithm pair to an offset value. This offset mapping would need to be configured identically on every router in the network. Figure 4 shows the formula for a router Y to compute its own unique node index value for each topology/algorithm pair. Y would then treat those computed node index values as if they were directly configured via CLI or via Netconf/Yang, advertising

them into the IGP and installing the appropriate label operations in the FIB.

$$\text{Node\_Index}(Y,T,A) = \text{Configured\_Offset}(T,A) + \text{Base\_Node\_Index}(Y)$$

Y is the computing router  
T is the topology  
A is the algorithm

Figure 4: Proposed configured offset mapping method to manage assignment of unique per-topology/per-algorithm node index values when using Option 1

We illustrate the operation of the configured offset mapping method with a specific example. In this example, the operator has a network with 500 nodes, and wants to support four different topologies using different algorithms. The default topology (topology=0) needs to support algorithms 0, 4, and 5. Topology 2 and topology 6 need to support algorithm 0, while topology 7 needs to support algorithm 2. There are a total of six topology/algorithm pairs. In order to avoid renumbering the network in the event of unanticipated increases in the number of nodes or the number of topology/algorithm pairs, the operator sizes the label offsets and overall label block size to accommodate 1000 nodes and 12 topology/algorithm pairs.

Figure 5 shows the configuration data required on each of the 500 routers using option 1 together with the configured offset mapping method to manage node index assignment.

```
base_node_index=123
label_block_size=12000
topology=0 algorithm=0 offset=0
topology=0 algorithm=4 offset=1000
topology=0 algorithm=5 offset=2000
topology=2 algorithm=0 offset=3000
topology=6 algorithm=0 offset=4000
topology=7 algorithm=2 offset=5000
```

Figure 5: Required configuration data using option 1

The `base_node_index` value is the unique node index for a given node, and will thus be different for each node. The other values define the overall size of the label block and associate topology algorithm pairs with an offset value. This set of values must be configured identically across all routers in the network in order avoid advertising duplicate node index values. Advertisement of duplicate



node index values would disrupt forwarding. The configuration above would result in R123 computing node index values of 123, 1123, 2123, 3123, 4123, and 5123 for the corresponding topology/algorithm pairs.

For comparison, Figure 6 shows the configuration data required on each of the 500 routers using option 2. Since the per-topology/per-algorithm label blocks are advertised independently by each node, option 2 requires no additional configuration beyond what is required for default topology shortest path forwarding (topology=0, algorithm=0).

```
node_index=123
label_block_size=1000
```

Figure 6: Required configuration data using option 2

## 6. Flexibility to create easy-to-interpret label values

For some applications, it may be desirable to arrange things so that the meaning of label values used for forwarding can be readily understood by people trouble-shooting the network. When using the configured offset mapping method with option 1, if one configures a meaningful base value for the single label block, then the configured offset values can also be chosen to provide understandable label values. In the example above with 500 nodes and 6 topology/algorithm pairs, if the single logically advertised label block consists of a single numerically contiguous label block from 20000 through 31999 across all routers in the network, then the label values corresponding to forwarding to R123 using different topology/algorithm pairs will be meaningful to a people. They will be 20123, 21123, 22123, 23123, 24123, and 25123 for the corresponding topology/algorithm pairs, so an operator who remembers the mapping between topology/algorithm pair and offset can tell that 25123 is the label corresponding to topology=7, algorithm=2, node=123.

When using option 2 (per-topology/per-algorithm label blocks) and requiring that the topology, algorithm, and node associated with a label value be easy to interpret, each topology/algorithm pair needs to have an associated label\_block\_base configured on every router. Figure 7 show an example configuration of a mapping from topology/algorithm pairs to label\_block\_base values.

```

node_index=123
label_block_size=1000
topology=0 algorithm=0 label_block_base=100000
topology=0 algorithm=4 label_block_base=104000
topology=0 algorithm=5 label_block_base=105000
topology=2 algorithm=0 label_block_base=120000
topology=6 algorithm=0 label_block_base=160000
topology=7 algorithm=2 label_block_base=172000

```

Figure 7: Configuration data for 500 node example with option 2,

Note in this example that we have taken advantage of the additional flexibility of option 2 to create label values that are more readable than from option 1. In this example, a first digit of "1" indicates that this is a SPRING node label. The second and third digits are readable as the topology and algorithm, while the last three digits encode the node number. So 172123 would indicate the node label for topology=7, algorithm=2, node=123.

In the above example, we have illustrated the flexibility of option 2 to create more readable labels in a hypothetical network with no constraints on label space. However, it is likely that in a multi-vendor network with multiple generations of hardware supporting different MPLS applications there will exist constraints regarding the location and size of contiguous label blocks for use by SPRING. This would impose constraints on one's ability to construct readable label values using option 1 with the configured offset mapping. Option 2 provides more flexibility to construct easy-to-interpret label values in such a network.

## 7. Robustness against misconfiguration

Option 2 is much more robust against misconfiguration than is option 1. This is true both in scenarios that require easy-to-interpret label values and in scenarios that do not.

In the simple case where the application does not require easy-to-interpret label values, option 2 has clear advantages over option 1 in terms of robustness against misconfiguration. Option 1 requires identical offset mapping configurations on all routers for proper forwarding. Option 2 requires no configuration, so there is nothing to misconfigure.

In scenarios requiring easy-to-interpret label values, where option 2 requires a label\_block\_base mapping configuration, option 2 is still more robust against misconfiguration than option 1. Misconfiguration of the label\_block\_base mapping in option 2 does not affect forwarding. The explicit advertisement of the per-topology/per-

algorithm label blocks ensures that forwarding will continue to work properly.

# 8. ISIS extensions to encode per-topology/per-algorithm label blocks

Below is a concrete proposal for encoding per-topology/per-algorithm label blocks in ISIS compatible with the encodings in [I-D.ietf-isis-segment-routing-extensions].

The newly-defined Topology-Algorithm-Label-Block sub-TLV is shown in Figure 8. It is carried in the IS-IS Router Capability TLV-242. It contains a 12-bit MT-ID field as well as an 8-bit Algorithm field which associates the SRGB Descriptor entries carried in the sub-TLV with a particular topology and algorithm. Otherwise, the structure and interpretation of the Topology-Algorithm-Label-Block sub-TLV is identical to that of the SR-Capabilities sub-TLV defined in section 3.1 of [I-D.ietf-isis-segment-routing-extensions].

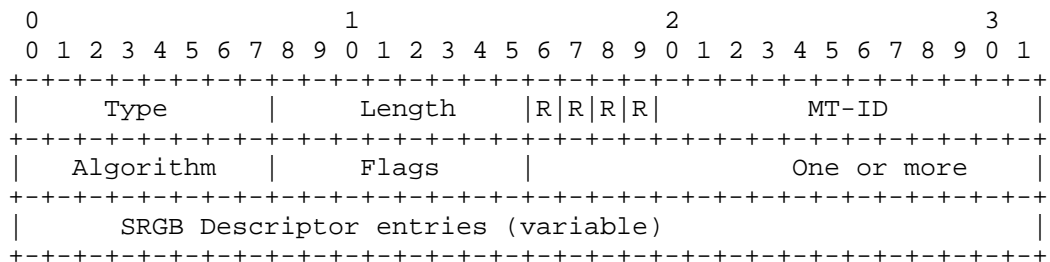


Figure 8: Topology-Algorithm-Label-Block sub-TLV

A single network must use either option 1 or option 2 for all routers. Mixed mode operation is not supported. When the Topology-Algorithm-Label-Block sub-TLV is present with a given pair of topology and algorithm values, routers MUST determine the label values associated with that topology/algorithm pair using the per-topology/per-algorithm label block method and the concatenated label block carried by the Topology-Algorithm-Label-Block sub-TLV. The node indices used in this calculation are those carried in Node-SID advertisements with algorithm value=0 in TLV-135(IPv4) or TLV-236(IPv6).

The Topology-Algorithm-Label-Block sub-TLV MUST NOT be advertised with both MT-ID=0 and Algorithm value=0. In this way, the concatenated label block used to compute the label values for the default topology and algorithm=0 can only be carried by the SR Capabilities sub-TLV.

When using the Topology-Algorithm-Label-Block sub-TLV in a network, nodes SHOULD only advertise a node index value corresponding to algorithm=0 in Node-SID advertisements in TLV-135(IPv4) and/or TLV-236(IPv6). Node index values (with algorithm=0 or any other value) SHOULD NOT be advertised in TLV-235(MT-IPv4) and TLV-237(MT-IPv6). If a node originates the Topology-Algorithm-Label-Block sub-TLV (meaning that it supports option 2), then it MUST ignore the receipt of node indices for non-zero algorithms in TLV-135 and TLV-236 and any node index values in TLV-235 and TLV-237.

9. OSPF extensions to encode per-topology/per-algorithm label blocks

OSPF extensions to encode per-topology/per-algorithm label blocks will be provided in a future version of this draft.

10. A note on algorithms and topologies

The example given in Section 4 supposes that at some point in the future the IETF defines algorithm=2 to mean shortest path routing based on latency. This simple example was chosen since it is easy to understand. However, the same result could also have been achieved by defining a second topology which uses latency as the metric for that topology, and running the default SPF algorithm on that second topology.

In general, when using other algorithms for computing next-hops for destination-based forwarding, it is not possible to achieve the same results by simply defining a new topology with modified metrics and running the default SPF algorithm. An example of such an algorithm is that used to compute Maximally Redundant Trees (MRTs), as defined in [I-D.ietf-rtgwg-mrt-frr-algorithm].

11. IANA Considerations

This document requests the following registration in the "sub-TLVs for TLV 242" registry.

Value: TBA (suggested value 20)

Description: Topology-Algorithm-Label-Block

Reference: This document (Section 8)

12. Management Considerations

This document proposes the use of per-topology/per-algorithm label blocks (option 2) to support destination-based forwarding along next-hops computed using different algorithms for different topologies.

The automated advertisement of per-topology/per-algorithm label blocks significantly simplifies network management compared to configuration and maintenance of unique per-topology/per-algorithm node indices.

### 13. Security Considerations

TBD

### 14. Acknowledgements

The authors thank John Scudder and Eric Rosen for their helpful review of this document and suggestions.

### 15. References

#### 15.1. Normative References

- [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, <<http://www.rfc-editor.org/info/rfc4915>>.
- [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, <<http://www.rfc-editor.org/info/rfc5120>>.

#### 15.2. Informative References

- [I-D.ietf-isis-segment-routing-extensions] Previdi, S., Filsfils, C., Bashandy, A., Gredler, H., Litkowski, S., Decraene, B., and J. Tantsura, "IS-IS Extensions for Segment Routing", draft-ietf-isis-segment-routing-extensions-05 (work in progress), June 2015.
- [I-D.ietf-ospf-segment-routing-extensions] Psenak, P., Previdi, S., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPF Extensions for Segment Routing", draft-ietf-ospf-segment-routing-extensions-05 (work in progress), June 2015.
- [I-D.ietf-rtgwg-mrt-frr-algorithm] Envedi, G., Csaszar, A., Atlas, A., Bowers, C., and A. Gopalan, "Algorithms for computing Maximally Redundant Trees for IP/LDP Fast- Reroute", draft-ietf-rtgwg-mrt-frr-algorithm-05 (work in progress), July 2015.

