

SPRING Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: May 6, 2020

Z. Ali  
R. Gandhi  
C. Filsfils  
F. Brockners  
N. Nainar  
C. Pignataro  
Cisco Systems, Inc.  
C. Li  
M. Chen  
Huawei  
G. Dawra  
LinkedIn  
November 3, 2019

Segment Routing Header encapsulation for In-situ OAM Data  
draft-ali-spring-ioam-srv6-02

## Abstract

OAM and PM information from the SR endpoints can be piggybacked in the data packet. The OAM and PM information piggybacking in the data packets is also known as In-situ OAM (IOAM). IOAM records operational and telemetry information in the data packet while the packet traverses a path between two points in the network. This document defines how IOAM data fields are transported as part of the Segment Routing with IPv6 data plane (SRv6) header.

## Status of This Memo

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

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

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

This Internet-Draft will expire on May 6, 2020.

## 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. Conventions . . . . .	3
2.1. Requirement Language . . . . .	3
2.2. Abbreviations . . . . .	3
3. OAM Metadata Piggybacked in Data Packets . . . . .	4
3.1 IOAM Data Field Encapsulation in SRH . . . . .	4
4. Procedure . . . . .	5
4.1. Ingress Node . . . . .	5
4.2. SR Segment Endpoint Node . . . . .	5
4.3. Egress Node . . . . .	6
5. IANA Considerations . . . . .	6
6. Security Considerations . . . . .	6
7. Acknowledgements . . . . .	6
8. References . . . . .	7
8.1. Normative References . . . . .	7
8.2. Informative References . . . . .	7
Authors' Addresses . . . . .	8

## 1. Introduction

OAM and PM information from the SR endpoints can be piggybacked in the data packet. The OAM and PM information piggybacking in the data packets is also known as In-situ OAM (IOAM). IOAM records OAM information within the packet while the packet traverses a particular network domain. The term "in-situ" refers to the fact that the IOAM data fields are added to the data packets rather than being sent within probe packets specifically dedicated to OAM.

This document defines how IOAM data fields are transported as part of the Segment Routing with IPv6 data plane (SRv6) header [I-D.6man-segment-routing-header].

The IOAM data fields carried are defined in [I-D.ietf-ippm-ioam-data], and can be used for various use-cases including Performance Measurement (PM) and Proof-of-Transit (PoT).

## 2. Conventions

### 2.1. Requirement Language

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

### 2.2. Abbreviations

Abbreviations used in this document:

IOAM      In-situ Operations, Administration, and Maintenance

OAM       Operations, Administration, and Maintenance

PM        Performance Measurement

PoT       Proof-of-Transit

SR        Segment Routing

SRH       SRv6 Header

SRv6      Segment Routing with IPv6 Data plane

### 3. OAM Metadata Piggybacked in Data Packets

OAM and PM information from the SR endpoints can be piggybacked in the data packet. The OAM and PM information piggybacking in the data packets is also known as In-situ OAM (IOAM). This section describes IOAM functionality in SRv6 network.

The IOAM data is carried in SRH.TLV. This enables the IOAM mechanism to build on the network programmability capability of SRv6. Specifically, the ability for an SRv6 endpoint to determine whether to process or ignore some specific SRH TLVs is based on the SID function. This enables collection of the IOAM information hardware friendly based on the intermediate endpoint capability. The nodes that are not capable of supporting the IOAM functionality does not have to look or process SRH TLV (i.e., such nodes can simply ignore the SRH IOAM TLV). This also enable collection of IOAM data only from segment endpoint.

#### 3.1 IOAM Data Field Encapsulation in SRH

The SRv6 encapsulation header (SRH) is defined in [I-D.ietf-6man-segment-routing-header]. IOAM data fields are carried in the SRH, using a single pre-allocated SRH TLV. The different IOAM data fields defined in [I-D.ietf-ippm-ioam-data] are added as sub-TLVs.

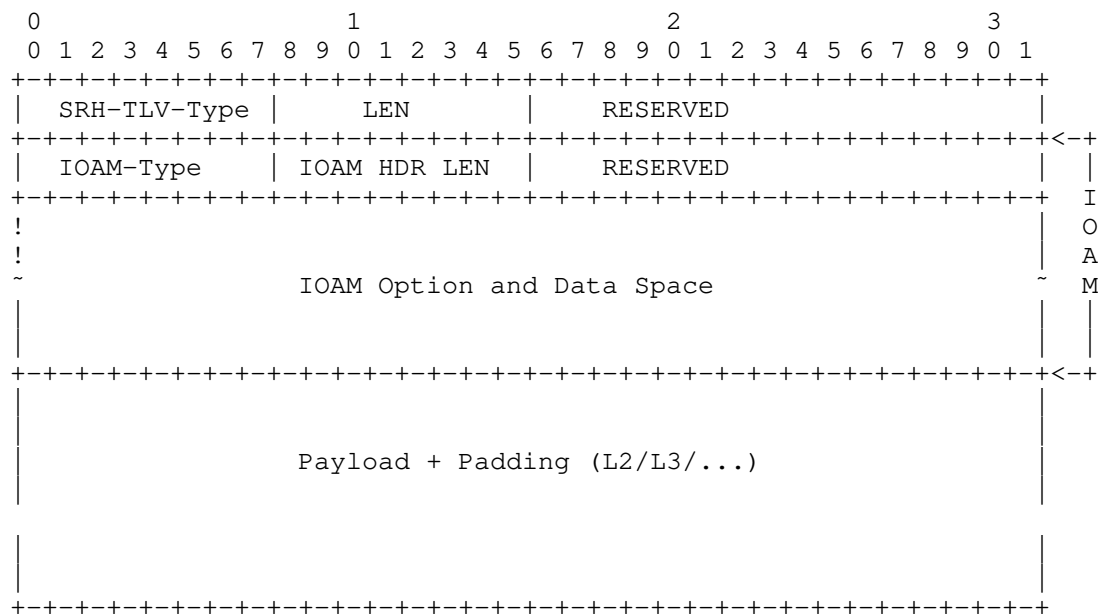


Figure 1: IOAM data encapsulation in SRH

SRH-TLV-Type: IOAM TLV Type for SRH is defined as TBA1.

The fields related to the encapsulation of IOAM data fields in the SRH are defined as follows:

IOAM-Type: 8-bit field defining the IOAM Option type, as defined in Section 7.2 of [I-D.ietf-ippm-ioam-data].

IOAM HDR LEN: 8-bit unsigned integer. Length of the IOAM HDR in 4-octet units.

RESERVED: 8-bit reserved field MUST be set to zero upon transmission and ignored upon receipt.

IOAM Option and Data Space: IOAM option header and data is present as defined by the IOAM-Type field, and is defined in Section 4 of [I-D.ietf-ippm-ioam-data].

#### 4. Procedure

This section summarizes the procedure for IOAM data encapsulation in SRv6 SRH. The SR nodes implementing the IOAM functionality follows the MTU and other considerations outlined in [I-D.6man-extension-header-insertion].

##### 4.1. Ingress Node

As part of the SRH encapsulation, the ingress node of an SR domain or an SR Policy [I-D.ietf-spring-segment-routing-policy] MAY add the IOAM TLV in the SRH of the data packet. If an ingress node supports IOAM functionality and, based on a local configuration, wants to collect IOAM data, it adds IOAM TLV in the SRH. Based on the size of the segment list (SL), the ingress node preallocates space in the IOAM TLV.

If IOAM data from the last node in the segment-list (Egress node) is desired, the ingress uses an Ultimate Segment Pop (USP) SID advertised by the Egress node.

The ingress node MAY also insert the IOAM data about the local information in the IOAM TLV in the SRH at index 0 of the preallocated IOAM TLV.

##### 4.2. Intermediate SR Segment Endpoint Node

The SR segment endpoint node is any node receiving an IPv6 packet where the destination address of that packet is a local SID. As part of the SR Header processing as described in [I-D.ietf-6man-segment-routing-header] and [I-D.ietf-spring-srv6-network-programming], the SR Segment Endpoint node performs the following IOAM operations.

If an intermediate SR segment endpoint node is not capable of processing IOAM TLV, it simply ignores it. I.e., it does not have to look or process SRH TLV.

If an intermediate SR segment endpoint node is capable of processing IOAM TLV and the local SID supports IOAM data recording, it checks if any SRH TLV is present in the packet using procedures defined in [I-D.ietf-6man-segment-routing-header]. If the node finds IOAM TLV in the SRH it finds the local index at which it is expected to record the IOAM data. The local index is found using the SRH.SL field. The node records the IOAM data at the desired preallocated space.

#### 4.3. Egress Node

The Egress node is the last node in the segment-list of the SRH. When IOAM data from the Egress node is desired, a USP SID advertised by the Egress node is used by the Ingress node.

The processing of IOAM TLV at the Egress node is similar to the processing of IOAM TLV at the SR Segment Endpoint Node. The only difference is that the Egress node may telemeter the IOAM data to an external entity.

#### 5. IANA Considerations

IANA is requested to allocate a mutable SRH TLV Type for IOAM TLV data fields under registry name "Segment Routing Header TLVs" requested by [I-D.6man-segment-routing-header].

SRH TLV Type	Description	Reference
TBA1 Greater than 128	TLV for IOAM Data Fields	This document

#### 6. Security Considerations

The security considerations of SRv6 are discussed in [I-D.spring-srv6-network-programming] and [I-D.6man-segment-routing-header], and the security considerations of IOAM in general are discussed in [I-D.ietf-ippm-ioam-data].

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

#### 7. Acknowledgements

The authors would like to thank Shwetha Bhandari and Vengada Prasad Govindan for the discussions on IOAM.

## 8. References

### 8.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", RFC 8174, May 2017.
- [I-D.spring-srv6-network-programming] Filsfils, C. et al. "SRv6 Network Programming", draft-filsfils-spring-srv6-network-programming, work in progress.
- [I-D.6man-segment-routing-header] Previdi, S., Filsfils, C. et al, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header, work in progress.
- [I-D.ietf-ippm-ioam-data] Brockners, F., Bhandari, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Mozes, D., Lapukhov, P., Chang, R., and Bernier, D., "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data, work in progress.
- [I-D.spring-segment-routing-policy] Filsfils, C., et al., "Segment Routing Policy Architecture", draft-ietf-spring-segment-routing-policy, work in progress.

### 8.2. Informative References

- [I-D.6man-extension-header-insertion] D. Voyer, et al., "Insertion of IPv6 Segment Routing Headers in a Controlled Domain", draft-voyer-6man-extension-header-insertion, work in progress.



Authors' Addresses

Zafar Ali  
Cisco Systems, Inc.

Email: zali@cisco.com

Rakesh Gandhi  
Cisco Systems, Inc.  
Canada

Email: rgandhi@cisco.com

Clarence Filsfils  
Cisco Systems, Inc.  
Belgium

Email: cf@cisco.com

Frank Brockners  
Cisco Systems, Inc.  
Germany

Email: fbrockne@cisco.com

Nagendra Kumar Nainar  
Cisco Systems, Inc.

Email: naikumar@cisco.com

Carlos Pignataro  
Cisco Systems, Inc.

Email: cpignata@cisco.com

Cheng Li  
Huawei

Email: chengli13@huawei.com

Mach(Guoyi) Chen  
Huawei

Email: mach.chen@huawei.com

Gaurav Dawra  
LinkedIn

Email: gdawra.ietf@gmail.com

SPRING Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: July 12, 2022

Z. Ali  
R. Gandhi  
C. Filsfils  
F. Brockners  
N. Nainar  
C. Pignataro  
Cisco Systems, Inc.  
C. Li  
M. Chen  
Huawei  
G. Dawra  
LinkedIn  
January 12, 2022

Segment Routing Header encapsulation for In-situ OAM Data  
draft-ali-spring-ioam-srv6-05

## Abstract

OAM and PM information from the SR endpoints can be piggybacked in the data packet. The OAM and PM information piggybacking in the data packets is also known as In-situ OAM (IOAM). IOAM records operational and telemetry information in the data packet while the packet traverses a path between two points in the network. This document defines how IOAM data fields are transported as part of the Segment Routing with IPv6 data plane (SRv6) header.

## 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 July 12, 2022.

## 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. Conventions . . . . .	3
2.1. Requirement Language . . . . .	3
2.2. Abbreviations . . . . .	3
3. OAM Metadata Piggybacked in Data Packets . . . . .	4
3.1 IOAM Data Field Encapsulation in SRH . . . . .	4
4. Procedure . . . . .	5
4.1. Ingress Node . . . . .	5
4.2. SR Segment Endpoint Node . . . . .	5
4.3. Egress Node . . . . .	6
5. IANA Considerations . . . . .	6
6. Security Considerations . . . . .	6
7. Acknowledgements . . . . .	6
8. References . . . . .	7
8.1. Normative References . . . . .	7
8.2. Informative References . . . . .	7
Authors' Addresses . . . . .	8

## 1. Introduction

OAM and PM information from the SR endpoints can be piggybacked in the data packet. The OAM and PM information piggybacking in the data packets is also known as In-situ OAM (IOAM). IOAM records OAM information within the packet while the packet traverses a particular network domain. The term "in-situ" refers to the fact that the IOAM data fields are added to the data packets rather than being sent within probe packets specifically dedicated to OAM.

This document defines how IOAM data fields are transported as part of the Segment Routing with IPv6 data plane (SRv6) header [I-D.6man-segment-routing-header].

The IOAM data fields carried are defined in [I-D.ietf-ippm-ioam-data], and can be used for various use-cases including Performance Measurement (PM) and Proof-of-Transit (PoT).

## 2. Conventions

### 2.1. Requirement Language

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

### 2.2. Abbreviations

Abbreviations used in this document:

IOAM	In-situ Operations, Administration, and Maintenance
OAM	Operations, Administration, and Maintenance
PM	Performance Measurement
PoT	Proof-of-Transit
SR	Segment Routing
SRH	SRv6 Header
SRv6	Segment Routing with IPv6 Data plane

### 3. OAM Metadata Piggybacked in Data Packets

OAM and PM information from the SR endpoints can be piggybacked in the data packet. The OAM and PM information piggybacking in the data packets is also known as In-situ OAM (IOAM). This section describes iOAM functionality in SRv6 network.

The IOAM data is carried in SRH.TLV. This enables the IOAM mechanism to build on the network programmability capability of SRv6. Specifically, the ability for an SRv6 endpoint to determine whether to process or ignore some specific SRH TLVs is based on the SID function. This enables collection of the IOAM information hardware friendly based on the intermediate endpoint capability. The nodes that are not capable of supporting the IOAM functionality does not have to look or process SRH TLV (i.e., such nodes can simply ignore the SRH IOAM TLV). This also enable collection of IOAM data only from segment endpoint.

### 3.1 IOAM Data Field Encapsulation in SRH

The SRv6 encapsulation header (SRH) is defined in [I-D.ietf-6man-segment-routing-header]. IOAM data fields are carried in the SRH, using a single pre-allocated SRH TLV. The different IOAM data fields defined in [I-D.ietf-ippm-ioam-data] are added as sub-TLVs.

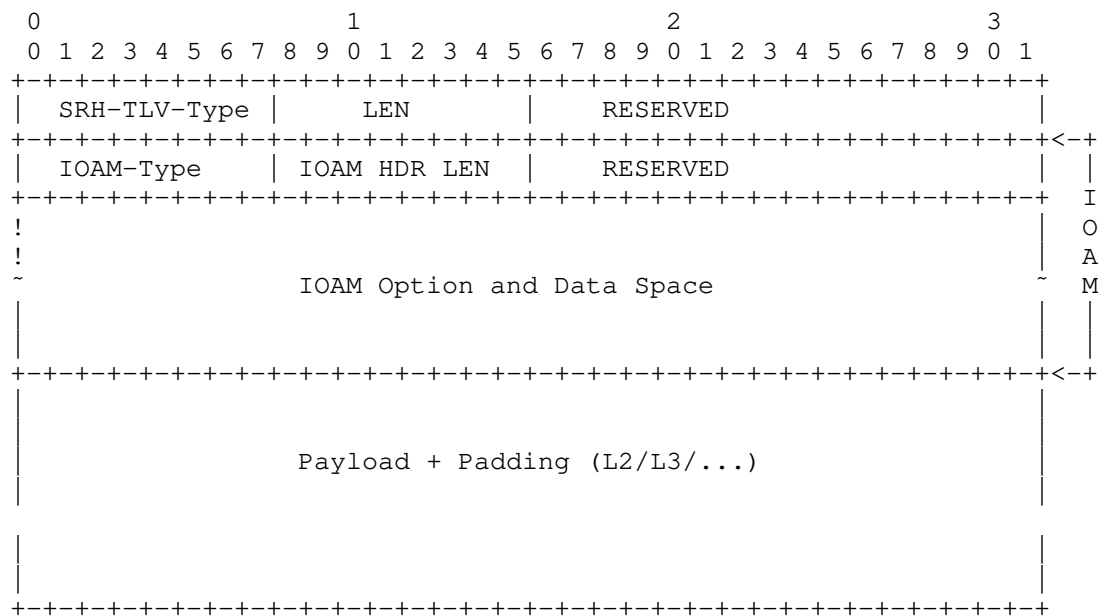


Figure 1: IOAM data encapsulation in SRH

SRH-TLV-Type: IOAM TLV Type for SRH is defined as TBA1.

The fields related to the encapsulation of IOAM data fields in the SRH are defined as follows:

IOAM-Type: 8-bit field defining the IOAM Option type, as defined in Section 7.2 of [I-D.ietf-ippm-ioam-data].

IOAM HDR LEN: 8-bit unsigned integer. Length of the IOAM HDR in 4-octet units.



RESERVED: 8-bit reserved field MUST be set to zero upon transmission and ignored upon receipt.

IOAM Option and Data Space: IOAM option header and data is present as defined by the IOAM-Type field, and is defined in Section 4 of [I-D.ietf-ippm-ioam-data].

#### 4. Procedure

This section summarizes the procedure for IOAM data encapsulation in SRv6 SRH. The SR nodes implementing the IOAM functionality follows the MTU and other considerations outlined in [I-D.6man-extension-header-insertion].

##### 4.1. Ingress Node

As part of the SRH encapsulation, the ingress node of an SR domain or an SR Policy [I-D.ietf-spring-segment-routing-policy] MAY add the IOAM TLV in the SRH of the data packet. If an ingress node supports IOAM functionality and, based on a local configuration, wants to collect IOAM data, it adds IOAM TLV in the SRH. Based on the size of the segment list (SL), the ingress node preallocates space in the IOAM TLV.

If IOAM data from the last node in the segment-list (Egress node) is desired, the ingress uses an Ultimate Segment Pop (USP) SID advertised by the Egress node.

The ingress node MAY also insert the IOAM data about the local information in the IOAM TLV in the SRH at index 0 of the preallocated IOAM TLV.

##### 4.2. Intermediate SR Segment Endpoint Node

The SR segment endpoint node is any node receiving an IPv6 packet where the destination address of that packet is a local SID. As part of the SR Header processing as described in [I-D.ietf-6man-segment-routing-header] and [I-D.ietf-spring-srv6-network-programming], the SR Segment Endpoint node performs the following IOAM operations.

If an intermediate SR segment endpoint node is not capable of processing IOAM TLV, it simply ignores it. I.e., it does not have to look or process SRH TLV.

If an intermediate SR segment endpoint node is capable of processing IOAM TLV and the local SID supports IOAM data recording, it checks if any SRH TLV is present in the packet using procedures defined in [I-D.ietf-6man-segment-routing-header]. If the node finds IOAM TLV in the SRH it finds the local index at which it is expected to record the IOAM data. The local index is found using the SRH.SL field. The node records the IOAM data at the desired preallocated space.



#### 4.3. Egress Node

The Egress node is the last node in the segment-list of the SRH. When IOAM data from the Egress node is desired, a USP SID advertised by the Egress node is used by the Ingress node.

The processing of IOAM TLV at the Egress node is similar to the processing of IOAM TLV at the SR Segment Endpoint Node. The only difference is that the Egress node may telemeter the IOAM data to an external entity.

#### 5. IANA Considerations

IANA is requested to allocate a mutable SRH TLV Type for IOAM TLV data fields under registry name "Segment Routing Header TLVs" requested by [I-D.6man-segment-routing-header].

SRH TLV Type	Description	Reference
TBA1 Greater than 128	TLV for IOAM Data Fields	This document

#### 6. Security Considerations

The security considerations of SRv6 are discussed in [I-D.spring-srv6-network-programming] and [I-D.6man-segment-routing-header], and the security considerations of IOAM in general are discussed in [I-D.ietf-ippm-ioam-data].

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

#### 7. Acknowledgements

The authors would like to thank Shwetha Bhandari and Vengada Prasad Govindan for the discussions on IOAM.

## 8. References

### 8.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", RFC 8174, May 2017.
- [I-D.spring-srv6-network-programming] Filsfils, C. et al. "SRv6 Network Programming", draft-filsfils-spring-srv6-network-programming, work in progress.
- [I-D.6man-segment-routing-header] Previdi, S., Filsfils, C. et al, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header, work in progress.
- [I-D.ietf-ippm-ioam-data] Brockners, F., Bhandari, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Mozes, D., Lapukhov, P., Chang, R., and Bernier, D., "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data, work in progress.
- [I-D.spring-segment-routing-policy] Filsfils, C., et al., "Segment Routing Policy Architecture", draft-ietf-spring-segment-routing-policy, work in progress.

### 8.2. Informative References

- [I-D.6man-extension-header-insertion] D. Voyer, et al., "Insertion of IPv6 Segment Routing Headers in a Controlled Domain", draft-voyer-6man-extension-header-insertion, work in progress.

Internet-Draft

In-situ OAM SRv6 encapsulation

Authors' Addresses

Zafar Ali  
Cisco Systems, Inc.

Email: zali@cisco.com

Rakesh Gandhi  
Cisco Systems, Inc.  
Canada

Email: rgandhi@cisco.com

Clarence Filsfils  
Cisco Systems, Inc.  
Belgium

Email: cf@cisco.com

Frank Brockners  
Cisco Systems, Inc.  
Germany

Email: fbrockne@cisco.com

Nagendra Kumar Nainar  
Cisco Systems, Inc.

Email: naikumar@cisco.com

Carlos Pignataro  
Cisco Systems, Inc.

Email: cpignata@cisco.com

Cheng Li  
Huawei

Email: chenglil13@huawei.com

Mach(Guoyi) Chen  
Huawei

Email: mach.chen@huawei.com

Gaurav Dawra  
LinkedIn

Email: gdawra.ietf@gmail.com



Internet Engineering Task Force  
Internet-Draft  
Intended status: Best Current Practice  
Expires: 1 May 2021

R. Geib, Ed.  
Deutsche Telekom  
28 October 2020

An MPLS SR OAM option reducing the number of end-to-end path validations  
draft-geib-spring-oam-opt-00

## Abstract

MPLS traceroute implementations validate dataplane connectivity and isolate faults by sending messages along every end-to-end Label Switched Path (LSP) combination between a source and a destination node. This requires a growing number of path validations in networks with a high number of equal cost paths between origin and destination. Segment Routing (SR) introduces MPLS topology awareness combined with Source Routing. By this combination, SR can be used to implement an MPLS traceroute option lowering the total number of LSP validations as compared to commodity MPLS traceroute.

## 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 1 May 2021.

## Copyright Notice

Copyright (c) 2020 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
1.1. Requirements Language . . . . .	5
2. MPLS OAM adding MPLS SR mechanisms . . . . .	5
2.1. Operation in an SR MPLS domain applying only IP-header based ECMP . . . . .	6
2.2. Operation in an SR MPLS domain additionally using incoming interface information for ECMP . . . . .	7
3. IANA Considerations . . . . .	8
4. Security Considerations . . . . .	9
5. References . . . . .	9
5.1. Normative References . . . . .	9
Author's Address . . . . .	9

## 1. Introduction

Commodity MPLS isn't topology aware and it offers no standard source routing methods. It is reasonable to validate connectivity and locate faults of MPLS LSPs by detecting and testing all existing LSP combinations between a source and a destination node. The source node originates all MPLS echo requests and evaluates all MPLS echo replies. Operational MPLS OAM implementations were present, when SR MPLS entered standardisation. They continue to work reliably in many cases. MPLS domains with a high number of equal cost paths between source and destination nodes push the detection capabilities of commodity MPLS OAM to the limit. So far, modes of MPLS OAM operation adding Segment Routing functionality to deal with limitations of commodity MPLS OAM have not been published within IETF.

This draft assumes readers to be aware of MPLS OAM functionality as specified by RFC 8029 [RFC8029] and RFC 8287 [RFC8287]. The function described in the following works for Shortest Path First Paths or Label stacks based on MPLS Node-SID and MPLS Adj-SIDs (if the latter are distributed by Interior Gateway Protocols).

Networks supporting a high number of equivalent cost paths between source and destination nodes require a high number of completed MPLS path validations. Consider a network with Multiple equal cost paths, as shown in figure 1.

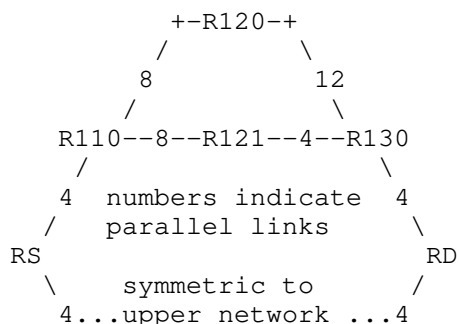


Figure 1

Figure 1: Multiple equal cost path example network.

The total number of MPLS LSP combinations between nodes RS and RD is multiplicative by the number of (equal cost, so to say) links per hop. That results in a maximum of  $4096 = 2^4 * (8 * 12 + 8 * 4) * 4$  path combinations which a commodity MPLS may try to validate. Assume RS to start an MPLS traceroute to RD containing a Multipath Data Sub-TLV requesting Multipath information for 32 IP-addresses. By Equal Cost Multipath routing (ECMP, [RFC2991]) traffic of likely 16 of these IP-addresses is forwarded via R110 as next hop (the other 16 addresses are assumed to be forwarded along the symmetric and equal cost paths in the lower half, which are omitted in the figure for brevity). R110 can be expected to respond by an MPLS echo reply indicating prefixes to address each of the 4 equal cost (sub-)paths between RS and R110.

R110 is able to forward traffic addressed by these 16 IP addresses via 16 equal cost paths. There's a fairly high probability that this will not be possible, as some of R110's available paths to forwards traffic to RD will be receive traffic of two or even three IP addresses, while others receive no traffic at all. The MPLS Echo Reply sent to RS will indicate that. A commodity solution is, to start an additional MPLS traceroute with 32 different IP-addresses. This may help to then enable forwarding traffic along all of R110's paths to RD via R120 and R121, respectively. With bad luck, R110 will forward only 14 or 15 addresses via R120. R120 forwards traffic along 12 equal cost paths to RD. Then again, there's a fair chance that more IP-addresses are required to forward at least one MPLS echo request along all of them. Per new set of addresses, the MPLS Echo-Request / Reply dialogue must be completed starting from RS to at least all routers along the path to R120.

R110 is able to forward traffic addressed by these 16 IP addresses via 16 equal cost paths. There's a fairly high probability that some of R110's available paths to forward traffic to RD will receive traffic of two or even three of these IP addresses, while other paths receive no traffic at all. The MPLS Echo Reply returned to RS will indicate that. A commodity solution is, to start an additional MPLS traceroute to check the routes for 32 different IP-addresses. Note that again traffic with 16 of these IP addresses is likely to be routed to the omitted symmetric part of the example network. The remaining additional 16 IP addresses forwarded to R110 however help to then enable forwarding traffic along all of R110's paths to RD via R120 and R121, respectively. With bad luck, R110 might forward only 14 or 15 addresses via R120. R120 forwards traffic along 12 equal cost paths to RD. Then again, there's a fair chance that more IP-addresses are required to forward at least one MPLS echo request along all of them. For every additional new set of IP-addresses, the MPLS Echo-Request / Reply dialogue must be completed starting from RS to at least all routers along the path to R120. In the example, roughly only a fourth of the addresses whose forwarding is validated starting from node RS will be routed via R120. The ECMP "filters away" 75% of the available addresses. If all 32 IP-addresses were reaching R120, the probability of being unable to forward at least one MPLS Echo request to each outgoing interface (or path, respectively) to node RD was rather small.

The reason for completing all MPLS Echo Request / Reply dialogues along the path between RS and R120 is figuring out, which IP addresses are routed from R110 to R120 to be available at the latter for forwarding traffic along paths to RD which can't be addressed otherwise. RFC 8029 section 4.1 'Dealing with Equal-Cost Multipath (ECMP)' concludes, that 'full coverage may not be possible' [RFC8029].

Segment Routing (SR) allows node RS to forward MPLS Echo Request packets with up to, e.g., 32 IP addresses to every node which RS detects on a path to node RD. Doing so reduces the number of path options to be checked to no more than the sum of the interfaces belonging to one of the ECMP routes between nodes RS and RD. In the case of the example network above, this sum is  $2 \cdot (4+8+8+12+4+4)=80$ . That means, that around 2% of the messages required with commodity MPLS traceroute implementations are sufficient to validate all local forwarding options (note that the calculations aren't exact, they rather indicate orders of magnitude). The commodity MPLS OAM implementations are neither broken nor not working. SR allows to add a new method to the router local MPLS OAM implementations to reliably validate also high numbers of ECMP routes. The method proposed results reduces the number of MPLS Echo-Request / -Reply dialogues to be stored and completed in that case.



The functions specified by this document do not require changes in the MPLS OAM protocol as specified by [RFC8029] and [RFC8287].

### 1.1. Requirements Language

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

## 2. MPLS OAM adding MPLS SR mechanisms

MPLS Segment Routing (SR) provides each node of an MPLS SR domain with this domain's MPLS Node-SID topology [RFC8402]. The SR source routing feature allows to forward packets to each individual node within a SR domain. Combining topology awareness and source routing allows complete validation of all local router shortest path forwarding options from an RS node to an RD node in a domain supporting ECMP.

Suppose SR to be deployed in the case of the example network and digits following the letter "R" to indicate the corresponding Node-SIDs. Assume "mixed operation" of commodity MPLS OAM and the option applying SR. RS starts a commodity MPLS Echo request to R110. After having received an MPLS Echo reply from R110 indicating local paths of R110 on which none of the packets with the remaining 16 IP addresses will be forwarded, RS creates an MPLS Echo Request which transports the original 32 IP addresses to R110. To do so, an additional top-Segment is pushed carrying the R110 Node-SID, 110. The message below that additional segment is coded as a standard RFC8287 MPLS Echo request. Two things are special: the TTL of the MPLS header containing the Node SID of RD is always set to 1. Further, a separate sequence number series needs to be started to distinguish the starting point of this SR using MPLS OAM sequence. Coding space for MPLS OAM Sender's Handle and Sequence Number offer sufficient coding space [RFC8029]. If PHP is active, the R110 Node-SID is implicitly present only on the link to a neighboring node. Still packets with all 32 IP-destination addresses are forwarded to R110. The chances to address all of the 16 ECMP paths of R110 to RD with the originally configured 32 IP-addresses increase. The same method is repeated for R120. Now the top Segment picked by node RS is the Node-SID of R120, again with a separate Sender's Handle and Sequence Number combination. Note, that the MPLS Echo request destined to R120 doesn't require execution of MPLS OAM functions in R110. That latter node simply forwards the packet to R120. Also R120 receives 32 IP-addresses (which is a significant increase as compared to commodity MPLS OAM).

As a result, the MPLS Echo reply tables maintained by RS likely indicate ECMP several forwarding masks of the same IP address range (discerned by the starting node receiving the MPLS Echo request with top Segment TTL=1). For every path at an intermediate node, to which the latter can't forward an MPLS Echo request due to the limited number of available IP-addresses, a suitable SR top segment is added for an additional next MPLS Echo request of node RS. This in the end allows to circumvent the IP-address filtering effect caused by ECMP.

Being able to forward a "complete" set of IP addresses to any interface along an end-to-end path is helpful in locating errors. Different MPLS OAM addressing options also offer more possibilities to test and unambiguously locate a faulty sub-path.

## 2.1. Operation in an SR MPLS domain applying only IP-header based ECMP

The basic operation is to transport an MPLS Echo request from the sender node sequentially to a next hop identified on any of the paths to a destination node. This is done by applying standard SR methodology, which here consists of pushing one additional Node-SID on top of the Label-stack to be validated by the sender node. The Node-SID is set to the value of the node, whose forwarding plane information is requested by the MPLS Echo request. This is illustrated by figure 2.

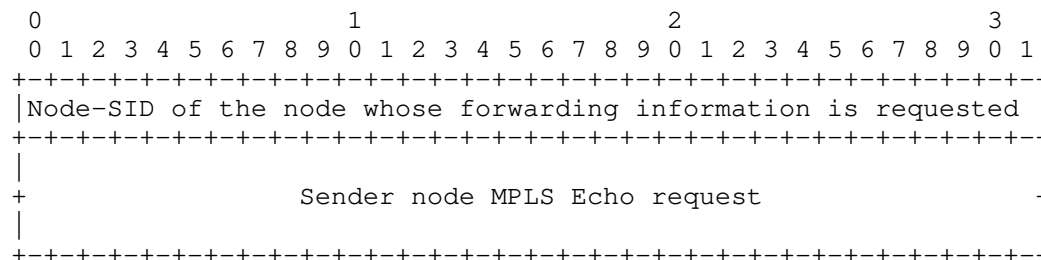


Figure 2

Figure 2: MPLS OAM Label Stack in the case of IP-header only based ECMP.

The added Node-SID is only added to use standard MPLS forwarding. The TTL of this added Node-SID set to the default value for traffic injected by the sending router. The MPLS-TC may be set to a value ensuring reliable transport up to the node, whose forwarding information is requested by the sender node (be aware of MPLS-TC treatment of the node popping this added Node-SID in that case).

The TTL of the top Label of the sender node MPLS Echo request which is contained below the added Node-SID initially is set to TTL=1. Other TTL values can be picked if LSPs from the intermediate node onwards to the destination node of that FEC are desired to be traced or pinged by MPLS OAM messages.

Two modes of operation exist: either applying legacy MPLS OAM and adding the described functionality as required or only applying the option specified here. Note that the exact path from the sender node to the intermediate node identified by the pushed Node-SID is only known to the node originating and maintaining the MPLS traceroute information, if only one path exists between that sender node and an intermediate node.