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

Authors' Addresses

Chris Bowers  
Juniper Networks  
1194 N. Mathilda Ave.  
Sunnyvale, CA 94089  
US

Email: cbowers@juniper.net

Pushpasis Sarkar  
Juniper Networks  
Embassy Business Park  
Bangalore, KA 560093  
India

Email: psarkar@juniper.net

Hannes Gredler  
Juniper Networks  
1194 N. Mathilda Ave.  
Sunnyvale, CA 94089  
US

Email: hannes@juniper.net

Uma Chunduri  
Ericsson Inc  
300 Holger Way  
San Jose, CA 95134  
US

Email: uma.chunduri@ericsson.com@

Network Working Group  
Internet-Draft  
Intended status: Informational  
Expires: September 6, 2019

C. Filsfils, Ed.  
S. Previdi  
Cisco Systems, Inc.  
G. Dawra, Ed.  
LinkedIn  
W. Henderickx  
Nokia  
D. Cooper  
Level 3  
March 5, 2019

Interconnecting Millions Of Endpoints With Segment Routing  
draft-filsfils-spring-large-scale-interconnect-13

Abstract

This document describes an application of Segment Routing to scale the network to support hundreds of thousands of network nodes, and tens of millions of physical underlay endpoints. This use-case can be applied to the interconnection of massive-scale DCs and/or large aggregation networks. Forwarding tables of midpoint and leaf nodes only require a few tens of thousands of entries. This may be achieved by inherent scaleable nature of Segment Routing and designed proposed in this document.

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 September 6, 2019.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

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

## Table of Contents

1. Introduction . . . . .	2
2. Terminology . . . . .	2
3. Reference Design . . . . .	3
4. Control Plane . . . . .	4
5. Illustration of the scale . . . . .	5
6. Design Options . . . . .	6
6.1. Segment Routing Global Block(SRGB) Size . . . . .	6
6.2. Redistribution of Agg nodes routes . . . . .	6
6.3. Sizing and hierarchy . . . . .	6
6.4. Local Segments to Hosts/Servers . . . . .	7
6.5. Compressed SRTE policies . . . . .	7
7. Deployment Model . . . . .	7
8. Benefits . . . . .	8
8.1. Simplified operations . . . . .	8
8.2. Inter-domain SLA . . . . .	8
8.3. Scale . . . . .	8
8.4. ECMP . . . . .	8
9. IANA Considerations . . . . .	8
10. Manageability Considerations . . . . .	9
11. Security Considerations . . . . .	9
12. Acknowledgements . . . . .	9
13. Contributors . . . . .	9
14. Informative References . . . . .	10
Authors' Addresses . . . . .	10

## 1. Introduction

This document describes how SR can be used to interconnect millions of endpoints. The following terminology is used in this document:

## 2. Terminology

The following terms and abbreviations are used in this document:



Term	Definition
Agg	Aggregation
BGP	Border Gateway Protocol
DC	Data Center
DCI	Data Center Interconnect
ECMP	Equal Cost MultiPathing
FIB	Forwarding Information Base
LDP	Label Distribution Protocol
LFIB	Label Forwarding Information Base
MPLS	Multi-Protocol Label Switching
PCE	Path Computation Element
PCEP	Path Computation Element Protocol
PW	Pseudowire
SLA	Service level Agreement
SR	Segment Routing
SRTE Policy	Segment Routing Traffic Engineering Policy
TE	Traffic Engineering
TI-LFA	Topology Independent - Loop Free Alternative

### 3. Reference Design

The network diagram here below describes the reference network topology used in this document:

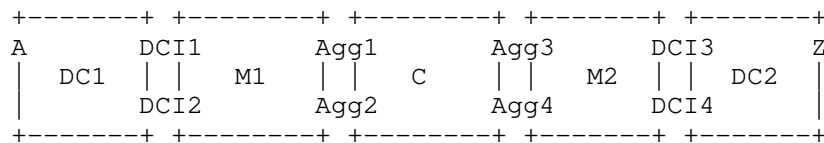


Figure 1: Reference Topology

The following applies to the reference topology above:

Independent ISIS-OSPF/SR instance in core (C) region.

Independent ISIS-OSPF/SR instance in Metro1 (M1) region.

Independent ISIS-OSPF/SR instance in Metro2 (M2) region.

BGP/SR in DC1.

BGP/SR in DC2.

Agg routes (Agg1, Agg2, Agg3, Agg4) are redistributed from C to M (M1 and M2) and from M to DC domains.

No other route is advertised or redistributed between regions.

The same homogeneous SRGB is used throughout the domains (e.g. 16000-23999).

Unique SRGB sub-ranges are allocated to each metro (M) and core (C) domains:

16000-16999 range is allocated to the core (C) domain/region.

17000-17999 range is allocated to the M1 domain/region.

18000-18999 range is allocated to the M2 domain/region.

Specifically, Agg1 router has SID 16001 allocated and Agg2 router has SID 16002 allocated.

Specifically, Agg3 router has SID 16003 allocated and the anycast SID for Agg3 and Agg4 is 16006.

Specifically, DCI3 router has SID 18003 allocated and the anycast SID for DCI3 and DCI4 is 18006.

Specifically, at Agg1 router Binding SID 4001 leads to DCI Pair DCI3, DCI4 via specific low-latency path {16002, 16003, 18006}.

The same SRGB sub-range is re-used within each DC (DC1 and DC2) region. for each DC: e.g. 20000-23999. Specifically, nodes A and Z both have SID 20001 allocated to them.

#### 4. Control Plane

This section provides a high-level description of a how a control plane could be implemented using protocol components already defined in other RFCs.

The mechanism through which SRTE Policies are defined, computed and programmed in the source nodes, are outside the scope of this document.

Typically, a controller or a service orchestration system programs node A with a pseudowire (PW) to a remote next-hop Z with a given SLA contract (e.g. low-latency path, be disjoint from a specific core plane, be disjoint from a different PW service, etc.).

Node A automatically detects that it does not have reachability to Z. It then automatically sends a PCEP request to an SR PCE for an SRTE policy that provides reachability to Z with the requested SLA.

The SR PCE [RFC4655] is made of two components. A multi-domain topology and a computation engine. The multi-domain topology is continuously refreshed through BGP-LS [RFC7752] feeds from each domain. The computing engine is designed to implement Traffic Engineering (TE) algorithms and provide output in SR Path format. Upon receiving the PCEP [RFC5440] request, the SR PCE computes the requested path. The path is expressed through a list of segments (e.g. {16003, 18006, 20001}) and provided to node A.

The SR PCE logs the request as a stateful query and hence is capable to recompute the path at each network topology change.

Node A receives the PCEP reply with the path (expressed as a segment list). Node A installs the received SRTE policy in the dataplane. Node A then automatically steers the PW into that SRTE policy.

## 5. Illustration of the scale

According to the reference topology described in Figure 1 the following assumptions are made:

There's 1 core domain and 100 leaf (metro) domains.

The core domain includes 200 nodes.

Two nodes connect each leaf (metro) domain. Each node connecting a leaf domain has a SID allocated. Each pair of nodes connecting a leaf domain also has a common anycast SID. This brings up to 300 prefix segments in total.

A core node connects only one leaf domain.

Each leaf domain has 6000 leaf node segments. Each leaf-node has 500 endpoints attached, thus 500 adjacency segments. In total, it is 3 millions endpoints for a leaf domain.

Based on the above, the network scaling numbers are as follows:

6,000 leaf node segments multiplied by 100 leaf domains: 600,000 nodes.

600,000 nodes multiplied by 500 endpoints: 300 millions of endpoints.

The node scaling numbers are as follows:

Leaf node segment scale: 6,000 leaf node segments + 300 core node segments + 500 adjacency segments = 6,800 segments

Core node segment scale: 6,000 leaf domain segments + 300 core domain segments = 6,300 segments

In the above calculations, the link adjacency segments are not taken into account. These are local segments and, typically, less than 100 per node.

It has to be noted that, depending on leaf node FIB capabilities, leaf domains could be split into multiple smaller domains. In the above example, the leaf domains could be split into 6 smaller domains so that each leaf node only need to learn 1000 leaf node segments + 300 core node segments + 500 adjacency segments which gives a total of 1,800 segments.

## 6. Design Options

This section describes multiple design options to the illustration of previous section.

### 6.1. Segment Routing Global Block (SRGB) Size

In the simplified illustrations of this document, we picked a small homogeneous SRGB range of 16000-23999. In practice, a large-scale design would use a bigger range such as 16000-80000, or even larger. Larger range provides allocations for various Traffic Engineering applications within a given domain

### 6.2. Redistribution of Agg nodes routes

The operator might choose to not redistribute the Agg nodes routes into the Metro/DC domains. In that case, more segments are required in order to express an inter-domain path.

For example, node A would use an SRTE Policy {DCI1, Agg1, Agg3, DCI3, Z} in order to reach Z instead of {Agg3, DCI3, Z} in the reference design.

### 6.3. Sizing and hierarchy

The operator is free to choose among a small number of larger leaf domains, a large number of small leaf domains or a mix of small and large core/leaf domains.

The operator is free to use a 2-tier design (Core/Metro) or a 3-tier (Core/Metro/DC).

#### 6.4. Local Segments to Hosts/Servers

Local segments can be programmed at any leaf node (e.g. node Z) in order to identify locally-attached hosts (or VM's). For example, if node Z has bound a local segment 40001 to a local host ZH1, then node A uses the following SRTE Policy in order to reach that host: {16006, 18006, 20001, 40001}. Such local segment could represent the NID (Network Interface Device) in the context of the SP access network, or VM in the context of the DC network.

#### 6.5. Compressed SRTE policies

As an example and according to Section 3, we assume node A can reach node Z (e.g., with a low-latency SLA contract) via the SRTE policy consisting of the path: Agg1, Agg2, Agg3, DCI3/4(anycast), Z. The path is represented by the segment list: {16001, 16002, 16003, 18006, 20001}.

It is clear that the control-plane solution can install an SRTE Policy {16002, 16003, 18006} at Agg1, collect the Binding SID allocated by Agg1 to that policy (e.g. 4001) and hence program node A with the compressed SRTE Policy {16001, 4001, 20001}.

From node A, 16001 leads to Agg1. Once at Agg1, 4001 leads to the DCI pair (DCI3, DCI4) via a specific low-latency path {16002, 16003, 18006}. Once at that DCI pair, 20001 leads to Z.

Binding SID's allocated to "intermediate" SRTE Policies allow to compress end-to-end SRTE Policies.

The segment list {16001, 4001, 20001} expresses the same path as {16001, 16002, 16003, 18006, 20001} but with 2 less segments.

The Binding SID also provides for an inherent churn protection.

When the core topology changes, the control-plane can update the low-latency SRTE Policy from Agg1 to the DCI pair to DC2 without updating the SRTE Policy from A to Z.

#### 7. Deployment Model

It is expected that this design be deployed as a green field but as well in interworking (brown field) with MPLS design across multiple domains.

## 8. Benefits

The design options illustrated in this document allow the interconnection on a very large scale. Millions of endpoints across different domains can be interconnected.

### 8.1. Simplified operations

Two protocols are not needed in this design: LDP and RSVP-TE. No new protocol has been introduced. The design leverages the core IP protocols: ISIS, OSPF, BGP, PCEP with straightforward SR extensions.

### 8.2. Inter-domain SLA

Fast reroute and resiliency is provided by TI-LFA with sub-50msec FRR upon Link/Node/SRLG failure. TI-LFA is described in [I-D.bashandy-rtgwg-segment-routing-ti-lfa].

The use of anycast SIDs also provides an improvement in availability and resiliency.

Inter-domain SLA's can be delivered, e.g., latency vs. cost optimized path, disjointness from backbone planes, disjointness from other services, disjointness between primary and backup paths.

Existing inter-domain solutions do not provide any support for SLA contracts. They just provide a best-effort reachability across domains.

### 8.3. Scale

In addition to having eliminated two control plane protocols, per-service midpoint states have also been removed from the network.

### 8.4. ECMP

Each policy (intra or inter-domain, with or without TE) is expressed as a list of segments. Since each segment is optimized for ECMP, then the entire policy is optimized for ECMP. The ECMP gain of anycast prefix segment should also be considered (e.g. 16001 load-shares across any gateway from M1 leaf domain to Core and 16002 load-shares across any gateway from Core to M1 leaf domain).

## 9. IANA Considerations

This document does not make any IANA request.

## 10. Manageability Considerations

This document describes an application 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] apply to the MPLS data plane when used with Segment Routing.

## 11. Security Considerations

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

## 12. Acknowledgements

We would like to thank Giles Heron, Alexander Preusche, Steve Braaten and Francis Ferguson for their contribution to the content of this document.

## 13. Contributors

The following people have substantially contributed to the editing of this document:

Dennis Cai  
Individual

Tim Laberge  
Individual

Steven Lin  
Google Inc.

Steven Lin  
Google Inc.

Bruno Decraene  
Orange

Luay Jalil  
Verizon

Jeff Tantsura  
Individual

Rob Shakir  
Google

## 14. Informative References

- [I-D.bashandy-rtgwg-segment-routing-ti-lfa]  
Bashandy, A., Filsfils, C., Decraene, B., Litkowski, S., Francois, P., daniel.voyer@bell.ca, d., Clad, F., and P. Camarillo, "Topology Independent Fast Reroute using Segment Routing", draft-bashandy-rtgwg-segment-routing-ti-lfa-05 (work in progress), October 2018.
- [RFC4655] Farrel, A., Vasseur, J., and J. Ash, "A Path Computation Element (PCE)-Based Architecture", RFC 4655, DOI 10.17487/RFC4655, August 2006, <<https://www.rfc-editor.org/info/rfc4655>>.
- [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>>.
- [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

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

Email: [cfilsfil@cisco.com](mailto:cfilsfil@cisco.com)

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

Email: [stefano@previdi.net](mailto:stefano@previdi.net)



Gaurav Dawra (editor)  
LinkedIn  
USA

Email: gdawra.ietf@gmail.com

Wim Henderickx  
Nokia  
Copernicuslaan 50  
Antwerp 2018  
Belgium

Email: wim.henderickx@nokia.com

Dave Cooper  
Level 3

Email: Dave.Cooper@Level3.com

Networking Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: April 16, 2016

L. Ginsberg  
P. Psenak  
S. Previdi  
Cisco Systems  
October 14, 2015

Segment Routing Conflict Resolution  
draft-ginsberg-spring-conflict-resolution-00.txt

Abstract

In support of Segment Routing (SR) routing protocols advertise a variety of identifiers used to define the segments which direct forwarding of packets. In cases where the information advertised by a given protocol instance is either internally inconsistent or conflicts with advertisements from another protocol instance a means of achieving consistent forwarding behavior in the network is required. This document defines the policies used to resolve these occurrences.

Requirements Language

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

Status of This Memo

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

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

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

This Internet-Draft will expire on April 16, 2016.

## Copyright Notice

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

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

This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

## Table of Contents

1. Introduction . . . . .	3
2. SR Global Block Inconsistency . . . . .	3
3. Segment Identifier Conflicts . . . . .	5
3.1. Conflict Types . . . . .	6
3.1.1. Prefix Conflict . . . . .	6
3.1.2. SID Conflict . . . . .	7
3.2. Processing conflicting entries . . . . .	8
3.2.1. Ignore conflicting entries . . . . .	8
3.2.2. Preference Algorithm . . . . .	8
3.2.3. Candidate Preference Algorithm . . . . .	9
3.2.4. Example Behavior . . . . .	9
3.2.5. Other Preference Factors to consider . . . . .	10
4. IANA Considerations . . . . .	11
5. Security Considerations . . . . .	11
6. Acknowledgements . . . . .	11
7. References . . . . .	11
7.1. Normative References . . . . .	11
7.2. Informational References . . . . .	12
Appendix A. Alternate Prefix Conflict Algorithm . . . . .	12

Authors' Addresses	14
--------------------	----

## 1. Introduction

Segment Routing (SR) as defined in [SR-ARCH] utilizes forwarding instructions called "segments" to direct packets through the network. Depending on the forwarding plane architecture in use, routing protocols advertise various identifiers which define the permissible values which can be used as segments, which values are assigned to specific prefixes, etc. Where segments have global scope it is necessary to have non-conflicting assignments - but given that the advertisements may originate from multiple nodes the possibility exists that advertisements may be received which are either internally inconsistent or conflicting with advertisements originated by other nodes. In such cases it is necessary to have consistent resolution of conflicts network-wide in order to avoid forwarding loops.

The problem to be addressed is protocol independent i.e., segment related advertisements may be originated by multiple nodes using different protocols and yet the conflict resolution **MUST** be the same on all nodes regardless of the protocol used to transport the advertisements.

The remainder of this document defines conflict resolution policies which meet these requirements. All protocols which support SR **MUST** adhere to the policies defined in this document.

## 2. SR Global Block Inconsistency

In support of an MPLS dataplane routing protocols advertise an SR Global Block (SRGB) which defines a set of label ranges reserved for use by the advertising node in support of SR. The details of how protocols advertise this information can be found in the protocol specific drafts e.g., [SR-OSPF] and [SR-IS-IS]. However the protocol independent semantics are illustrated by the following example:

The originating router advertises the following ranges:

```
Range 1: (100, 199)
Range 2: (1000, 1099)
Range 3: (500, 5990)
```

The receiving routers concatenate the ranges and build the Segment Routing Global Block (SRGB) as follows:

```
SRGB = (100, 199)
       (1000, 1099)
       (500, 599)
```

The indexes span multiple ranges:

```
index=0 means label 100
...
index 99 means label 199
index 100 means label 1000
index 199 means label 1099
...
index 200 means label 500
...
```

Note that the ranges are an ordered set - what labels are mapped to a given index depends on the placement of a given label range in the set of ranges advertised.

For the set of ranges to be usable the ranges MUST be disjoint. The question then arises what receiving routers should do if they receive an SRGB which includes overlapping ranges. In such a case the following rule is defined:

Each range is examined in the order it was advertised. If it does not overlap with any advertised range which preceded it the advertised range is used. If the range overlaps with any preceding range it MUST NOT be used and all ranges advertised after the first encountered overlapping range also MUST NOT be used.

Consider the following example:

The originating router advertises the following ranges:

Range 1: (100, 199]  
Range 2: (1000, 1099)  
Range 3: (100, 599)  
Range 4: (2000, 2099)

Range 3 overlaps with Range 1.  
Only Ranges #1 and #2 are usable.  
Ranges #3 and #4 are ignored.  
Note that Range #4 is not used even though it does not overlap with any of the other ranges.

### 3. Segment Identifier Conflicts

In support of an MPLS dataplane Segment identifiers (SIDs) are advertised and associated with a given prefix. SIDs may be advertised in the prefix reachability advertisements originated by a routing protocol. SIDs may also be advertised by a Segment Routing Mapping Server (SRMS).

A generalized mapping entry can be represented using the following definitions:

Pi - Initial prefix  
Pe - End prefix  
L - Prefix length  
Lx - Maximum prefix length (32 for IPv4, 128 for IPv6)  
Si - Initial SID value  
Se - End SID value  
R - Range value

Mapping Entry is then the tuple: (Pi/L, Si, R)  
 $Pe = (Pi + ((R-1) \ll (Lx-L)))$   
 $Se = Si + (R-1)$

Note that the SID advertised in a prefix reachability advertisement can be more generally represented as a mapping entry with a range of 1.

Conflicts in SID advertisements may occur as a result of misconfiguration. Conflicts may occur either in the set of advertisements originated by a single node or between advertisements originated by different nodes. When conflicts occur, it is not possible for routers to know which of the conflicting advertisements is "correct". If a router chooses to use one of the conflicting

entries forwarding loops and/or blackholes may result unless it can be guaranteed that all other routers in the network make the same choice. Making the same choice requires that all routers have identical sets of advertisements and that they all use the same selection algorithm.

### 3.1. Conflict Types

Various types of conflicts may occur.

#### 3.1.1. Prefix Conflict

When different SIDs are assigned to the same prefix we have a "prefix conflict". Consider the following set of advertisements:

```
(192.0.2.120/32, 200, 1)
(192.0.2.120/32, 30, 1)
```

The prefix 192.0.2.120/32 has been assigned two different SIDs - 200 by the first advertisement - 30 by the second advertisement.

Prefix conflicts may also occur as a result of overlapping prefix ranges. Consider the following set of advertisements:

```
(192.0.2.1/32, 200, 200)
(192.0.2.121/32, 30, 10)
```

Prefixes 192.0.2.121/32 - 192.0.2.130/32 are assigned two different SIDs - 320 through 329 by the first advertisement - 30 through 39 by the second advertisement.

The second example illustrates a complication - only part of the range advertised in the first advertisement is in conflict. It is logically possible to isolate the conflicting portion and try to use the non-conflicting portion(s) at the cost of increased implementation complexity. The algorithm defined here does NOT attempt to support use of a partial range.

A variant of the overlapping prefix range is a case where we have overlapping prefix ranges but no actual SID conflict.

```
(192.0.2.1/32, 200, 200)
(192.0.2.121/32, 320, 10)
```

Although there is prefix overlap between the two entries the same SID is assigned to all of the shared prefixes by the two entries. It is possible to utilize both entries but it complicates the implementation of the database required to support this. See

Appendix A for a more complete discussion of this case. An alternative is to ensure at the nodes which originate these advertisements that no such overlap is allowed to be configured. Such overlaps can then be considered as a conflict if they are received. This allows a simpler and more efficient implementation of the database. This is the approach assumed in this document.

Given two mapping entries:

(P1/L1, S1, R1) and (P2/L2, S2, R2)

a prefix conflict exists if all of the following are true:

- 1) The prefixes are in the same address family.
- 2)  $L1 == L2$
- 3)  $((P1 < P2) \ \&\& \ (P1e \geq P2)) \ || \ ((P2 < P1) \ \&\& \ (P2e \geq P1))$

### 3.1.2. SID Conflict

When the same SID has been assigned to multiple prefixes we have a "SID conflict". Consider the following example:

(192.0.2.1/32, 200, 1)  
(192.0.2.222/32, 200, 1)

SID 200 has been assigned to 192.0.2.1/32 by the first advertisement. The second advertisement assigns SID 200 to 192.0.2.222/32.

SID conflicts may also occur as a result of overlapping SID ranges. Consider the following set of advertisements:

(192.0.2.1/32, 200, 200)  
(192.1.2.1/32, 300, 10)

SIDs 300 - 309 have been assigned to two different prefixes. The first advertisement assigns these SIDs to 192.0.2.101/32 - 192.0.2.110/32. The second advertisement assigns these SIDs to 192.1.2.1/32 - 192.1.2.10/32.