If the method is added to commodity MPLS OAM functions, the originator IP-address of an MPLS Echo-reply indicating a lack of IP-addresses to forward traffic along all ECMP egress interfaces at that intermediate node can be used to derive the Node-SID to be pushed by the MPLS Echo request sender node.

## 2.2. Operation in an SR MPLS domain additionally using incoming interface information for ECMP

This option can only be applied, if the Segment Routing domain's Adj-SID topology is known to the node originating MPLS Echo Request messages. Configuring the the Interior Gateway Protocol to distribute Adj-SIDs conveniently enables that. If ECMP is additionally using the incoming interface of a packet for path selection, an Adj-SID is added between the Node-SID and the MPLS Echo request. As the idea is to determine the incoming interface of the node, whose ECMP path choices are requested by MPLS OAM, the additionally pushed Node-SID here is that of the node preceding the intermediate node, whose forwarding information is requested. The Adj-SID is chosen to correspond to a specific incoming interface of the intermediate node whose forwarding information is requested. As the aim of that test is to ensure that every incoming to outgoing interface path choice of the intermediate node can be addressed, the topology information required to identify the upstream Adj-SID corresponding to an incoming interface of the intermediate node is assumed to be present and maintained in the originating node. This additional MPLS to IP topology excerpt information results from prior MPLS path validations of the same basic set of MPLS path validations between the source node and the destination node (this is to express, that no extra measurement effort is caused, as correlation of available information is sufficient). The resulting label stack is illustrated by figure 3.

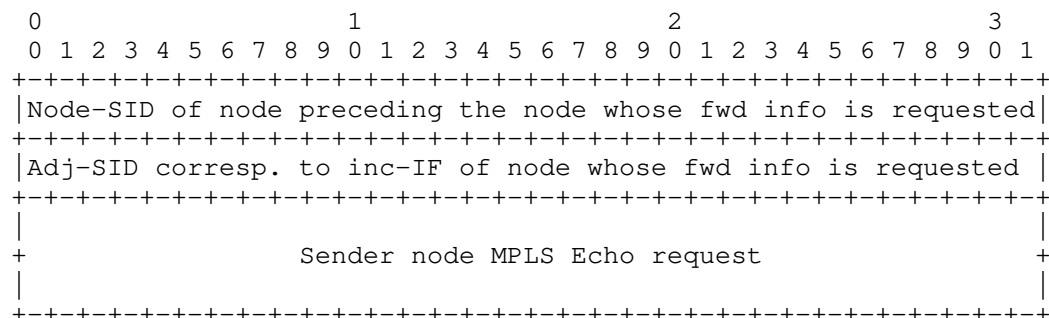


Figure 3

Figure 3: MPLS OAM Label Stack applying SR features if ECMP is additionally based on incoming interfaces.

In the network example of figure 1, node RS picks the Node-SID of R110 and an Adj-SID of R110 corresponding to a particular incoming interface of R120, if the latter's ECMP path also depends on the incoming interface, by which the MPLS Echo request was received.

Here, the full set of original IP-addresses can be forwarded individually per incoming interface of the router whose MPLS forwarding information is requested. In the example above, it is node R120 (not node R110.) Monitoring incoming interface based ECMP results in a higher number of MPLS OAM validations, no matter whether commodity MPLS OAM is applied or the option specified here. The overall sum of tests now is determined by the sum of per node incoming \* outgoing paths (or interfaces, respectively). If the method specified here is applied in the case of the example network,  $2 \cdot (4 \cdot 8 + 4 \cdot 8 + 8 \cdot 12 + 8 \cdot 4 + 12 \cdot 4 + 4 \cdot 4) = 512$  MPLS Echo-Request / Response validations are required. Note that this is still a smaller number than the original 4096 path validations in the case of commodity MPLS OAM required for a domain applying ECMP based on IP-address information only. Note that the number of required MPLS OAM path validations is increasing significantly, if ECMP forwarding is in addition based on incoming interfaces and the product of a nodes incoming \* outgoing interfaces is high.

### 3. IANA Considerations

This memo includes no request to IANA.

#### 4. Security Considerations

This document does not introduce new functionality. It combines Segment Routing functions with those of MPLS OAM. The related security sections apply, see [RFC8029] and [RFC8402].

#### 5. References

##### 5.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC2991] Thaler, D. and C. Hopps, "Multipath Issues in Unicast and Multicast Next-Hop Selection", RFC 2991, DOI 10.17487/RFC2991, November 2000, <<https://www.rfc-editor.org/info/rfc2991>>.
- [RFC8029] Kompella, K., Swallow, G., Pignataro, C., Kumar Nainar, N., Aldrin, S., and M. Chen, "Detecting Multiprotocol Label Switched (MPLS) Data-Plane Failures", RFC 8029, DOI 10.17487/RFC8029, March 2017, <<https://www.rfc-editor.org/info/rfc8029>>.
- [RFC8287] Kumar Nainar, N., Pignataro, C., Swallow, G., Akiya, N., Kini, S., and M. Chen, "Label Switched Path (LSP) Ping/Traceroute for Segment Routing (SR) IGP-Prefix and IGP-Adjacency Segment Identifiers (SIDs) with MPLS Data Planes", RFC 8287, DOI 10.17487/RFC8287, December 2017, <<https://www.rfc-editor.org/info/rfc8287>>.
- [RFC8402] Filsfils, C., Previdi, S., 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>>.

##### Author's Address

Ruediger Geib (editor)  
Deutsche Telekom  
Heinrich Hertz Str. 3-7  
64295 Darmstadt  
Germany  
  
Phone: +49 6151 5812747  
Email: [Ruediger.Geib@telekom.de](mailto:Ruediger.Geib@telekom.de)

Internet Engineering Task Force  
Internet-Draft  
Intended status: Best Current Practice  
Expires: 28 October 2022

R. Geib, Ed.  
Deutsche Telekom  
26 April 2022

An MPLS SR OAM option reducing the number of end-to-end path validations  
draft-geib-spring-oam-opt-03

## Abstract

MPLS traceroute implementations validate dataplane connectivity and isolate faults by sending messages along every end-to-end Label Switched Path (LSP) combination between a source and a destination node. This requires a growing number of path validations in networks with a high number of equal cost paths between origin and destination. Segment Routing (SR) introduces MPLS topology awareness combined with Source Routing. By this combination, SR can be used to implement an MPLS traceroute option lowering the total number of LSP validations as compared to commodity MPLS traceroute.

## 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 28 October 2022.

## Copyright Notice

Copyright (c) 2022 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 Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

## Table of Contents

1. Introduction . . . . .	2
1.1. Requirements Language . . . . .	5
2. MPLS OAM adding MPLS SR mechanisms . . . . .	5
2.1. Operation in an SR MPLS domain applying only IP-header based ECMP . . . . .	6
2.2. Operation in an SR MPLS domain additionally using incoming interface information for ECMP . . . . .	7
3. IANA Considerations . . . . .	9
4. Security Considerations . . . . .	9
5. References . . . . .	9
5.1. Normative References . . . . .	9
Author's Address . . . . .	9

## 1. Introduction

Commodity MPLS isn't topology aware and it doesn't support standardized source routing methods. It is reasonable to validate connectivity and locate faults of MPLS LSPs by detecting and testing all existing LSP combinations between a source and a destination node. The source node originates all MPLS echo requests and evaluates all MPLS echo replies. Operational MPLS OAM implementations were present, when SR MPLS entered standardisation. They continue to work reliably in many cases. MPLS domains with a high number of equal cost paths between source and destination nodes push the detection capabilities of commodity MPLS OAM to the limit. So far, modes of MPLS OAM operation adding Segment Routing functionality to deal with limitations of commodity MPLS OAM have not been published within IETF.

This draft assumes readers to be aware of MPLS OAM functionality as specified by RFC 8029 [RFC8029] and RFC 8287 [RFC8287]. The function described in the following works for Shortest Path First Paths or Label stacks based on MPLS Node-SID and MPLS Adj-SIDs (if the latter are distributed by Interior Gateway Protocols).

Networks supporting a high number of equivalent cost paths between source and destination nodes require a high number of completed MPLS path validations. Consider a network with Multiple equal cost paths, as shown in figure 1.

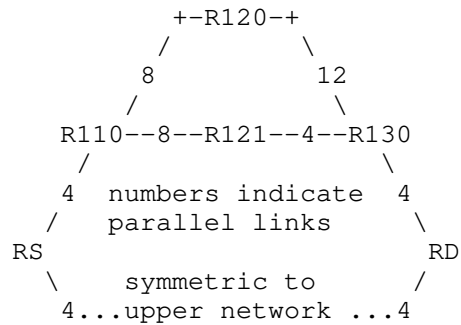


Figure 1

Figure 1: Multiple equal cost path example network.

The total number of MPLS LSP combinations between nodes RS and RD is multiplicative by the number of (equal cost, so to say) links per hop. That results in a maximum of  $4096 = 2 \times 4 \times (8 \times 12 + 8 \times 4) \times 4$  path combinations which a commodity MPLS may try to validate. Assume node RS to start an MPLS traceroute to node RD, containing a Multipath Data Sub-TLV requesting Multipath information for 32 IP-addresses. By Equal Cost Multipath routing (ECMP, [RFC2991]) traffic of likely 16 of these IP-addresses is forwarded via R110 as next hop (the other 16 addresses are assumed to be forwarded along the symmetric and equal cost paths in the lower half, which are omitted in the figure for brevity). R110 can be expected to respond by an MPLS echo reply indicating prefixes to address each of the 4 equal cost (sub-)paths between RS and R110.



R110 is able to forward traffic addressed by these 16 IP addresses via 16 equal cost paths. There's a fairly high probability that this will not be possible, as some of R110's available paths to forward traffic to RD will receive traffic of two or even three MPLS echo request destination IP addresses resulting in an MPLS Echo request being sent from RS to R110 and ahead, while other equal cost paths of R110 receive no traffic at all. The MPLS Echo Reply returned to RS will indicate that. A commodity solution is, to start an additional MPLS traceroute from RS with another 32 destination IP-addresses. This may help to then enable forwarding of MPLS Echo requests along all of R110's paths to RD via R120 and R121, respectively. With bad luck, R110 will forward only 14 or 15 addresses via R120. R120 forwards MPLS Echo requests along 12 equal cost paths to RD. Then again, there's a fair chance that more destination IP-addresses are required to forward at least one MPLS echo request along all of R120 equal cost paths to RD. Each new MPLS Echo Request containing additional IP destination addresses requires completion of the MPLS Echo-Request / Reply dialogue starting from RS to at least all routers along the path to R120.

In the example, roughly only a fourth of the addresses whose forwarding is validated starting from node RS will be routed via R120. ECMP load balancing "filters away" 75% of MPLS Echo requests carrying the destination IP-addresses whose forwarding is to be determined. If MPLS Echo requests carrying a full set of 32 destination IP-addresses were reaching R120, the probability of being unable to forward at least one MPLS Echo request to each outgoing interface (or path, respectively) at R110 destined to node RD was rather small.

The reason for completing all MPLS Echo Request / Reply dialogues along the path between RS and R120 is figuring out, which destination IP-addresses are routed from R110 to R120 to be available at the latter for local traffic forwarding along paths to RD which can't be addressed otherwise. RFC 8029 section 4.1 'Dealing with Equal-Cost Multipath (ECMP)' concludes, that 'full coverage may not be possible' [RFC8029].

Segment Routing (SR) allows node RS to forward MPLS Echo Request packets with up to, e.g., 32 IP addresses to every node which RS detects on a path to node RD. Doing so reduces the number of local router path options to be checked to no more than the sum of the interfaces belonging to one of the ECMP routes between nodes RS and RD. In the case of the example network above, this sum is  $2 \cdot (4+8+8+12+4+4) = 80$  different local router interfaces of routers RS, R110, R120, R121 and R130. That means, that around 2% of the messages and MPLS Label Switched Path checks required with commodity MPLS traceroute implementations are sufficient to validate all local

forwarding options for paths from RS to RD (note that the calculation isn't exact, it rather indicates the order of magnitude). The commodity MPLS OAM implementations are neither broken nor not working. SR allows an additional router local MPLS OAM method to validate high numbers of ECMP routes reliably and fast. The method proposed here reduces the number of MPLS Echo-Request / -Reply dialogues to be stored and completed by the origing of the path validation and it reduces the number of MPLS Echo-Request / -Reply processing at intermediate nodes.

The functions specified by this document do not require changes in the MPLS OAM protocol as specified by [RFC8029] and [RFC8287].

### 1.1. Requirements Language

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

## 2. MPLS OAM adding MPLS SR mechanisms

MPLS Segment Routing (SR) provides each node of an MPLS SR domain with this domain's MPLS Node-SID topology [RFC8402]. The SR source routing feature allows to forward packets to each individual node within a SR domain. Combining topology awareness and source routing allows complete validation of all local router shortest path forwarding options from an RS node to an RD node in a domain supporting ECMP.

Suppose SR to be deployed in the case of the example network and digits following the letter "R" to indicate the corresponding Node-SIDs. Assume "mixed operation" of commodity MPLS OAM and the option applying SR. RS starts a commodity MPLS Echo request to R110. After having received an MPLS Echo reply from R110 indicating local paths of R110 on which none of the packets with the remaining 16 IP addresses will be forwarded, RS creates an MPLS Echo Request which transports the original 32 IP addresses to R110. To do so, an additional top-Segment is pushed carrying the R110 Node-SID, 110. The message below this additional segment is coded as a standard RFC8287 MPLS Echo request. Two things are special: the TTL of the MPLS header containing the Node SID of RD is always set to 1. Further, a separate sequence number series needs to be started to distinguish the starting point of this SR using MPLS OAM sequence. Coding space for MPLS OAM Sender's Handle and Sequence Number offer sufficient coding space [RFC8029]. If PHP is active, the R110 Node-SID is implicitly present only on the link to a neighboring node. Still packets with all 32 IP-destination addresses are forwarded to R110. The chances to address all of the 16 ECMP paths of R110 to RD with

the originally configured 32 IP-addresses increase. The same method is repeated for R120. Now the top Segment picked by node RS is the Node-SID of R120, again with a separate Sender's Handle and Sequence Number combination. Note, that the MPLS Echo request destined to R120 doesn't require execution of MPLS OAM functions in R110. That latter node simply forwards the packet to R120. Also R120 receives 32 IP-addresses (which is a significant increase as compared to commodity MPLS OAM).

As a result, the MPLS Echo reply tables maintained by RS likely indicate several forwarding masks correlated to the same IP address range (discerned by the intermediate node receiving an MPLS Echo request with top Segment TTL=1). For every path at an intermediate node, to which the latter can't forward an MPLS Echo request due to the limited number of available IP-addresses, a suitable SR top segment is added for an additional next MPLS Echo request of node RS. This in the end allows to circumvent the IP-address filtering effect caused by ECMP.

Being able to forward a "complete" set of IP addresses to any interface along an end-to-end path is helpful in locating errors. Different MPLS OAM addressing options also offer more possibilities to test and unambiguously locate a failed sub-path.

## 2.1. Operation in an SR MPLS domain applying only IP-header based ECMP

The basic operation is to transport an MPLS Echo request from the sender node sequentially to a next hop identified on any of the paths to a destination node. This is done by applying standard SR methodology, which here consists of pushing one additional Node-SID on top of the Label-stack to be validated by the sender node. The Node-SID is set to the value of the node, whose forwarding plane information is requested by the MPLS Echo request. This is illustrated by figure 2.

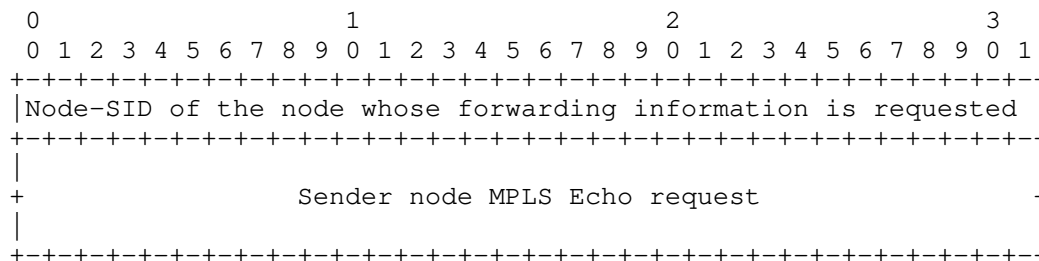


Figure 2

Figure 2: MPLS OAM Label Stack in the case of IP-header only based ECMP.

The added Node-SID is only added to use standard MPLS forwarding. The TTL of this added Node-SID set to the default value for traffic injected by the sending router. The MPLS-TC may be set to a value ensuring reliable transport up to the node, whose forwarding information is requested by the sender node (be aware of MPLS-TC treatment of the node popping this added Node-SID in that case).

The TTL of the top Label of the sender node MPLS Echo request which is contained below the added Node-SID initially is set to TTL=1. Other TTL values can be picked if LSPs from the intermediate node onwards to the destination node of that FEC are desired to be traced or pinged by MPLS OAM messages.

Two modes of operation exist: either applying legacy MPLS OAM and adding the described functionality as required or only applying the option specified here. Note that the exact path from the sender node to the intermediate node identified by the pushed Node-SID is only known to the node originating and maintaining the MPLS traceroute information, if only one path exists between that sender node and an intermediate node.

If the method is added to commodity MPLS OAM functions, the originator IP-address of an MPLS Echo-reply indicating a lack of IP-addresses to forward traffic along all ECMP egress interfaces at that intermediate node can be used to derive the Node-SID to be pushed by the MPLS Echo request sender node.

## 2.2. Operation in an SR MPLS domain additionally using incoming interface information for ECMP

This option can only be applied, if the Segment Routing domain's Adj-SID topology is known to the node originating MPLS Echo Request messages. Configuring the the Interior Gateway Protocol to distribute Adj-SIDs conveniently enables that. If ECMP is additionally using the incoming interface of a packet for path selection, an Adj-SID is added between the Node-SID and the MPLS Echo request. As the idea is to determine the incoming interface of the node, whose ECMP path choices are requested by MPLS OAM, the additionally pushed Node-SID here is that of the node preceding the intermediate node, whose forwarding information is requested. The Adj-SID is chosen to correspond to a specific incoming interface of the intermediate node whose forwarding information is requested. As the aim of that test is to ensure that every incoming to outgoing interface path choice of the intermediate node can be addressed, the topology information required to identify the upstream Adj-SID

corresponding to an incoming interface of the intermediate node is assumed to be present at and maintained by the node originating the MPLS data plane failure test. This additional MPLS to IP topology information excerpt results from prior MPLS path validations of the same basic set of MPLS path validations between the source node and the destination node (this is to express, that no extra measurement effort is caused, as correlation of available information is sufficient). The resulting label stack is illustrated by figure 3.

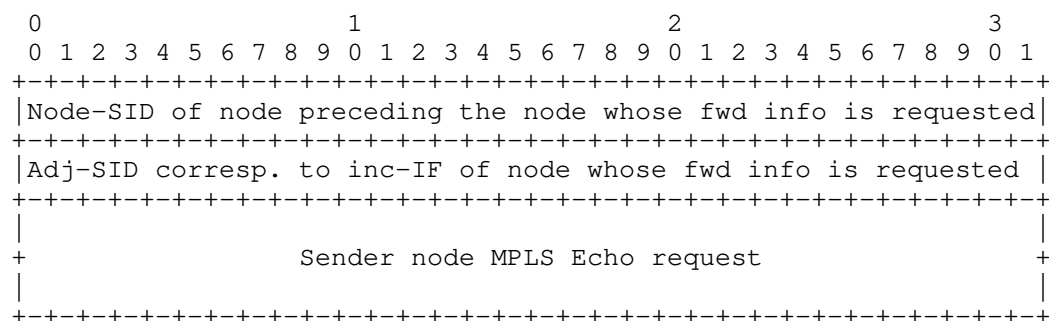


Figure 3

Figure 3: MPLS OAM Label Stack applying SR features if ECMP is additionally based on incoming interfaces.

In the network example of figure 1, node RS picks the Node-SID of R110 and an Adj-SID of R110 corresponding to a particular incoming interface of R120, if the latter's ECMP path also depends on the incoming interface, by which the MPLS Echo request was received.

Here, the full set of original IP-addresses can be forwarded individually per incoming interface of the router whose MPLS forwarding information is requested. In the example above, it is node R120 (not node R110.) Monitoring incoming interface based ECMP results in a higher number of MPLS OAM validations, no matter whether commodity MPLS OAM is applied or the option specified here. The overall sum of tests now is determined by the sum of per node incoming \* outgoing paths (or interfaces, respectively). If the method specified here is applied in the case of the example network,  $2 \cdot (4 \cdot 8 + 4 \cdot 8 + 8 \cdot 12 + 8 \cdot 4 + 12 \cdot 4 + 4 \cdot 4) = 512$  MPLS Echo-Request / Response validations are required. Note that this is still a smaller number as compared to the original 4096 path validations resulting in the case of commodity MPLS OAM based on IP-address information only deployed by a domain applying ECMP. Note that the number of required MPLS OAM path validations is increasing significantly, if ECMP forwarding is in addition based on incoming interfaces and the product of a nodes incoming \* outgoing interfaces is high.

### 3. IANA Considerations

This memo includes no request to IANA.

### 4. Security Considerations

This document does not introduce new functionality. It combines Segment Routing functions with those of MPLS OAM. The related security sections apply, see [RFC8029] and [RFC8402].

### 5. References

#### 5.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC2991] Thaler, D. and C. Hopps, "Multipath Issues in Unicast and Multicast Next-Hop Selection", RFC 2991, DOI 10.17487/RFC2991, November 2000, <<https://www.rfc-editor.org/info/rfc2991>>.
- [RFC8029] Kompella, K., Swallow, G., Pignataro, C., Kumar Nainar, N., Aldrin, S., and M. Chen, "Detecting Multiprotocol Label Switched (MPLS) Data-Plane Failures", RFC 8029, DOI 10.17487/RFC8029, March 2017, <<https://www.rfc-editor.org/info/rfc8029>>.
- [RFC8287] Kumar Nainar, N., Pignataro, C., Swallow, G., Akiya, N., Kini, S., and M. Chen, "Label Switched Path (LSP) Ping/Traceroute for Segment Routing (SR) IGP-Prefix and IGP-Adjacency Segment Identifiers (SIDs) with MPLS Data Planes", RFC 8287, DOI 10.17487/RFC8287, December 2017, <<https://www.rfc-editor.org/info/rfc8287>>.
- [RFC8402] Filsfils, C., Previdi, S., 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>>.

Author's Address

Ruediger Geib (editor)  
Deutsche Telekom  
Heinrich Hertz Str. 3-7  
64295 Darmstadt  
Germany  
Phone: +49 6151 5812747  
Email: Ruediger.Geib@telekom.de

SPRING Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: May 6, 2021

X. Geng  
M. Chen  
F. Yang  
Huawei Technologies  
November 2, 2020

Segment Routing for Redundancy Protection  
draft-geng-spring-sr-redundancy-protection-00

Abstract

Redundancy protection is one of the mechanisms to achieve service protection, following the principle of PREOF (Packet Replication/ Elimination/Ordering Function). To empower the Segment Routing with the capability of redundancy protection, two types of Segment including Redundancy Segment and Merging Segment are introduced. The instantiation of Redundancy and Merging Segments can be applied to both segment routing over MPLS (SR-MPLS) and segment routing over IPv6 (SRv6).

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 .

Status of This Memo

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

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

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

This Internet-Draft will expire on May 6, 2021.



## Copyright Notice

Copyright (c) 2020 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 and Conventions . . . . .	3
3. Redundancy Protection in Segment Routing Scenario . . . . .	3
4. Segment to Support Redundancy Protection . . . . .	4
4.1. Redundancy Segment . . . . .	5
4.2. Merging Segment . . . . .	5
5. Meta Information to Support Redundancy Protection . . . . .	6
6. Segment Routing Policy to Support Redundancy Protection . . . . .	6
7. IANA Considerations . . . . .	6
8. Security Considerations . . . . .	7
9. Acknowledgements . . . . .	7
10. References . . . . .	7
10.1. Normative References . . . . .	7
10.2. Informative References . . . . .	7
Authors' Addresses . . . . .	7

## 1. Introduction

Service protection defined in [RFC8655] is initially required from the use cases in a variety of industries described in [RFC8578]. Together with other two techniques Resource allocation and Explicit routes, targets to provide the deterministic flow transmission. Meanwhile, with the emerge of Cloud VR, Cloud Game, HDV (high-definition video) applications running in the Internet, SLA (Service Level Agreement) guarantee becomes an important issue which requires new technologies and solutions for network.

Redundancy Protection is one of the mechanisms to achieve service protection, following the principle of PREOF (Packet Replication/Elimination/Ordering Function) defined in [RFC8655]. Specifically, replicates the packets of flows into two or more copies, transports

different copies through different path in parallel, eliminates and orders the packets at end to provide redundancy protection.

Segment Routing (SR) leverages the source routing paradigm. An ingress node steers a packet through an ordered list of instructions, called "segments". A segment can be associated to an arbitrary processing of the packet in the node identified by the segment.

This document extends the capabilities in SR paradigm to support the redundancy protection, including the definitions of new Segments and a variation of Segment Routing Policy. The new mechanism applies equally to both segment routing with MPLS data plane (SR-MPLS) and segment routing with IPv6 data plane (SRv6).

## 2. Terminology and Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [I-D.ietf-spring-srv6-network-programming] and[RFC2119].

**Redundancy Node:** the start point of redundancy protection, which is a network device that could implement packet replication

**Merging Node:** the end point of redundancy protection, which is a network node that could implement packet elimination and ordering (optionally)

**Redundancy Policy:** an extended SR policy which includes more than one active segment lists to support redundancy protection

**Flow Identification:** information in SR data service to indicate one flow

**Sequence Number:** information in SR data service to indicate the packet sequence of one flow

**Editor's Note:** Similar mechanism is defined as "Service Protection" in the [RFC8655]. In this document, we define a new term "Redundancy Protection" to distinguish with other service protection method. Some of the terms are similar as [RFC8655].

## 3. Redundancy Protection in Segment Routing Scenario

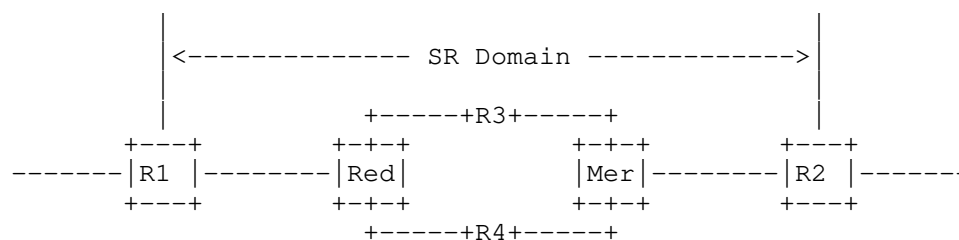


Figure 1: Example Scenario of Redundancy Protection in SR Domain

This figure shows the process of redundancy protection when a flow is sent into SR domain:

- 1) R1 receives the packet flow and encapsulates with segment to destination R2, either instantiated in a stack of MPLS labels or Segment Routing Extension Header (SRH) defined in [RFC8754];
- 2) When the packet flow arrives in Red node, known as Redundancy Node, one flow is replicated to two copies with the same flow identifier; For each packet in one flow, sequence number is marked to indicate the packet sequence; the flow identifier and sequence number of each packet can alternatively be marked at the ingress edge R1 of SR domain;
- 3) Two replicated flows go through different paths till Mer node, known as Merging Node; When there is any failures happened in one path, the service continues to deliver through the other path without break;
- 4) The first received packet of the flow is transmitted from Merging Node to R2, and the redundant packets are eliminated;
- 5) Sometimes, the packet will arrive out of order because of redundancy protection, the function of reordering may be necessary in the Merging Node;

In this example, service protection is supported by utilizing at least two packet flows transmitted over two candidate paths. For a unidirectional flow, Red node supports replication function, and Mer node supports elimination and ordering functions.

#### 4. Segment to Support Redundancy Protection

To achieve the Packet Replication/ Elimination/Ordering Function, Redundancy Segment and Merging Segment are introduced.

#### 4.1. Redundancy Segment

Redundancy Segment is associated with a Redundancy Policy on redundancy node. Redundancy Segment is associated with service instructions, indicating the following operations:

- o Steering the packet into the corresponding redundancy policy
- o Packet replication based on the redundancy policy, e.g., the number of replication copies
- o Encapsulate the packet with necessary meta data (e.g., flow identification, sequence number) if it is not included in the original packet

In the case of SRv6, a new behavior End.R for Redundancy Segment is defined.

When N receives a packet whose IPv6 DA is S and S is a Redundancy SID, N does:

```

S01. When an SRH is processed {
S02.   If (Segments Left>0) {
S03.     Create two headers IPv6+SRH-1 and IPv6+SRH-2
S04.     Insert different policy-instructed segment lists into SRH-1 and SRH
S05.     Add Flow Identification and Sequence Number to SRH-1 and SRH-2
S06.     Remove the incoming outer IPv6+SRH header
S07.     Create a duplication of the incoming packet payload
S08.     Encapsulate the original packet with IPv6+SRH-1 header
S09.     Encapsulate the duplicate packet with IPv6+SRH-2 header
S10.     Set IPv6 SA as the local address of this node
S11.     Set IPv6 DA of IPv6+SRH-1 to the first segment of SRv6 Policy in SR
S12.     Set IPv6 DA of IPv6+SRH-2 to the first segment of SRv6 Policy in SR
S13.   }
S14.   ELSE {
S15.     Drop the packet
S16.   }

```

#### 4.2. Merging Segment

Merging Segment is associated with service instructions, indicates the following operations:

- o Packet merging and elimination: forward the first received packets and eliminate the redundant packets
- o Packet ordering(optional): reorder the packets if the packet arrives out of order

In the case of SRv6, a new behavior End.M for Merging Segment is defined.

When N receives a packet whose IPv6 DA is S and S is a Merging SID, N does:

```
S01.  When an SRH is processed {
S02.      If (Segments Left>0) & "the packet is not a redundant packet" {
S03.          Do not decrement SL nor update the IPv6 DA with SRH[SL]
S04.          Create a new outer IPv6+SRH-3 header
S05.          Insert the policy-instructed segment lists in the newly created SRH
S06.          Remove the incoming outer IPv6+SRH header
S07.          Set IPv6 DA of IPv6+SRH-3 to the first segment of SRv6 Policy in SR
S08.      }
S09.      ELSE {
S10.          Drop the packet
S11.      }
```

## 5. Meta Information to Support Redundancy Protection

To distinguish the different copies replicated at Redundancy node, and distinguish the different packets in the same flow to perform elimination and ordering, the definition of Flow Identification and Sequence Number is required.

Flow Identification and Sequence Number can be defined as MPLS labels in SR over MPLS data plane, or as option TLV in SRH header in SR over IPv6 data plane. This information must be carried along the path to Merging node. Merging node removes Flow Identifier and Sequence Number once the elimination and ordering is accomplished.

## 6. Segment Routing Policy to Support Redundancy Protection

Redundancy Policy is a variation of SR Policy, which is identified through the tuple <redundancy node, redundancy ID, merging node>. Redundancy Policy extends SR policy to include more than one ordered lists of segments between redundancy node and merging node to steer the same flow through different paths in SR domain. In Redundancy Policy, Redundancy Segment MUST be specified, and the last segment of each ordered list of segments SHOULD be Merging Segment.

## 7. IANA Considerations

This document makes no request of IANA.

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

## 8. Security Considerations

TBD

## 9. Acknowledgements

TBD

## 10. References

### 10.1. Normative References

[I-D.ietf-spring-srv6-network-programming]  
Filsfils, C., Camarillo, P., Leddy, J., Voyer, D.,  
Matsushima, S., and Z. Li, "SRv6 Network Programming",  
draft-ietf-spring-srv6-network-programming-24 (work in  
progress), October 2020.

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

### 10.2. Informative References

[RFC8578] Grossman, E., Ed., "Deterministic Networking Use Cases",  
RFC 8578, DOI 10.17487/RFC8578, May 2019,  
<<https://www.rfc-editor.org/info/rfc8578>>.

[RFC8655] Finn, N., Thubert, P., Varga, B., and J. Farkas,  
"Deterministic Networking Architecture", RFC 8655,  
DOI 10.17487/RFC8655, October 2019,  
<<https://www.rfc-editor.org/info/rfc8655>>.

[RFC8754] Filsfils, C., Ed., Dukes, D., Ed., Previdi, S., Leddy, J.,  
Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header  
(SRH)", RFC 8754, DOI 10.17487/RFC8754, March 2020,  
<<https://www.rfc-editor.org/info/rfc8754>>.

## Authors' Addresses

Xuesong Geng  
Huawei Technologies

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

Mach (Guoyi) Chen  
Huawei Technologies

Email: mach.chen@huawei.com

Fan Yang  
Huawei Technologies

Email: shirley.yangfan@huawei.com

SPRING Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: 3 February 2022

X. Geng  
M. Chen  
F. Yang, Ed.  
Huawei Technologies  
P. Camarillo  
Cisco Systems, Inc.  
G. Mishra  
Verizon Inc.  
2 August 2021

SRv6 for Redundancy Protection  
draft-geng-spring-sr-redundancy-protection-05

Abstract

Redundancy Protection is a generalized protection mechanism to achieve the high reliability of service transmission in Segment Routing network. The mechanism inherits the "Live-Live" methodology, targeting to enhance the functionalities of Segment Routing over IPv6. Inspired by DetNet Packet Replication and Packet Elimination functions, two new Segments are introduced to provide replication and elimination functions on specific network nodes by leveraging SRv6 Segment programming capabilities.

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 .

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 3 February 2022.



## Copyright Notice

Copyright (c) 2021 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 . . . . .	3
2.1. Requirements Language . . . . .	3
2.2. Terminology and Conventions . . . . .	3
3. Redundancy Protection in Segment Routing Scenario . . . . .	4
4. Segment to Support Redundancy Protection . . . . .	5
4.1. Redundancy Segment . . . . .	5
4.2. Merging Segment . . . . .	6
5. Meta Data to Support Redundancy Protection . . . . .	7
6. Segment Routing Policy to Support Redundancy Protection . . . . .	7
7. IANA Considerations . . . . .	7
8. Security Considerations . . . . .	8
9. Acknowledgements . . . . .	8
10. References . . . . .	8
10.1. Normative References . . . . .	8
10.2. Informative References . . . . .	8
Authors' Addresses . . . . .	8

## 1. Introduction

Redundancy Protection is a generalized protection mechanism to achieve the high reliability of service transmission in Segment Routing network. Specifically, packets of flows are replicated at a network node into two or more copies, which are transported via different paths in parallel. When copies of packets are received and merged at one network node, the redundant packets are determined and further eliminated to guarantee only one copy of packets is transmitted. The mechanism inherits the "Live-Live" methodology, targeting to enhance the functionalities of Segment Routing over IPv6 [RFC8986]. Inspired by DetNet [RFC8655] Packet Replication and Packet Elimination Functions, two new Segments are introduced to provide the replication and elimination functions on specific network

nodes by leveraging SRv6 Segment programming capabilities. As it is unnecessary to perform switchover between different paths triggered by failure detection, redundancy protection can facilitate to achieve zero packet loss target when failure on either path happens.

Redundancy protection provides ultra reliable protection to many services, for example Cloud VR/Game, IPTV service and other type of video services, high value private line service etc. In this document, redundancy protection is applied to point-to-point service. The mechanism for P2MP service stays out of the scope of this document.

Segment Routing (SR) leverages the source routing paradigm. An ingress node steers a packet through an ordered list of instructions, called "segments". A segment can be associated to an arbitrary processing of the packet in the node identified by the segment.

This document extends the Segment Routing capabilities to support the redundancy protection in an SRv6 environment, including the definitions of two new Segments, meta data encapsulation, and a variation of Segment Routing Policy.

## 2. Terminology

### 2.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

### 2.2. Terminology and Conventions

SR: Segment Routing

URLLC: Ultra-Reliable Low-Latency Communication

VR: Virtual Reality

Red Node: Redundancy Node

Mer Node: Merging Node

FID: Flow IDentification

SN: Sequence Number

### 3. Redundancy Protection in Segment Routing Scenario

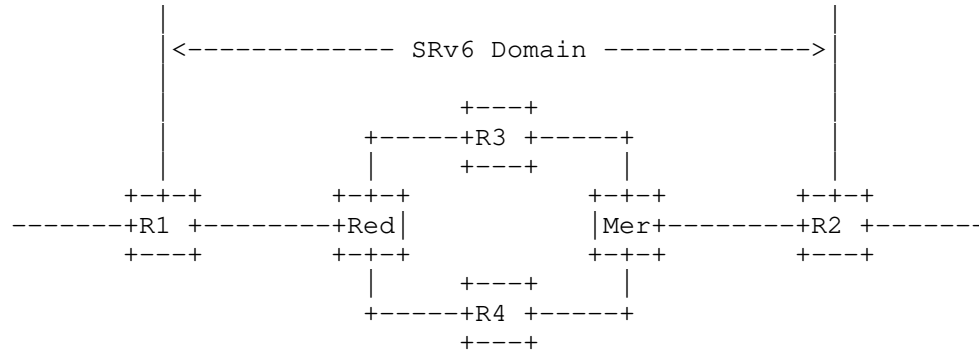


Figure 1: Example Scenario of Redundancy Protection in SRv6 Domain

This figure shows an example of redundancy protection used in SRv6 domain. R1, R2, R3, R4, Red and Mer are SR-capable nodes. When a flow is sent into SRv6 domain, the process is:

- 1) R1 receives the traffic flow and encapsulates packets with a list of segments destined to R2, which is instantiated as an ordered list of SRv6 SIDs.
- 2) When the packet flow arrives at Red node, known as Redundancy Node, each packet is replicated into two or more copies. Each copy of the packet is encapsulated with a new segment list, which represents different disjoint forwarding paths.
- 3) Meta data information such as flow identification (FID) and sequence number (SN) is used to facilitate the packet elimination on Merging node (Mer). Flow identification identifies the specific flow, and sequence number distinguishes the packet sequence of a flow. Meta data is either carried in the packet before it arrives at Red node, or added to each of the replicas at Red node.
- 4) The multiple replicas go through different paths until the Mer node. The first received packet of the flow is transmitted from Merging Node to R2, and the redundant packets are eliminated.
- 5) When there is any failures or packet loss in one path, the service continues undisrupted through the other path without break.

6) Sometimes, the packet will arrive out of order because of redundancy protection, the function of reordering may be also necessary on Merging Node. In such case the Merging node may include a reordering function, which is implementation specific and out of the scope of this document.

In this example, service protection is supported by utilizing two packet flows transmitted over two forwarding paths. It is noted that there is no limitation of the number of replicas. For a unidirectional flow, Red node supports replication function, and Mer node supports elimination function. Reordering function MAY be required in combination of elimination function on merging node. To minimize the jitter caused by random packet loss, the disjoint paths are recommended to have similar path forwarding delay.

#### 4. Segment to Support Redundancy Protection

To achieve the packet replication and elimination functions, Redundancy Segment and Merging Segment, as well as the related SRv6 Endpoint Behavior are introduced.

##### 4.1. Redundancy Segment

Redundancy Segment is the identifier of packets which need the replication function on redundancy node. It is also a variation of Binding SID, and associated with a Redundancy Policy to provide segment lists of disjoint paths. Thus, Redundancy segment is associated with service instructions, indicating the following operations:

- \* Steers the packet into the corresponding redundancy policy
- \* Encapsulates flow identification and sequence number in packets if the two information is not carried in packets
- \* Packet replication and segment encapsulation based on the information of redundancy policy, e.g., the number of replication copies, an ordered list of segments with a topological instruction

In the case of SRv6, a new behavior End.R for Redundancy Segment is defined. An instance of a redundancy SID is associated with a redundancy policy B and a source address A. In the following description, End.R behavior is specified in the encapsulation mode. The End.R behavior in the insertion mode is for further study.

When an SRv6-capable node (N) receives an IPv6 packet whose destination address matches a local IPv6 address instantiated as an SRv6 SID (S), and S is a Redundancy SID, N does:

```
S01. When an SRH is processed {
S02.   If (Segments Left>0)   {
S03.     Decrement IPv6 Hop Limit by 1
S04.     Decrement Segments Left by 1
S05.     Update IPv6 DA with Segment List[Segments Left]
S06.     Add flow identification and sequence number if indicated*
S07.     Duplicate the packets (as number of active SID lists in B)
S08.     Push the new IPv6 headers to each replica. The IPv6 header
        contains an SRH with the SID list in B
S09.     Set the outer IPv6 SA to A
S10.     Set the outer IPv6 DA to the first SID of new SRH SL
S11.     Set the outer Payload Length, Traffic Class, Flow Label,
        Hop Limit and Next-Header fields
S12.     Submit the packet to the egress IPv6 FIB lookup
        for transmission to the new destination
S13.   }
S14. }
```

\* Adding flow identification and sequence number is an optional behavior for Redundancy Segment. The instruction execution is determined and explicitly indicated by SR policy or Segment itself.

#### 4.2. Merging Segment

Merging Segment is associated with service instructions, indicates the following operations:

- \* Packet merging and elimination: forward the first received packets and eliminate the redundant packets

In order to eliminate the redundant packet of a flow, merging node utilizes sequence number to evaluate the redundant status of a packet. Note that implementation specific mechanism could be applied to control the amount of state monitored on sequence number, so that system memory usage can be limited at a reasonable level.

As merging node needs to maintain the state of flows, a centralized controller should have a knowledge of merging nodes capability, and never provision the redundancy policy to redundancy node when the computation result goes beyond the flow recovery capability of merging node. The capability advertisement of merging node will be specified separately elsewhere, which is not within the scope of this document.

In the case of SRv6, a new behavior End.M for Merging Segment is defined.

When an SRv6-capable node (N) receives an IPv6 packet whose destination address matches a local IPv6 address instantiated as an SRv6 SID (S), and S is a Merging SID, N does:

```
S01. When an SRH is processed {
S02.   If (Segments Left > 0) {
S03.     Acquire the sequence number of received packet and
        look it up in table
S04.     If (this sequence number does not exist in the table) {
S05.       Store this sequence number in table
S06.       Remove the outer IPv6+SRH header
S07.       Decrement IPv6 Hop Limit by 1 in inner SRH
S08.       Decrement Segments Left by 1 in inner SRH
S09.       Update IPv6 DA with Segment List[Segments Left] in inner SRH
S10.       Submit the packet to the egress IPv6 FIB lookup and transmit
S11.     }
S12.   ELSE {
S13.     Drop the packet
S14.   }
S15. }
S16. }
```

## 5. Meta Data to Support Redundancy Protection

To support the redundancy protection function, flow identification and sequence number are required. Flow identification identifies one specific flow of redundancy protection, and is usually allocated from centralized controller to the SR ingress node or redundancy node in SR network. Sequence number distinguishes the packets within a flow by specifying the order of packets. It is usually generated at SR ingress node. If necessary, redundancy node can also facilitate to add sequence number if required. Thus, encapsulations of flow identification and sequence number should be specified accordingly.

## 6. Segment Routing Policy to Support Redundancy Protection

Redundancy Policy is a variation of SR Policy to conduct the replicas to multiple disjoint paths for redundancy protection. It extends SR policy to include more than one ordered lists of segments between redundancy node and merging node, and all the ordered lists of segments are used at the same time to steer the copies of flow into different disjoint paths.

## 7. IANA Considerations

This document requires registration of End.R behavior and End.M behavior in "SRv6 Endpoint Behaviors" sub-registry of "Segment Routing Parameters" registry.

## 8. Security Considerations

TBD

## 9. Acknowledgements

The authors would like to thank Bruno Decraene, Ron Bonica, James Guichard, Jeffrey Zhang, Balazs Varga for their valuable comments and discussions.

## 10. References

### 10.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8986] Filsfils, C., Ed., Camarillo, P., Ed., Leddy, J., Voyer, D., Matsushima, S., and Z. Li, "Segment Routing over IPv6 (SRv6) Network Programming", RFC 8986, DOI 10.17487/RFC8986, February 2021, <<https://www.rfc-editor.org/info/rfc8986>>.

### 10.2. Informative References

- [RFC8655] Finn, N., Thubert, P., Varga, B., and J. Farkas, "Deterministic Networking Architecture", RFC 8655, DOI 10.17487/RFC8655, October 2019, <<https://www.rfc-editor.org/info/rfc8655>>.

## Authors' Addresses

Xuesong Geng  
Huawei Technologies  
China

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

Mach(Guoyi) Chen  
Huawei Technologies  
China

Email: mach.chen@huawei.com

Fan Yang  
Huawei Technologies  
China

Email: shirley.yangfan@huawei.com

Pablo Camarillo Garvia  
Cisco Systems, Inc.  
Spain

Email: pcamaril@cisco.com

Gyan Mishra  
Verizon Inc.

Email: gyan.s.mishra@verizon.com



SPRING  
Internet-Draft  
Intended status: Standards Track  
Expires: March 25, 2021

S. Hegde  
C. Bowers  
Juniper Networks Inc.  
X. Xu  
Alibaba Inc.  
A. Gulko  
Refinitiv  
A. Bogdanov  
Google Inc.  
J. Uttaro  
ATT  
L. Jalil  
Verizon  
September 21, 2020

Seamless Segment Routing  
draft-hegde-spring-mpls-seamless-sr-02

Abstract

In order to operate networks with large numbers of devices, network operators organize networks into multiple smaller network domains. Each network domain typically runs an IGP which has complete visibility within its own domain, but limited visibility outside of its domain. Seamless Segment Routing (Seamless SR) provides flexible, scalable and reliable end-to-end connectivity for services across independent network domains. Seamless SR accommodates domains using SR, LDP, and RSVP for MPLS label distribution as well as domains running IP without MPLS (IP-Fabric).

Requirements Language

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

Status of This Memo

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

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

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

This Internet-Draft will expire on March 25, 2021.

#### Copyright Notice

Copyright (c) 2020 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 . . . . .	3
2. Terminology . . . . .	4
3. Use Cases . . . . .	5
3.1. Service provider network . . . . .	5
3.2. Large scale WAN networks . . . . .	7
3.3. Data Center Interconnect (DCI) Networks . . . . .	8
3.4. Multicast Use cases . . . . .	8
4. Requirements . . . . .	9
4.1. MPLS Transport . . . . .	9
4.2. SLA Guarantee . . . . .	10
4.3. Scalability . . . . .	10
4.4. Availability . . . . .	10
4.5. Operations . . . . .	10
4.6. Service Mapping . . . . .	11
5. Seamless Segment Routing architecture . . . . .	11
5.1. Solution Concepts . . . . .	11
5.2. BGP Classful Transport . . . . .	12
5.3. Automatically Creating Transport Classes . . . . .	17
5.3.1. Automatically Creating Transport Classes for BGP-SR- TE Intra-domain Tunnels . . . . .	17
5.3.2. Automatically Creating Transport Classes for Flex- Algo Tunnels . . . . .	17
5.3.3. Auto-deriving Transport Classes for PCEP . . . . .	18
5.4. Inter-domain flex-algo with BGP-CT . . . . .	18

5.5. Applicability to color-only policies . . . . .	18
5.6. Data sovereignty . . . . .	18
5.7. Interconnecting IP Fabric Data Centers . . . . .	20
5.8. Translating Transport Classes across Domains . . . . .	22
5.9. SLA Guarantee . . . . .	23
5.9.1. Low latency . . . . .	23
5.9.2. Traffic Engineering (TE) constraints . . . . .	23
5.9.3. Bandwidth constraints . . . . .	24
5.10. Scalability . . . . .	24
5.10.1. Access node scalability . . . . .	24
5.10.2. Label stack depth . . . . .	24
5.10.3. Label Resources . . . . .	25
5.11. Availability . . . . .	28
5.11.1. Intra domain link and node protection . . . . .	28
5.11.2. Egress link and node protection . . . . .	28
5.11.3. Border Node protection . . . . .	28
5.12. Operations . . . . .	29
5.12.1. MPLS ping and Traceroute . . . . .	29
5.12.2. Counters and Statistics . . . . .	29
5.13. Service Mapping . . . . .	29
5.14. Migrations . . . . .	30
5.15. Interworking with v6 transport technologies . . . . .	30
5.16. BGP based Multicast . . . . .	30
6. Backward Compatibility . . . . .	31
7. Security Considerations . . . . .	31
8. IANA Considerations . . . . .	31
9. Acknowledgements . . . . .	31
10. Contributors . . . . .	31
11. References . . . . .	31
11.1. Normative References . . . . .	31
11.2. Informative References . . . . .	32
Authors' Addresses . . . . .	35

## 1. Introduction

Evolving wireless access technology and cloud applications are expected to place new requirements on the packet transport networks. These services are contributing to significantly higher bandwidth throughput which in turn leads to a growing number of transport network devices. As an example, 5G networks are expected to require up to 250Gbps in the fronthaul and up to 400Gbps in the backhaul. There is a desire to allow many network functions to be virtualized and cloud native. In order to support latency-sensitive cloud-native network functions, packet transport networks should be capable of providing low-latency paths end-to-end. Some services will require low-latency paths while others may require different QoS properties. The network should be able to differentiate between the services and provide corresponding SLA transport paths. In addition, as these

applications become more sensitive and less loss tolerant, more and more emphasis is placed on overall service availability and reliability.

The Seamless SR architecture builds upon the Seamless MPLS architecture and caters to new requirements imposed by the 5G transport networks and the cloud applications. [I-D.ietf-mpls-seamless-mpls], contains a good description of the Seamless MPLS architecture. Although [I-D.ietf-mpls-seamless-mpls] has not been published as an RFC, it serves as a useful description of the Seamless MPLS architecture. [I-D.ietf-mpls-seamless-mpls] describes the Seamless MPLS architecture, which uses LDP and/or RSVP for intra-domain label distribution, and BGP-LU [RFC3107] for end-to-end label distribution. Seamless SR focuses on using segment routing for intra-domain label distribution. The mechanisms described in this document are equally applicable to intra-domain tunneling mechanisms deployed using RSVP and/or LDP.

By using segment routing for intra-domain label distribution, Seamless SR is able to easily support both SR-MPLS on IPv4 and IPv6 networks. This overcomes a limitation of the classic Seamless MPLS architecture, which was limited to run MPLS on IPv4 networks in practice. Seamless SR (like Seamless MPLS) can use BGP-LU (RFC 3107) to stitch different domains. However, Seamless SR can also take advantage of BGP Prefix-SID [RFC8669] to provide predictable and deterministic labels for the inter-domain connectivity.

The basic functionality of the Seamless SR architecture does not require any enhancements to existing protocols. However, in order to support end-to-end service requirements across multiple domains, protocol extensions may be needed. This draft discusses use cases, requirements, and potential protocol enhancements.

## 2. Terminology

This document uses the following terminology

- o Access Node (AN): An access node is a node which processes customers frames or packets at Layer 2 or above. This includes but is not limited to DSLAMs and Cell Site Routers in 5G networks. Access nodes have only limited MPLS functionalities in order to reduce complexity in the access network.
- o Pre-Aggregation Node (P-AGG): A pre-aggregation node (P-AGG) is a node which aggregates several access nodes (ANs).
- o Aggregation Node (AGG): A aggregation node (AGG) is a node which aggregates several pre-aggregation nodes (P-AGG).
- o Area Border Router (ABR): Router between aggregation and core domain.
- o Label Switch Router (LSR): Label Switch router are pure transit nodes. ideally have no customer or service state and are therefore decoupled from service creation.
- o Use Case: Describes a typical network including service creation points and distribution of remote node loopback prefixes.

Figure 1: Terminology

### 3. Use Cases

#### 3.1. Service provider network

Service provider transport networks use multiple domains to support scalability. For this analysis, we consider a representative network design with four level of hierarchy: access domains, pre-aggregation domains, aggregation domains and a core. (See Figure 2). The 5G transport networks in particular are expected to scale to very large number of access nodes due to the shorter range of the 5G radio technology. The networks are expected to scale up to one million nodes.

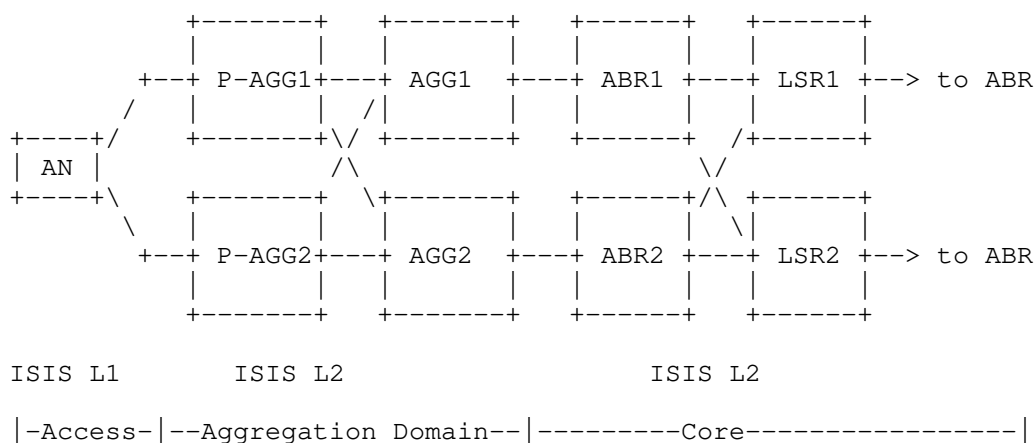


Figure 2: 5G network

Many network functions in a 5G network will be virtualized/containerized and distributed across multiple data centers. Virtualized network functions are instantiated dynamically across different compute resources. This requires that the underlying transport network supports the stringent SLA on end-to-end paths.

5G networks support variety of service use cases that require end-to-end slicing. In certain cases the end-to-end connectivity requires differentiated forwarding capabilities. Seamless SR architecture should provide the ability to establish end-to-end paths that satisfy the required SLAs. For example, end user requirement could be to establish a low latency path end-to-end. The System Architecture for the 5G System [TS.23.501-3GPP] currently defines four standardized Slice/Service Types: Enhanced Mobile Broadband (eMBB), Ultra-Reliable Low Latency Communication (URLLC), massive Internet of Things (mIoT), Vehicle to everything (V2X). The Seamless SR should support end-to-end Service Level Objectives(SLO) to allow the creation of network slices with these four Slice/Service Types.

Many deployments consist of ring topologies in the access and aggregation networks. In the ring topologies, there are at most two forwarding paths for the traffic, whereas the core networks consist of nodes with more denser connectivity compared to ring topologies. Thus core networks may have a larger number of TE paths while access networks will have a smaller number of TE paths. The Seamless SR architecture should support the ability to have more TE paths in one domain and lesser number of TE paths in another domain and provide the ability to effectively connect the domains end-to-end while satisfying end-to-end constraints.

### 3.2. Large scale WAN networks

As WAN networks grow beyond several thousand nodes, it is often useful to divide the network into multiple IGP domains, as illustrated in Section 3.2. Separate IGP domains increase service availability by establishing a constrained failure domain. Smaller IGP domains may also improve network performance and health by reducing the device scale profile (including protocol and FIB scale).

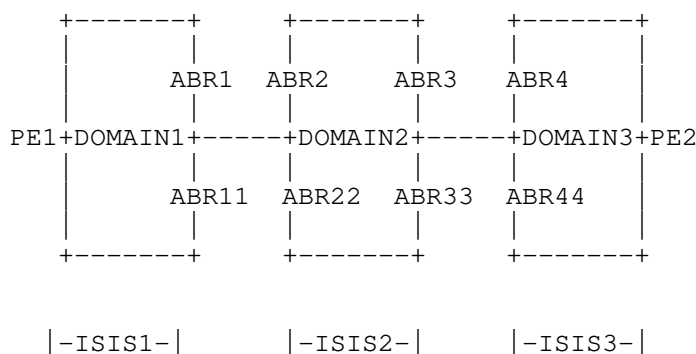


Figure 3: WAN Network

These Large WAN networks often cross national boundaries. In order to meet data sovereignty requirements, operators need to maintain strict control over end-to-end traffic-engineered(TE) paths. Segment Routing provides two main solutions to implement highly constrained TE paths. Flex-algo (defined in [I-D.ietf-lsr-flex-algo]) uses prefix-SIDs computed by all nodes in the IGP domain using the same pruned topology. Highly constrained TE paths for the data sovereignty use case can also be implemented using SR-TE policies ([I-D.ietf-spring-segment-routing-policy]) built using unprotected adjacency SIDs.

Both of these approaches work well for intra-domain TE paths. However, they both have limitations when one tries to extend them to the creation of highly constrained inter-domain TE paths. A goal of seamless SR is to be able to create highly constrained inter-domain TE paths in a scalable manner.

Some deployments may use a centralized controller to acquire the topologies of multiple domains and build end-to-end constrained paths. This can be scaled with hierarchical controllers. However, there is still significant risk of a loss of network connectivity to one or more controllers, which can result in a failure to satisfy the

strict requirements of data sovereignty. The network should have pre-established TE paths end-to-end that don't rely on controllers in order to address these failure scenarios.

### 3.3. Data Center Interconnect (DCI) Networks

Data centers are playing an increasingly important role in providing access to information and applications. Geographically diverse data centers usually connect via a high speed, reliable and secure core network.

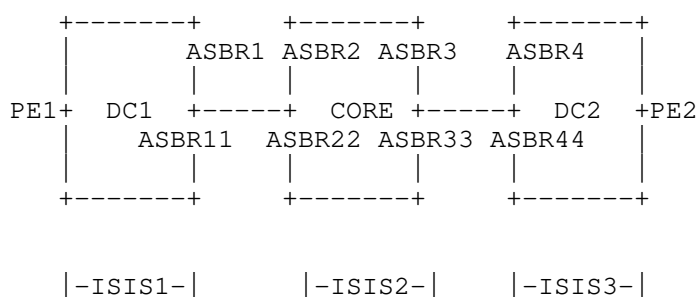


Figure 4: DCI Network

In many Data Center deployments, applications require end-to-end path diversity and/or end-to-end low latency paths. It is desirable to have a uniform technology deployed in the core as well as in the Data Centers to create these SLA paths. Such uniformity simplifies the network to a great extent. It is desirable for a solution to only require service-related configurations on the access end-points where services are attached, avoiding service-related configurations on the ABR/ASBR nodes.

### 3.4. Multicast Use cases

Multicast services such as IPTV and multicast also need to be support across a multi-domain service provider network. Multicast services such as IPTV, multicast VPN etc need to be supported in a service provider network.



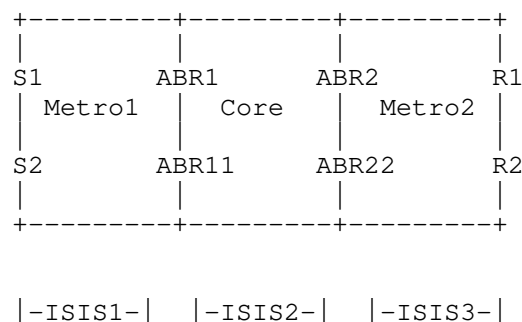


Figure 5: Multicast usecases

Figure 5 shows a simplified multi-domain network supporting multicast. Multicast sources S1 and S2 lie in a different domain from the receivers R1 and R2. Using multiple IGP domains presents a problem for the establishment of multicast replication trees. Typically, a multicast receiver does a reverse path forwarding (RPF) lookup for a multicast source. One solution is to leak the routes for multicast sources across the IGP domains. However, this can compromise the scaling properties of the multi-domain architecture. SR-P2MP [I-D.voyer-pim-sr-p2mp-policy] offers a solution for both intra-domain and inter-domain multicast. However, it does not accommodate deployments using existing intra-domain multicast technology, such as mLDP [RFC6388] in some of the domains. A solution should accommodate a mixture of existing and newer technologies to better facilitate coexistence and migration.

#### 4. Requirements

This section provides a summary of requirements derived from the use cases described in previous sections.

##### 4.1. MPLS Transport

The architecture SHOULD provide MPLS transport between two service endpoints regardless of whether the two end-points are in the same IGP domain, different IGP domains, or in different autonomous systems.

The MPLS transport SHOULD be supported on IPv4, IPv6, and dual-stack networks.

#### 4.2. SLA Guarantee

The architecture SHOULD allow the creation of paths that support end-to-end SLAs. The paths should for example obey constraints related to latency, diversity, bandwidth and availability.

The architecture SHOULD support end-to-end network slicing as described by 5G transport requirements [TS.23.501-3GPP].

#### 4.3. Scalability

The architecture SHOULD be able to support up to 1 million nodes.

The architecture SHOULD facilitate the use of access nodes with low RIB/FIB and low CPU capabilities.

The architecture SHOULD facilitate the use of access nodes with low label stacking capability.

The architecture SHOULD allow for a scalable response to network events. An individual node SHOULD only need to respond to a limited subset of network events.

Service routes on the border nodes SHOULD be minimized.

#### 4.4. Availability

Traffic SHOULD be Fast Reroute (FRR) protected against link, node, and SRLG failures within a domain.

Traffic SHOULD be Fast Reroute (FRR) protected against border node failures.

Traffic SHOULD be Fast Reroute (FRR) protected against egress node and egress link failures.

#### 4.5. Operations

Each domain SHOULD be independent and SHOULD not depend on the transport technology in another domain. This allows for more flexible evolution of the network.

Basic MPLS OAM mechanisms described in [RFC8029] SHOULD be supported.

End-to-end mpls ping and traceroute procedures SHOULD be supported.

Ability to validate the path inside each domain SHOULD be supported.

Statistics for inter-domain paths on the ingress and egress PE nodes as well as border nodes SHOULD be supported.

#### 4.6. Service Mapping

The architecture SHOULD support the automated steering of traffic on to transport paths based on communities carried in the service prefix advertisements.

The architecture SHOULD support the steering of traffic on to transport paths based on the DSCP value carried in IPv4/IPv6 packets.

Traffic steering based on EXP bits in the mpls header SHOULD be supported.

Traffic steering based on 5-tuple packet filter SHOULD be supported. Source address, destination address, source port, destination port and protocol fields should be allowed.

All traffic steering mechanisms SHOULD be supported for all kinds of service traffic including VPN traffic as well as global internet traffic.

The core domain is expected to have more traffic engineering constraints as compared to metros. The ability to map the services to appropriate transport tunnels at service attachment points SHOULD be supported.

### 5. Seamless Segment Routing architecture

#### 5.1. Solution Concepts

The solution described below makes use of the following concepts.

- o Transport Class (TC): A Transport Class is defined as a collection of end-to-end MPLS paths that satisfy a set of constraints or Service Level Agreements.
- o BGP-Classful Transport (BGP-CT): A new BGP family used to establish Transport Class paths across different domains.
- o Route Distinguisher (RD): The Route Distinguisher is defined in RFC4364. In BGP-CT, the RD is used in BGP advertisements to differentiate multiple paths to the same loopback address. It may be useful to automatically generate RDs in order to simplify configuration.
- o Route Target (RT): The Route Target extended community is carried in BGP-CT advertisements. The RT represents the Transport Class of an advertised path. Note that the RT is only carried in the BGP-CT advertisements. No BGP-VPN related configuration or VPN family advertisements are needed when BGP-CT transport paths are used to carry non-VPN traffic.
- o Mapping Community (MC): The Mapping Community is the BGP extended community as defined in RFC4360. In the Seamless SR architecture, an MC is carried by a BGP-CT route and/or a service route. The MC is used to identify the specific local policy used to map traffic for a service route to different Transport Class paths. When a mapping community is advertised in a BGP-CT route it identifies the specific local policy used to map the BGP-CT route to the intra-domain tunnels. The local policy can include additional traffic steering properties for placing traffic on different Transport Class paths. The values of the MCs and the corresponding local policies for service mapping are defined by the network operator.

Figure 6: Solution Concepts

## 5.2. BGP Classful Transport

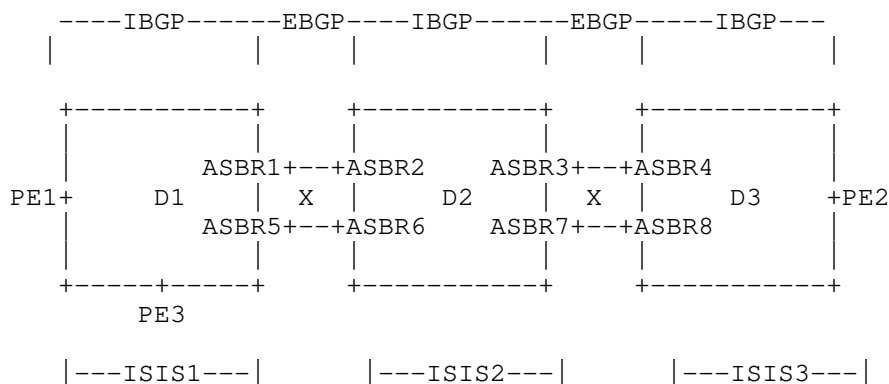


Figure 7: WAN Network

The above diagram shows a WAN network divided into 3 different domains. Within each domain, BGP sessions are established between the PE nodes and the border nodes as well as between border nodes. BGP sessions are also established between border nodes across domains. The goal is for PE1 to have MPLS connectivity to PE2, satisfying specific characteristics. Multiple MPLS paths from PE1 to PE2 are required in order to satisfy different SLAs.

[I-D.kaliraj-idr-bgp-classful-transport-planes] defines a new BGP family called BGP-Classful Transport. The NLRI for this new family consists of a prefix and a Route Distinguisher. The prefix corresponds to the loopback of the destination PE, and RD is used to distinguish different paths to the same PE loopback. The BGP-CT advertisement also carries a Route Target. The RT specifies the Transport Class to which the BGP-CT advertisement belongs. BGP-CT mechanisms are applicable to single ownership networks that are organized into multiple domains. It is also applicable to multiple ASes with different ownership but closely co-operating administration. BGP-CT mechanisms are not expected to be applied on the internet peering or between domains that have completely independent administrations.

## BGP-CT advertisements for red Transport Class

Prefix:PE2	Prefix:PE2	Prefix:PE2	Prefix:PE2	Prefix:PE2
RD:RD1	RD:RD1	RD:RD1	RD:RD1	RD:RD1
RT:Red	RT:Red	RT:Red	RT:Red	RT:Red(100)
nh:ASBR1	nh:ASBR2	nh:ASBR3	nh:ASBR4	nh:PE2
Label:L1	Label:L2	Label:L3	Label:L4	Label:L5

PE1-----ASBR1-----ASBR2-----ASBR3-----ASBR4-----PE2

VPNa Prefix:  
10.1.1.1/32  
RD: RD50  
RT: RT-VPNa  
ext-community:  
Red(100)  
nh: PE2  
Label: S1

+-----+		+-----+		+-----+
IL71		IL72		IL73
+-----+	+-----+	+-----+	+-----+	+-----+
L1	L2	L3	L4	L5
+-----+	+-----+	+-----+	+-----+	+-----+
S1	S1	S1	S1	S1
+-----+	+-----+	+-----+	+-----+	+-----+

Label stacks along end-to-end path

S1 is the end-to-end service label.

IL71, IL72, and IL73 are intra-domain labels corresponding to red intra-domain paths.

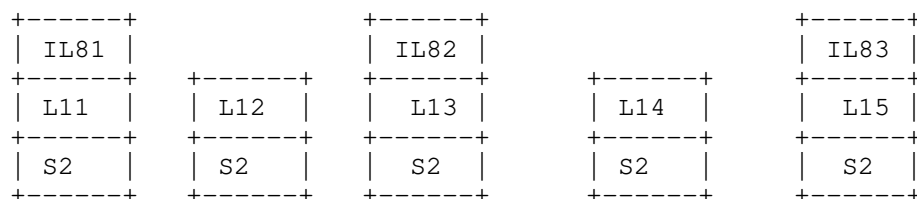
Figure 8: BGP-CT Advertisements and Label Stacks

## BGP-CT advertisements for blue Transport Class

Prefix:PE2	Prefix:PE2	Prefix:PE2	Prefix:PE2	Prefix:PE2
RD:RD2	RD:RD2	RD:RD2	RD:RD2	RD:RD2
RT:Blue	RT:Blue	RT:Blue	RT:Blue	RT:Blue (200)
nh:ASBR1	nh:ASBR2	nh:ASBR3	nh:ASBR4	nh:PE2
Label:L11	Label:L12	Label:L13	Label:L14	Label:L15

PE1-----ASBR1----ASBR2-----ASBR3-----ASBR4-----PE2

VPNb Prefix:  
10.1.1.1/32  
RD: RD51  
RT: RT-VPNb  
ext-community:  
Blue (200)  
nh: PE2  
Label: S2



Label stacks along end-to-end path  
S2 is the end-to-end service label.