The second example illustrates a complication - only part of the range advertised in the first advertisement is in conflict. It is logically possible to isolate the conflicting portion and try to use the non-conflicting portion(s) at the cost of increased implementation complexity. The algorithm defined here does NOT attempt to support use of a partial range.



SID conflicts are independent of address-family and independent of prefix len. A SID conflict occurs when a mapping entry which has previously been checked to have no prefix conflict assigns one or more SIDs that are assigned by another entry which also has no prefix conflicts.

### 3.2. Processing conflicting entries

Two general approaches can be used to process conflicting entries.

1. Conflicting entries can be ignored
2. A standard preference algorithm can be used to choose which of the conflicting entries will be used

The following sections discuss these two approaches in more detail.

Note: This document does not discuss any implementation details i.e. what type of data structure is used to store the entries (trie, radix tree, etc.) nor what type of keys may be used to perform lookups in the database.

#### 3.2.1. Ignore conflicting entries

In cases where entries are in conflict none of the conflicting entries are used i.e., the network operates as if the conflicting advertisements were not present.

Implementation requires identifying the conflicting entries and ensuring that they are not used. The occurrence of conflicts is easily diagnosed from the behavior of the network as the forwarding of traffic which would, in the absence of conflicts, utilize segments no longer does so. Which prefixes are impacted is easily seen and therefore the entries which are misconfigured are easily identified. Unintended traffic flow will never occur.

The downside of ignoring conflicting entries is that forwarding of all packets with destinations covered by the conflicting entries will always be negatively impacted.

#### 3.2.2. Preference Algorithm

For entries which are in conflict properties of the advertisement (e.g. prefix value, prefix length, SID value, etc.) are used to determine which of the conflicting entries are used in forwarding and which are ignored.

This approach requires that conflicting entries first be identified and then evaluated based on the preference rule. Based on which entry is preferred this in turn may impact what other entries are considered in conflict i.e. if A conflicts with B and B conflicts with C - it is possible that A does NOT conflict with C. Hence if as a result of the evaluation of the conflict between A and B, entry B is not used the conflict between B and C will not be considered.

As at least some of the traffic continues to be forwarded after the conflict is detected, the presence of the conflict may be harder to diagnose based on traffic flow than when using the ignore policy.

The upside of the preference algorithm is that in some cases forwarding of traffic may continue to be correct despite the existence of the conflict. If the preference algorithm happens to prefer the intended configuration traffic will still be successfully delivered. Whether this will occur is a random outcome since the preference algorithm cannot know which of the conflicting entries is the "correct" entry.

### 3.2.3. Candidate Preference Algorithm

The following algorithm is proposed. Evaluation is made in the order specified.

1. IPv4 entry wins over IPv6 entry
2. Smaller prefix length wins
3. Smaller starting address (considered as an unsigned integer value) wins
4. Smaller starting SID wins
5. Smaller range wins
6. non-attached entries preferred over attached entries (SRMS attached flag)

### 3.2.4. Example Behavior

Consider the following simple case. The following mapping entries exist:

1. (192.0.2.1/32, 100, 200)
2. (192.0.2.200/32, 150, 300) !Prefix conflict with entry #1

### 3. (193.3.3.3/32, 400, 100) !SID conflict with entry #2

Using the Ignore conflicts behavior we would not use any of the above entries.

Using a preference rule which favors smaller prefixes, entries #1 and #3 would be used.

- o Entry #1 would be used as 192.0.200.1 is less than 192.0.2.200.
- o Entry #3 would be used because once Entry #2 has been excluded, entry #3 no longer conflicts with any entry which is being used. (An example of lack of transitivity of conflicts)

If we now add

### 4. (192.0.1.1/32, 50, 100) ! Prefix conflict with #1

Using ignore policy still none of the entries would be used.

Using a preference rule which favors smaller prefixes , entries #4 and #2 would be used.

- o Entry #4 would be used in preference to entry #1 because 192.0.1.1 < 192.0.2.1.
- o Entry #2 would be used because once Entry #1 is excluded entry #2 no longer has a prefix conflict with any active entry.
- o Entry #3 would NOT be used because once Entry #2 becomes active entry #3 loses due to the SID conflict with Entry #2 since the latter has a smaller prefix.

### 3.2.5. Other Preference Factors to consider

Prefix to SID mapping is based on a variety of sources.

- o SIDs can be configured locally for prefixes assigned to interfaces on the router itself
- o SIDs can be received in prefix reachability advertisements from protocol peers. These advertisements may originate from peers local to the area or be leaked from other areas and/or redistributed from other routing protocols
- o SIDs can be received from SRMS advertisements - these advertisements can originate from routers local to the area or leaked from other areas

SIDs configured locally for prefixes associated with interfaces on the router itself are only used by the originating router in prefix advertisements - they are not installed in the forwarding plane locally. Therefore, they do not need to be considered in conflict resolution.

For other sources, It may seem intuitive to assign priority based on point of origination (e.g. intra-area preferred over inter-area, prefix reachability advertisements preferred over SRMS advertisements, etc.). However, any such policy makes it more likely that inconsistent choices will be made by routers in the network and increase the likelihood of forwarding loops or blackholes. The algorithms defined in this document assume that prefix reachability advertisements are part of the set of entries considered when determining conflicts and conflict resolution and no preference is associated with prefix reachability advertisements over SRMS advertisements.

It is common to use the identity of the advertising source router (e.g. router ID) as a tie breaker. However, in the case of SID advertisements it is possible that the source ID is not known. For example, when leaking SRMS advertisements the source ID may appear to be the Area Border Router (ABR) which performed the leaking. But this means that the relative preference of the SIDs associated with the leaked advertisements will have a different priority in different areas. Therefore router ID is not used in the algorithms discussed above.

#### 4. IANA Considerations

None.

#### 5. Security Considerations

TBD

#### 6. Acknowledgements

The authors would like to thank Martin Pilka for sharing his knowledge of algorithm implementation and complexity.

#### 7. References

##### 7.1. Normative References

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

## 7.2. Informational References

[SR-ARCH] "Segment Routing Architecture, draft-ietf-spring-segment-routing-05(work in progress)", September 2015.

[SR-IS-IS] "IS-IS Extensions for Segment Routing, draft-ietf-isis-segment-routing-extensions-05(work in progress)", June 2015.

[SR-OSPF] "OSPF Extensions for Segment Routing, draft-ietf-ospf-segment-routing-extensions-05(work in progress)", June 2015.

## Appendix A. Alternate Prefix Conflict Algorithm

It is possible when encountering overlapping prefix ranges to allow use of such entries when there is no actual SID conflict. This case can be avoided if configuration of such entries is blocked at the source - which allows a far simpler implementation when processing received entries. The latter model is what is described in the body of this document. What is described below is the algorithm and consequences of trying to use overlapping ranges when the SID assignments do not conflict.

The algorithm used to determine if two entries are in conflict is as follows.

Given two mapping entries:

(P1/L1, S1, R1) and (P2/L2, S2, R2)

a prefix conflict exists if all of the following are true:

- 1) The prefixes are in the same address family.
- 2)  $L1 == L2$
- 3)  $(P1 < P2) \ \&\& \ (P1e \geq P2) \ \&\& \ (((P2-P1) \gg (Lx-L)) + S1) \neq S2$   
 $\quad \quad \quad ||$   
 $\quad \quad \quad (P2 < P1) \ \&\& \ (P2e \geq P1) \ \&\& \ (((P1-P2) \gg (Lx-L)) + S2) \neq S1$

From an implementation standpoint, the complexity comes in supporting lookup of the SID for a given prefix in the presence of overlapping ranges which are in use. Consider the following simple example:

Entry #1: (192.0.2.1/32, 100, 250)

Entry #2: (192.0.2.101/32, 200, 10)

Entry #3: (192.0.2.201/32, 300, 100)

Using the conflict detection algorithm described in this section there is no conflict in the three ranges since all prefixes which overlap between the entries are assigned the same SID in all ranges.

If however we want to find the SID for the prefix 192.0.2.150, here are the steps one might go through:

1) Find the entry with the largest value of starting prefix which is less than or equal to 192.0.2.150. In the example this would be Entry #2 above.

2) Determine if the range covers 192.0.2.150. In the example the answer to this would be "no".

If we were not required to support overlapping entries at this point we would be done and conclude that no SID was assigned. But because we know that we may have overlapping entries we have to walk backwards (conceptually) and look at all of the entries with starting prefixes (of prefix length 32) which are less than 192.0.2.150 and check if their range is large enough to include that prefix. In the presence of a large # of entries this would be very slow. To avoid this problem we could build a local entry which contained all of overlapping entries (in this example all three of the entries shown) and use that in our active database to do the lookups. In this example we would generate:

(192.0.2.1/32, 100, 300)

Then we could use steps #1 and #2 above and be confident in the answer we get.

However, since we are no longer using the entries we received in our active database we would need to maintain a link between the generated entry and the set of overlapping entries we received which caused its generation so that if one of those entries was removed or modified we could properly update our active database.

Authors' Addresses

Les Ginsberg  
Cisco Systems  
510 McCarthy Blvd.  
Milpitas, CA 95035  
USA

Email: [ginsberg@cisco.com](mailto:ginsberg@cisco.com)

Peter Psenak  
Cisco Systems  
Apollo Business Center Mlynske nivy 43  
Bratislava 821 09  
Slovakia

Email: [ppsenak@cisco.com](mailto:ppsenak@cisco.com)

Stefano Previdi  
Cisco Systems  
Via Del Serafico 200  
Rome 0144  
Italy

Email: [sprevidi@cisco.com](mailto:sprevidi@cisco.com)

Network Working Group  
Internet-Draft  
Intended status: Informational  
Expires: March 7, 2016

Z. Li  
N. Wu  
Huawei  
September 4, 2015

Tunnel Segment in Segment Routing  
draft-li-spring-tunnel-segment-00

## Abstract

This document introduces a new type of segment, Tunnel Segment, for the segment routing (SR). Tunnel segment can be used to reduce SID stack depth of SR path, span the non-SR domain or provide differentiated services. Forwarding mechanisms and requirements of control plane and data models for tunnel segments are also defined.

## Requirements Language

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

## Status of This Memo

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

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

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

This Internet-Draft will expire on March 7, 2016.

## Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of



publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	2
2. Terminology . . . . .	2
3. Usecases . . . . .	3
3.1. Reducing SID Stack Depth . . . . .	3
3.2. Passing through Non-SR Domain . . . . .	4
3.3. Differentiated Services . . . . .	5
4. Comparison with Agency Segment . . . . .	6
5. Forwarding Mechanisms . . . . .	6
6. Requirement of Control Plane and Yang Models . . . . .	7
7. IANA Considerations . . . . .	7
8. Security Considerations . . . . .	8
9. References . . . . .	8
9.1. Normative References . . . . .	8
9.2. Informative References . . . . .	8
Authors' Addresses . . . . .	8

## 1. Introduction

Segment Routing (SR), introduced by [I-D.ietf-spring-segment-routing], leverages the source routing paradigm. A packet can be steered through an ordered list of instructions, which are also called segments. The node segment, adjacency segment, etc. have been proposed for different usecases.

This document introduces a new type of segment, Tunnel Segment, for the segment routing. Tunnel segment can be used to reduce SID stack depth of SR path, span the non-SR domain or provide differentiated services. Forwarding mechanisms and requirements of control plane and data models for tunnel segments are also defined.