IL81, IL82, and IL83 are intra-domain labels corresponding to blue intra-domain paths.

Figure 9: BGP-CT Advertisements and Label Stacks

For example, consider the diagram in Figure 8 and Figure 9 . The diagram shows the BGP-CT advertisements corresponding to two different end-to-end paths between PE1 and PE2. The two different paths belong to two different Transport Classes, red and blue.

The inter-domain paths created by BGP-CT Transport Classes can be used by any traffic that can be steered using BGP next-hop resolution, including vanilla IPv4 and IPv6, L2VPN, L3VPN, and eVPN. In the example above, we show how traffic from two different L3VPNs (VPN<sub>a</sub> and VPN<sub>b</sub>) is mapped onto two different BGP-CT Transport Classes (Red and Blue). The L3VPN advertisements for VPN<sub>a</sub> and VPN<sub>b</sub> are originated by PE2 as usual. PE1 receives these L3VPN advertisements

and uses the next-hop in the L3VPN advertisements to determine the path to use. In the absence of any BGP-CT Transport Classes in the network, PE1 would likely resolve the L3VPN next-hop over BGP-LU routes corresponding to the BGP best path. However, when BGP-CT Transport Classes are used, PE1 will resolve the L3VPN next-hop over a BGP-CT route.

In the example above, PE2 originates BGP-CT advertisements for the Red and Blue Transport Classes. These BGP-CT advertisements propagate across the multiple domains, causing forwarding state for the two Transport Classes to be installed at ABRs along the way. In order to create unique NLRIs for the two advertisements, PE2 uses two different RDs. In the example above, the red BGP-CT advertisement has an RD of RD1 and the blue BGP-CT advertisement has an RD of RD2. Note that the RD values used in the BGP-CT advertisement are completely independent of the RD values used in the L3VPN advertisements. In both cases, the RD values are simply a mechanism to guarantee uniqueness of a prefix/RD pair.

The RT values used in the BGP-CT advertisements are unrelated to the RT values used on the L3VPN advertisements. The L3VPN RT values identify VPN membership, as usual. The BGP-CT RT values identify Transport Class membership. In order to be able to easily map VPN traffic into BGP-CT Transport classes, it can be useful however to make an association between BGP-CT RT values and color extended community values in the L3VPN advertisements. In the example above, the RT value carried in the BGP-CT advertisement originated from PE2 for the red Transport Class is configured to correspond to the color extended community advertised in the VPN advertisement for VPNa. Similarly, the RT value for the blue Transport Class corresponds to the color extended community for VPNb. In this way, traffic on PE1 for each VPN can be mapped to a transport class path by associating the value of the color extended community carried in the VPN advertisement with an RT value carried in a BGP-CT advertisement.

The example above also shows the label stacks at different points along the end-to-end paths for the forwarding entries which are established by the two advertisements. Labels L1-L4 are red BGP-CT labels advertised by border nodes ASBR1,2,3, and 4, while label L5 is advertised by PE2 for the red Transport Class. Labels L11-L14 are blue BGP-CT labels advertised by border nodes ASBR1,2,3, and 4, while label L15 is advertised by PE2 for the blue Transport Class.

IL71, IL72, and IL73 represent tunnels internal to the domains 1, 2, and 3 which correspond to the red Transport Class. IL81, IL82, and IL83 represent tunnels internal to the domains 1, 2, and 3 which correspond to the blue Transport Class. In this example, we assume that the intra-domain tunnels correspond to SRTE policies having red



SRTE-policy-color and blue SRTE-policy-color. Service labels are represented by S1 and S2.

Note that this example focuses on how signalling originated by PE2 results in forwarding state used by PE1 to reach PE2 on a specific Transport Class path. The solution supports the establishment of forwarding state for an arbitrary number of PEs to reach PE2. For example, PE3 in Figure 8 can reach PE2 on a red Transport Class path established using the same BGP-CT signalling. The signalling and forwarding state from ASBR1 all the way to PE2 is common to the paths used by both PE1 and PE3. This merging of signalling and forwarding state is essentially to the good scaling properties of the Seamless SR architecture. Millions of end-to-end Transport Class paths can be established in a scalable manner.

### 5.3. Automatically Creating Transport Classes

In order to simplify the creation of inter-domain paths, it may be desirable to automatically advertise a BGP-CT Transport Class based on the existence of an intra-domain tunnel. The RT value used on the BGP-CT advertisement is automatically derived from a property of the intra-domain tunnel that triggered its creation. How the Transport Class RT value is derived for different types of intra-domain tunnels is discussed below.

#### 5.3.1. Automatically Creating Transport Classes for BGP-SR-TE Intra-domain Tunnels

When the intra-domain tunnel is a BGP-SR-TE policy [I-D.ietf-idr-segment-routing-te-policy], the value of the Transport Class RT in the corresponding BGP-CT advertisement is derived from the Policy Color contained in SR Policy NLRI. The 32-bit Policy Color is directly converted to a 32-bit Transport Class RT.

#### 5.3.2. Automatically Creating Transport Classes for Flex-Algo Tunnels

When the intra-domain tunnel is created using Flex-Algo [I-D.ietf-lsr-flex-algo], the value of the Transport Class RT in the corresponding BGP-CT advertisement is derived from the 8-bit Algorithm value carried in SR-Algorithm sub-TLV (RFC8667). The conversion from 8-bit Algorithm value to 32-bit Transport Class RT is done by treating both as unsigned integers. Note that this definition allows for intra-domain tunnels created via standardized algorithm (0-127) as well as flex-algo (128-255).

### 5.3.3. Auto-deriving Transport Classes for PCEP

When the intra-domain tunnel is created using PCEP, the value of the Transport Class RT in the corresponding BGP-CT advertisement is derived from the Color of the SR Policy Identifiers TLV defined in [I-D.ietf-pce-segment-routing-policy-cp]. The 32-bit Color is directly converted to a 32-bit Transport Class RT.

### 5.4. Inter-domain flex-algo with BGP-CT

Flex-algo (defined in [I-D.ietf-lsr-flex-algo]) provides a mechanism to separate routing planes. Multiple algorithms are defined and prefix-SIDs are advertised for each algorithm. BGP-CT can be used to advertise these flex-algo SIDs in BGP-CT. BGP Prefix-SID (RFC 8669) is an attribute and can be carried in the BGP-CT NLRI. Multiple transport classes that correspond to each of the flex-algo in IGP domain are defined. These Transport Classes advertise the IGP flex-algo SIDs in the prefix-SIDs attribute in the BGP-CT NLRI.

### 5.5. Applicability to color-only policies

Color-only policies consist of (nullEndpoint, color) as specified in [I-D.ietf-spring-segment-routing-policy]. Special steering mechanisms are defined with "CO" flags defined in the color extended community [I-D.ietf-idr-segment-routing-te-policy]. Color-only policies can be advertised in BGP-CT with the prefix being NULL (0.0.0.0/32 or 0::0/128). Separate RD will be advertised for each NULL advertisement with different color. The Route target carries the Policy Color contained in SR Policy NLRI. The steering mechanisms defined in [I-D.ietf-spring-segment-routing-policy] MUST be honoured while resolving services prefixes on the BGP-CT advertisements.

### 5.6. Data sovereignty

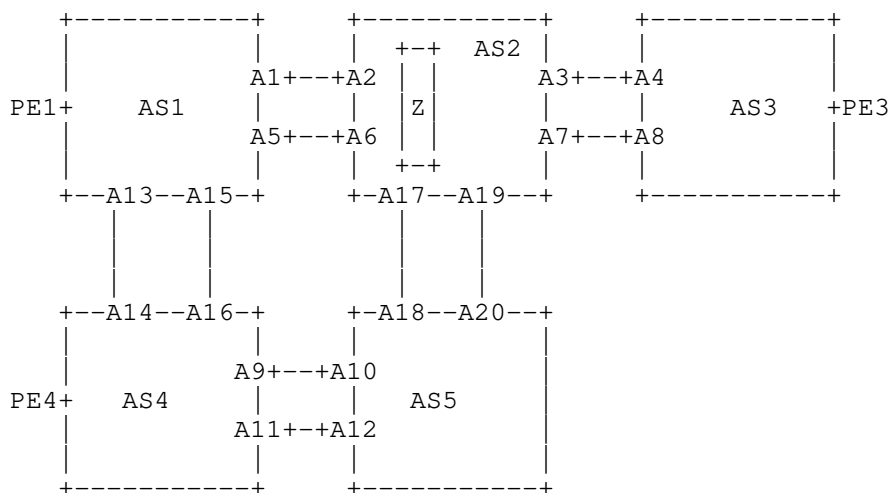


Figure 10: Multi domain Network

Consider a WAN network with multiple ASes as shown in the diagram Figure 10. The ASes roughly correspond to the geographical location of the nodes. In this example, we assume that each AS corresponds to a continent. The data sovereignty requirement in this example is that certain traffic from PE1 (in AS1) to PE3 (in AS3) must not cross through country Z in AS2. As indicated by the location of country Z in the diagram, all paths that go directly from AS1 to AS3 through AS2 necessarily pass through country Z. Using BGP-LU to provide connectivity from PE1 to PE3 would generally result in a path that goes from AS1 to AS2 to AS3, which does not satisfy the data sovereignty requirement in this example. Instead, the solution using BGP-CT will go from AS1 to AS4 to AS5 to AS2 to AS3. BGP-CT will ensure that when the traffic passes through AS2, only intra-domain paths satisfying the data sovereignty requirement will be used.

Within AS2, there are several different intra-domain TE mechanisms that can be used to exclude links that pass through country Z. For example, RSVP-TE or flex-algo can be used to create intra-domain paths that satisfy the data sovereignty requirement. BGP-CT allows the constrained intra-domain paths to satisfy requirements for end-to-end inter-domain paths. LSPs created by RSVP-TE or Flex-algo that satisfy the "exclude country Z" constraint are associated with a color Green. A Green Transport Class is defined on border nodes in all ASes. This Green Transport Class is associated with a mapping community called Not-Z.

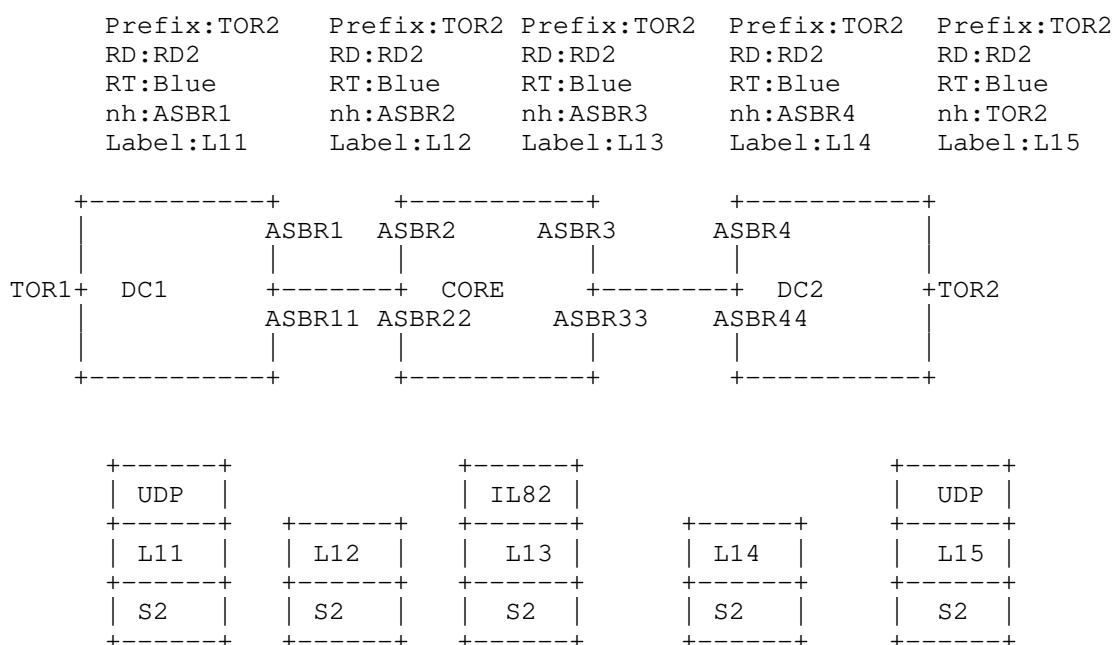
In AS2, the ASBRs are configured such that the presence of the mapping community Not-Z in BGP-CT routes results in a strict route resolution mechanism for those routes. A BGP-CT route carrying the color extended community Not-Z will only resolve on the Green Transport Class. So it will only use Green intra-domain tunnels.

In AS1, AS3, AS4, and AS5, no links pass through country Z, so all intra-domain paths automatically satisfy the data sovereignty requirement. So there is no need for the creation of Green intra-domain tunnels. In these ASes, the presence of the mapping community Not-Z in BGP-CT routes results in resolution on best-effort paths. Even though the ASBRs in these ASes do not need to create Green intra-domain tunnels, they still need to allocate labels to identify traffic using the Green Transport Class. These labels will be used by the ASBRs in AS2 to put traffic on the Green intra-domain tunnels in AS2.

The requirement is that only a subset of traffic honor the data sovereignty requirement. The service prefixes from PE1 to PE2 that need to honor the data sovereignty requirement will be associated with Green extended color community in the service advertisements. This will result in PE1 using the BGP-CT labels corresponding to {PE2, Green} to forward the traffic. BGP-CT labels corresponding to {PE2, Green} will exist at every ASBR along the path. The traffic originating on PE1, will be associated with Green color community. The bottom-most label in the packet consists of a VPN label. Above the VPN label, BGP-CT label is imposed. Above BGP-CT label, the intra-domain transport label is imposed. Let us assume the traffic from PE1 needs to go to PE2 through AS1, AS4, AS5, AS2, and AS3. The BGP-CT label for {PE2, Green} will be swapped at the border nodes.

Note that end-to-end inter-domain data sovereignty can in principle be accomplished using BGP-LU with multiple loopbacks and associating those loopbacks to appropriate transport tunnels at every border node in every domain. This is very configuration intensive and require multiple loopbacks. BGP-CT builds on the basic mechanisms of BGP-LU while greatly simplifying such use cases.

#### 5.7. Interconnecting IP Fabric Data Centers



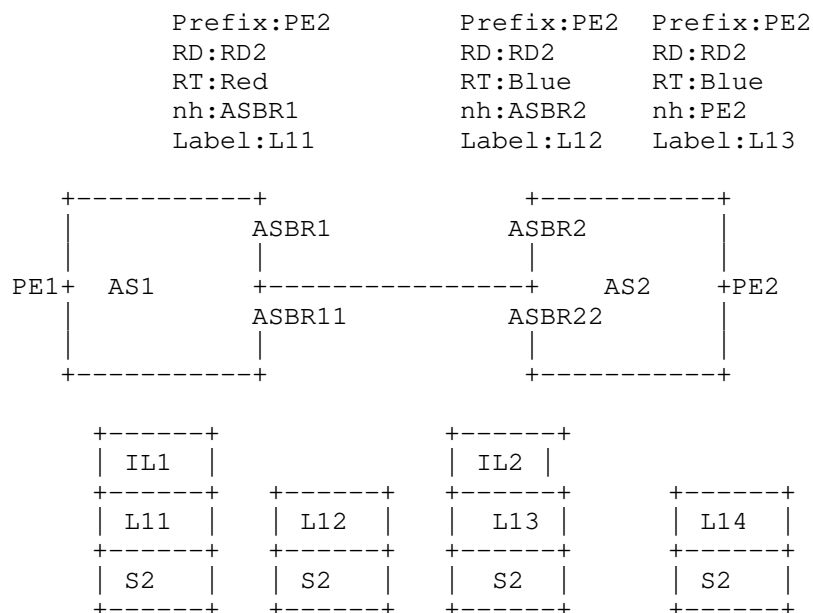
Label stacks along end-to-end path  
 S2 is the end-to-end service label.  
 IL82, is intra-domain labels corresponding to  
 blue intra-domain paths.

Figure 11: Operation in IP fabric

Many data center networks consist of IP fabrics which do not have MPLS packet processing capability. A common requirement is that traffic originated from an IP Fabric data center needs to satisfy certain constraints in the MPLS-enable core, for example, only using a subset of links (blue links). It is useful for the traffic originating in an IP Fabric DC to carry information that allows the MPLS-enable core to treat it accordingly. MPLSoUDP, as defined in [RFC7510], is a mechanism where a UDP header is imposed on an MPLS packets on the border nodes. In Figure 11 above, the traffic needs to take blue paths in the core. The Blue Transport Class is defined on the ASBRs. In the core, Blue intra-domain tunnels are created. The BGP-CT advertisements for the Blue Transport Class are as shown in the diagram. The BGP-CT advertisements originate at TOR2 and propagate through all the ASBRs, until finally reaching TOR1. Within DC1, traffic is encapsulated with a UDP header. Traffic with the UDP header gets decapsulated at ASBR1. The traffic follows Blue paths in

the core. At ASBR4, the MPLS packet gets encapsulated with a UDP header. The UDP header is removed at TOR2, and the lookup will be done for the service label.

#### 5.8. Translating Transport Classes across Domains



Label stacks along end-to-end path  
 S2 is the end-to-end service label.  
 IL1 and IL2 are intra-domain labels corresponding to  
 red intra-domain path in AS1 and Blue intra-domain  
 path in AS2.

Figure 12: Translating Transport Classes across Domains

In certain scenarios, the TE intent represented by Transport Classes may differ from one domain to another. This could be the result of two independent organizations merging into one. It could also occur when two ASes are under different administration, but use BGP-CT to provide an end-to-end service. In both scenarios, the same color may represent different intent in each domain. When the traffic needs to satisfy certain TE characteristic, the colors need to be mapped correctly at the border. In the example in Figure 12, there are two ASes. The low latency TE intent is represented with the Red

Transport Class in AS1 and with the Blue Transport Class in AS2. PE2 advertises a BGP-CT prefix with RT of Blue. ASBR2 sets the nexthop to self and advertises a new label L12. On ASBR1, the Blue BGP-CT advertisement is imported into the Red Transport RIB and the advertisement from ASBR1 will carry a Red RT. This ensures that the BGP-CT prefix for PE2 resolves on a Red intra-domain path in AS1.

## 5.9. SLA Guarantee

### 5.9.1. Low latency

Many network functions are virtualized and distributed. Certain functions are time and latency sensitive. In inter-domain networks, End-to-End latency measurement is required. Inside a domain, latency measurement mechanisms such as TWAMP [RFC5357] are used and link latency is advertised in IGP using extensions described in [RFC8570] and [RFC7471].

[I-D.ietf-idr-performance-routing] extends the BGP AIGP attribute [RFC7311] by adding a sub TLV to carry an accumulated latency metric. The BGP best path selection algorithm used for a Transport Class requiring low latency will consider the accumulated latency metric to choose the lowest latency path.

### 5.9.2. Traffic Engineering (TE) constraints

TE constraints generally include the ability to send traffic via certain nodes or links or avoid using certain nodes or links. In the Seamless SR architecture, the intra-domain transport technology is responsible for ensuring the TE constraints inside the domain, BGP-CT ensures that the end-to-end path is constructed from intra-domain paths and inter-AS links that individually satisfy the TE constraints.

For example, in order to construct a pair of diverse paths, we can define a red and a blue Transport Class. Within each domain, the red and blue Transport Class path are realized using intra-domain path diversity mechanisms. For example, in a domain using flex-algo, red and blue Transport Classes are realized using red and blue flex-algo definitions (FAD) which don't share any links. To maintain path diversity on inter-AS links, BGP policies are used to associate two inter-AS peers with the red Transport Class and another two inter-AS peers with the blue Transport Class.

### 5.9.3. Bandwidth constraints

The Seamless SR architecture does not natively support end-to-end bandwidth reservations. In this architecture, the bandwidth utilization characteristics of each domain are managed independently. The intra-domain bandwidth management can make use of a variety of tools.

Link bandwidth extended community as defined in [I-D.ietf-idr-link-bandwidth] allows for efficient weighted load-balancing of traffic on multiple BGP-CT paths that belong to the same Transport Class. For optimized path placement, a centralized TE system may be deployed with BGP policies/communities used for path placement.

## 5.10. Scalability

### 5.10.1. Access node scalability

The Seamless SR architecture needs to be able to accommodate very large numbers of access devices. These access devices are expected to be low-end devices with limited FIB capacity. The Seamless MPLS architecture, as described in [I-D.ietf-mpls-seamless-mpls], recommends the use of LDP DOD mode to limit the size of both the RIB and the FIB needed on the access devices. In the Seamless SR architecture, networks use IGP-based label distribution and do not have this selective label request mechanism. However, RIB scalability of access nodes has not been a problem for real seamless MPLS deployments. In cases where access devices are low on CPU and memory and unable to support large a RIB, BGP filtering policies can be applied at the ABR/ASBR routers to restrict the number of BGP-CT advertisements towards the access devices. The access devices will receive only the PE loopbacks that it needs to connect to.

### 5.10.2. Label stack depth

The ability for a device to push multiple MPLS labels on a packet depends on hardware capabilities. Access devices are expected to have limited label stack push capabilities. Assuming shortest path SR-MPLS in the access domain, the access domain transport will use a single label. Lightweight traffic-engineering and slicing could also be achieved with a single label as described in [I-D.ietf-lsr-flex-algo]. The Seamless SR architecture can provide cross-domain MPLS connectivity with a single label. Assuming the use of a service label, end-to-end connectivity is provided by pushing one service label, one BGP-CT label, and one intra-domain transport label (which could also be a Binding-SID). Therefore, access nodes will only need to be able to push 3 labels for most applications.



### 5.10.3. Label Resources

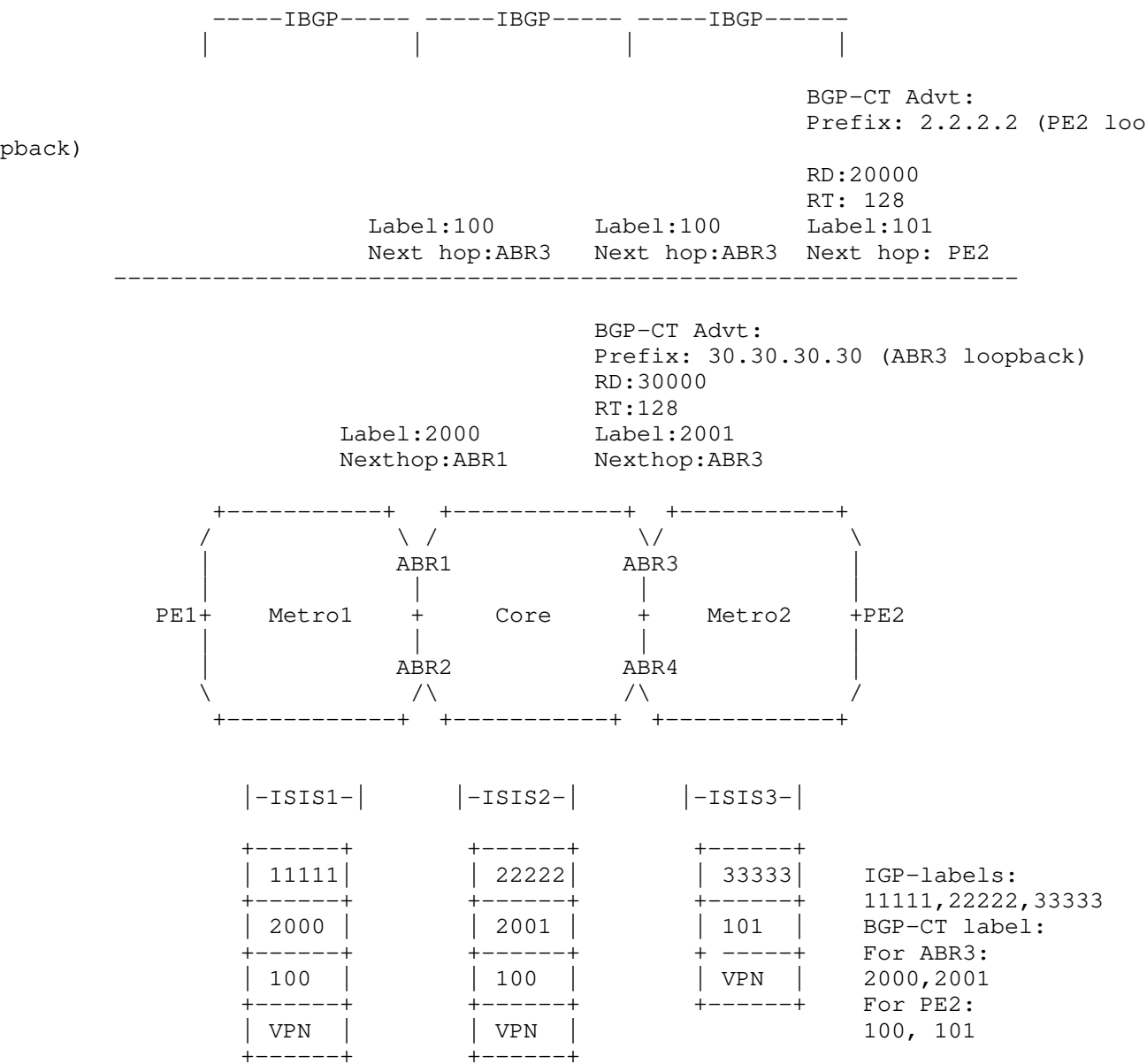


Figure 13: Recursive Route Resolution

The label resources are an important consideration in MPLS networks. On access devices, labels are consumed by services as well as for transport loopbacks inside IGP domain where the access device resides. For example, in the above diagram PE1 would have to allocate label resources equal to the number of customers connecting (i.e. the number of L2/L3 VPNs). Based on the size of the IGP domain that PE1 resides in, it will also have to allocate labels for IGP loopbacks. This number is at most a few thousands. So overall a typical access device should have adequate label resources in Seamless SR architecture. The P routers need to allocate labels for IGP loopbacks. This number again is small. At most it will be a few thousand based on number of nodes in the largest IGP domains. The metro networks connect to the core network through ABRs. It is possible that a given ABR may end up having to maintain forwarding entries for a large subset of the transport loopback routes. There may be a large number of metro networks connecting to a given ABR, and in this case, the ABR will need forwarding entries for every access node in the directly connected metros. So, this ABR may have to maintain on the order of 100k routes. With BGP-CT each Transport Class will have to be separately allocated a label. So, in the above example, the ABR1 would have to use 300k labels if there were 3 Transport Classes. This large number of label forwarding entries could be problematic.

In highly scaled scenarios, it is therefore desirable to reduce the forwarding state on the ABRs. This reduction can be achieved with label stacking as a result of recursive route resolution. Figure 13 illustrates how the forwarding state on ABRs can be greatly reduced by removing forward state for PEs in remote domains from the ABRs. In this example, we assume that we are setting up end-to-end paths for a single Transport Class, for example red. PE2 advertises a BGP-CT prefix of 2.2.2.2 with nexthop of 2.2.2.2 and label 101. 2.2.2.2 is PE2's loopback. ABR3 advertises label 100 for BGP-CT prefix 2.2.2.2 and changes the nexthop to self. When ABR1 receives the BGP-CT advertisement for 2.2.2.2, it does not change the nexthop and advertises same label advertised by ABR3. When PE1 receives the BGP-CT advertisement for 2.2.2.2 with a nexthop of ABR3, it resolves the route using reachability to ABR3.

The reachability of ABR3 has been learned by PE1 as the result of a BGP-CT advertisement originated by ABR3. As shown in Figure 13, ABR3 advertises BGP-CT prefix 30.30.30.30 with label 2001. ABR1 advertises label 2000 for BGP-CT prefix 30.30.30.30 and sets nexthop to self. PE1 constructs the service data packet with a VPN label at the bottom followed by 2 BGP-CT labels 100 and 2000. The top most label 2000 is the transport label for the metro domain. Removing the forwarding state for PEs in remote domains on the ABRs comes at the expense of one additional BGP-CT label on the data packet.

Recursive route resolution provides significant forwarding state reduction on the ABRs. ABRs have to allocate label resources only for the PEs in their local domain. The number of PEs in the same domain as a given ABR is much lower than the total number of PEs in the network.

The examples in this draft generally show VPN routes resolving on BGP-CT prefixes. However, the mechanisms are equally applicable to non-VPN routes.

#### 5.11. Availability

Transport layer availability is very important in latency and loss sensitive networks. Any link or node failure must be repaired with 50ms convergence time. 50 ms convergence time can be achieved with Fast ReRoute (FRR) mechanisms. The seamless SR architecture provides protection against intra-domain link and node failures, Protection against border node failures and the egress link and node failures are also provided. Details of the FRR techniques are described in the sections below.

##### 5.11.1. Intra domain link and node protection

In the seamless SR architecture, protection against node and link failure is achieved with the relevant FRR techniques for the corresponding transport mechanism used inside the domain. In the case of an IP fabric, ECMP FRR or LFA can be used. In SR networks, TI-LFA [I-D.ietf-rtgwg-segment-routing-ti-lfa] provides link and node protection. For SR-TE transport ([I-D.ietf-spring-segment-routing-policy]), link and node protection can be achieved using TI-LFA, combined with mechanisms described in [I-D.hegde-spring-node-protection-for-sr-te-paths].

##### 5.11.2. Egress link and node protection

[RFC8679] describes the mechanisms for providing protection for border nodes and PE devices where services are hosted. The mechanism can be further simplified operationally with anycast SIDs and anycast service labels, as described in [I-D.hegde-rtgwg-egress-protection-sr-networks].

##### 5.11.3. Border Node protection

Border node protection is very important in a network consisting of multiple domains. Seamless SR architecture can achieve 50ms FRR protection in the event of node failure using anycast addresses for the ABR/ASBRs. This requires that a set of ABRs advertise the same

label for a given BGP-CT Prefix. The detailed mechanism is described in [I-D.hegde-rtgw-egress-protection-sr-networks].

## 5.12. Operations

### 5.12.1. MPLS ping and Traceroute

The Seamless SR Architecture consists of 3 layers: the service layer, intra-domain transport, and BGP-CT transport. Within each layer, connectivity can be verified independently. Within the BGP-CT transport layer, end-to-end connectivity can be verified using a new OAM FEC for BGP-CT defined in draft [I-D.kaliraj-idr-bgp-classful-transport-planes]. The draft describes end-to-end connectivity verification as well as fault isolation. BGP-CT verification happens only on the BGP nodes. The intra-domain connectivity verification and fault isolation will be based on the technology deployed in that domain as defined in [RFC8029] and [RFC8287].

### 5.12.2. Counters and Statistics

Traffic accounting and the ability to build demand matrix for PE to PE traffic is very important. With BGP-CT, per-label transit counters should be supported on every transit router. Per-label transit counters provide details of total traffic towards a remote PE measured at every BGP transit router. Per-label egress counters should be supported on ingress PE router. Per-label egress counters provide total traffic from ingress PE to the specific remote PE.

## 5.13. Service Mapping

Service mapping is an important aspect of any architecture. It provides means to translate end users SLA requirements into operator's network configurations. Seamless SR architecture supports automatic steering with extended color community. The Transport Class and the route target carried by the BGP-CT advertisement directly map to the extended color community. Services that require specific SLA carry the extended color community which maps to the Transport Class to which the BGP-CT advertisement belongs.

Other types of traffic steering such as DSCP based forwarding is expressed with mapping-community. Mapping community is a standard BGP community and is completely generic and user defined. The mapping community will have a specific service mapping feature associated with it along with required fallback behaviour when the primary transport goes down. The below list provides a general guideline into the different service mapping features and fallback options an implementation should provide.

DSCP based mapping with each DSCP mapping to a Transport Class.

DSCP based mapping with default mapping to a best-effort transport

DSCP based mapping with fallback to best-effort when primary transport tunnel goes down.

Extended color community based mapping with fallback to best effort

Fallback options with specific protocol during migrations

Fallback options to a different Transport Class.

No Fallback permitted.

#### 5.14. Migrations

Networks that migrate from Seamless MPLS architecture to Seamless SR architecture, require that all the border nodes and PE devices be upgraded and enabled with new family on the BGP session. In cases where legacy nodes that cannot be upgraded, exporting from BGP-LU into BGP-CT and vice versa SHOULD be supported. Once the entire network is migrated to support BGP-CT, there is no need to run BGP-LU family on the BGP sessions. BGP-CT itself can advertise a best effort Transport Class and BGP-LU family can be removed.

#### 5.15. Interworking with v6 transport technologies

A later version of this document will address interworking with other v6 technologies, including SRv6, SRm6, and MPLS over GRE6.

#### 5.16. BGP based Multicast

BGP based multicast as described in draft [I-D.zzhang-bess-bgp-multicast] serves two main purposes. It can replace PIM/ mLDLP inside a domain to natively do a BGP based multicast. It can also serve as an overlay stitching protocol to stitch multiple P2MP LSPs across the domain. This gives the ability to easily transition each domain independently from one technology to the other. BGP based multicast defines a new SAFI for carrying the MULTICAST TREE SAFI. Different route types are defined to support the various usecases.

## 6. Backward Compatibility

## 7. Security Considerations

TBD

## 8. IANA Considerations

## 9. Acknowledgements

Many thanks to Kireeti Kompella, Ron Bonica, Krzysztof Szarcowitz, Srihari Sangli, Julian Lucek, Ram Santhanakrishnan for discussions and inputs. Thanks to Joel Halpern for review and comments.

## 10. Contributors

## 1. Kaliraj Vairavakkalai

Juniper Networks

kaliraj@juniper.net

## 2. Jeffrey Zhang

Juniper Networks

zzhang@juniper.net

## 11. References

## 11.1. Normative References

[I-D.hegde-rtgwg-egress-protection-sr-networks]

Hegde, S. and W. Lin, "Egress Protection for Segment Routing (SR) networks", draft-hegde-rtgwg-egress-protection-sr-networks-00 (work in progress), March 2020.

[I-D.ietf-idr-performance-routing]

Xu, X., Hegde, S., Talaulikar, K., Boucadair, M., and C. Jacquenet, "Performance-based BGP Routing Mechanism", draft-ietf-idr-performance-routing-02 (work in progress), October 2019.

[I-D.kaliraj-idr-bgp-classful-transport-planes]

Vairavakkalai, K., Venkataraman, N., and B. Rajagopalan, "BGP Classful Transport Planes", draft-kaliraj-idr-bgp-classful-transport-planes-01 (work in progress), July 2020.

- [I-D.zzhang-bess-bgp-multicast]  
Zhang, Z., Giuliano, L., Patel, K., Wijnands, I., mishra, m., and A. Gulko, "BGP Based Multicast", draft-zzhang-bess-bgp-multicast-03 (work in progress), October 2019.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC3107] Rekhter, Y. and E. Rosen, "Carrying Label Information in BGP-4", RFC 3107, DOI 10.17487/RFC3107, May 2001, <<https://www.rfc-editor.org/info/rfc3107>>.
- [RFC8669] Previdi, S., Filsfils, C., Lindem, A., Ed., Sreekantiah, A., and H. Gredler, "Segment Routing Prefix Segment Identifier Extensions for BGP", RFC 8669, DOI 10.17487/RFC8669, December 2019, <<https://www.rfc-editor.org/info/rfc8669>>.

## 11.2. Informative References

- [I-D.hegde-spring-node-protection-for-sr-te-paths]  
Hegde, S., Bowers, C., Litkowski, S., Xu, X., and F. Xu, "Node Protection for SR-TE Paths", draft-hegde-spring-node-protection-for-sr-te-paths-07 (work in progress), July 2020.
- [I-D.ietf-idr-link-bandwidth]  
Mohapatra, P. and R. Fernando, "BGP Link Bandwidth Extended Community", draft-ietf-idr-link-bandwidth-07 (work in progress), March 2018.
- [I-D.ietf-idr-segment-routing-te-policy]  
Previdi, S., Filsfils, C., Talaulikar, K., Mattes, P., Rosen, E., Jain, D., and S. Lin, "Advertising Segment Routing Policies in BGP", draft-ietf-idr-segment-routing-te-policy-09 (work in progress), May 2020.
- [I-D.ietf-idr-tunnel-encaps]  
Patel, K., Velde, G., Sangli, S., and J. Scudder, "The BGP Tunnel Encapsulation Attribute", draft-ietf-idr-tunnel-encaps-19 (work in progress), September 2020.
- [I-D.ietf-lsr-flex-algo]  
Psenak, P., Hegde, S., Filsfils, C., Talaulikar, K., and A. Gulko, "IGP Flexible Algorithm", draft-ietf-lsr-flex-algo-11 (work in progress), September 2020.



- [I-D.ietf-mpls-seamless-mpls]  
Leymann, N., Decraene, B., Filsfils, C., Konstantynowicz, M., and D. Steinberg, "Seamless MPLS Architecture", draft-ietf-mpls-seamless-mpls-07 (work in progress), June 2014.
- [I-D.ietf-pce-segment-routing-policy-cp]  
Koldychev, M., Sivabalan, S., Barth, C., Peng, S., and H. Bidgoli, "PCEP extension to support Segment Routing Policy Candidate Paths", draft-ietf-pce-segment-routing-policy-cp-00 (work in progress), June 2020.
- [I-D.ietf-rtgwg-segment-routing-ti-lfa]  
Litkowski, S., Bashandy, A., Filsfils, C., Decraene, B., Francois, P., Voyer, D., Clad, F., and P. Camarillo, "Topology Independent Fast Reroute using Segment Routing", draft-ietf-rtgwg-segment-routing-ti-lfa-04 (work in progress), August 2020.
- [I-D.ietf-spring-segment-routing-policy]  
Filsfils, C., Talaulikar, K., Voyer, D., Bogdanov, A., and P. Mattes, "Segment Routing Policy Architecture", draft-ietf-spring-segment-routing-policy-08 (work in progress), July 2020.
- [I-D.voyer-pim-sr-p2mp-policy]  
Voyer, D., Filsfils, C., Parekh, R., Bidgoli, H., and Z. Zhang, "Segment Routing Point-to-Multipoint Policy", draft-voyer-pim-sr-p2mp-policy-02 (work in progress), July 2020.
- [RFC1997] Chandra, R., Traina, P., and T. Li, "BGP Communities Attribute", RFC 1997, DOI 10.17487/RFC1997, August 1996, <<https://www.rfc-editor.org/info/rfc1997>>.
- [RFC4364] Rosen, E. and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, DOI 10.17487/RFC4364, February 2006, <<https://www.rfc-editor.org/info/rfc4364>>.
- [RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, DOI 10.17487/RFC5357, October 2008, <<https://www.rfc-editor.org/info/rfc5357>>.
- [RFC6388] Wijnands, IJ., Ed., Minei, I., Ed., Kompella, K., and B. Thomas, "Label Distribution Protocol Extensions for Point-to-Multipoint and Multipoint-to-Multipoint Label Switched Paths", RFC 6388, DOI 10.17487/RFC6388, November 2011, <<https://www.rfc-editor.org/info/rfc6388>>.

- [RFC7311] Mohapatra, P., Fernando, R., Rosen, E., and J. Uttaro, "The Accumulated IGP Metric Attribute for BGP", RFC 7311, DOI 10.17487/RFC7311, August 2014, <<https://www.rfc-editor.org/info/rfc7311>>.
- [RFC7471] Giacalone, S., Ward, D., Drake, J., Atlas, A., and S. Previdi, "OSPF Traffic Engineering (TE) Metric Extensions", RFC 7471, DOI 10.17487/RFC7471, March 2015, <<https://www.rfc-editor.org/info/rfc7471>>.
- [RFC7510] Xu, X., Sheth, N., Yong, L., Callon, R., and D. Black, "Encapsulating MPLS in UDP", RFC 7510, DOI 10.17487/RFC7510, April 2015, <<https://www.rfc-editor.org/info/rfc7510>>.
- [RFC8029] Kompella, K., Swallow, G., Pignataro, C., Ed., Kumar, N., Aldrin, S., and M. Chen, "Detecting Multiprotocol Label Switched (MPLS) Data-Plane Failures", RFC 8029, DOI 10.17487/RFC8029, March 2017, <<https://www.rfc-editor.org/info/rfc8029>>.
- [RFC8287] Kumar, N., Ed., Pignataro, C., Ed., Swallow, G., Akiya, N., Kini, S., and M. Chen, "Label Switched Path (LSP) Ping/Traceroute for Segment Routing (SR) IGP-Prefix and IGP-Adjacency Segment Identifiers (SIDs) with MPLS Data Planes", RFC 8287, DOI 10.17487/RFC8287, December 2017, <<https://www.rfc-editor.org/info/rfc8287>>.
- [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>>.
- [RFC8570] Ginsberg, L., Ed., Previdi, S., Ed., Giacalone, S., Ward, D., Drake, J., and Q. Wu, "IS-IS Traffic Engineering (TE) Metric Extensions", RFC 8570, DOI 10.17487/RFC8570, March 2019, <<https://www.rfc-editor.org/info/rfc8570>>.
- [RFC8679] Shen, Y., Jeganathan, M., Decraene, B., Gredler, H., Michel, C., and H. Chen, "MPLS Egress Protection Framework", RFC 8679, DOI 10.17487/RFC8679, December 2019, <<https://www.rfc-editor.org/info/rfc8679>>.
- [TS.23.501-3GPP] 3rd Generation Partnership Project (3GPP), "System Architecture for 5G System; Stage 2, 3GPP TS 23.501 v16.4.0", March 2020.

## Authors' Addresses

Shraddha Hegde  
Juniper Networks Inc.  
Exora Business Park  
Bangalore, KA 560103  
India

Email: shraddha@juniper.net

Chris Bowers  
Juniper Networks Inc.

Email: cbowers@juniper.net

Xiaohu Xu  
Alibaba Inc.  
Beijing  
China

Email: xiaohu.xxh@alibaba-inc.com

Arkadiy Gulko  
Refinitiv

Email: arkadiy.gulko@refinitiv.com

Alex Bogdanov  
Google Inc.

Email: bogdanov@google.com

Jim Uttaro  
ATT

Email: jul738@att.com

Luay Jalil  
Verizon

Email: luay.jalil@verizon.com

SPRING  
Internet-Draft  
Intended status: Informational  
Expires: 28 March 2022

S. Hegde  
C. Bowers  
Juniper Networks Inc.  
X. Xu  
Alibaba Inc.  
A. Gulko  
EdwardJones  
A. Bogdanov  
Google Inc.  
J. Uttaro  
ATT  
L. Jalil  
Verizon  
M. Khaddam  
Cox communications  
A. Alston  
Liquid Telecom  
LM. Contreras  
Telefonica  
24 September 2021

Seamless SR Problem Statement  
draft-hegde-spring-mpls-seamless-sr-06

Abstract

This draft documents a set of use cases and requirements for end-to-end intent-based paths spanning multi-domain packet networks. The document explicitly focuses on use cases that require high scale and availability, which will likely benefit from distributed solutions. It is intended that the requirements in this document serve as a basis for future IETF work to develop distributed solutions for inter-domain intent-based transport paths.

Requirements Language

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

Status of This Memo

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

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

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

This Internet-Draft will expire on 28 March 2022.

#### Copyright Notice

Copyright (c) 2021 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 . . . . .	3
2. Large scale networks . . . . .	4
2.1. Service provider networks . . . . .	4
2.2. Cloud provider WAN networks . . . . .	5
2.3. Data Center WAN Networks . . . . .	6
3. Use Cases for Inter-domain Intent-based Transport . . . . .	7
3.1. Inter-domain Data Sovereignty . . . . .	7
3.2. Inter-domain Low-Latency Services . . . . .	7
3.3. Network Mergers . . . . .	8
3.4. Inter-domain Service Function Chaining . . . . .	8
3.5. AS Confederation . . . . .	9
3.6. Inter-domain Multicast Use cases . . . . .	9
4. Requirements . . . . .	10
4.1. AS and IGP Domain Models . . . . .	10
4.1.1. Multiple ASes connected with E-BGP . . . . .	10
4.1.2. Single AS multiple IGP domains . . . . .	10
4.1.3. Single AS, Multiple IGP domains with no common border node . . . . .	11
4.2. Transport tunneling Requirements . . . . .	12
4.2.1. Unicast tunneling Requirements . . . . .	12

4.2.2. Multicast tunneling Requirements . . . . .	12
4.3. Inter-domain SLA Requirements . . . . .	13
4.3.1. Latency, Delay Variation, and Link Loss Constraints . . . . .	13
4.3.2. Bandwidth Constraints . . . . .	14
4.3.3. Link Inclusion/Exclusion Constraints . . . . .	14
4.3.4. Node Inclusion/Exclusion Constraints . . . . .	14
4.3.5. Domain Inclusion/Exclusion Constraints . . . . .	14
4.3.6. Diverse Paths . . . . .	15
4.3.7. Constraint applicability to a subset of domains . . . . .	16
4.3.8. Service function chaining . . . . .	16
4.4. Multicast specific requirements . . . . .	16
4.5. Interoperate with BGP-LU . . . . .	16
4.6. Merger and Migration Requirements . . . . .	16
4.6.1. Option A and Option B Usecases . . . . .	16
4.6.2. Inter-Domain Intent Translation . . . . .	17
4.6.3. Native Support for Best Effort Paths . . . . .	17
4.6.4. Interoperate with Other tunneling Mechanisms . . . . .	17
4.7. Scalability Requirements . . . . .	17
4.8. Availability Requirements . . . . .	18
4.9. Operations and Automation Requirements . . . . .	18
4.10. Service Mapping Requirements . . . . .	19
4.10.1. Traffic service mapping . . . . .	19
4.10.2. 1 to N service mapping . . . . .	20
4.11. Interaction with Other Approaches . . . . .	20
5. Backward Compatibility . . . . .	20
6. Security Considerations . . . . .	21
7. IANA Considerations . . . . .	21
8. Acknowledgements . . . . .	21
9. Contributors . . . . .	21
10. References . . . . .	21
10.1. Normative References . . . . .	21
10.2. Informative References . . . . .	22
Authors' Addresses . . . . .	26

## 1. Introduction

Evolving trends in wireless access technology, cloud applications, virtualization, and network consolidation all contribute to the increasing demands being placed on a common packet network. In order to meet these demands, a given network will need to scale horizontally in terms of its bandwidth, absolute number of nodes, and geographical extent. The same network will need to scale vertically in terms of the different services that it needs to simultaneously support.

In order to operate networks with large numbers of devices, network operators organize networks into multiple smaller network domains. Each network domain typically runs an IGP which has complete visibility within its own domain, but limited visibility outside of its domain. Network operators will continue to use multiple domains to scale horizontally. These multi-domain networks will also need to scale vertically, to allow a common multi-domain network to carry all of an organization's services.

Evolving network requirements (e.g., 5G, native cloud) motivate network operators to deploy tunnels that span multiple AS's and maintain specific transport characteristics (e.g., bandwidth, latency). There is a need to provide flexible, scalable, and reliable end-to-end connectivity for multiple services across independent network domains.

## 2. Large scale networks

### 2.1. Service provider networks

Service Provider networks can contain many nodes distributed over a large geographic area. 5G networks can include as many as one million nodes, with the majority of those being radio access nodes. Radio and access nodes may be constrained by their memory and processing capabilities.

Service provider transport networks use multiple domains to support scalability. For this analysis, we consider a representative network design with four level of hierarchy: access domains, pre-aggregation domains, aggregation domains and a core. (See Figure 1). The separation of domains internal to the service provider can be performed by using either IGP or BGP.

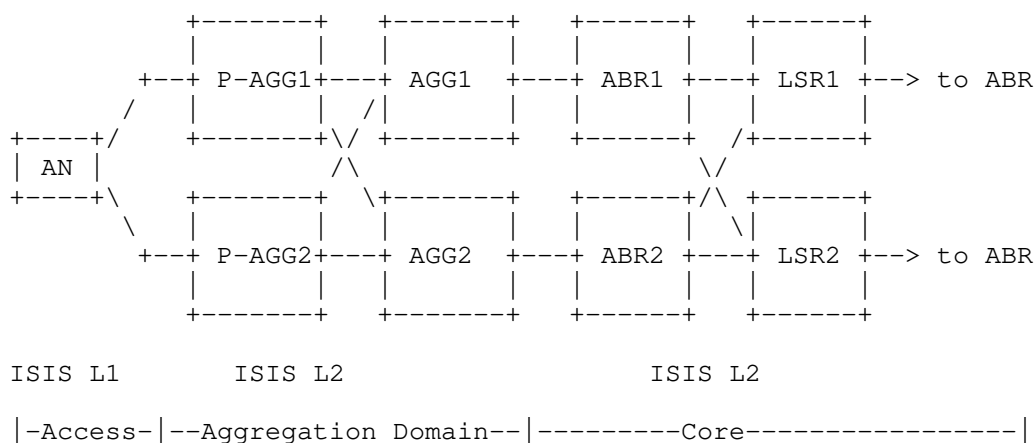


Figure 1: 5G network

5G networks support a variety of service use cases that require end-to-end slicing. In certain cases the end-to-end connectivity requires the ability to forward over intent-based paths. The inter-domain solution should support end-to-end Service Level Objectives(SLO) to allow the creation of network slices.

## 2.2. Cloud provider WAN networks

As WAN networks grow beyond several thousand nodes, it is often useful to divide the network into multiple IGP domains, as illustrated in Figure 2. Separate IGP domains increase service availability by establishing a constrained failure domain. Smaller IGP domains may also improve network performance and health by reducing the device scale profile (including protocol and FIB scale).

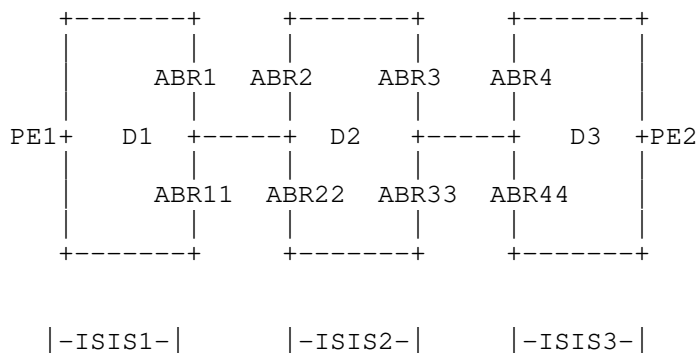




Figure 2: WAN Network

These large WAN networks often cross national boundaries. In order to meet data sovereignty requirements, operators need to maintain strict control over end-to-end traffic-engineered (TE) paths. A goal of a distributed inter-domain solution is to be able to create highly constrained inter-domain TE paths in a scalable manner.

Some deployments may use a centralized controller to acquire the topologies of multiple domains and build end-to-end constrained paths. This centralized approach can be scaled with hierarchical controllers. However, there is still significant risk of a loss of network connectivity to one or more controllers, which can result in a failure to satisfy the strict requirements of data sovereignty. The network should have pre-established TE paths end-to-end that don't rely on controllers in order to address these failure scenarios.

### 2.3. Data Center WAN Networks

Data centers are playing an increasingly important role in providing access to information and applications. Geographically diverse data centers usually connect via a high speed, reliable and secure DC WAN core network.

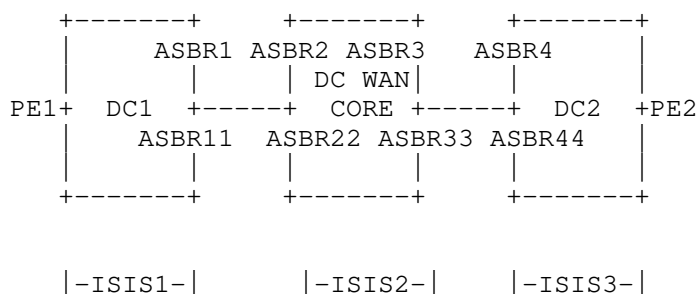


Figure 3: DCI Network

In many DC WAN deployments, applications require end-to-end path diversity and end-to-end low latency paths. The DC WAN networks may consist of large number of devices owing to global presence. In some DC WAN deployments the tunneling mechanisms used within the data centers are the same as those used in the DC WAN core. For example, a network may use MPLS in both data center and DC WAN core. Or it may use SRv6 in both data center and DC WAN core. This can simplify network deployments.

However, in some DC WAN deployments the traffic within data centers and the traffic over the DC WAN core use different tunneling mechanisms, such as SRv6 in the data center and MPLS in the DC WAN core. It is important for DC WAN network operators to have flexibility in the choice of tunneling mechanisms across domains.

### 3. Use Cases for Inter-domain Intent-based Transport

The use cases for inter-domain intent-based packet transport described in this section are intended to provide motivation for the requirements that follow.

#### 3.1. Inter-domain Data Sovereignty

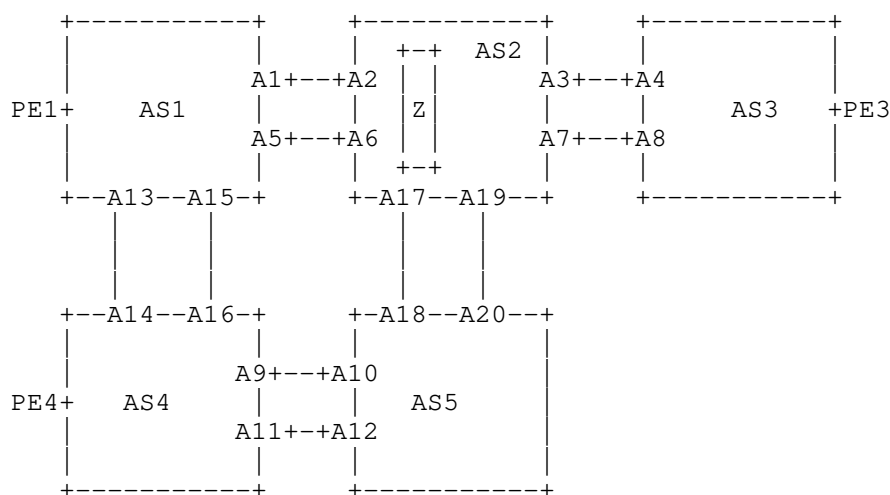


Figure 4: Multi domain Network

Figure Figure 4 depicts a WAN with multiple ASes. Each AS is resides serves a continent (e.g., Asia). Certain traffic from PE1 (in AS1) to PE3 (in AS3) must not traverse country Z in AS2. However, all paths from AS1 to AS3 traverse AS 2. The inter-domain solution should provide end-to-end path creation that traverses AS 2 but avoids country Z.

#### 3.2. Inter-domain Low-Latency Services

Service provider networks running L2 and L3VPNs carry traffic for particular VPNs on low-latency paths that traverse multiple domains.

### 3.3. Network Mergers

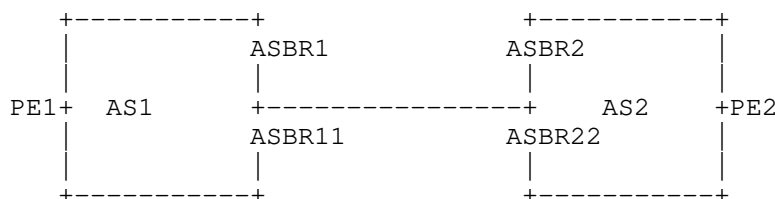


Figure 5: Network Mergers

In diagram Figure 5 above, AS1 and AS2 which were previously under independent administration, merge to come under a single administration. The network operator may decide to merge the two domains into single AS which would need bigger operational effort. Network operators may also retain the two ASes and build end-to-end paths across the two Ases. In this case, the paths in AS1 and AS2 corresponding to the same intent may use different representations in the two ASes. In some cases, organizations may continue to use option A or option B [RFC4364] style interconnectivity in which case the inter-domain solution should satisfy intent of the path on inter-domain links for the service prefixes. In other cases, organizations may prefer to use option C style connectivity from PE1 to PE2. In this case, an inter-domain solution should provide effective mechanisms to translate intent across domains without requiring renumbering of the intent representation.

### 3.4. Inter-domain Service Function Chaining

[RFC7665] defines service function chaining as an ordered set of service functions and automated steering of traffic through this set of service functions. There could be a variety of service functions such as firewalls, parental control, CGNAT etc. In 5G networks these functions may be completely virtualized or could be a mix of virtualized functions and physical appliances. It is required that the inter-domain solution caters to the service function chaining requirements. The service functions may be virtualized and spread across different data centers attached to different domains.

### 3.5. AS Confederation

BGP confederation allows the division of a public AS in multiple sub-AS, usually with private identifiers. From outside, the confederation is seen as a single and common AS, the public one. BGP sessions are maintained among sub-AS. In the internals of the confederation, each sub-AS can be configured and run autonomously, even though some BGP parameters (like e.g. LOCAL\_PREF or MED) are preserved across sub-AS. Thus, it can be of interest to define end-to-end paths of specific characteristics in terms of SLOs across the sub-AS as well as internally to each sub-AS.

### 3.6. Inter-domain Multicast Use cases

Multicast services such as IPTV and multicast VPN also need to be supported across a multi-domain service provider network.

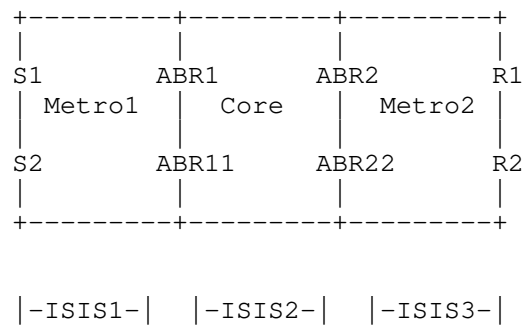


Figure 6: Multicast usecases

Figure 6 shows a simplified multi-domain network supporting multicast. Multicast sources S1 and S2 lie in a different domain from the receivers R1 and R2. Using multiple IGP domains presents a problem for the establishment of multicast replication trees. Typically, a multicast receiver does a reverse path forwarding (RPF) lookup for a multicast source. One solution is to leak the routes for multicast sources across the IGP domains. However, this can compromise the scaling properties of the multi-domain architecture. A distributed inter-domain solution should accommodate a mixture of existing and newer technologies to better facilitate coexistence and migration. A distributed solution should avoid leaking RPF routes into the IGP domains.

#### 4. Requirements

The requirements described in this document are mostly applicable to network under a single administrative domain that are organized into multiple network domains. The requirements are also applicable to multi-AS networks with closely cooperating administration.

##### 4.1. AS and IGP Domain Models

This section describes three different ways that multi-domain networks are organized today. The requirements in subsequent sections are applicable to all three types of multi-domain networks described below.

###### 4.1.1. Multiple ASes connected with E-BGP

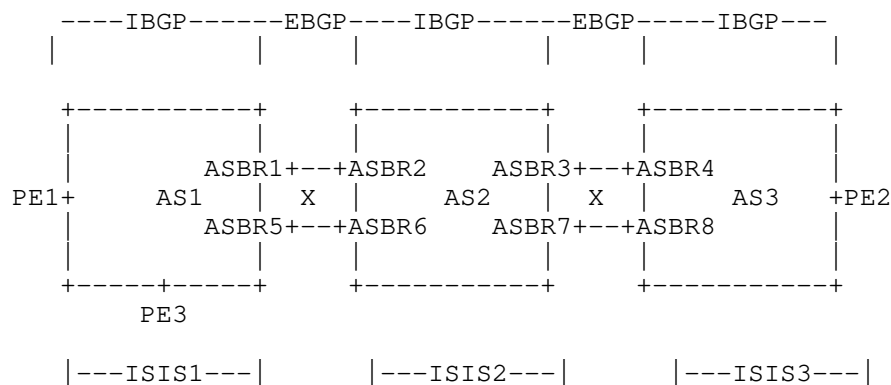


Figure 7: Multiple ASes connected with E-BGP

The above diagram Figure 7 shows three different ASes (AS1, AS2 and AS3.) ASBR1 to ASBR8 are border nodes between the ASes. A given ASBR runs E-BGP sessions with the ASBRs in adjacent ASes. The ASBR also runs I-BGP sessions with other ASBRs in the same AS. Route reflectors can also be used to achieve this full mesh of I-BGP information exchange. Similar scenario applies when considering BGP confederations [RFC5065].

###### 4.1.2. Single AS multiple IGP domains

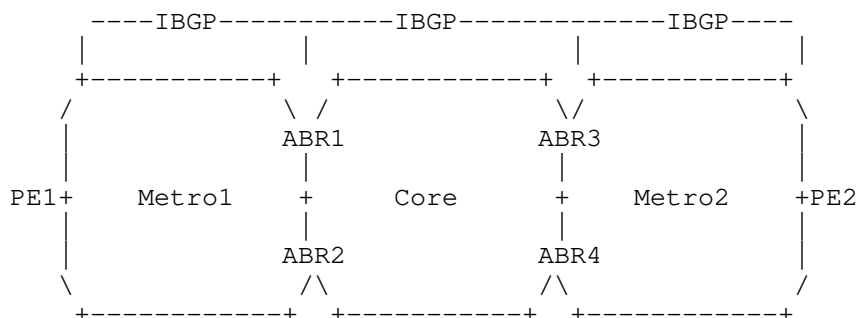


Figure 8: Single AS with Multiple IGP domains

The above diagram Figure 8 shows three different IGP domains, Metro1, Core, and Metro2. The three IGP domains may be realized with multiple levels in ISIS or multiple areas in OSPF. They can also be realized using separate IGP instances.

This single-AS network uses I-BGP sessions. ABRs and PEs achieve a full mesh of I-BGP information sharing by configuring the ABRs as inline route reflectors.

#### 4.1.3. Single AS, Multiple IGP domains with no common border node

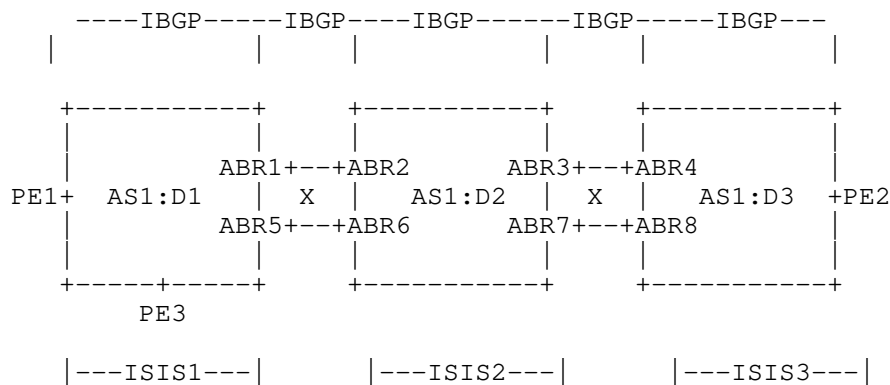


Figure 9: Single AS multiple IGP domains with no common Border node

The above diagram Figure 9 shows a single AS1 with three different IGP domains, D1, D2, and D3. ABR1 to ABR8 are border nodes for the IGP domains and they participate in only one IGP domain.

This single-AS network uses I-BGP sessions. ABRs and PEs achieve a full mesh of I-BGP information sharing by configuring the ABRs as inline route reflectors.

## 4.2. Transport tunneling Requirements

### 4.2.1. Unicast tunneling Requirements

The inter-domain solution should support the following unicast tunneling mechanisms:

- SR-MPLS tunnels with IPv4 underlay

- SR-MPLS tunnels with IPv6 underlay

- SR-MPLS tunnels with dual stack underlay

- SRv6 tunneling end-to-end

- Segment routing TE tunnels and color-only policies as described in [I-D.ietf-idr-segment-routing-te-policy] (both SR-MPLS and SRv6)

- Flex-algo [I-D.ietf-lsr-flex-algo] (both SR-MPLS and SRv6)

- Pure IP fabric (incapable of supporting MPLS or SRv6 tunneling mechanisms)

- RSVP and LDP based tunnels

The inter-domain solution should support the ability to have different domains running different unicast tunneling mechanisms.

The solution should support inter-domain paths that fulfil a common intent using different unicast tunneling mechanisms in different domains.

### 4.2.2. Multicast tunneling Requirements

The inter-domain solution should support the following multicast tunneling mechanisms:

- All of the unicast tunneling mechanisms described in Section 4.2.1 should be supported for multicast service for the purpose of ingress replication.

SR-P2MP as defined in [I-D.voyer-pim-sr-p2mp-policy]

PIM based multicast

RSVP-P2MP and mLDP [RFC6388] based tunnels

BGP based multicast (hop-by-hop or controller-driven, for native IP, labelled, or SRv6 forwarding planes)

The inter-domain solution should support the ability to have different domains running different multicast tunneling mechanisms and should not require to leak RPF routes into IGP domains.

The solution should support inter-domain paths that fulfil a common intent using different multicast tunneling mechanisms in different domains.

#### 4.3. Inter-domain SLA Requirements

This section discusses the end-to-end constraints that intent-based inter-domain path may have to adhere to. The requirements described in this section are applicable to the three types of AS and IGP domain partitioning described in Section 4.1.

##### 4.3.1. Latency, Delay Variation, and Link Loss Constraints

Link delay, delay variation and link loss values can be advertised within a domain using the IGP as described in [RFC8570]. Within an IGP domain, minimum latency, minimum delay variation, and minimum link loss paths can be built using flex-algo [I-D.ietf-lsr-flex-algo]. The end-to-end low latency, low delay variation, or low link loss path requires accumulated metrics for latency, delay variation, and link loss.

The solution should allow the creation of inter-domain paths with low values of latency as calculated over the end-to-end path. It is not necessary that the solution produce the absolute minimum end-to-end latency, delay variation, or link loss path. However, the solution should provide the ability to balance scalability with optimality.

Best path selection at any intermediate border node should be allowed.

The inter-domain solution should allow advertising multiple paths end-to-end and compare the accumulated metric across all of the paths at the ingress.



#### 4.3.2. Bandwidth Constraints

A distributed solution should support the creation of inter-domain paths using intra-domain bandwidth guaranteed paths.

A distributed solution may support optimized path placement with end-to-end bandwidth guarantees.

#### 4.3.3. Link Inclusion/Exclusion Constraints

The links are associated with link-affinity or admin-groups. The link-affinity can be used to indicate a characteristic of a link, such as capacity, encryption, geography, etc. The inter-domain solution should support the creation of paths across different domains that satisfy link inclusion/exclusion constraints. The link constraints should also be satisfied for inter-domain links, such as those between ASBRs.

#### 4.3.4. Node Inclusion/Exclusion Constraints

Creating an inter-domain path that includes or excludes a certain set of nodes in each domain should be supported. The inter-domain solution should be independent of the mechanisms used to achieve the node inclusion/exclusion constraints within a domain. For example, an RSVP-based domain may use link affinities to achieve node exclusion constraints, while an SR-based domain may use flex-algo, which natively supports excluding nodes.

#### 4.3.5. Domain Inclusion/Exclusion Constraints

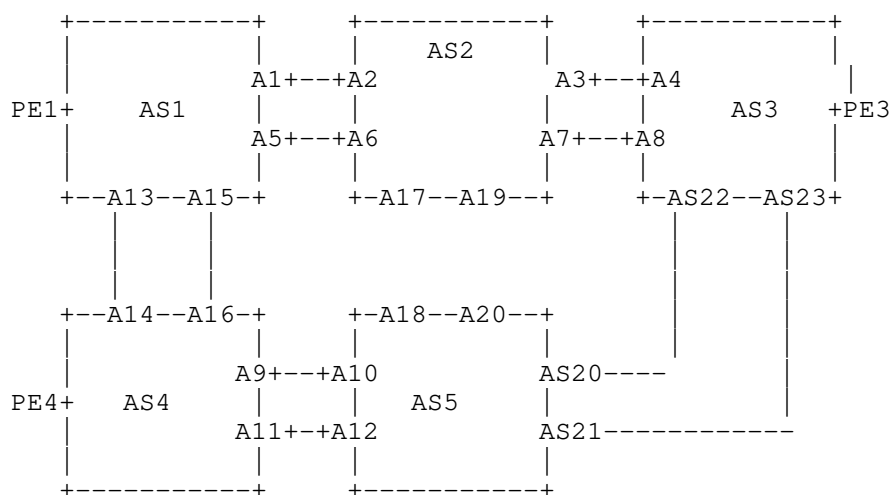


Figure 10: Multi-domain Network

Diagram Figure 10 shows a multi-domain, multi-AS network with the possibility for AS-diverse paths. The inter-domain solution should support creation of end-to-end paths that includes/excludes a certain domain entirely. For example, a network operator should be able to use the solution to create a path from PE1 to PE3 that automatically avoids passing through nodes belonging to AS2.