## 2. Terminology

- o SID: Segment ID
- o SR: Segment Routing
- o SR Path: Segment Routing Path
- o SR-TE Path: Segment Routing Traffic Engineering Path

- o MSD: Maximally SID Depth

The terms "Tunnel Segment" and "Tunnel SID" are the generic names for a segment attached to a specific tunnel. A tunnel segment can be used to steer traffic into the corresponding tunnel along the SR path.

### 3. Usecases

#### 3.1. Reducing SID Stack Depth

It is possible that a SR path has to take an explicit path with multiple hops instead of the shortest path for the purpose of traffic engineering. As a result, the ingress node has to push lots of segments to steer the packet, which could be a challenge for the forwarding plane, since the depth of this segment stack may be beyond the capability of their forwarding engines. The tunnel segment introduced in this document will be helpful to mitigate the pain in these scenarios.

Taking Figure 1 below as an example, the SR-TE path is created from SR-Node-1(ingress) to SR-Node-2(egress). The original SID stack, {A, B, X, E, F, G, H, Y, J, K}, is too overwhelming for the path MSD. With help of the tunnel segment, the tunnel from Gateway-Node-1 to Gateway-Node-2 can be represented by a dedicated SID, saying Z. So the SR-TE path can be represented as {A, B, X, Z, J, K}. Comparing with the original SR-TE path, the SID stack depth is reduced.

The SR-TE tunnel can be created through two ways:

1. Manually configure on ingress node (Gateway-Node-1) and designate the SID binding to it. This binding relationship needs to be propagated to PCE/controller or advertised to other nodes in the network.
2. With the knowledge of all MSD along the path, a PCE/controller can calculate SR-TE tunnels using for reduce SID stack depth and determine ingress/egress gateway nodes dynamically. Those SR-TE tunnels can be created through PCE initiated style. The corresponding tunnel segment and the binding relationship can be propagated to ingress nodes and other nodes if necessary. As shown in Figure 1, ingress (SR-Node-1) can receive update messages from PCE/controller about the binding relationship. And SR-Node-1 can calculate the SR-TE path with the SR-TE tunnel segment without the help of PCE/controller in a centralized manner.

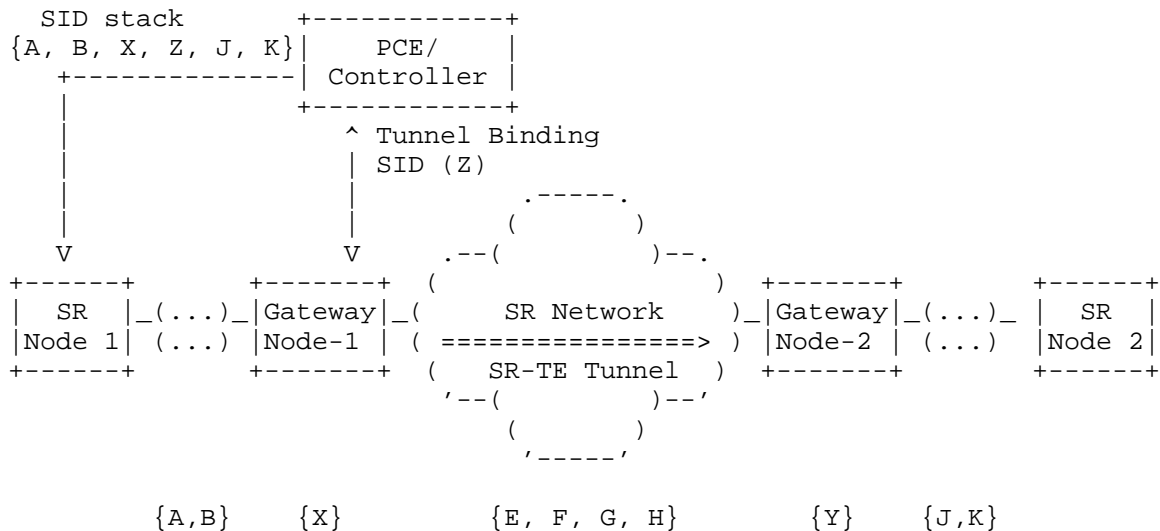


Figure 1 Usecase for Reducing SID Stack Depth

### 3.2. Passing through Non-SR Domain

The tunnel segment can also be used in those scenarios that traffic has to pass through non-SR domains. In another word, tunnel segment can be used to connect SR islands.

As shown in Figure 2, traffic from SR-Node-1 to SR-Node-2 has to pass through a traditional IP/MPLS network. Usually a RSVP-TE tunnel or IP tunnel will be created between two gateway nodes. By allocating SID for this tunnel, saying Z, the SR path from SR-Node-1 to SR-Node-2 can be represented as {A, B, X, Z, J, K}.

In this scenario, the RSVP-TE tunnel or IP tunnel can be involved into SR networks through two ways:

1. Manually configure on ingress node (Gateway-Node-1) and designate the SID binding to it. This binding relationship needs to be propagated to PCE/controller or advertised to other nodes in the network.
2. With the knowledge of topology of non-SR domain, a PCE/controller can calculate RSVP-TE tunnels or IP tunnels and determine ingress/egress gateway nodes dynamically. Those RSVP-TE tunnels or IP tunnels can be created through PCE initiated style. The corresponding tunnel segment and the binding relationship can be propagated to ingress nodes and other nodes if necessary. As shown in Figure 2, ingress (SR-Node-1) can receive update

messages from PCE/controller about the binding relationship. And SR-Node-1 can calculate the SR-TE path which can pass through non-SR domain without the help of PCE/controller in a centralized manner.

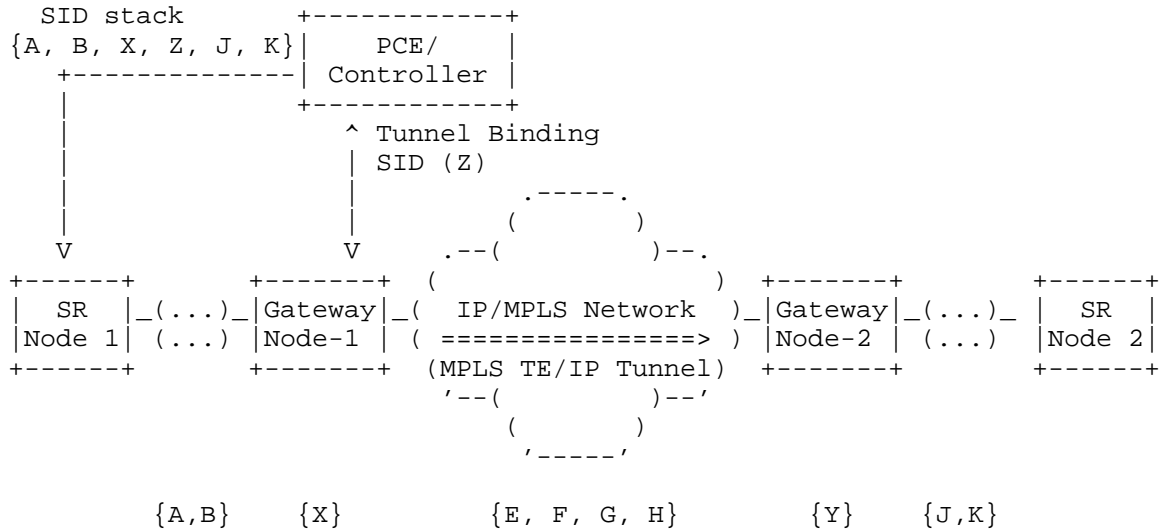


Figure 2 Usecase for Passing through Non-SR Domain

### 3.3. Differentiated Services

It is necessary to create multiple tunnels between the same pair of gateway nodes to support different services, since different tunnels can have different attributes. As a result, different SIDs have to be assigned per tunnel. Then an End-to-End SR path can choose different SIDs at ingress according to the service requirement when passing through the network between gateway nodes.

As depicted in Figure 3, two RSVP-TE tunnels, say RSVP-TE-tunnel1 and RSVP-TE-tunnel2, are created in MPLS network to provide different bandwidth guarantee services. And two SIDs, Z1 and Z2, are allocated and mapped to these two tunnels separately. These two SIDs can be utilized by a PCE/controller when defining the SR path at ingress. Since different traffic will transport through different tunnels, differentiated services can be guaranteed.

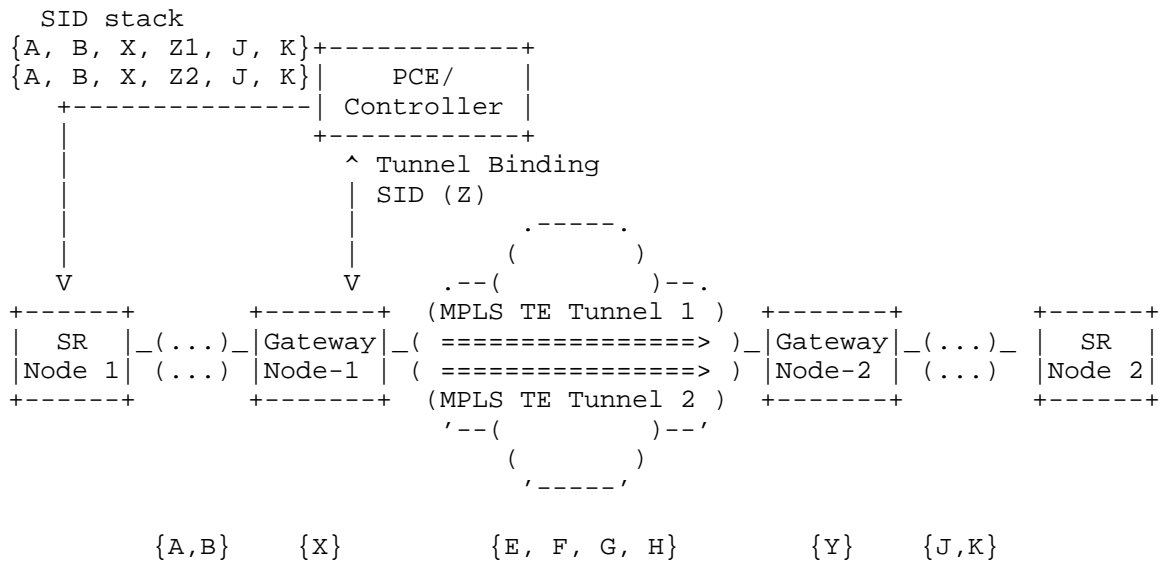


Figure 3 Usecase for Differentiated Services

#### 4. Comparison with Agency Segment

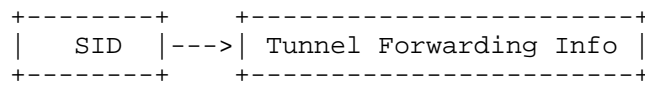
As described in [I-D.ietf-spring-segment-routing], a tunnel can be represented by an Adj-SID or as a Forwarding Adjacency. One obvious benefit of the method is to unify the process. But it may be necessary to differentiate a tunnel segment from other adjacency segment in some scenarios since there are more attributes attached to a tunnel.

By introducing the tunnel segment, this document expects not only to inform the binding relationship between a tunnel and a SID but also to learn tunnel information as much as possible. For example, it will be helpful for SR-capable nodes to know the detail of an explicit path that passes through non-SR networks.

In addition, one tunnel will need an IP address if handled as an adjacency (a borrowed IP address at least). While a tunnel binding to a Tunnel-SID does not have to contain an IP address, only an ingress node and an egress node is enough.