#### 4.3.6. Diverse Paths

The solution should support the creation of node and link-diverse inter-domain paths.

The intra-domain portion of the end-to-end paths should make use of existing mechanisms for computing and instantiating diverse paths within a domain.

Inter-domain links (such as those connecting ASBRs) should also be taken into account for diverse inter-domain paths.

The solution should support SRLG-aware inter-domain diverse paths.

#### 4.3.7. Constraint applicability to a subset of domains

Use cases such as data sovereignty described in Section 3.1 require that the paths with certain constraints are applicable to only a subset of domains. In domains where a constraint is not applicable, the end-to-end path should not create any state on the internal nodes.

#### 4.3.8. Service function chaining

Support the case where the set of service functions to be applied are deployed in single domain.

Support the case where the set of service functions to be applied are deployed across multiple domains.

Support virtualized service functions as well as service functions based on physical appliances.

Support the movement of a virtualized service function from one location to another.

Support high availability for service functions.

#### 4.4. Multicast specific requirements

Many of the requirements above are applicable to multicast traffic as well. Some requirements need to be refined with respect to multicast. Multicast also has some unique requirements not shared by unicast. These requirements will be covered in a future version of this document.

#### 4.5. Interoperate with BGP-LU

Seamless MPLS architecture is widely deployed and BGP-LU [RFC3107] is used to connect different domains. The inter-domain solution for intent-based paths should be interoperable with BGP-LU.

#### 4.6. Merger and Migration Requirements

##### 4.6.1. Option A and Option B Usecases

Options A and B require additional state on border nodes, so they are typically less scalable than option C. However, options A and B can be advantageous when it is necessary to do filtering or policing on border nodes. When option A or B is deployed, the solution should still meet the SLA requirements described in Section 4.3.

#### 4.6.2. Inter-Domain Intent Translation

In cases where two network domains previously under different administrations merge to come under a single administration, it may be preferable to use option C connectivity between the domains. The paths that fulfill the same intent may be represented using different conventions in each domain. The inter-domain solution should support efficient translation of intent from one representation to another.

#### 4.6.3. Native Support for Best Effort Paths

The inter-domain solution for intent-based paths should also provide the ability to create end-to-end best effort paths with accumulated IGP metric across the domains. A deployment should not require two different mechanisms to be deployed for best effort and intent-based paths.

#### 4.6.4. Interoperate with Other tunneling Mechanisms

As described in Section 4.2.1 and Section 3.6 the inter-domain solution should support one domain having one type of tunneling mechanism and another domain having another type of tunneling mechanism. The different tunneling mechanisms may completely differ in control plane and data plane operations (e.g. SRv6 and MPLS.) The inter-domain solution should provide interoperability between various tunneling mechanisms and provide the ability to create end-to-end intent-based paths.

#### 4.7. Scalability Requirements

The inter-domain solution should be able to support up to 1 million nodes.

The inter-domain solution should facilitate the use of access nodes with low RIB/FIB and low CPU capabilities.

The inter-domain solution should facilitate the use of access nodes with low label stacking capability.

The inter-domain solution should allow for a scalable response to network events. An individual node should only need to respond to a limited subset of network events.

Service routes on the border nodes should be minimized.

Non-MPLS versions of the inter-domain solution should support summarization of prefixes in order achieve scalability.

The inter-domain solution should facilitate filtering in order to ensure the access nodes need to receive and process only the endpoint prefixes that the access node needs to send traffic to.

The inter-domain solution should minimize state on the border nodes in order to reduce label and FIB resource consumption on border nodes.

#### 4.8. Availability Requirements

Traffic should be Fast Reroute (FRR) protected against link, node, and SRLG failures within a domain.

Traffic should be FRR protected against border node failures.

Traffic should be FRR protected against inter-domain link failures.

Traffic should be FRR protected against egress node and egress link failures.

#### 4.9. Operations and Automation Requirements

Each domain should be independent and should not depend on the transport technology in another domain. This allows for more flexible evolution of the network.

Basic MPLS OAM mechanisms described in [RFC8029] should be supported for MPLS based solutions.

End-to-end ping and traceroute procedures should be supported.

The ability to validate the path inside each domain should be supported.

Statistics for inter-domain intent-based paths should be supported on a per path basis on the ingress and egress PE nodes as well as border nodes.

The choice of transport tunnels that make up the inter-domain path should be derived automatically from the intent that the path fulfills.

The intent defined as color in the SR-TE architecture [I-D.ietf-idr-segment-routing-te-policy] should map automatically for all controller to router protocols such as BGP-SR-TE [I-D.ietf-idr-segment-routing-te-policy], PCEP-SR [I-D.ietf-pce-segment-routing-policy-cp], and NETCONF.

The intent should be mapped automatically from flex-algo number [I-D.ietf-lsr-flex-algo].

When access devices have CPU and memory constraints, it is useful to be able to filter prefix advertisements using policies as described in Section 4.7. For large networks it is operationally a tedious and erroneous process to manage this. The inter-domain solution should facilitate filtering the advertisements automatically, based on the service prefixes it receives from end-points.

#### 4.10. Service Mapping Requirements

The above requirements focus on the service independent aspects of inter-domain intent-based paths. In order for different services to effectively use these paths, flexible service mapping is required. The sections below summarize the requirements needed to achieve flexible service mapping.

##### 4.10.1. Traffic service mapping

Automated steering of traffic onto transport paths based on communities carried in the service prefix advertisements should be supported.

Steering of traffic on to transport paths based on the DSCP value carried in IPv4/IPv6 packets should be supported.

Traffic steering based on EXP bits in the MPLS header should be supported.

Traffic steering based on 5-tuple packet filter should be supported. Source address, destination address, source port, destination port and protocol fields should be allowed.

All the above traffic steering mechanisms should be supported for all common types of service traffic, including L2 VPN and L3 VPN traffic and global internet traffic.

When a path that fulfills the desired intent is not available, fallback to a path that fulfills a secondary intent should be supported.

When a path that fulfills the desired intent is not available, fallback to a best-effort path should be supported.

When a path that fulfills the desired intent is not available, the option of not using a fallback path (i.e. dropping the traffic) should be supported.

#### 4.10.2. 1 to N service mapping

The core domain is expected to have more traffic engineering constraints as compared to metros. The ability to map the services to appropriate transport tunnels at service attachment points should be supported.

#### 4.11. Interaction with Other Approaches

This document focuses on use cases and requirements that may benefit from a distributed solution. Many of these same use cases and requirements can be addressed with centralized approaches or other distributed TE solutions. One example of a centralized approach is described in "Interconnecting Millions of Endpoints with Segment Routing" ([RFC8604]).

Distributed and centralized approaches have inherent tradeoffs. Some networks may use a single approach. Other networks may choose to use both distributed and centralized approaches to get the benefits of both. A distributed inter-domain solution should support the requirements below:

Support scenarios where some traffic uses paths created using a centralized approach, and other traffic uses paths created using the distributed solution.

Support scenarios where part of the distributed inter-domain path is created using a centralized approach.

Support scenarios where traffic uses a centralized inter-domain solution for primary traffic, and uses a distributed inter-domain solution as a backup.

The distributed solution should not have any inherent dependencies on centralized approaches.

The distributed solution should co-exist with other distributed TE solutions.

#### 5. Backward Compatibility

## 6. Security Considerations

TBD

## 7. IANA Considerations

## 8. Acknowledgements

Many thanks to Kireeti Kompella, Ron Bonica, Krzysztof Szarcowitz, Srihari Sangli, Julian Lucek, Ram Santhanakrishnan, for discussions and inputs. Thanks to Colby Barth, John Scudder, Joel Halpern for review and comments.

## 9. Contributors

1. Kaliraj Vairavakkalai

Juniper Networks

kaliraj@juniper.net

2. Jeffrey Zhang

Juniper Networks

zzhang@juniper.net

## 10. References

### 10.1. Normative References

[I-D.hegde-rtgwg-egress-protection-sr-networks]

Hegde, S., Lin, W., and P. Shaofu, "Egress Protection for Segment Routing (SR) networks", Work in Progress, Internet-Draft, draft-hegde-rtgwg-egress-protection-sr-networks-01, 15 November 2020, <<https://www.ietf.org/archive/id/draft-hegde-rtgwg-egress-protection-sr-networks-01.txt>>.

[I-D.ietf-idr-performance-routing]

Xu, X., Hegde, S., Talaulikar, K., Boucadair, M., and C. Jacquenet, "Performance-based BGP Routing Mechanism", Work in Progress, Internet-Draft, draft-ietf-idr-performance-routing-03, 22 December 2020, <<https://www.ietf.org/archive/id/draft-ietf-idr-performance-routing-03.txt>>.



- [I-D.kaliraj-idr-bgp-classful-transport-planes]  
Vairavakkalai, K., Venkataraman, N., Rajagopalan, B.,  
Mishra, G., Khaddam, M., Xu, X., Szarecki, R. J., and D.  
J. Gowda, "BGP Classful Transport Planes", Work in  
Progress, Internet-Draft, draft-kaliraj-idr-bgp-classful-  
transport-planes-12, 25 August 2021,  
<[https://www.ietf.org/archive/id/draft-kaliraj-idr-bgp-  
classful-transport-planes-12.txt](https://www.ietf.org/archive/id/draft-kaliraj-idr-bgp-classful-transport-planes-12.txt)>.
- [I-D.zzhang-bess-bgp-multicast]  
Zhang, Z., Giuliano, L., Patel, K., Wijnands, I., Mishra,  
M., and A. Gulko, "BGP Based Multicast", Work in Progress,  
Internet-Draft, draft-zzhang-bess-bgp-multicast-03, 29  
October 2019, <[https://www.ietf.org/archive/id/draft-  
zzhang-bess-bgp-multicast-03.txt](https://www.ietf.org/archive/id/draft-zzhang-bess-bgp-multicast-03.txt)>.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate  
Requirement Levels", BCP 14, RFC 2119,  
DOI 10.17487/RFC2119, March 1997,  
<<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC3107] Rekhter, Y. and E. Rosen, "Carrying Label Information in  
BGP-4", RFC 3107, DOI 10.17487/RFC3107, May 2001,  
<<https://www.rfc-editor.org/info/rfc3107>>.
- [RFC8669] Previdi, S., Filsfils, C., Lindem, A., Ed., Sreekantiah,  
A., and H. Gredler, "Segment Routing Prefix Segment  
Identifier Extensions for BGP", RFC 8669,  
DOI 10.17487/RFC8669, December 2019,  
<<https://www.rfc-editor.org/info/rfc8669>>.

## 10.2. Informative References

- [I-D.hegde-spring-node-protection-for-sr-te-paths]  
Hegde, S., Bowers, C., Litkowski, S., Xu, X., and F. Xu,  
"Node Protection for SR-TE Paths", Work in Progress,  
Internet-Draft, draft-hegde-spring-node-protection-for-sr-  
te-paths-07, 30 July 2020,  
<[https://www.ietf.org/archive/id/draft-hegde-spring-node-  
protection-for-sr-te-paths-07.txt](https://www.ietf.org/archive/id/draft-hegde-spring-node-protection-for-sr-te-paths-07.txt)>.
- [I-D.ietf-idr-link-bandwidth]  
Mohapatra, P. and R. Fernando, "BGP Link Bandwidth  
Extended Community", Work in Progress, Internet-Draft,  
draft-ietf-idr-link-bandwidth-07, 5 March 2018,  
<[https://www.ietf.org/archive/id/draft-ietf-idr-link-  
bandwidth-07.txt](https://www.ietf.org/archive/id/draft-ietf-idr-link-bandwidth-07.txt)>.

[I-D.ietf-idr-segment-routing-te-policy]

Previdi, S., Filsfils, C., Talaulikar, K., Mattes, P., Rosen, E., Jain, D., and S. Lin, "Advertising Segment Routing Policies in BGP", Work in Progress, Internet-Draft, draft-ietf-idr-segment-routing-te-policy-13, 7 June 2021, <<https://www.ietf.org/archive/id/draft-ietf-idr-segment-routing-te-policy-13.txt>>.

[I-D.ietf-idr-tunnel-encaps]

Patel, K., Velde, G. V. D., Sangli, S. R., and J. Scudder, "The BGP Tunnel Encapsulation Attribute", Work in Progress, Internet-Draft, draft-ietf-idr-tunnel-encaps-22, 7 January 2021, <<https://www.ietf.org/archive/id/draft-ietf-idr-tunnel-encaps-22.txt>>.

[I-D.ietf-lsr-flex-algo]

Psenak, P., Hegde, S., Filsfils, C., Talaulikar, K., and A. Gulko, "IGP Flexible Algorithm", Work in Progress, Internet-Draft, draft-ietf-lsr-flex-algo-17, 6 July 2021, <<https://www.ietf.org/archive/id/draft-ietf-lsr-flex-algo-17.txt>>.

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

Leymann, N., Decraene, B., Filsfils, C., Konstantynowicz, M., and D. Steinberg, "Seamless MPLS Architecture", Work in Progress, Internet-Draft, draft-ietf-mpls-seamless-mpls-07, 28 June 2014, <<https://www.ietf.org/archive/id/draft-ietf-mpls-seamless-mpls-07.txt>>.

[I-D.ietf-pce-segment-routing-policy-cp]

Koldychev, M., Sivabalan, S., Barth, C., Peng, S., and H. Bidgoli, "PCEP extension to support Segment Routing Policy Candidate Paths", Work in Progress, Internet-Draft, draft-ietf-pce-segment-routing-policy-cp-05, 23 May 2021, <<https://www.ietf.org/archive/id/draft-ietf-pce-segment-routing-policy-cp-05.txt>>.

[I-D.ietf-rtgwg-segment-routing-ti-lfa]

Litkowski, S., Bashandy, A., Filsfils, C., Francois, P., Decraene, B., and D. Voyer, "Topology Independent Fast Reroute using Segment Routing", Work in Progress, Internet-Draft, draft-ietf-rtgwg-segment-routing-ti-lfa-07, 29 June 2021, <<https://www.ietf.org/archive/id/draft-ietf-rtgwg-segment-routing-ti-lfa-07.txt>>.

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

Filsfils, C., Talaulikar, K., Voyer, D., Bogdanov, A., and P. Mattes, "Segment Routing Policy Architecture", Work in

Progress, Internet-Draft, draft-ietf-spring-segment-routing-policy-13, 28 May 2021, <<https://www.ietf.org/archive/id/draft-ietf-spring-segment-routing-policy-13.txt>>.

[I-D.ietf-spring-sr-service-programming]

Clad, F., Xu, X., Filsfils, C., Bernier, D., Li, C., Decraene, B., Ma, S., Yadlapalli, C., Henderickx, W., and S. Salsano, "Service Programming with Segment Routing", Work in Progress, Internet-Draft, draft-ietf-spring-sr-service-programming-05, 10 September 2021, <<https://www.ietf.org/archive/id/draft-ietf-spring-sr-service-programming-05.txt>>.

[I-D.ietf-spring-srv6-network-programming]

Filsfils, C., Garvia, P. C., Leddy, J., Voyer, D., Matsushima, S., and Z. Li, "Segment Routing over IPv6 (SRv6) Network Programming", Work in Progress, Internet-Draft, draft-ietf-spring-srv6-network-programming-28, 29 December 2020, <<https://www.ietf.org/archive/id/draft-ietf-spring-srv6-network-programming-28.txt>>.

[I-D.saad-sr-fa-link]

Saad, T., Beeram, V. P., Barth, C., and S. Sivabalan, "Segment-Routing over Forwarding Adjacency Links", Work in Progress, Internet-Draft, draft-saad-sr-fa-link-03, 15 February 2021, <<https://www.ietf.org/archive/id/draft-saad-sr-fa-link-03.txt>>.

[I-D.voyer-pim-sr-p2mp-policy]

Voyer, D., Filsfils, C., Parekh, R., Bidgoli, H., and Z. Zhang, "Segment Routing Point-to-Multipoint Policy", Work in Progress, Internet-Draft, draft-voyer-pim-sr-p2mp-policy-02, 10 July 2020, <<https://www.ietf.org/archive/id/draft-voyer-pim-sr-p2mp-policy-02.txt>>.

[RFC1997] Chandra, R., Traina, P., and T. Li, "BGP Communities Attribute", RFC 1997, DOI 10.17487/RFC1997, August 1996, <<https://www.rfc-editor.org/info/rfc1997>>.

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

- [RFC4684] Marques, P., Bonica, R., Fang, L., Martini, L., Raszuk, R., Patel, K., and J. Guichard, "Constrained Route Distribution for Border Gateway Protocol/MultiProtocol Label Switching (BGP/MPLS) Internet Protocol (IP) Virtual Private Networks (VPNs)", RFC 4684, DOI 10.17487/RFC4684, November 2006, <<https://www.rfc-editor.org/info/rfc4684>>.
- [RFC5065] Traina, P., McPherson, D., and J. Scudder, "Autonomous System Confederations for BGP", RFC 5065, DOI 10.17487/RFC5065, August 2007, <<https://www.rfc-editor.org/info/rfc5065>>.
- [RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, DOI 10.17487/RFC5357, October 2008, <<https://www.rfc-editor.org/info/rfc5357>>.
- [RFC6388] Wijnands, IJ., Ed., Minei, I., Ed., Kompella, K., and B. Thomas, "Label Distribution Protocol Extensions for Point-to-Multipoint and Multipoint-to-Multipoint Label Switched Paths", RFC 6388, DOI 10.17487/RFC6388, November 2011, <<https://www.rfc-editor.org/info/rfc6388>>.
- [RFC7311] Mohapatra, P., Fernando, R., Rosen, E., and J. Uttaro, "The Accumulated IGP Metric Attribute for BGP", RFC 7311, DOI 10.17487/RFC7311, August 2014, <<https://www.rfc-editor.org/info/rfc7311>>.
- [RFC7471] Giacalone, S., Ward, D., Drake, J., Atlas, A., and S. Previdi, "OSPF Traffic Engineering (TE) Metric Extensions", RFC 7471, DOI 10.17487/RFC7471, March 2015, <<https://www.rfc-editor.org/info/rfc7471>>.
- [RFC7510] Xu, X., Sheth, N., Yong, L., Callon, R., and D. Black, "Encapsulating MPLS in UDP", RFC 7510, DOI 10.17487/RFC7510, April 2015, <<https://www.rfc-editor.org/info/rfc7510>>.
- [RFC7665] Halpern, J., Ed. and C. Pignataro, Ed., "Service Function Chaining (SFC) Architecture", RFC 7665, DOI 10.17487/RFC7665, October 2015, <<https://www.rfc-editor.org/info/rfc7665>>.
- [RFC8029] Kompella, K., Swallow, G., Pignataro, C., Ed., Kumar, N., Aldrin, S., and M. Chen, "Detecting Multiprotocol Label Switched (MPLS) Data-Plane Failures", RFC 8029, DOI 10.17487/RFC8029, March 2017, <<https://www.rfc-editor.org/info/rfc8029>>.

- [RFC8287] Kumar, N., Ed., Pignataro, C., Ed., Swallow, G., Akiya, N., Kini, S., and M. Chen, "Label Switched Path (LSP) Ping/Traceroute for Segment Routing (SR) IGP-Prefix and IGP-Adjacency Segment Identifiers (SIDs) with MPLS Data Planes", RFC 8287, DOI 10.17487/RFC8287, December 2017, <<https://www.rfc-editor.org/info/rfc8287>>.
- [RFC8402] Filtsfils, 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>>.
- [RFC8570] Ginsberg, L., Ed., Previdi, S., Ed., Giacalone, S., Ward, D., Drake, J., and Q. Wu, "IS-IS Traffic Engineering (TE) Metric Extensions", RFC 8570, DOI 10.17487/RFC8570, March 2019, <<https://www.rfc-editor.org/info/rfc8570>>.
- [RFC8604] Filtsfils, C., Ed., Previdi, S., Dawra, G., Ed., Henderickx, W., and D. Cooper, "Interconnecting Millions of Endpoints with Segment Routing", RFC 8604, DOI 10.17487/RFC8604, June 2019, <<https://www.rfc-editor.org/info/rfc8604>>.
- [RFC8679] Shen, Y., Jeganathan, M., Decraene, B., Gredler, H., Michel, C., and H. Chen, "MPLS Egress Protection Framework", RFC 8679, DOI 10.17487/RFC8679, December 2019, <<https://www.rfc-editor.org/info/rfc8679>>.
- [TS.23.501-3GPP] 3rd Generation Partnership Project (3GPP), "System Architecture for 5G System; Stage 2, 3GPP TS 23.501 v16.4.0", March 2020.

#### Authors' Addresses

Shraddha Hegde  
Juniper Networks Inc.  
Exora Business Park  
Bangalore 560103  
KA  
India

Email: [shraddha@juniper.net](mailto:shraddha@juniper.net)

Chris Bowers  
Juniper Networks Inc.

Email: cbowers@juniper.net

Xiaohu Xu  
Alibaba Inc.  
Beijing  
China

Email: xiaohu.xxh@alibaba-inc.com

Arkadiy Gulko  
EdwardJones

Email: arkadiy.gulko@edwardjones.com

Alex Bogdanov  
Google Inc.

Email: bogdanov@google.com

James Uttaro  
ATT

Email: jul738@att.com

Luay Jalil  
Verizon

Email: luay.jalil@verizon.com

Mazen Khaddam  
Cox communications

Email: mazen.khaddam@cox.com

Andrew Alston  
Liquid Telecom

Email: andrew.alston@liquidtelecom.com

Luis M. Contreras  
Telefonica  
Ronda de la Comunicacion, s/n  
Sur-3 building, 3rd floor  
28050 Madrid  
Spain

Email: [luismiguel.contrerasmurillo@telefonica.com](mailto:luismiguel.contrerasmurillo@telefonica.com)  
URI: <http://lmcontreras.com/>

Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: April 26, 2021

Z. Hu  
Huawei Technologies  
H. Chen  
Futurewei  
J. Yao  
Huawei Technologies  
C. Bowers  
Juniper Networks  
Y. Zhu  
China Telecom  
October 23, 2020

SR-TE Path Midpoint Protection  
draft-hu-spring-segment-routing-proxy-forwarding-12

Abstract

Segment Routing Traffic Engineering (SR-TE) supports explicit paths using segment lists containing adjacency-SIDs, node-SIDs and binding-SIDs. The current SR FRR such as TI-LFA provides fast re-route protection for the failure of a node along a SR-TE path by the direct neighbor or say point of local repair (PLR) to the failure. However, once the IGP converges, the SR FRR is no longer sufficient to forward traffic of the path around the failure, since the non-neighbors of the failure will no longer have a route to the failed node. This document describes a mechanism for fast re-route protection against the failure of a SR-TE path after the IGP converges. It provides fast re-route protection for an adjacency segment, a node segment and a binding segment of the path.

Requirements Language

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

Status of This Memo

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

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



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

This Internet-Draft will expire on April 26, 2021.

#### Copyright Notice

Copyright (c) 2020 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 . . . . .	3
2. Proxy Forwarding . . . . .	3
3. Extensions to IGP for Proxy Forwarding . . . . .	4
3.1. Extensions to OSPF . . . . .	4
3.1.1. Advertising Proxy Forwarding . . . . .	4
3.1.2. Advertising Binding Segment . . . . .	7
3.2. Extensions to IS-IS . . . . .	10
3.2.1. Advertising Proxy Forwarding . . . . .	10
3.2.2. Advertising Binding Segment . . . . .	12
4. Building Proxy Forwarding Table . . . . .	13
4.1. Advertising Proxy Forwarding . . . . .	15
4.2. Building Proxy Forwarding Table . . . . .	15
5. Node Protection for Segment List . . . . .	15
5.1. Next Segment is an Adjacency Segment . . . . .	16
5.2. Next Segment is a Node Segment . . . . .	16
5.3. Next Segment is a Binding Segment . . . . .	17
6. Security Considerations . . . . .	18
7. IANA Considerations . . . . .	18
7.1. OSPFv2 . . . . .	18
7.2. OSPFv3 . . . . .	19
7.3. IS-IS . . . . .	19
8. Acknowledgements . . . . .	20
9. References . . . . .	20
9.1. Normative References . . . . .	20

9.2. Informative References . . . . .	21
Authors' Addresses . . . . .	22

## 1. Introduction

Segment Routing Traffic Engineering (SR-TE) is a technology that implements traffic engineering using a segment list. SR-TE supports the creation of explicit paths using adjacency-SIDs, node-SIDs, anycast-SIDs, and binding-SIDs. A node-SID in the segment list defining an SR-TE path indicates a loose hop that the SR-TE path should pass through. When the node fails, the network may no longer be able to properly forward traffic on that SR-TE path.

[I-D.ietf-rtgwg-segment-routing-ti-lfa] describes an SR FRR mechanism that provides fast re-route protection for the failure of a node on a SR-TE path by the direct neighbor or say point of local repair (PLR) to the failure. However, once the IGP converges, the SR FRR is no longer sufficient to forward traffic of the path around the failure, since the non-neighbors of the failure will no longer have a route to the failed node and drop the traffic.

To solve this problem, [I-D.ietf-spring-segment-protection-sr-te-paths] proposes that a hold timer should be configured on every router in a network. After the IGP converges on the event of a node failure, if the node-SID of the failed node becomes unreachable, the forwarding changes should not be communicated to the forwarding planes on all configured routers (including PLRs for the failed node) until the hold timer expires. This solution may not work for some cases such as some of nodes in the network not supporting this solution.

This document describes a proxy protection/forwarding mechanism, which provides more protection coverages. It considers the fast re-route protection capability of every node in the network and supports the fast re-route protection of the binding-SIDs on a failed node.

## 2. Proxy Forwarding

In the proxy forwarding mechanism, each neighbor of a possible failed node advertises its SR proxy forwarding capability in its network domain when it has the capability. This capability indicates that the neighbor (the Proxy Forwarder) will forward traffic on behalf of the failed node. A router receiving the SR Proxy Forwarding capability from neighbors of a failed node will send traffic using the node-SID of the failed node to the nearest Proxy Forwarder after the IGP converges on the event of the failure.

Once the affected traffic reaches a Proxy Forwarder, it sends the traffic on the post-failure shortest path to the node immediately following the failed node in the segment list.

For a binding segment of a possible failed node, the node advertises the information about the binding segment, including the binding SID and the list of SIDs associated with the binding SID, to its direct neighbors only. Note that the information is not advertised in the network domain.

After the node fails and the IGP converges on the failure, the traffic with the binding SID of the failed node will reach its neighbor having SR Proxy Forwarding capability. Once receiving the traffic, the neighbor swaps the binding SID with the list of SIDs associated with the binding SID and sends the traffic along the post-failure shortest path to the first node in the segment list.

### 3. Extensions to IGP for Proxy Forwarding

This section defines extensions to IGP for advertising the SR proxy forwarding capability of a node in a network domain and the information about each binding segment (including its binding SID and the list of SIDs associated) of a node to its direct neighbors.

#### 3.1. Extensions to OSPF

##### 3.1.1. Advertising Proxy Forwarding

When a node P has the capability to do a SR proxy forwarding for all its neighboring nodes for protecting the failures of these nodes, node P advertises its SR proxy forwarding capability in its router information opaque LSA, which contains a Router Functional Capabilities TLV of the format as shown in Figure 1.

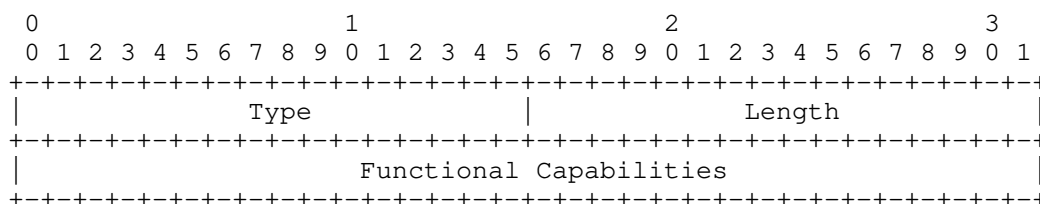


Figure 1: Router Functional Capabilities TLV

One bit (called PF bit) in the Functional Capabilities field of the TLV is used to indicate node P's SR proxy forwarding capability. When this bit is set to one by node P, it indicates that node P is capable of doing a SR proxy forwarding for its neighboring nodes.

For a node X in the network, it learns the prefix/node SID of node N, which is originated and advertised by node N. It creates a proxy prefix/node SID of node N for node P if node P is capable of doing SR proxy forwarding for node N. The proxy prefix/node SID of node N for node P is a copy of the prefix/node SID of node N originated by node N, but stored under (or say, associated with) node P.

In normal operations, node X prefers to use the prefix/node SID of node N. When node N fails, node X prefers to use the proxy prefix/node SID of node N. Thus node X will forward the traffic targeting to the prefix/node SID of node N to node P when node N fails, and node P will do a SR proxy forwarding for node N and forwarding the traffic to its final destination without going through node N. After node N fails, node X will keep the FIB entry to the proxy prefix/node SID of node N for a given period of time.

Note that the behaviors of normal IP forwarding and routing convergences in a network are not changed at all by the SR proxy forwarding. For example, the next hop used by BGP is an IP address (or prefix). The IGP and BGP converge in normal ways for changes in the network. The packet with its IP destination to this next hop is forwarded according to the IP forwarding table (FIB) derived from IGP and BGP routes.

If node P can not do a SR proxy forwarding for all its neighboring nodes, but for some of them, then it advertises the node SID of each of the nodes as a proxy node SID, indicating that it is able to do proxy forwarding for the node SID.

A new TLV, called Proxy Node SIDs TLV, is defined for node P to advertise the node SIDs of some of its neighboring nodes. It has the format as shown in Figure 2.

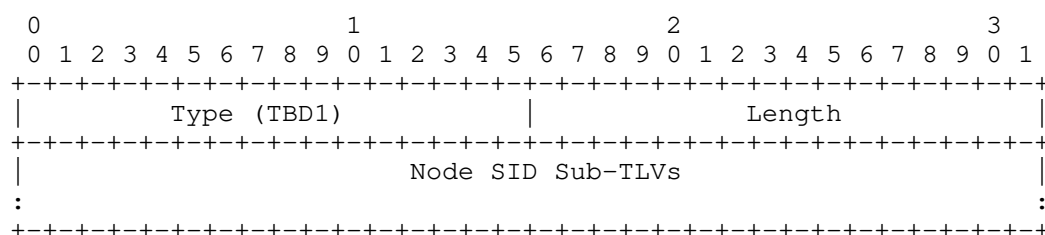


Figure 2: OSPF Proxy Node SIDs TLV

The Type (TBD1) is to be assigned by IANA. The TLV contains a number of Node SID Sub-TLVs. The Length is the total size of the Node SID Sub-TLVs included in the TLV. A Node SID Sub-TLV is the Prefix SID Sub-TLV defined in [I-D.ietf-ospf-segment-routing-extensions].

A proxy forwarding node P originates an Extended Prefix Opaque LSA containing this new TLV. The TLV includes the Node SID Sub-TLVs for the node SIDs of some of P's neighboring nodes. For each of some of P's neighboring nodes, the Node SID Sub-TLV for its prefix/node SID is included the TLV. This prefix/node SID is called a proxy prefix/node SID.

A proxy forwarding node will originate an Extended Prefix Opaque LSA, which includes a Proxy Node SIDs TLV. The format of the LSA is shown in Figure 3.

For a proxy forwarding node P, having a number of neighboring nodes, P originates and maintains an Extended Prefix Opaque LSA, which includes a Proxy Node SIDs TLV. The TLV contains the Prefix/Node SID Sub-TLV for each of some of the neighboring nodes after node P creates the corresponding proxy forwarding entries for protecting the failure of some of the neighboring nodes.

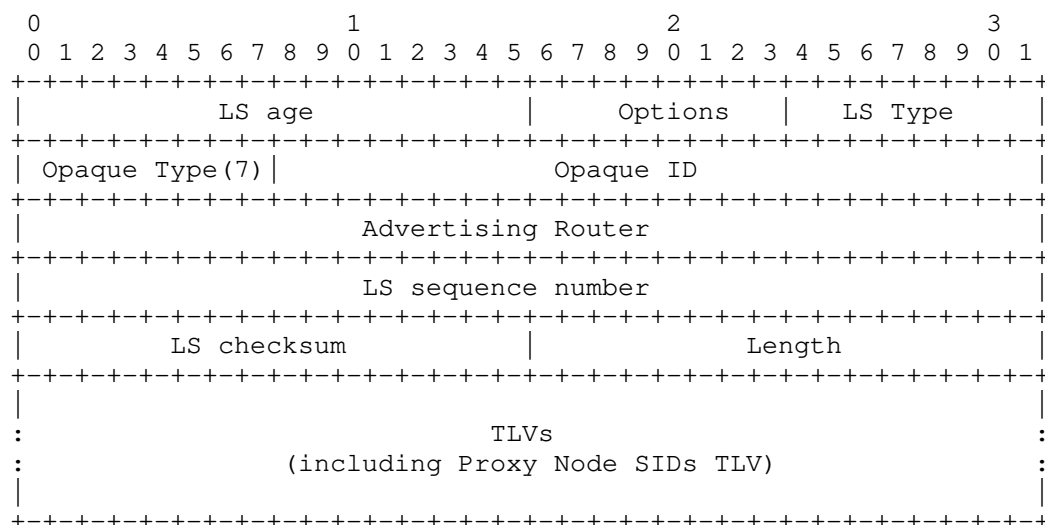


Figure 3: OSPFv2 Extended Prefix Opaque LSA

When an neighboring node fails, P maintains the LSA with the TLV containing the Prefix/Node SID Sub-TLV for the neighboring node for a given period of time. After the given period of time, the Prefix/Node SID Sub-TLV for the neighboring node is removed from the TLV in the LSA and then after a given time the corresponding proxy forwarding entries for protecting the failure of the neighboring node is removed.

For a node X in the network, it learns the prefix/node SID of node N and the proxy prefix/node SID of node N. The former is originated and advertised by node N, and the latter is originated and advertised by the proxy forwarding node P of node N. Note that the proxy Prefix/Node SID Sub-TLV for node N does not contain a prefix of node N, and the prefix is the prefix associated with the prefix/node SID of node N originated by node N.

In normal operations, node X prefers to use the prefix/node SID of node N. When node N fails, node X prefers to use the proxy prefix/node SID of node N. Thus node X will forward the traffic targeting to node N to node P when node N fails, and node P will do a proxy forwarding for node N and forwarding the traffic to its destination without going through node N.

### 3.1.2. Advertising Binding Segment

For a binding segment (or binding for short) on a node A, which consists of a binding SID and a list of segments, node A advertises an LSA containing the binding (i.e., the binding SID and the list of the segments). The LSA is advertised only to each of the node A's neighboring nodes. For OSPFv2, the LSA is a opaque LSA of LS type 9 (i.e., a link local scope LSA).

A binding segment is represented by binding segment TLV of the format as shown in Figure 4.

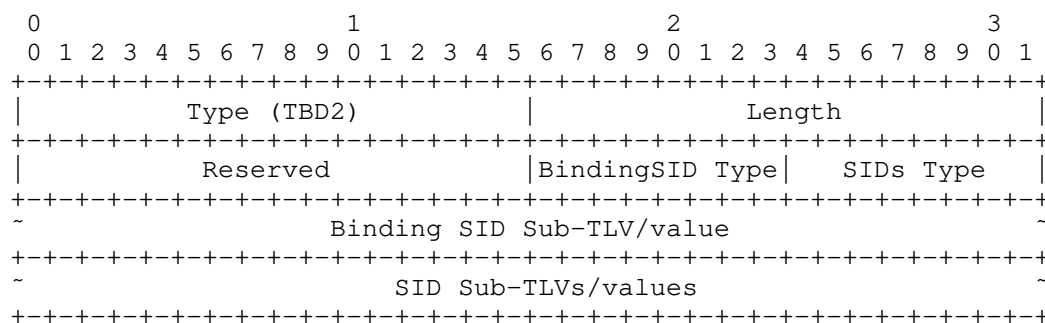


Figure 4: OSPF Binding Segment TLV

It comprises a binding SID and a list of segments (SIDs). The fields of this TLV are defined as follows:

Type: 2 octets, its value (TBD2) is to be assigned by IANA.

Length: 2 octets, its value is (4 + length of Sub-TLVs/values).

Binding SID Type (BT): 1 octet indicates whether the binding SID is represented by a Sub-TLV or a value included in the TLV. For the binding SID represented by a value, it indicates the type of binding SID. The following BT values are defined:

- o BT = 0: The binding SID is represented by a Sub-TLV (i.e., Binding SID Sub-TLV) in the TLV. A binding SID Sub-TLV is a SID/Label Sub-TLV defined in [I-D.ietf-ospf-segment-routing-extensions]. BT != 0 indicates that the binding SID is represented by a value.

- o BT = 1: The binding SID value is a label, which is represented by the 20 rightmost bits. The length of the value is 3 octets.

- o BT = 2: The binding SID value is a 32-bit SID. The length of the value is 4 octets.

SIDs Type (ST): 1 octet indicates whether the list of segments (SIDs) are represented by Sub-TLVs or values included in the TLV. For the SIDs represented by values, it indicates the type of SIDs. The following ST values are defined:

- o ST = 0: The SIDs are represented by Sub-TLVs (i.e., SID Sub-TLVs) in the TLV. A SID Sub-TLV is an Adj-SID Sub-TLV, a Prefix-SID Sub-TLV or a SID/Label Sub-TLV defined in [I-D.ietf-ospf-segment-routing-extensions]. ST != 0 indicates that the SIDs are represented by values.

- o ST = 1: Each of the SID values is a label, which is represented by the 20 rightmost bits. The length of the value is 3 octets.

- o ST = 2: Each of the SID values is a 32-bit SID. The length of the value is 4 octets.

The opaque LSA of LS Type 9 containing the binding segment (i.e., the binding SID and the list of the segments) has the format as shown in Figure 5. It may have Opaque Type of x (the exact type is to be assigned by IANA) for Binding Segment Opaque LSA.

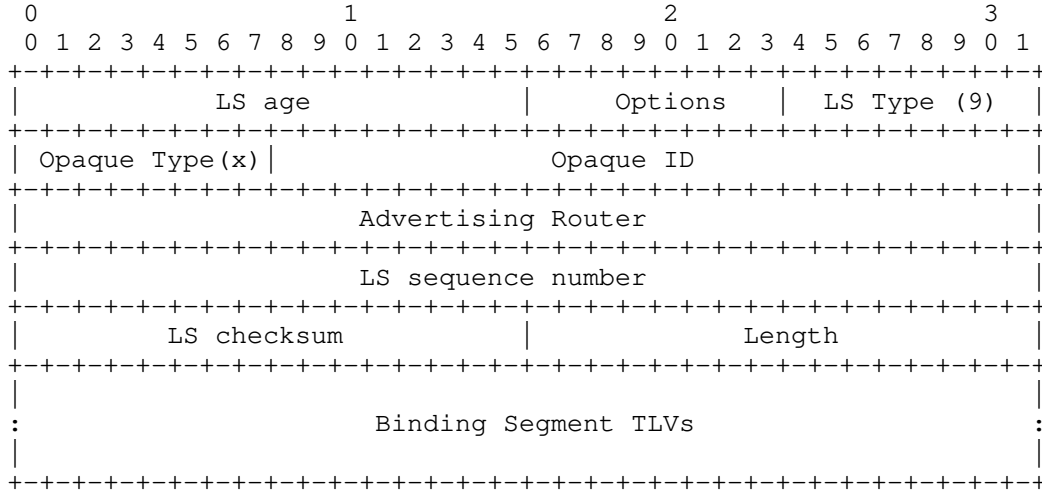


Figure 5: OSPFv2 Binding Segment Opaque LSA

For every binding on a node A, the LSA originated by A contains a binding segment TLV for it.

For node A running OSPFv3, it originates a link-local scoping LSA of a new LSA function code (TBD3) containing binding segment TLVs for the bindings on it. The format of the LSA is illustrated in Figure 6.

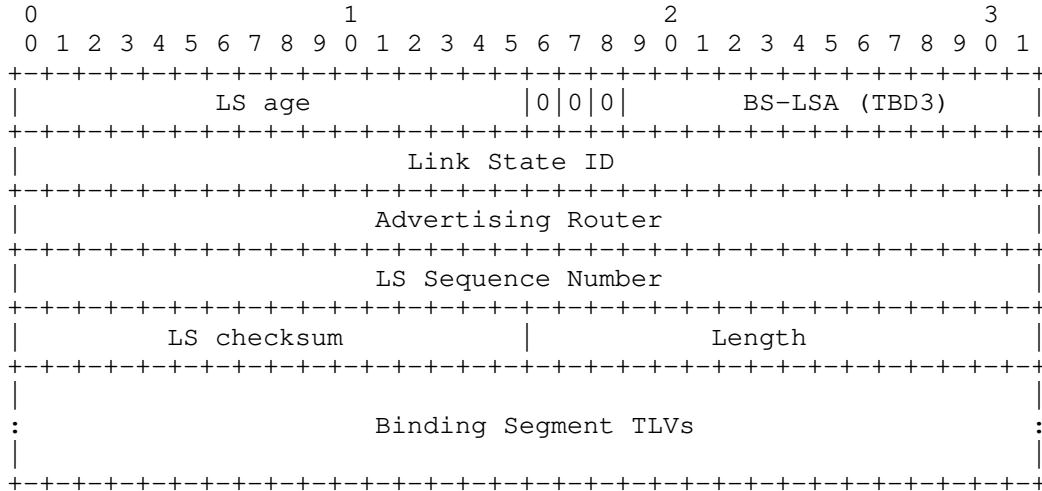


Figure 6: OSPFv3 Binding Segment Opaque LSA



The U-bit is set to 0, and the scope is set to 00 for link-local scoping.

### 3.2. Extensions to IS-IS

#### 3.2.1. Advertising Proxy Forwarding

When a node P has the capability to do a SR proxy forwarding for its neighboring nodes for protecting the failures of them, node P advertises its SR proxy forwarding capability in its LSP, which contains a Router Capability TLV of Type 242 including a SR capabilities sub-TLV of sub-Type 2.

One bit (called PF bit as shown in Figure 7) in the Flags field of the SR capabilities sub-TLV is defined to indicate node P's SR proxy forwarding capability. When this bit is set to one by node P, it indicates that node P is capable of doing a SR proxy forwarding for its neighboring nodes.

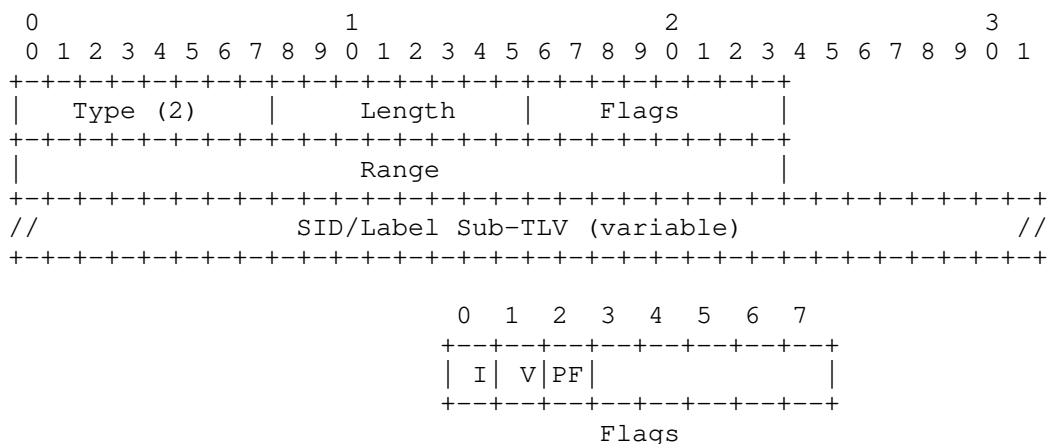


Figure 7: SR Capabilities sub-TLV

If node P can not do a SR proxy forwarding for all its neighboring nodes, but for some of them, then it advertises the node SID of each of the nodes as a proxy node SID, indicating that it is able to do proxy forwarding for the node SID.

The IS-IS SID/Label Binding TLV (suggested value 149) is defined in [I-D.ietf-isis-segment-routing-extensions]. A Proxy Forwarder uses the SID/Label Binding TLV to advertise the node SID of its neighboring node. The Flags field of the SID/Label Binding TLV is extended to include a P flag as shown in Figure 8. The prefix/node

SID in prefix/node SID Sub-TLV included in SID/Label Binding TLV is identified as a proxy forwarding prefix/node SID.

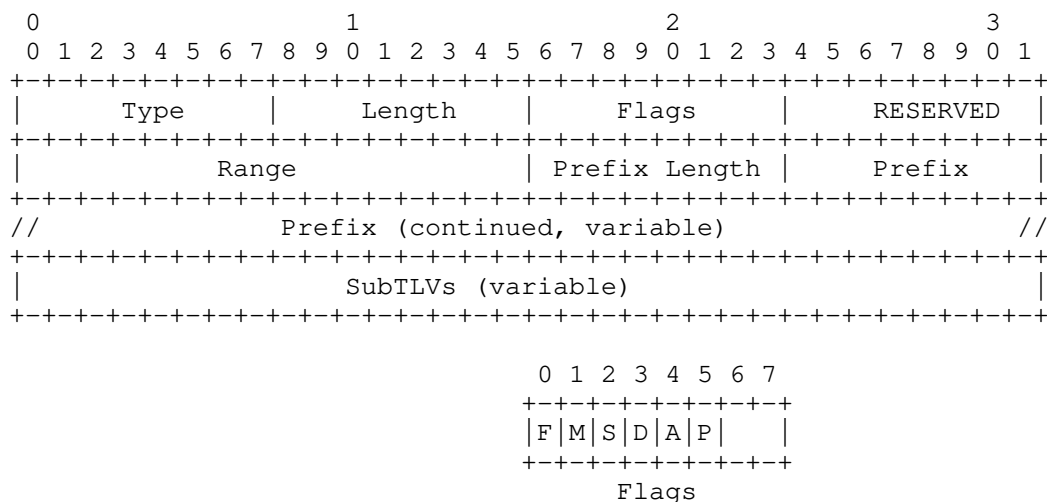


Figure 8: SID/Label Binding TLV

Where:

**P-Flag:** Proxy forwarding flag. If set, this prefix/node SID is advertised by the proxy node. This TLV is used to announce that the node has the ability to proxy forward the prefix/node SID.

When the P-flag is set in the SID/Label Binding TLV, the following usage rules apply.

The Range, Prefix Length and Prefix field are not used. They should be set to zero on transmission and ignored on receipt.

SID/Label Binding TLV contains a number of prefix/node SID Sub-TLVs. The TLV advertised by a proxy forwarding node P contains prefix/node SID Sub-TLVs for the node SIDs of P's neighbor nodes. Each of the Sub-TLVs is a prefix/node SID Sub-TLV defined in [I-D.ietf-isis-segment-routing-extensions]. From the SID in a prefix/node SID Sub-TLV advertised by the Proxy Forwarding node, its prefix can be obtained through matching corresponding prefix/node SID advertised by the neighbor/protected node using TLV-135 (or 235, 236, or 237) together with the prefix/node SID Sub-TLV.

### 3.2.2. Advertising Binding Segment

[I-D.ietf-spring-segment-routing-policy] has defined the usage of binding-SID. For supporting binding SID proxy forwarding, a new IS-IS TLV, called Binding Segment TLV, is defined. It contains a binding SID and a list of segments (SIDs). This TLV may be advertised in IS-IS Hello (IIH) PDUs, LSPs, or in Circuit Scoped Link State PDUs (CS-LSP) [RFC7356]. Its format is shown in Figure 9.

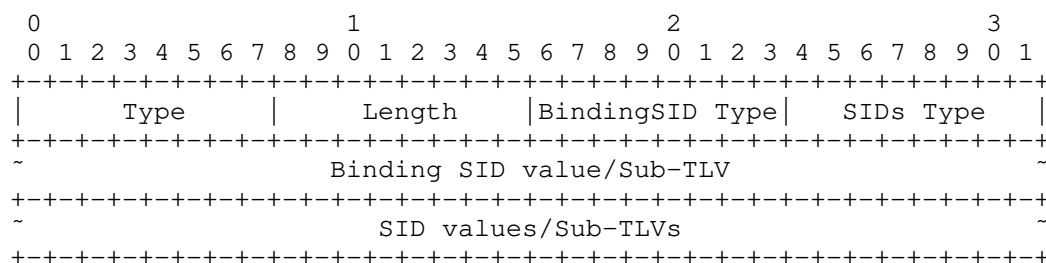


Figure 9: IS-IS Binding Segment TLV

The fields of this TLV are defined as follows:

Type: 1 octet Suggested value 152 (to be assigned by IANA)

Length: 1 octet (2 + length of Sub-TLVs/values).

Binding SID Type (BT): 1 octet indicates whether the binding SID is represented by a Sub-TLV or a value included in the TLV. For the binding SID represented by a value, it indicates the type of binding SID. The following BT values are defined:

- o BT = 0: The binding SID is represented by a Sub-TLV (i.e., binding SID Sub-TLV) in the TLV. A binding SID Sub-TLV is a SID/Label Sub-TLV defined in [I-D.ietf-isis-segment-routing-extensions]. BT != 0 indicates that the binding SID is represented by a value.

- o BT = 1: The binding SID value is a label, which is represented by the 20 rightmost bits. The length of the value is 3 octets.

- o BT = 2: The binding SID value is a 32-bit SID. The length of the value is 4 octets.

SIDs Type (ST): 1 octet indicates whether the SIDs are represented by Sub-TLVs or values included in the TLV. For the SIDs represented by values, it indicates the type of SIDs. The following ST values are defined:

- o ST = 0: The SIDs are represented by Sub-TLVs (i.e., SID Sub-TLVs) in the TLV. A SID Sub-TLV is an Adj-SID Sub-TLV, a Prefix-SID Sub-TLV or a SID/Label Sub-TLV defined in [I-D.ietf-isis-segment-routing-extensions]. ST != 0 indicates that the SIDs are represented by values.
- o ST = 1: Each of the SID values is a label, which is represented by the 20 rightmost bits. The length of the value is 3 octets.
- o ST = 2: Each of the SID values is a 32-bit SID. The length of the value is 4 octets.

#### 4. Building Proxy Forwarding Table

Figure 10 is used to illustrate the SR proxy forwarding approach. Each node N has SRGB = [N000-N999]. RT1 is an ingress node of SR domain. RT3 is a failure node. RT2 is a Point of Local Repair (PLR) node, i.e., a proxy forwarding node. Three label stacks are shown in the figure. Label Stack 1 uses only adjacency-SIDs and represents the path RT1->RT2->RT3->RT4->RT5. Label Stack 2 uses only node-SIDs and represents the ECMP-aware path RT1->RT3->RT4->RT5. Label Stack 3 uses a node-SID and a binding SID. The Binding-SID with label=100 at RT3 represents the ECMP-aware path RT3->RT4->RT5. So Label Stack 3, which consists of the node-SID for RT3 following by Binding-SID=100, represents the ECMP-aware path RT1->RT3->RT4->RT5.

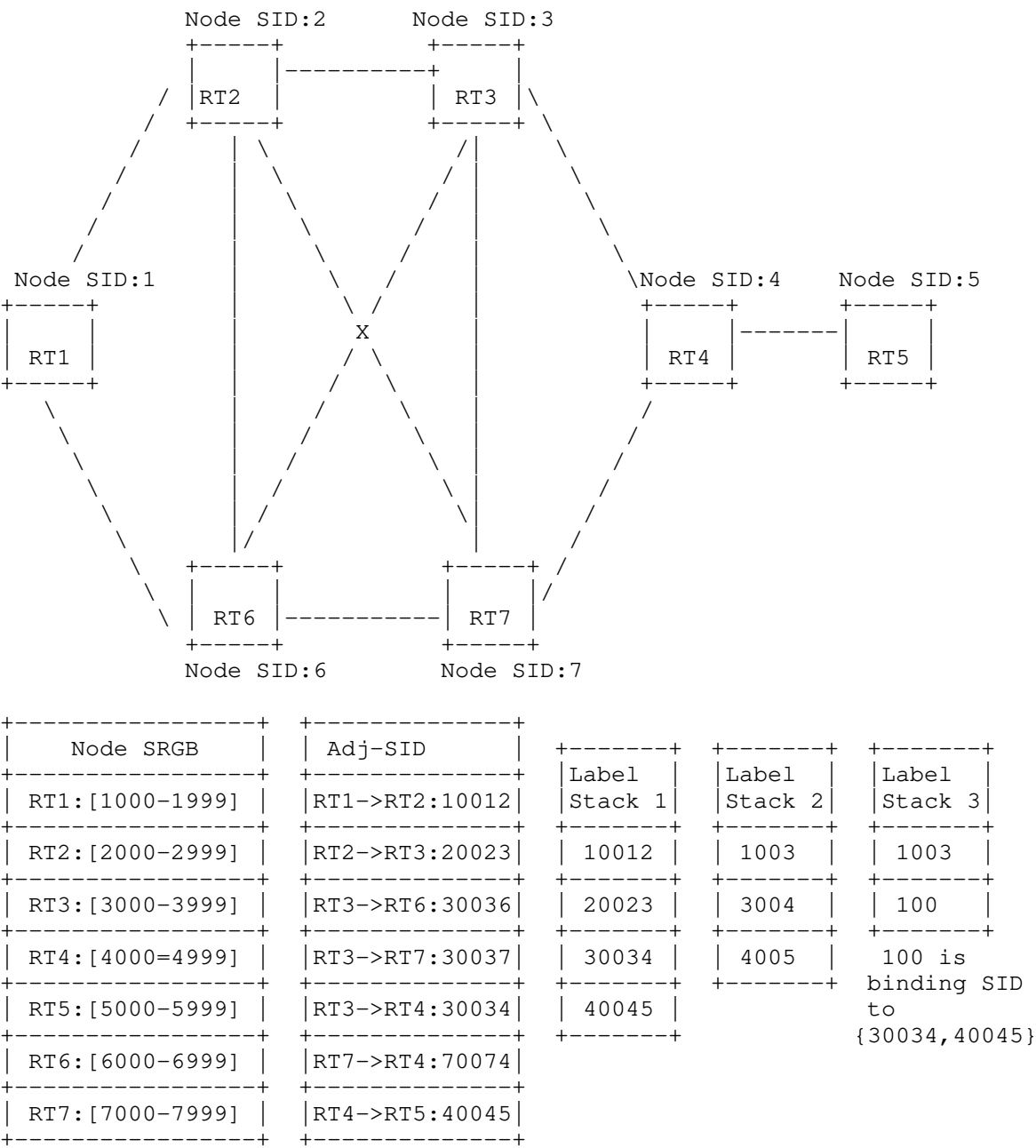


Figure 10: Topology of SR-TE Path

#### 4.1. Advertising Proxy Forwarding

If the Point of Local Repair (PLR), for example, RT2, has the capability to do a SR proxy forwarding for all its neighboring nodes, it must advertise this capability. If the PLR can not do a SR proxy forwarding for all its neighboring nodes, but for some of them, for example, RT3, then it uses proxy Node SIDs TLV to advertise the prefix-SID learned from RT3. The TLV contains the Sub-TLV/value for the prefix/node SID of RT3 as a proxy SID. When RT3 fails, RT2 needs to maintain the Sub-TLV/value for a period of time. When the proxy forwarding table corresponding to the fault node is deleted (see section 3.2), the Sub-TLV/value is withdrawn. The nodes in the network (for example, RT1) learn the prefix/node SID TLV advertised by RT3 and the proxy Node SIDs TLV advertised by RT2. When RT3 is normal, the nodes prefer prefix/node SID TLV. When the RT3 fails, the proxy prefix/node SIDs TLV advertised by RT2 is preferred.

#### 4.2. Building Proxy Forwarding Table

A SR proxy node P needs to build an independent proxy forwarding table for each neighbor N. The proxy forwarding table for node N contains the following information:

- 1: Node N's SRGB range and the difference between the SRGB start value of node P and that of node N;
- 2: All adjacency-SID of N and Node-SID of the node pointed to by node N's adjacency-SID.
- 3: The binding-SID of N and the label stack associated with the binding-SID.

Node P (PLR) uses a proxy forwarding table based on the next segment to find a node N as a backup forwarding entry to the adj-SID and Node-SID of node N. When node N fails, the proxy forwarding table needs to be maintained for a period of time, which is recommended for 30 minutes.

Node RT3 in the topology of Figure 1 is node N, and node RT2 is node P (PLR). RT2 builds the proxy forwarding table for RT3. The structure of the table and how to build the table is a local implementation issue.

#### 5. Node Protection for Segment List

Segment Routing Traffic Engineering supports the creation of explicit paths using adjacency-SIDs, node-SIDs, and binding-SIDs. The label stack is a combination of one or more of adjacency-SIDs, node-SIDs,

and binding-SIDs. This Section shows how a proxy node uses the SR proxy forwarding mechanism to protect traffic to the destination node when the next segment of label stack is adjacency-SIDs, node-SIDs, or binding-SIDs, respectively.

#### 5.1. Next Segment is an Adjacency Segment

As shown in Figure 1, Label Stack 1 {10012, 20023, 30034, 40045} represents SR-TE strict explicit path RT1->RT2->RT3->RT4->RT5. When RT3 fails, node RT2 acts as a PLR, and uses next adj-SID (30034) of the label stack to lookup the proxy forwarding table built by RT2 locally for RT3. The path returned is the label forwarding path to RT3's next hop node RT4, which bypasses RT3. The specific steps are as follows:

- a. RT1 pops top adj-SID 10012, and forwards the packet to RT2;
- b. RT2 uses the label 20023 to identify the next hop node RT3, which has failed. RT2 pops label 20023 and queries the Proxy Forwarding Table corresponding to RT3 with label 30034. The Proxy Forwarding Table corresponding to RT3 returns an outgoing interface and label stack representing a path to RT4 that does not pass through RT3. In this case, outgoing interface to RT7 with label stack 7004, satisfies this requirement.
- c. So the packet leaves RT2 out the interface to RT7 with label stack {7004, 40045}. RT4 forwards it to RT4, where the original path is rejoined.
- d. RT2 forwards packets to RT7. RT7 queries the local routing table to forward the packet to RT4.

#### 5.2. Next Segment is a Node Segment

As shown in Figure 1, Label Stack 2 {1003, 3004, 4005} represents SR-TE loose path RT1->RT3->RT4->RT5, where 1003 is the node SID of RT3.

When the node RT3 fails, the proxy forwarding TLV advertised by the RT2 is preferred to direct the traffic of the RT1 to the PLR node RT2. Node RT2 acts as a PLR node and queries the proxy forwarding table locally built for RT3. The path returned is the label forwarding path to RT3's next hop node RT4, which bypasses RT3. The specific steps are as follows:

- a. RT1 swaps label 1003 to out-label 2003 to RT3.

- b. RT2 receives the label forwarding packet whose top label of label stack is 2003, and searches for the local Routing Table, the behavior found is to lookup Proxy Forwarding table due to RT3 failure.
- c. RT2 uses 2003 as the in-label to lookup Proxy Forwarding table, and the query result is forwarding the packet to RT4.
- d. Then RT2 queries the Routing Table to RT4, using the primary or backup path to RT4. The next hop is RT7.
- e. RT2 forwards the packet to RT7. RT7 queries the local routing table to forward the packet to RT4.
- f. After RT1 convergences, node SID 1003 is preferred to the proxy SID implied/advertised by RT2.

### 5.3. Next Segment is a Binding Segment

As shown in Figure 1, Label Stack 3 {1003, 100} represents SR-TE loose path RT1->RT3->RT4->RT5, where 100 is a Binding-SID, which represents segment list {30034, 40045}.

When the node RT3 fails, the proxy forwarding SID implied or advertised by the RT2 is preferred to forward the traffic of the RT1 to the PLR node RT2. Node RT2 acts as a PLR node and uses Binding-SID to query the proxy forwarding table locally built for RT3. The path returned is the label forwarding path to RT3's next hop node (RT4), which bypasses RT3. The specific steps are as follows:

- a. RT1 swaps label 1003 to out-label 2003 to RT3.
- b. RT2 receives the label forwarding packet whose top label of label stack is 2003, and searches for the local Routing Table, the behavior found is to lookup Proxy Forwarding table due to RT3 failure.
- c. RT2 uses Binding-SID:100 (label 2003 has pop) as the in-label to lookup the Next Label record of the Proxy Forwarding Table, the behavior found is to swap to Segment list {30034, 40045}.
- d. RT2 swaps Binding-SID:100 to Segment list {30034, 40045}, and uses the 3034 to lookup the Next Label record of the Proxy Forwarding table again. The behavior found is to forward the packet to RT4.
- e. RT2 queries the Routing Table to RT4, using primary or backup path to RT4. The next hop is RT7.
- f. RT2 forwards packets to RT7. RT7 queries the local routing table to forward the packet to RT4.



## 6. Security Considerations

The extensions to OSPF and IS-IS described in this document result in two types of behaviors in data plane when a node in a network fails. One is that for a node, which is a upstream (except for the direct upstream) node of the failed node along a SR-TE path, it continues to send the traffic to the failed node along the SR-TE path for an extended period of time. The other is that for a node, which is the direct upstream node of the failed node, it fast re-routes the traffic around the failed node to the direct downstream node of the failed node along the SR-TE path. These behaviors are internal to a network and should not cause extra security issues.

## 7. IANA Considerations

### 7.1. OSPFv2

Under Subregistry Name "OSPF Router Functional Capability Bits" within the "Open Shortest Path First v2 (OSPFv2) Parameters" [RFC7770], IANA is requested to assign one bit for Proxy Forwarding Capability as follows:

Bit number	Capability Name	Reference
31	Proxy Forwarding	This document

Under Registry Name "OSPFv2 Extended Prefix Opaque LSA TLVs" [RFC7684], IANA is requested to assign one new TLV value for OSPF Proxy Node SIDs as follows:

TLV Value	TLV Name	Reference
2	Proxy Node SIDs TLV	This document

Under Registry Name "Opaque Link-State Advertisements (LSA) Option Types" [RFC5250], IANA is requested to assign new Opaque Type registry values for Binding Segment LSA as follows:

Registry Value	Opaque Type	Reference
10	Binding Segment	This document

IANA is requested to create and maintain new registries:

- o OSPFv2 Binding Segment Opaque LSA TLVs

Initial values for the registry are given below. The future assignments are to be made through IETF Review [RFC5226].

Value	TLV Name	Definition
-----	-----	-----
0	Reserved	
1	Binding Segment TLV	This Document
2-32767	Unassigned	
32768-65535	Reserved	

## 7.2. OSPFv3

Under Registry Name "OSPFv3 LSA Function Codes", IANA is requested to assign new registry values for Binding Segment LSA as follows:

Value	LSA Function Code Name	Reference
-----	-----	-----
16	Binding Segment LSA	This document
-----	-----	-----

IANA is requested to create and maintain new registries:

- o OSPFv3 Binding Segment LSA TLVs

Initial values for the registry are given below. The future assignments are to be made through IETF Review [RFC5226].

Value	TLV Name	Definition
-----	-----	-----
0	Reserved	
1	Binding Segment TLV	This Document
2-32767	Unassigned	
32768-65535	Reserved	

## 7.3. IS-IS

Under Registration "Segment Routing Capability" in the "sub-TLVs for TLV 242" registry [I-D.ietf-isis-segment-routing-extensions], IANA is requested to assign one bit flag for Proxy Forwarding Capability as follows:

Bit number	Capability Name	Reference
2	Proxy Forwarding (PF)	This document

Under Registration "Segment Identifier/Label Binding TLV 149"  
[I-D.ietf-isis-segment-routing-extensions], IANA is requested to  
assign one bit P-Flag as follows:

Bit number	Flag Name	Reference
5	P-Flag	This document

Under Registry Name: IS-IS TLV Codepoints, IANA is requested to  
assign one new TLV value for IS-IS Binding Segment as follows:

Value	TLV Name	Reference
152	Binding Segment TLV	This Document

## 8. Acknowledgements

The authors would like to thank Peter Psenak, Acee Lindem, Les Ginsberg, Bruno Decraene and Jeff Tantsura for their comments to this work.

## 9. References

### 9.1. Normative References

- [I-D.ietf-isis-segment-routing-extensions]  
Previdi, S., Ginsberg, L., Filsfils, C., Bashandy, A., Gredler, H., and B. Decraene, "IS-IS Extensions for Segment Routing", draft-ietf-isis-segment-routing-extensions-25 (work in progress), May 2019.
- [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-27 (work in progress), December 2018.

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 5226, DOI 10.17487/RFC5226, May 2008, <<https://www.rfc-editor.org/info/rfc5226>>.
- [RFC5250] Berger, L., Bryskin, I., Zinin, A., and R. Coltun, "The OSPF Opaque LSA Option", RFC 5250, DOI 10.17487/RFC5250, July 2008, <<https://www.rfc-editor.org/info/rfc5250>>.
- [RFC7356] Ginsberg, L., Previdi, S., and Y. Yang, "IS-IS Flooding Scope Link State PDUs (LSPs)", RFC 7356, DOI 10.17487/RFC7356, September 2014, <<https://www.rfc-editor.org/info/rfc7356>>.
- [RFC7684] Psenak, P., Gredler, H., Shakir, R., Henderickx, W., Tantsura, J., and A. Lindem, "OSPFv2 Prefix/Link Attribute Advertisement", RFC 7684, DOI 10.17487/RFC7684, November 2015, <<https://www.rfc-editor.org/info/rfc7684>>.
- [RFC7770] Lindem, A., Ed., Shen, N., Vasseur, JP., Aggarwal, R., and S. Shaffer, "Extensions to OSPF for Advertising Optional Router Capabilities", RFC 7770, DOI 10.17487/RFC7770, February 2016, <<https://www.rfc-editor.org/info/rfc7770>>.

## 9.2. Informative References

- [I-D.ietf-rtgwg-segment-routing-ti-lfa]  
Litkowski, S., Bashandy, A., Filsfils, C., Decraene, B., Francois, P., Voyer, D., Clad, F., and P. Camarillo, "Topology Independent Fast Reroute using Segment Routing", draft-ietf-rtgwg-segment-routing-ti-lfa-04 (work in progress), August 2020.
- [I-D.ietf-spring-segment-protection-sr-te-paths]  
Hegde, S., Bowers, C., Litkowski, S., Xu, X., and F. Xu, "Segment Protection for SR-TE Paths", draft-ietf-spring-segment-protection-sr-te-paths-00 (work in progress), September 2020.

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

Filsfils, C., Talaulikar, K., Voyer, D., Bogdanov, A., and P. Mattes, "Segment Routing Policy Architecture", draft-ietf-spring-segment-routing-policy-08 (work in progress), July 2020.

[I-D.sivabalan-pce-binding-label-sid]

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

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

#### Authors' Addresses

Zhibo Hu  
Huawei Technologies  
Huawei Bld., No.156 Beiqing Rd.  
Beijing 100095  
China

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

Huaimo Chen  
Futurewei  
Boston, MA  
USA

Email: [Huaimo.chen@futurewei.com](mailto:Huaimo.chen@futurewei.com)

Junda Yao  
Huawei Technologies  
Huawei Bld., No.156 Beiqing Rd.  
Beijing 100095  
China

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

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

Email: cbowers@juniper.net

Yongqing  
China Telecom  
109, West Zhongshan Road, Tianhe District  
Guangzhou 510000  
China

Email: zhuyq8@chinatelecom.cn

Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: 13 October 2022

Z. Hu  
Huawei Technologies  
H. Chen  
Futurewei  
J. Yao  
Huawei Technologies  
C. Bowers  
Juniper Networks  
Y. Zhu  
China Telecom  
Y. Liu  
China Mobile  
11 April 2022

SR-TE Path Midpoint Restoration  
draft-hu-spring-segment-routing-proxy-forwarding-19

Abstract

Segment Routing Traffic Engineering (SR-TE) supports explicit paths using segment lists containing adjacency-SIDs, node-SIDs and binding-SIDs. The current SR FRR such as TI-LFA provides fast re-route protection for the failure of a node along a SR-TE path by the direct neighbor or say point of local repair (PLR) to the failure. However, once the IGP converges, the SR FRR is no longer sufficient to forward traffic of the path around the failure, since the non-neighbors of the failure will no longer have a route to the failed node. This document describes a mechanism for the restoration of the routes to the failure of a SR-MPLS TE path after the IGP converges. It provides the restoration of the routes to an adjacency segment, a node segment and a binding segment of the path. With the restoration of the routes to the failure, the traffic is continuously sent to the neighbor of the failure after the IGP converges. The neighbor as a PLR fast re-routes the traffic around the failure.