#### 5. Forwarding Mechanisms

In the gateway node, when received the packet with the tunnel segment SID as the topmost SID, it will use the forwarding mechanism shown in the following figure to steering the traffic to the corresponding tunnel.



SID: Segment ID

Figure 4 Forwarding Mechanisms for Tunnel Segment

## 6. Requirement of Control Plane and Yang Models

According to the procedures of the above usecases, following requirements of control plane and Yang models for Tunnel Segment are proposed:

- o REQ 01: IGP extensions SHOULD be introduced to advertise the binding relationship between a SID/label and the corresponding tunnel. Attributes of the tunnel MAY be carried optionally.
- o REQ 02: BGP Link-State extension SHOULD be introduced to advertise the binding relationship between a label and the corresponding tunnel. Attributes of the tunnel MAY be carried optionally.
- o REQ 03: PCEP extensions SHOULD be introduced to advertise the binding relationship between a SID/label and the corresponding tunnel from a PCC to a PCE. Attributes of the tunnel MAY be carried optionally.
- o REQ 04: PCE SHOULD support initiated IP tunnel.
- o REQ 05: PCE SHOULD support to allocate SID/label for the corresponding tunnel dynamically.
- o REQ 06: PCEP extensions SHOULD be introduced to distribute the binding relationship between a SID/label and the corresponding tunnel from a PCE to a PCC. Attributes of the tunnel MAY be carried optionally.
- o REQ 07: An I2RS interface SHOULD be available for allocating SID/label to the corresponding tunnel. And augmentation on segment routing YANG models SHOULD be introduced.

## 7. IANA Considerations

This document makes no request of IANA.

## 8. Security Considerations

This document does not introduce new security threat.

## 9. References

### 9.1. Normative References

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

### 9.2. Informative References

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

## Authors' Addresses

Zhenbin Li  
Huawei  
Huawei Bld., No.156 Beiqing Rd.  
Beijing 100095  
China

Email: [lizhenbin@huawei.com](mailto:lizhenbin@huawei.com)

Nan Wu  
Huawei  
Huawei Bld., No.156 Beiqing Rd.  
Beijing 100095  
China

Email: [eric.wu@huawei.com](mailto:eric.wu@huawei.com)

SPRING Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: April 11, 2016

P. Sarkar, Ed.  
Juniper Networks, Inc.  
H. Gredler  
Individual Contributor  
C. Filsfils  
S. Previdi  
Cisco Systems, Inc.  
B. Decraene  
Orange  
M. Horneffer  
Deutsche Telekom  
October 9, 2015

Anycast Segments in MPLS based Segment Routing  
draft-psarkar-spring-mpls-anycast-segments-01

Abstract

Instead of forwarding to a specific device or to all devices in a group, anycast addresses, let network devices forward a packet to (or steer it through) one or more topologically nearest devices in a specific group of network devices. The use of anycast addresses has been extended to the Segment Routing (SR) network, wherein a group of SR-capable devices can represent a anycast address, by having the same Segment Routing Global Block (SRGB) provisioned on all the devices and each one of them advertising the same anycast prefix segment (or Anycast SID).

This document describes a proposal for implementing anycast prefix segments in a MPLS-based SR network, without the need to have the same SRGB block (label ranges) provisioned across all the member devices in the group. Each node can be provisioned with a separate SRGB from the label range supported by the specific hardware platform.

Requirements Language

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

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute



working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on April 11, 2016.

#### Copyright Notice

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

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

#### Table of Contents

1. Introduction . . . . .	3
2. Problem Statement . . . . .	4
3. Solution . . . . .	7
3.1. Definitions . . . . .	7
3.1.1. Common Anycast SRGB (CA-SRGB) . . . . .	7
3.1.2. Common Anycast Prefix Segment Label (CAPSL) . . . . .	8
3.1.3. Anycast Prefix Segment Label (APSL) . . . . .	8
3.2. Procedures . . . . .	9
3.2.1. Label Stack Computation . . . . .	9
3.2.2. Virtual SID Label Lookup Table . . . . .	10
3.2.3. Advertising Anycast Prefix Segments . . . . .	13
3.2.4. Programming Anycast Prefix Segments . . . . .	14
3.3. Packet Flow . . . . .	16
4. Acknowledgements . . . . .	17
5. IANA Considerations . . . . .	17
6. Security Considerations . . . . .	17
7. References . . . . .	17
7.1. Normative References . . . . .	17
7.2. Informative References . . . . .	17
Authors' Addresses . . . . .	18

## 1. Introduction

Anycast is a network addressing scheme and routing methodology in which packets from a single source device are forwarded to the topologically nearest node in a group of potential receiving devices, all identified by the same anycast address. There are various useful usecases of anycast addresses, and discussion of the same are outside the scope of this document.

[I-D.ietf-spring-segment-routing] extended the use of anycast addresses to SR networks. An operator may combine a group of SR-enabled nodes to form a anycast group, by picking a anycast address and a segment identifier (hereon referred to as SID) to represent the group, and then provisioning all the nodes with the same address and SID. Once provisioned, each device in the group advertises the corresponding anycast address in its IGP link-state advertisements along with the SID provisioned. Source devices on receiving such anycast prefix segment advertisements, finds out the topologically nearest device that originated the anycast segment and forwards packets destined to the same on the shortest-path to the nearest device.

[I-D.ietf-spring-segment-routing] requires all devices in a given anycast group to implement the exact same SRGB block(s). This requirement will always be met in SR network deployed over IPV6 forwarding plane [I-D.previdi-6man-segment-routing-header]. For SR over MPLS dataplane [I-D.ietf-spring-segment-routing-mpls], while this is a simple (and hence more desirable) solution, the same may not be possible in a multi-vendor networks deploying devices with varying hardware capabilities.

In MPLS-based SR deployments, the segments on a given source router are actually mapped to a MPLS labels allocated from the local label pool carved out by the device for accomodating the SRGB. In multi-vendor deployments with various types of devices deployed in the same network topology, such a anycast group may contain a good combination of devices from different vendors and have different internal hardware capabilities. In such environments it is not sufficient to assume that all the devices in a anycast group will be able to allocate exactly the same range of labels for implementing the SRGB. In reality, getting a common range of labels among all the various vendors may not be feasible.

This documents provides mechanisms to implement anycast segments with any kind of device in a multi-vendor network deployment without requiring to provision the same exact range of labels for SRGB on all the devices.

## 2. Problem Statement

To better illustrate the problem let us consider an example topology using anycast segments as shown in Figure 1 below.

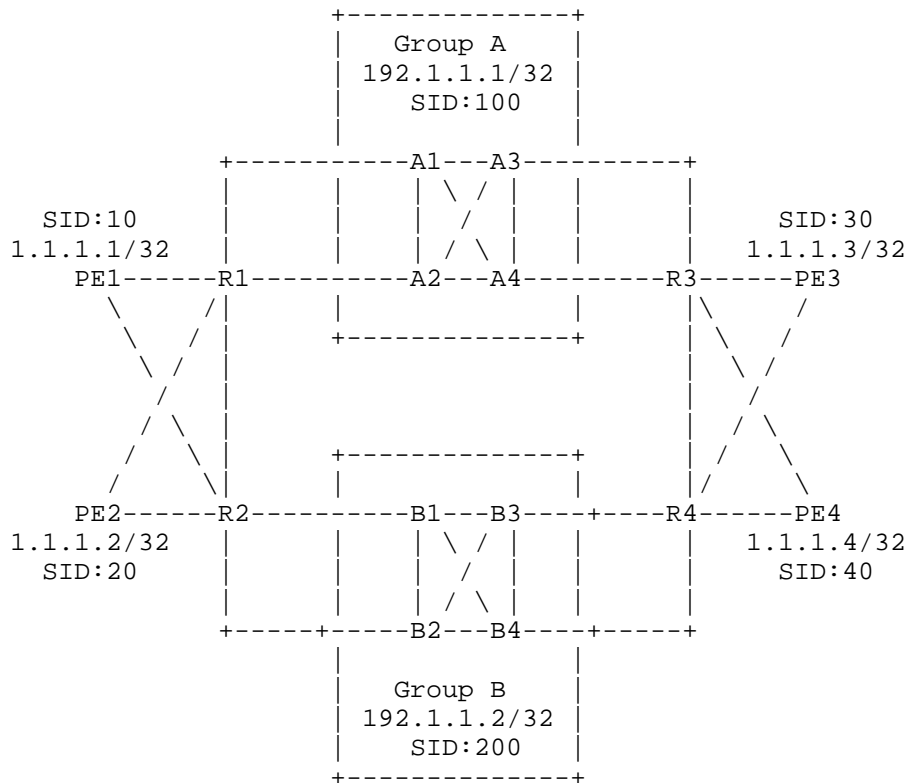


Figure 1: Topology 1

In Figure 1 above, there are two groups of transit devices. Group A consists of devices {A1, A2, A3 and A4}. They are all provisioned with the anycast address 192.1.1.1/32 and the anycast SID 100. Similarly, group B consists of devices {B1, B2, B3 and B4} and are all provisioned with the anycast address 192.1.1.2/32, anycast SID 200. In the above network topology, each PE device is connected to two routers in each of the groups A and B.

Following are all the possible ECMP paths between the various pairs of PE devices.

- o P1: via {R1, A1, A3, R3}
- o P2: via {R1, A1, A4, R3}
- o P3: via {R1, A2, A3, R3}
- o P4: via {R1, A2, A4, R3}
- o P5: via {R2, B1, B3, R4}
- o P6: via {R2, B1, B4, R4}
- o P7: via {R2, B2, B3, R4}
- o P8: via {R2, B2, B4, R4}

As seen above, there is always eight ECMP paths between each of pair of PE devices. The network operator may not wish to utilize all possible ECMP paths for all possible types of traffic flowing between a given pair of PE devices. It may be more useful for use paths P1, P2, P3 and P4 for certain types of traffic and use paths P5, P6, P7 and P8 for all other types of traffic between the same PE devices. If so desired, operators may use these anycast groups A and B and the corresponding anycast segment to impose a segment-list (refer to [I-D.ietf-spring-segment-routing]) to forward the respective traffic flows over the desired specific paths as shown below. Figure 2 below depicts an expanded view of the paths via group A. The range labels allocated for SRGB on each of the devices in group A are also mentioned in this diagram.

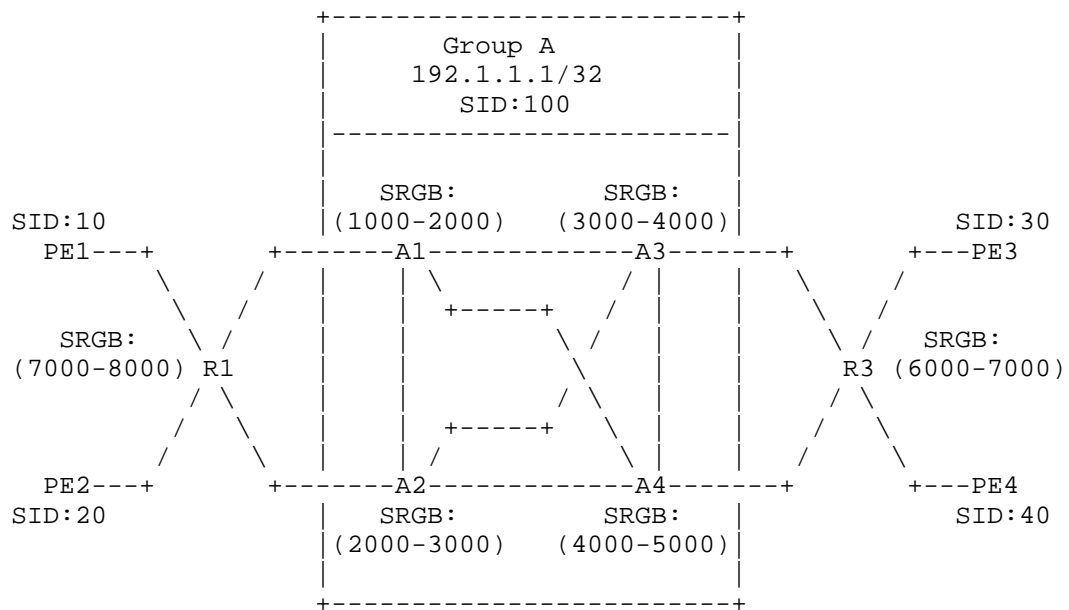


Figure 2: Transit paths via anycast group A

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 a MPLS payload with the following stack of labels.

- o Label allocated R1 for anycast SID 100 (outer label)
- o Label allocated by the nearest router in group A for SID 30 (for destination PE3)

While the first label is easy to compute, in this case since there are more than one topologically nearest devices (A1 and A2), unless A1 and A2 implement same exact SRGB, determining the second label is impossible. In all likeness, devices A1 and A2 may be devices from different hardware vendors and it may not implement the same exact SRGB label ranges. In such cases, separate labels are allocated by A1 and A2 (1030 and 2030 respectively, in the above example). 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.

### 3. Solution

#### 3.1. Definitions

##### 3.1.1. Common Anycast SRGB (CA-SRGB)

This document introduces the term 'Common-Anycast SRGB' (hereafter referred to as the CA-SRGB) to define the SRGB implemented by the majority of the devices in the network, that are participating in one or more anycast segments. Each device MUST implement provisions to let the operators assign the CA-SRGB on the device. Each vendor implementation MUST implement provisions to configure the CA-SRGB at all configuration levels (per-routing-instance/per-protocol/per-topology etc) wherein provisions to configure the local SRGB label ranges has also been implemented. Essentially, for each SRGB configured on the device, vendor implementations MUST allow configuring a corresponding CA-SRGB value.

For each configuration level (per-routing-instance/per-protocol/per-topology etc) supported, the operator MUST set the same exact CA-SRGB on all devices across the entire IGP domain (including different IS-IS levels and OSPF areas). This ensures the proposal specified in Section 3.2.1 works flawlessly across all devices in any multi-vendor network deployment.

However assigning the CA-SRGB (for a given routing-instance/protocol/topology etc.) on the device, does not mean the label ranges allocated by the device for the corresponding SRGB has to belong to the CA-SRGB defined. The device may dynamically allocate the corresponding SRGB label ranges, or allocate the range provisioned by the operator, through an appropriate separate configuration (please refer to [I-D.ietf-spring-sr-yang] for more details).

For devices that has the local SRGB to be exact same as the 'CA-SRGB' applicable for the entire network, operators need not explicitly set the corresponding CA-SRGB values. In such case, the vendor implementations MUST assume the local SRGB values to be the corresponding CA-SRGB values defined on the specific device.

If the CA-SRGB defined on a device does not absolutely match the corresponding SRGB label ranges allocated (or provisioned) on the same device (i.e. the CA-SRGB is not an exact copy of the corresponding SRGB label ranges), and the device is provisioned with one or more anycast prefix segments, the device MUST implement all the additional functionalities specified in Section 3.2.2, Section 3.2.3 and Section 3.2.4. On devices, where the SRGB label ranges is an exact copy of the corresponding CA-SRGB defined, the

device need not implement these additional functionalities (Section 3.2.2, Section 3.2.3 and Section 3.2.4).

### 3.1.2. Common Anycast Prefix Segment Label (CAPSL)

For each anycast prefix segment, this document also defines a 'Common Anycast Prefix Segment Label' (hereafter referred to CAPSL). The value of this label is derived by applying the SID index associated with the prefix segment as an offset to the CA-SRGB configured on the specific device. Since the operator MUST configure the same CA-SRGB values on all devices in the IGP domain, all devices shall associate the same CAPSL label value for a given anycast prefix segment. Table 1 below shows the CAPSL labels allocated by any device for the various prefix segments found in Figure 2, with CA-SRGB set to 3000-4000 on all devices.

SID	CA-SRGB Range	CAPSL Value
10	2000-3000	2010
20	2000-3000	2020
30	2000-3000	2030
40	2000-3000	2040
100	2000-3000	2100

Table 1: Common Anycast Prefix Segment Label Allocation

### 3.1.3. Anycast Prefix Segment Label (APSL)

This document also introduces the term 'Anycast Prefix Segment Label' (hereafter referred to as APSL) to define the label allocated by a device to advertise reachability for the specific anycast prefix segment. The value of this label is derived by applying the SID index associated with the anycast prefix segment as an offset to the SRGB of the specific device. Table 2 below shows the labels allocated by the various devices in Figure 2 for the anycast prefix segment with SID 100.

Anycast-SID	Device	SRGB	APSL-Label
100	R1	7000-8000	7100
100	A1	1000-2000	1100
100	A2	2000-3000	2100
100	A3	3000-4000	3100
100	A4	4000-5000	4100
100	R3	6000-7000	6100

Table 2: Anycast Prefix Segment Label Allocation

### 3.2. Procedures

#### 3.2.1. Label Stack Computation

A MPLS device that tries to encapsulate any kind of traffic into a SR-based MPLS payload (hereafter referred to as the ingress device) and steer it through a series of SR adjacency and/or unicast/anycast prefix segments, needs to compute an appropriate stack of MPLS labels and put it in the outgoing packet. Alternatively, in a SDN environment, the SDN controller may need to compute the label stack and install it on the ingress device.

However in both cases, as illustrated in Section 2, for a given ingress device (e.g. PE1 or PE2), there maybe multiple topologically nearest devices in a specific anycast group (e.g. A1 and A2), even through there is only out-going link from the source device(e.g. PE1->R1 or PE2-R1). In such case, when the ingress device (or the SDN controller) wants to steer a packet through the anycast group A, it can use the anycast segment label advertised by the downstream neighbor of the ingress device for the specific anycast prefix segment. Since the packet may reach any one of the multiple devices in the group and each of them may have a separate SRGB label range, choosing the MPLS label for the next segment providing reachability to the final destination. Also, since the packet steered through a anycast segment can reach of any of the member device in the anycast group, it is sufficient to assume that the ingress (or the controller) cannot place an adjacency segment immediately after a anycast segment in the outgoing packet.

This document proposes the ingress device (or the SDN controller) to derive the label for a prefix segment that immediately follows a given anycast segment, to be the CAPSL label associated with the corresponding SID index (refer to Section 3.1.2). Note the prefix segment immediately following the given anycast segment may itself be another anycast segment.



The ingress (or the SDN controller) MUST follow the algorithm below to compute the label-stack, that it must use to steer a packet through a list of SR segments.

- o Set previous\_segment ==> NONE.
- o Set label\_stack ==> {EMPTY}.
- o For [all segment in Segment\_List]
  - \* If {segment.type == Adjacency\_Segment}
    - + Set label ==> segment.Adjacency\_Segment\_Label.
  - \* Else
    - + If {previous\_segment.type == Anycast\_Prefix\_Segment}
      - Set label ==> CAPSL\_Label(segment.SID\_index).
    - + Else
      - Set label ==> segment.Prefix\_Segment\_Label.
  - \* Add label to label\_stack.
  - \* Set previous\_segment ==> segment.

### 3.2.2. Virtual SID Label Lookup Table

When a MPLS packet on the wire first hits a device, the forwarding hardware reads the topmost label in the MPLS header and looks up the default label lookup table associated with the interface on which the label has been received. This table is generally called LFIB. The range of labels found in the LFIB constitutes the default label space.

This document introduces a separate virtual label lookup table (hereafter referred to as Virtual LFIB or V-LFIB), that represents a label space which is also separate from the actual label space represented by the default LFIB. The label value may be present in both the default and Virtual LFIB. However the forwarding semantics associated with the label under the default and Virtual LFIB may not be same. Following are the fields of a typical entry of this table.

- o CAPSL-Label: The CAPSL label value derived from the SID index associated with a given prefix segment originated by another

device in the same network. Refer to Section 3.1.2 for more details. This is also the key field for this table.

- o Forwarding Semantics: This is once again one or more tuples of following items.
  - \* Outgoing-Label: The label(s) allocated by the neighbor device(s) on the shortest-path to the topologically nearest originator(s) of the prefix segment.
  - \* Outgoing-link: The link(s) connecting the device to the neighbor device(s) on the shortest path to the topologically nearest originator(s) of the prefix segment.

This document proposes that, any device, when provisioned with one or more anycast prefix segment (address and SID), and the CA-SRGB defined by the operator is not an exact copy of the corresponding SRGB label ranges allocated by the device, it MUST create a Virtual LFIB table.

Such a device MUST add an entry in the Virtual LFIB for each unicast and anycast prefix segments learnt from a remote device, if and only if the same prefix has not been provisioned on the device. The device SHOULD NOT add an entry for any of the Anycast or Node prefix segments that it has advertised itself. However if the device has learnt any anycast prefix segment from a remote device, and the same is not provisioned on this device, the device MUST include the same in the Virtual LFIB table.

In cases where a prefix segment is reachable via multiple shortest paths on a given device, the corresponding entry for the prefix SID MUST have as many forwarding entries in the Virtual LFIB table as the number of shortest-paths found for the corresponding prefix on the device.

Figure 3 below shows how the Virtual LFIB table on each of devices in group A should look like. Please note that some of the prefix segments has multiple forwarding semantics associated with them. For example, on device A1, the prefix SID 10 (originated by PE3) is reachable through its neighbors A3 and A4. And as per the SRGB advertised by A3 and A4, the labels allocated by A3 and A4 are 3030 and 4030 respectively. Hence A1 has added two forwarding entries for the prefix SID 30 in its Virtual LFIB table.

CA-SRGB configured on all devices: {2000-3000}			
Device	CAPSL-Label	Forwarding Semantics	
		Outgoing-Label	Outgoing-Link
A1	2010	7010	A1->R1
	2020	7020	A1->R1
	2030	3030	A1->A3
		4030	A1->A4
	2040	3040	A1->A3
		4040	A1->A4
A2	No V-LFIB Table created since CA-SRGB is identical to SRGB allocated locally		
A3	2010	1010	A3->A1
		2010	A3->A2
	2020	1020	A3->A1
		2020	A3->A2
	2030	6030	A3->R3
	2040	6040	A3->R3
A4	2010	1010	A4->A1
		2010	A4->A2
	2020	1020	A4->A1
		2020	A4->A2
	2030	6030	A4->R3
	2040	6040	A4->R3

Figure 3: Virtual LFIB Table Setup

Please note that node A2 has not created a Virtual LFIB table since the CA-SRGB (2000-3000) is identical to the SRGB provisioned on it.

Also please note that none of the devices in the anycast group have included the anycast SID 100 in the Virtual LFIB table, since the same has already been provisioned on these devices.

When a device receives a MPLS packet with the anycast segment label associated with one of the anycast prefix segments provisioned on the same device, and the CA-SRGB defined by the operator is not an exact copy of the corresponding SRGB label ranges allocated by it, it MUST use the Virtual LFIB table to lookup the next label that follows the anycast segment label in the stack of labels found in the MPLS header. Refer to Section 3.2.4 for more details.

Following forwarding instructions MUST be installed in the MPLS data-plane for each entry in the Virtual LFIB entry.

- o If the label at the top of the stack matches any of the prefix SIDs in the Virtual LFIB table,
  - \* If there are multiple forwarding tuples associated with matching table entry,
    - + Select one forwarding tuple. (Criteria to select one is outside the scope of this document.)
  - \* Else,
    - + Select the single forwarding tuple available.
  - \* Replace the next label (should be a CAPSL label) found at top of the MPLS label stack in the incoming packet, with the 'Outgoing-label' from the selected forwarding tuple.
  - \* Forward the modified packet onto the 'Outgoing-link' as specified in the selected forwarding tuple.
  - \* If the prefix SID is another anycast segment,
    - + Ensure the next label lookup is launched again on the Virtual LFIB table.
  - \* Else,
    - + Ensure the next label lookup is launched on the default LFIB table.

### 3.2.3. Advertising Anycast Prefix Segments

Like unicast prefix segments, anycast prefix segments SHOULD be advertised in IGP Link-state advertisements using IGP protocol extension for SR specified in [I-D.ietf-isis-segment-routing-extensions], [I-D.ietf-ospf-segment-routing-extensions] and

[I-D.ietf-ospf-ospfv3-segment-routing-extensions]. This document does not propose any protocol extension for advertising anycast prefix segments.

However when advertising the anycast segments, and the CA-SRGB defined by the operator is not an exact copy of the corresponding SRGB label ranges allocated by the originating device, it MUST set the corresponding P-Flag(No-PHP) in ISIS Prefix-SID SubTLV and/or the NP-Flag (No-PHP) in OSPFv2 and OSPFv3 Prefix-SID SubTLV to 1 and the E-Flag in the same SubTLVs to 0. Please refer to following for more details on usage of these flags.

- o ISIS Prefix-SID SubTLV [I-D.ietf-isis-segment-routing-extensions]
- o OSPFv2 Prefix-SID SubTLV [I-D.ietf-ospf-segment-routing-extensions]
- o OSPFv3 Prefix-SID SubTLV [I-D.ietf-ospf-ospfv3-segment-routing-extensions]

The proposal above, ensures that a MPLS packet sent to (or taking transit through) a given anycast group, when reaching at a topologically nearest device in the group where CA-SRGB does not match SRGB provisioned on it, always arrives with the APSL-label that is derived from the device's SRGB, and the SID associated with the corresponding anycast prefix segment. Note in the above topology, assuming domain-wide CA-SRGB is set to (2000-3000) on all nodes, while nodes A1, A3 and A4 will advertise the SID 100 with P-Flag(No-PHP) set to 1, node A2 will advertise the same anycast prefix SID with P-Flag unset. This is because on node A1 the domain-wide CA-SRGB is identical to the local SRGB provisioned on A2.

In Figure 2, when PE1 or PE2 intends to steer a packet destined for PE3 or PE4, through the anycast group A (SID 100), it needs to forward the packet to R1 (SRGB:7000-8000), after putting the label 7100 (derived from R1's SRGB), at top of the label stack in the MPLS header. However when the same packet is forwarded to A1 (one of the topologically nearest devices in group A), R1 shall not POP (or remove) the label 7100. Instead R1 shall replace it with the label 1100 while forwarding to A1. While forwarding to A2, since A2 would have advertised the anycast SID 100 with P-Flag (No-PHP) unset, R1 shall POP the incoming label 7100 before forwarding it to R1.

#### 3.2.4. Programming Anycast Prefix Segments

The proposal specified in Section 3.2.3, ensures that a MPLS packet destined to (or steered via) a anycast prefix segment always arrives at the nearest device in the anycast group with a label derived from

the device's SRGB and the SID associated with the corresponding anycast prefix segment, as the top-most label label stack in its MPLS header. If this label is also the bottom-most label ( $S=1$ ), it means packet has been destined to the anycast segment, and should be consumed by the local device. If the label is not the bottom-most label ( $S=0$ ), the packet must be forwarded to the next segment, for which the next label in the stack should be consulted. However Section 3.2.1 specifies that the next label in such case, shall be a label belonging to the CA-SRGB defined by the operator, derived from the SID associated with the next segment. Since the CAPSL label for the SID index associated with a prefix segment may directly collide with another label in the default LFIB table, Section 3.2.2 also proposed to have a Virtual LFIB table to provide a separate label-space for looking up the next label.

This document specifies that a device provisioned with a given prefix segment index MUST implement following forwarding semantics for the anycast segment label (refer to Section 3.1.3) associated with the anycast prefix segment, if the CA-SRGB label ranges defined is not an exact copy of the corresponding SRGB label range(s) locally allocated/provisioned on the device.

- o If the label at the top the stack is a anycast segment label, and the CA-SRGB defined is not an exact copy of the corresponding SRGB label range(s),
  - \* Pop the label.
  - \* If bottom-most label in the stack ( $S=1$ ),
    - + Send it to host stack for local consumption, as usual.
  - \* Else if not the bottom-most label in the stack ( $S=0$ ),
    - + Set the Virtual LFIB table as the lookup table for the next label lookup.
    - + Launch a lookup for the next label in the stack (should be a CAPSL label).
- o Else
  - \* Lookup the label in the default LFIB table as usual.

## 3.3. Packet Flow

Figure 4 below illustrate how SR-based MPLS packets destined for PE3 and sourced by PE1 are expected to flow through when PE1 encapsulates the packet with an appropriate label stack to steer it through group A devices only

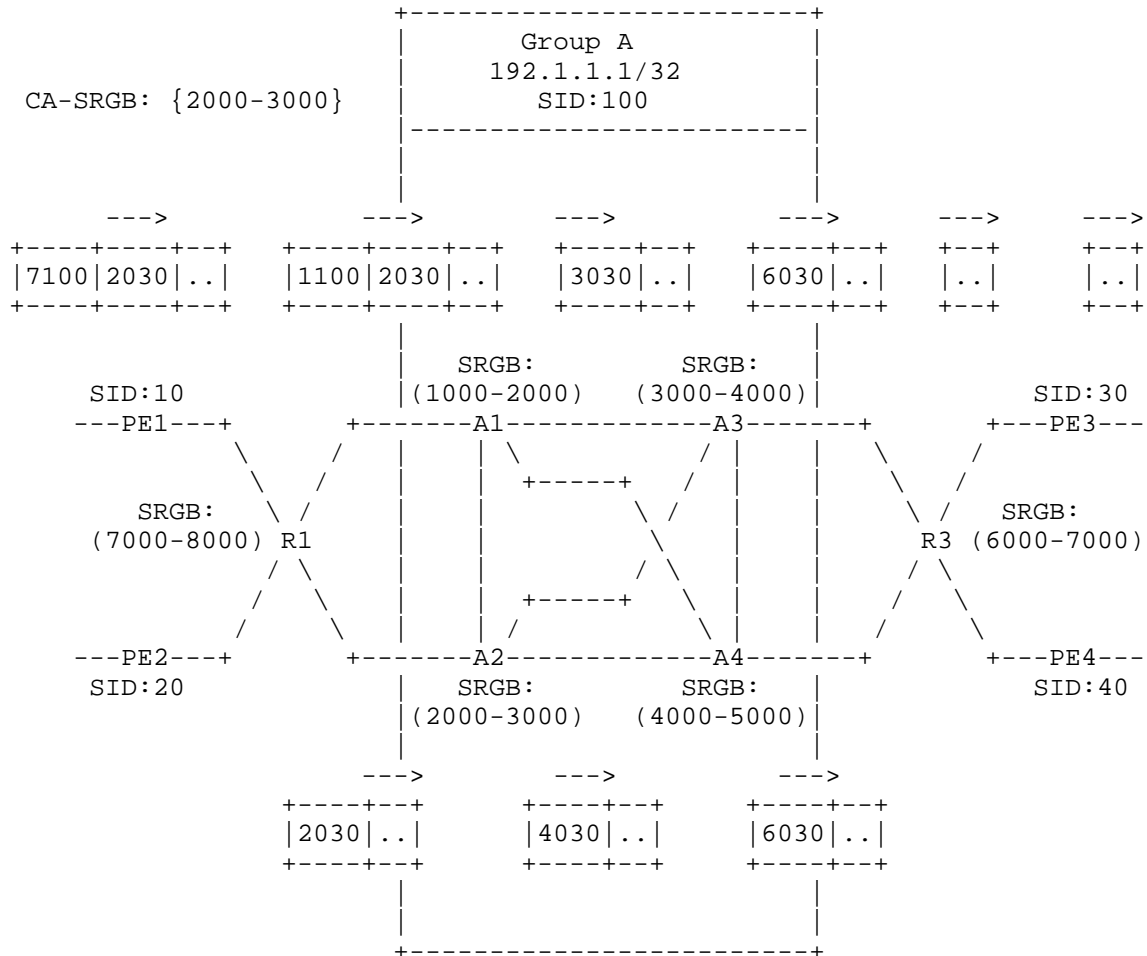


Figure 4: Packet Flow through MPLS-based SR Anycast Segments

#### 4. Acknowledgements

Many many thanks to Shraddha Hegde, Eric Rosen, Chris Bowers and Stephane Litkowski for their valuable inputs.

#### 5. IANA Considerations

N/A. - No protocol changes are proposed in this document.

#### 6. Security Considerations

This document does not introduce any change in any of the protocol specifications.

#### 7. References

##### 7.1. Normative References

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

##### 7.2. Informative References

[I-D.ietf-isis-segment-routing-extensions]  
Previdi, S., Filsfils, C., Bashandy, A., Gredler, H., Litkowski, S., Decraene, B., and J. Tantsura, "IS-IS Extensions for Segment Routing", draft-ietf-isis-segment-routing-extensions-04 (work in progress), May 2015.

[I-D.ietf-ospf-ospfv3-segment-routing-extensions]  
Psenak, P., Previdi, S., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPFv3 Extensions for Segment Routing", draft-ietf-ospf-ospfv3-segment-routing-extensions-02 (work in progress), February 2015.

[I-D.ietf-ospf-segment-routing-extensions]  
Psenak, P., Previdi, S., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPF Extensions for Segment Routing", draft-ietf-ospf-segment-routing-extensions-04 (work in progress), February 2015.

[I-D.ietf-spring-segment-routing]  
Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", draft-ietf-spring-segment-routing-03 (work in progress), May 2015.



[I-D.ietf-spring-segment-routing-mpls]

Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., Litkowski, S., Horneffer, M., Shakir, R., Tantsura, J., and E. Crabbe, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls-01 (work in progress), May 2015.

[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-00 (work in progress), July 2015.

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

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

Authors' Addresses

Pushpasis Sarkar (editor)  
Juniper Networks, Inc.  
Electra, Exora Business Park  
Bangalore, KA 560103  
India

Email: psarkar@juniper.net; pushpasis.ietf@gmail.com

Hannes Gredler  
Individual Contributor

Email: hannes@gredler.at

Clarence Filsfils  
Cisco Systems, Inc.  
Brussels  
BE

Email: cfilsfil@cisco.com

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

Email: sprevidi@cisco.com

Bruno Decraene  
Orange

Email: bruno.decraene@orange.com

Martin Horneffer  
Deutsche Telekom  
Hammer Str. 216-226  
Muenster 48153  
DE

Email: Martin.Horneffer@telekom.de