Requirements Language

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

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 13 October 2022.

#### Copyright Notice

Copyright (c) 2022 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 Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

#### Table of Contents

1. Introduction . . . . .	3
1.1. Terminology . . . . .	3
2. Proxy Forwarding . . . . .	4
3. Protocol Extensions/Re-uses for Proxy Forwarding . . . . .	4
3.1. Advertising Binding Segment . . . . .	4
3.2. Advertising Proxy Forwarding . . . . .	5
4. Proxy Forwarding Example . . . . .	6
4.1. Advertising Proxy Forwarding . . . . .	8
4.2. Building Proxy Forwarding Table . . . . .	8
4.3. Proxy Forwarding for Binding Segment . . . . .	9
5. Security Considerations . . . . .	10
6. Acknowledgements . . . . .	10
7. References . . . . .	10
7.1. Normative References . . . . .	10
7.2. Informative References . . . . .	11
Appendix A. Proxy Forwarding for Adjacency and Node Segment . . . . .	11
A.1. Next Segment is an Adjacency Segment . . . . .	11
A.2. Next Segment is a Node Segment . . . . .	12
Authors' Addresses . . . . .	13



## 1. Introduction

Segment Routing Traffic Engineering (SR-TE) is a technology that implements traffic engineering using a segment list. SR-TE supports the creation of explicit paths using adjacency-SIDs, node-SIDs, anycast-SIDs, and binding-SIDs. A node-SID in the segment list defining an SR-TE path indicates a loose hop that the SR-TE path should pass through. When the node fails, the network may no longer be able to properly forward traffic on that SR-TE path.

[I-D.ietf-rtgwg-segment-routing-ti-lfa] describes an SR FRR mechanism that provides fast re-route protection for the failure of a node on a SR-TE path by the direct neighbor or say point of local repair (PLR) to the failure. However, once the IGP converges, the SR FRR is no longer sufficient to forward traffic of the path around the failure, since the non-neighbors of the failure will no longer have a route to the failed node and drop the traffic.

To solve this problem,

[I-D.ietf-spring-segment-protection-sr-te-paths] proposes that a hold timer should be configured on every router in a network. After the IGP converges on the event of a node failure, if the node-SID of the failed node becomes unreachable, the forwarding changes should not be communicated to the forwarding planes on all configured routers (including PLRs for the failed node) until the hold timer expires. This solution may not work for some cases such as some of nodes in the network not supporting this solution.

This document describes a proxy forwarding mechanism for the restoration of the routes to the failure of a SR-MPLS TE path after the IGP converges. It provides the restoration of the routes to an adjacency segment, a node segment and a binding segment on a failed node along the path. With the restoration of the routes to the failure, the traffic for the SR-MPLS TE path is continuously sent to the neighbor of the failure after the IGP converges. The neighbor as a PLR fast re-routes the traffic around the failure.

### 1.1. Terminology

SR: Segment Routing.

PLR: Point of Local Repair.

LSP: Link State Protocol Data Unit (PDU) in IS-IS.

LSA: Link State Advertisement in OSPF.

LS: Link State, which is LSP or LSA.

## 2. Proxy Forwarding

In the proxy forwarding mechanism, each neighbor of a possible failed node advertises its SR proxy forwarding capability in its network domain when it has the capability. This capability indicates that the neighbor (the Proxy Forwarder) will forward traffic on behalf of the failed node. A router receiving the SR Proxy Forwarding capability from the neighbors of a failed node will send traffic using the node-SID of the failed node to the nearest Proxy Forwarder after the IGP converges on the event of the failure.

Once the affected traffic reaches a Proxy Forwarder, it sends the traffic on the post-failure shortest path to the node immediately following the failed node in the segment list.

For a binding segment of a possible failed node, the node advertises the information about the binding segment, including the binding SID and the list of SIDs/segments associated with the binding SID, to its direct neighbors only. Note that the information is not advertised in the network domain.

After the node fails and the IGP converges on the failure, the traffic with the binding SID of the failed node will reach its neighbor having SR Proxy Forwarding capability. Once receiving the traffic, the neighbor swaps the binding SID with the list of SIDs/segments associated with the binding SID and sends the traffic along the post-failure shortest path to the first node in the segment list.

## 3. Protocol Extensions/Re-uses for Proxy Forwarding

This section describes the semantic of protocol extensions/re-uses for advertising the information about each binding segment (including its binding SID and the list of SIDs/segments associated with the binding SID) of a node to its direct neighbors and the SR proxy forwarding capability of a node in a network domain.

### 3.1. Advertising Binding Segment

For a binding segment (or binding for short) on a node A, which consists of a binding SID and a list of SIDs/segments, node A advertises an LS containing the binding (i.e., the binding SID and the list of the SIDs/segments) in a binding segment TLV. The LS is advertised only to each of the node A's neighboring nodes. For OSPFv2, the LS is a opaque LSA of LS type 9 (i.e., a link local scope LSA). For IS-IS, the TLV is advertised in Circuit Scoped Link State PDUs (CS-LSP) [RFC7356].

Alternatively, when a protocol (such as PCE or BGP running on a controller) supports sending a binding on a node A to A, this protocol may be extended to send the binding with node A to A's neighbors if the controller knows the neighbors and there are protocol (PCE or BGP) sessions between the controller and the neighbors.

Note: how to send bindings of node A to A's neighbors via which protocol is out of the scope of this document.

### 3.2. Advertising Proxy Forwarding

When a node P is able to do SR proxy forwarding for its neighboring nodes for protecting the failures of these nodes, P advertises its SR proxy forwarding capability for these nodes. The mirror SID [RFC8402] for a node N (Neighbor of P) advertised by P using IS-IS extensions [RFC8667] indicates the capability of P for N.

For a node X in the network, it learns the prefix/node SID of node N, which is originated and advertised by node N. It creates a proxy prefix/node SID of node N for node P if node P is capable of doing SR proxy forwarding for node N. The proxy prefix/node SID of node N for node P is a copy of the prefix/node SID of node N originated by node N, but stored under (or say, associated with) node P. The route to the proxy prefix/node SID is through proxy forwarding capable nodes.

In normal operations, node X prefers to use the prefix/node SID of node N. When node N fails, node X prefers to use the proxy prefix/node SID of node N. Thus node X will forward the traffic targeting to the prefix/node SID of node N to node P when node N fails, and node P will do a SR proxy forwarding for node N and forward the traffic towards its final destination without going through node N.

Note that the behaviors of normal IP forwarding and routing convergences in a network are not changed at all by the SR proxy forwarding. For example, the next hop used by BGP is an IP address (or prefix). The IGP and BGP converge in normal ways for changes in the network. The packet with its IP destination to this next hop is forwarded according to the IP forwarding table (FIB) derived from IGP and BGP routes.

Similar to IS-IS [RFC8667], OSPF should be extended for advertising mirror SID to indicate the capability. Note that OSPF extensions is out of the scope of this document.

#### 4. Proxy Forwarding Example

This section illustrates the proxy forwarding for a binding SID through an example. The proxy forwarding for a node SID and an adjacency SID can refer to [I-D.ietf-spring-segment-protection-sr-te-paths] or Appendix. Figure 1 is an example network topology used to illustrate the proxy forwarding mechanism for a binding SID. Each node N has SRGB = [N000-N999]. RT1 is an ingress node of SR domain. RT3 is a failure node. RT2 is a Point of Local Repair (PLR) node, i.e., a proxy forwarding node. Label Stack 1 uses a node-SID and a binding SID. The Binding-SID with label=100 at RT3 represents the ECMP-aware path RT3->RT4->RT5. So Label Stack 1, which consists of the node-SID for RT3 following by Binding-SID=100, represents the ECMP-aware path RT1->RT3->RT4->RT5.

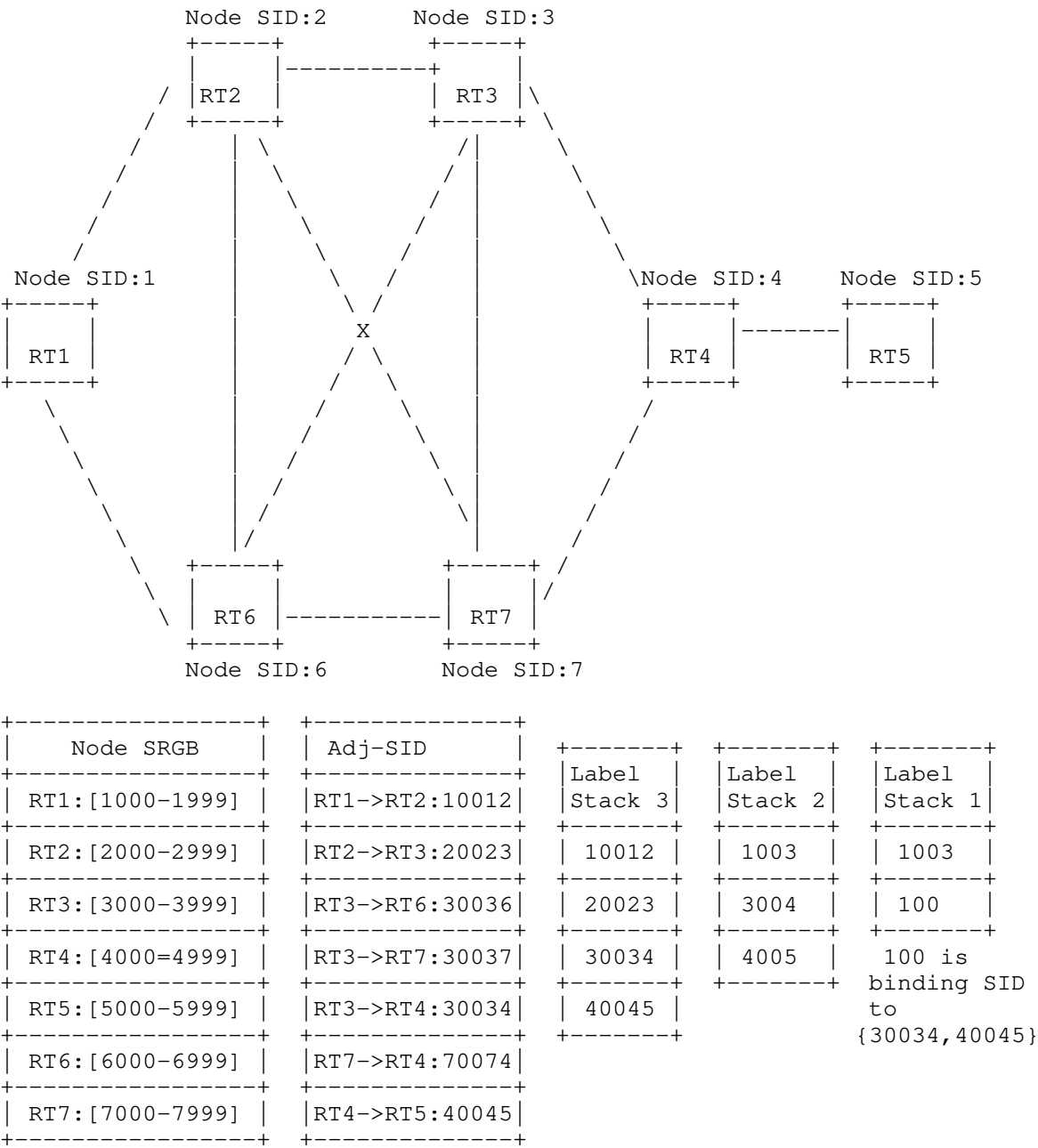


Figure 1: Topology of SR-TE Path

#### 4.1. Advertising Proxy Forwarding

If the Point of Local Repair (PLR), for example, RT2, has the capability to do SR proxy forwarding for its neighboring nodes such as RT3, it must advertise this capability. When RT3 fails, RT2 needs to maintain its SR proxy forwarding capability for a period of time. When the proxy forwarding table corresponding to the fault node is deleted, the capability is withdrawn. The nodes in the network (for example, RT1) learn the prefix/node SID advertised by RT3 and the proxy forwarding capability for RT3 advertised by RT2. When RT3 is normal, the nodes prefer prefix/node SID. When the RT3 fails, the proxy prefix/node SIDs of RT3 for RT2 is preferred.

For binding-SID 100, which is associated with segment list {30034, 40045}, RT3 advertises the binding (i.e., 100 bond to {30034, 40045}) to its neighbors RT2, RT4 and RT7. RT2 as PLR uses the binding to build an entry for proxy forwarding for binding-SID 100 in its Proxy Forwarding Table for RT3. The entry is used when RT3 fails.

#### 4.2. Building Proxy Forwarding Table

A SR proxy node P needs to build an independent proxy forwarding table for each neighbor N. The proxy forwarding table for node N contains the following information:

- 1: Node N's SRGB range and the difference between the SRGB start value of node P and that of node N;
- 2: Every adjacency-SID of N and Node-SID of the node pointed to by node N's adjacency-SID.
- 3: Every binding-SID of N and the label stack associated with the binding-SID.

Node P (PLR) uses a proxy forwarding table based on the next segment to find a node N as a backup forwarding entry to the adjacency-SID and Node-SID of node N. When node N fails, the proxy forwarding table needs to be maintained for a period of time, which is recommended for 30 minutes.

Node RT3 in Figure 1 is node N, and node RT2 is node P (PLR). RT2 builds the proxy forwarding table for RT3. RT2 calculates the proxy forwarding table for RT3, as shown in Figure 2.

In-label	SRGBDiffValue	Next Label	Action	Map Label
2003	-1000	30034	Fwd to RT4	2004
		30036	Fwd to RT6	2006
		30037	Fwd to RT7	2007
		100	Swap to { 30034, 40045 }	

Figure 2: RT2's Proxy Forwarding Table for RT3

#### 4.3. Proxy Forwarding for Binding Segment

This Section shows through example how a proxy node uses the SR proxy forwarding mechanism to forward traffic to the destination node when a node fails and the next segment of label stack is a binding-SID.

As shown in Figure 1, Label Stack 1 {1003, 100} represents SR-TE loose path RT1->RT3->RT4->RT5, where 100 is a Binding-SID, which represents segment list {30034, 40045}.

When the node RT3 fails, the proxy forwarding SID implied or advertised by the RT2 is preferred to forward the traffic of the RT1 to the PLR node RT2. Node RT2 acts as a PLR node and uses Binding-SID to query the proxy forwarding table locally built for RT3. The path returned is the label forwarding path to RT3's next hop node (RT4), which bypasses RT3. The specific steps are as follows:

- a. RT1 swaps label 1003 to out-label 2003 to RT3.
- b. RT2 receives the label forwarding packet whose top label of label stack is 2003, and searches for the local Routing Table, the behavior found is to lookup Proxy Forwarding table due to RT3 failure.
- c. RT2 uses Binding-SID:100 (label 2003 has pop) as the in-label to lookup the Next Label record of the Proxy Forwarding Table, the behavior found is to swap to Segment list {30034, 40045}.
- d. RT2 swaps Binding-SID:100 to Segment list {30034, 40045}, and uses the 30034 to lookup the Next Label record of the Proxy Forwarding table again. The behavior found is to forward the packet to RT4.
- e. RT2 queries the Routing Table to RT4, using primary or backup path to RT4. The next hop is RT7.

f. RT2 forwards packets to RT7. RT7 queries the local routing table to forward the packet to RT4.

## 5. Security Considerations

The extensions to OSPF and IS-IS described in this document result in two types of behaviors in data plane when a node in a network fails. One is that for a node, which is a upstream (except for the direct upstream) node of the failed node along a SR-TE path, it continues to send the traffic to the failed node along the SR-TE path for an extended period of time. The other is that for a node, which is the direct upstream node of the failed node, it fast re-routes the traffic around the failed node to the direct downstream node of the failed node along the SR-TE path. These behaviors are internal to a network and should not cause extra security issues.

## 6. Acknowledgements

The authors would like to thank Peter Psenak, Acee Lindem, Les Ginsberg, Bruno Decraene and Jeff Tantsura for their comments to this work.

## 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, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC7356] Ginsberg, L., Previdi, S., and Y. Yang, "IS-IS Flooding Scope Link State PDUs (LSPs)", RFC 7356, DOI 10.17487/RFC7356, September 2014, <<https://www.rfc-editor.org/info/rfc7356>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <<https://www.rfc-editor.org/info/rfc8402>>.



[RFC8667] Previdi, S., Ed., Ginsberg, L., Ed., Filsfils, C., Bashandy, A., Gredler, H., and B. Decraene, "IS-IS Extensions for Segment Routing", RFC 8667, DOI 10.17487/RFC8667, December 2019, <<https://www.rfc-editor.org/info/rfc8667>>.

## 7.2. Informative References

[I-D.ietf-rtgwg-segment-routing-ti-lfa]  
Litkowski, S., Bashandy, A., Filsfils, C., Francois, P., Decraene, B., and D. Voyer, "Topology Independent Fast Reroute using Segment Routing", Work in Progress, Internet-Draft, draft-ietf-rtgwg-segment-routing-ti-lfa-08, 21 January 2022, <<https://www.ietf.org/archive/id/draft-ietf-rtgwg-segment-routing-ti-lfa-08.txt>>.

[I-D.ietf-spring-segment-protection-sr-te-paths]  
Hegde, S., Bowers, C., Litkowski, S., Xu, X., and F. Xu, "Segment Protection for SR-TE Paths", Work in Progress, Internet-Draft, draft-ietf-spring-segment-protection-sr-te-paths-03, 7 March 2022, <<https://www.ietf.org/archive/id/draft-ietf-spring-segment-protection-sr-te-paths-03.txt>>.

[I-D.ietf-spring-segment-routing-policy]  
Filsfils, C., Talaulikar, K., Voyer, D., Bogdanov, A., and P. Mattes, "Segment Routing Policy Architecture", Work in Progress, Internet-Draft, draft-ietf-spring-segment-routing-policy-22, 22 March 2022, <<https://www.ietf.org/archive/id/draft-ietf-spring-segment-routing-policy-22.txt>>.

## Appendix A. Proxy Forwarding for Adjacency and Node Segment

This Section shows through example how a proxy node forward traffic to the destination node when a node fails and the next segment of label stack is an adjacency-SID or node-SID.

### A.1. Next Segment is an Adjacency Segment

As shown in Figure 1, Label Stack 3 {10012, 20023, 30034, 40045} uses only adjacency-SIDs and represents the SR-TE strict explicit path RT1->RT2->RT3->RT4->RT5. When RT3 fails, node RT2 acts as a PLR, and uses next adjacency-SID (30034) of the label stack to lookup the proxy forwarding table built by RT2 locally for RT3. The path returned is the label forwarding path to RT3's next hop node RT4, which bypasses RT3. The specific steps are as follows:

- a. RT1 pops top adjacency-SID 10012, and forwards the packet to RT2;
- b. RT2 uses the label 20023 to identify the next hop node RT3, which has failed. RT2 pops label 20023 and queries the Proxy Forwarding Table corresponding to RT3 with label 30034. The query result is 2004. RT2 uses 2004 as the incoming label to query the label forwarding table. The next hop is RT7, and the incoming label is changed to 7004.
- c. So the packet leaves RT2 out the interface to RT7 with label stack {7004, 40045}. RT7 forwards it to RT4, where the original path is rejoined.
- d. RT2 forwards packets to RT7. RT7 queries the local routing table to forward the packet to RT4.

#### A.2. Next Segment is a Node Segment

As shown in Figure 1, Label Stack 2 {1003, 3004, 4005} uses only node-SIDs and represents the ECMP-aware path RT1->RT3->RT4->RT5, where 1003 is the node SID of RT3.

When the node RT3 fails, the proxy forwarding TLV advertised by the RT2 is preferred to direct the traffic of the RT1 to the PLR node RT2. Node RT2 acts as a PLR node and queries the proxy forwarding table locally built for RT3. The path returned is the label forwarding path to RT3's next hop node RT4, which bypasses RT3. The specific steps are as follows:

- a. RT1 swaps label 1003 to out-label 2003 to RT3.
- b. RT2 receives the label forwarding packet whose top label of label stack is 2003, and searches for the local Routing Table, the behavior found is to lookup Proxy Forwarding table due to RT3 failure, RT2 pops label 2003.
- c. RT2 uses 3004 as the in-label to lookup Proxy Forwarding table, The value of Map Label calculated based on SRGBDiffValue is 2004. and the query result is forwarding the packet to RT4.
- d. Then RT2 queries the Routing Table to RT4, using the primary or backup path to RT4. The next hop is RT7.
- e. RT2 forwards the packet to RT7. RT7 queries the local routing table to forward the packet to RT4.
- f. After RT1 convergences, node SID 1003 is preferred to the proxy SID implied/advertised by RT2.

## Authors' Addresses

Zhibo Hu  
Huawei Technologies  
Huawei Bld., No.156 Beiqing Rd.  
Beijing  
100095  
China  
Email: huzhibo@huawei.com

Huaimo Chen  
Futurewei  
Boston, MA,  
United States of America  
Email: Huaimo.chen@futurewei.com

Junda Yao  
Huawei Technologies  
Huawei Bld., No.156 Beiqing Rd.  
Beijing  
100095  
China  
Email: yaojunda@huawei.com

Chris Bowers  
Juniper Networks  
1194 N. Mathilda Ave.  
Sunnyvale, CA, 94089  
United States of America  
Email: cbowers@juniper.net

Yongqing  
China Telecom  
109, West Zhongshan Road, Tianhe District  
Guangzhou  
510000  
China  
Email: zhuyq8@chinatelecom.cn

Yisong  
China Mobile  
510000  
China

Email: [liuyisong@chinamobile.com](mailto:liuyisong@chinamobile.com)

Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: January 29, 2021

Z. Hu  
Huawei  
H. Chen  
Futurewei  
H. Chen  
China Telecom  
P. Wu  
Huawei  
M. Toy  
Verizon  
C. Cao  
T. He  
China Unicom  
L. Liu  
Fujitsu  
X. Liu  
Volta Networks  
July 28, 2020

SRv6 Path Egress Protection  
draft-ietf-rtgwg-srv6-egress-protection-01

Abstract

This document describes protocol extensions for protecting the egress node of a Segment Routing for IPv6 (SRv6) path or tunnel.

Requirements Language

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

Status of This Memo

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

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

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

This Internet-Draft will expire on January 29, 2021.

#### Copyright Notice

Copyright (c) 2020 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. Terminologies . . . . .	3
3. SR Path Egress Protection . . . . .	4
3.1. Mechanism . . . . .	4
3.2. Example . . . . .	6
4. Extensions to IGP for Egress Protection . . . . .	8
4.1. Extensions to IS-IS . . . . .	8
4.2. Extensions to OSPF . . . . .	10
5. Security Considerations . . . . .	12
6. IANA Considerations . . . . .	12
6.1. IS-IS . . . . .	12
6.2. OSPFv3 . . . . .	12
7. Acknowledgements . . . . .	13
8. References . . . . .	13
8.1. Normative References . . . . .	13
8.2. Informative References . . . . .	14
Authors' Addresses . . . . .	15

#### 1. Introduction

The fast protection of a transit node of a Segment Routing (SR) path or tunnel is described in [I-D.ietf-rtgwg-segment-routing-ti-lfa] and [I-D.hu-spring-segment-routing-proxy-forwarding]. [RFC8400] specifies the fast protection of egress node(s) of an MPLS TE LSP tunnel including P2P TE LSP tunnel and P2MP TE LSP tunnel in details. However, these documents do not discuss the fast protection of the egress node of a Segment Routing for IPv6 (SRv6) path or tunnel.

This document fills that void and presents protocol extensions for the fast protection of the egress node of an SRv6 path or tunnel. Egress node and egress, fast protection and protection as well as SRv6 path and SRv6 tunnel will be used exchangeably below.

There are a number of topics related to the egress protection, which include the detection of egress node failure, the relation between egress protection and global repair, and so on. These are discussed in details in [RFC8679].

## 2. Terminologies

The following terminologies are used in this document.

SR: Segment Routing

SRv6: SR for IPv6

SRH: Segment Routing Header

SID: Segment Identifier

LSA: Link State Advertisement in OSPF

LSP: Label Switched Path in MPLS or Link State Protocol PDU in IS-IS

PDU: Protocol Data Unit

LS: Link State, which is LSA in OSPF or LSP in IS-IS

TE: Traffic Engineering

SA: Source Address

DA: Destination Address

P2MP: Point-to-MultiPoint

P2P: Point-to-Point

CE: Customer Edge

PE: Provider Edge

LFA: Loop-Free Alternate

TI-LFA: Topology Independent LFA

BFD: Bidirectional Forwarding Detection

VPN: Virtual Private Network

L3VPN: Layer 3 VPN

VRF: Virtual Routing and Forwarding

FIB: Forwarding Information Base

PLR: Point of Local Repair

BGP: Border Gateway Protocol

IGP: Interior Gateway Protocol

OSPF: Open Shortest Path First

IS-IS: Intermediate System to Intermediate System

### 3. SR Path Egress Protection

This section describes the mechanism of SR path egress protection and illustrates it through an example.

### 3.1. Mechanism

Figure 1 is used to explain the mechanism of SR path egress node protection.

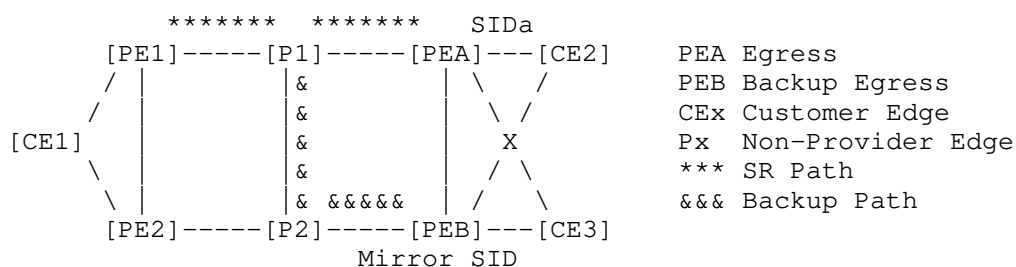


Figure 1: PEB Protects Egress PEA of SR Path

Where node PEA is the egress of the SR path from PE1 to PEA, and has SIDA which is the active segment in the packet from the SR path at PEA. Node PEB is the backup egress (or say protector) to provide the protection for egress (or say primary egress) PEA. Node P1 is the direct previous hop of egress PEA and acts as PLR to support the protection for PEA.



When PEB is selected as a backup egress to protect the egress PEA, a Mirror SID (refer to Section 5.1 of [RFC8402]) is configured on PEB to protect PEA. PEB advertises this information through IGP, which includes the Mirror SID and the egress PEA. The information is represented by <PEB, PEA, Mirror SID>, which indicates that PEB protects PEA with Mirror SID.

After PEA receives the information <PEB, PEA, Mirror SID>, it may send the forwarding behavior of the SIDA at PEA to PEB with the Mirror SID using some protocols such as BGP if PEB can not obtain this behavior from other approaches if PEB wants to protect SIDA of PEA. How to send the forwarding behavior of the SIDA to PEB is out scope of this document.

When PEB gets the forwarding behavior of the SIDA of PEA from PEA or other means, it adds a forwarding entry for the SIDA according to the behavior into the forwarding table for node PEA. This table is identified by the Mirror SID, which indicates node PEA's context. Using the forwarding entry for SIDA in this table, a packet with SIDA will be transmitted by PEB to the same destination as it is transmitted by PEA. For example, assume that the packet with SIDA is transmitted by PEA to CE2 through the forwarding behavior of the SIDA in PEA. The packet will be transmitted by PEB to the same CE2 through looking up the table identified by the Mirror SID.

After P1 as PLR receives the information <PEB, PEA, Mirror SID> and knows that PEB wants to protect SIDA of PEA, it computes a shortest path to PEB. A Repair List RL is obtained based on the path. It is one of the followings:

- o RL = <Mirror SID> if the path does not go through PEA; or
- o RL = <S1, ..., Sn, Mirror SID> if the path goes through PEA, where <S1, ..., Sn> is the TI-LFA Repair List to PEB computed by P1.

When PEA fails, P1 as PLR sends the packet with SIDA carried by the SR path to PEB, but encapsulates the packet before sending it by executing H.Encaps with the Repair List RL and a Source Address T.

Suppose that the packet received by P1 is represented by Pkt = (S, SIDA)Pkt0, where SA = S and DA = SIDA, and Pkt0 is the rest of the packet.

The execution of H.Encaps pushes an IPv6 header to Pkt and sets some fields in the outer and inner IPv6 header to produce an encapsulated packet Pkt'. Pkt' will be one of the followings:

- o Pkt' = (T, Mirror SID) (S, SIDA)Pkt0 if RL = <Mirror SID>; or

- o  $\text{Pkt}' = (T, S1)(\text{Mirror SID}, Sn, \dots, S1; SL=n) (S, SIDA)\text{Pkt0}$  if  $RL = \langle S1, \dots, Sn, \text{Mirror SID} \rangle$ .

When PEB receives the re-routed packet, which is (T, Mirror SID) (S, SIDA)Pkt0, it decapsulates the packet and forwards the decapsulated packet using the forwarding table identified by Mirror SID through executing End.DT6.

It obtains the Mirror SID in the outer IPv6 header of the packet, removes this outer IPv6 header with all its extension headers, and then processes the inner IPv6 packet (i.e., (S, SIDA)Pkt0, the packet without the outer IPv6 header). PEB finds the forwarding table for node PEA using the Mirror SID as the context ID, and submits the packet to this forwarding table lookup and transmission to the same destination as PEA does.

### 3.2. Example

Figure 2 shows an example of protecting egress PE3 of a SR path, which is from ingress PE1 to egress PE3.

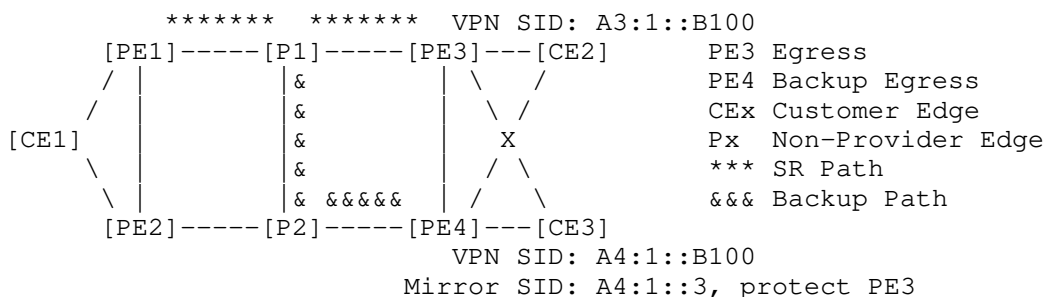


Figure 2: PE4 Protects Egress PE3 of SR Path

Where node P1's pre-computed backup path for PE3 is from P1 to PE4 via P2. In normal operations, after receiving a packet with destination PE3, P1 forwards the packet to PE3 according to its FIB. When PE3 receives the packet, it sends the packet to CE2.

When PE3 fails, P1 as PLR detects the failure through using a failure detection mechanism such as BFD and forwards the packet to PE4 via the backup path. When PE4 receives the packet, it sends the packet to the same CE2.

In Figure 2, Both CE2 and CE3 are dual home to PE3 and PE4. PE3 has a VPN SID A3:1::B100. PE4 has a VPN SID A4:1::B100. A Mirror SID A4:1::3 is configured on PE4 for protecting PE3.

After the configuration, PE4 advertises this information through an IGP LS (i.e., LSA in OSPF or LSP in IS-IS), which includes PE4's ID, PE3's ID and Mirror SID A4:1::3. Every node in the SR domain will receive this IGP LS, which indicates that PE4 wants to protect PE3 with Mirror SID A4:1::3.

When PE4 (e.g., BGP on PE4) receives a prefix whose VPN SID belongs to PE3 that is protected by PE4 through Mirror SID A4:1::3, it finds PE4's VPN SID corresponding to PE3's VPN SID. For example, local PE4 has Prefix 1.1.1.1 with VPN SID A4:1::B100, when PE4 receives prefix 1.1.1.1 with remote PE3's VPN SID A3:1::B100, it knows that they are for the same VPN.

The forwarding behaviors for these two VPN SIDs are the same from function's point of view. If the behavior for PE3's VPN SID in PE3 forwards the packet with it to CE2, then the behavior for PE4's VPN SID in PE4 forwards the packet to the same CE2; and vice versa. PE4 creates a forwarding entry for PE3's VPN SID A3:1::B100 in the table (or FIB) identified by Mirror SID A4:1::3 according to the forwarding behavior for PE4's VPN SID A4:1::B100.

Node P1's pre-computed backup path for destination PE3 is from P1 to PE4 having mirror SID A4:1::3. When P1 receives a packet destined to PE3's VPN SID A3:1::B100, in normal operations, it forwards the packet with source A1:1:: and destination PE3's VPN SID A3:1::B100 according to the FIB using the destination PE3's VPN SID A3:1::B100.

When PE3 fails, P1 as PLR sends the packet to PE4 via the backup path pre-computed. P1 encapsulates the packet using H.Encaps before sending it to PE4.

Suppose that the packet received by P1 is represented by  $\text{Pkt} = (\text{SA} = \text{A1:1::}, \text{DA} = \text{A3:1::B100})\text{Pkt0}$ , where  $\text{DA} = \text{A3:1::B100}$  is PE3's VPN SID, and Pkt0 is the rest of the packet. The encapsulated packet Pkt' will be one of the followings:

- o  $\text{Pkt}' = (\text{T}, \text{Mirror SID A4:1::3}) (\text{A1:1::}, \text{A3:1::B100})\text{Pkt0}$  if backup path not via PE3; or (otherwise)
- o  $\text{Pkt}' = (\text{T}, \text{S1}) (\text{Mirror SID A4:1::3}, \text{Sn}, \dots, \text{S1}; \text{SL=n}) (\text{A1:1::}, \text{A3:1::B100})\text{Pkt0}$ .

where T is a Source Address,  $\langle \text{S1}, \dots, \text{Sn} \rangle$  is the TI-LFA Repair List to PE4 computed by P1 when the backup path to PE4 goes through PE3.

When PE4 receives the re-routed packet, it decapsulates the packet and forwards the decapsulated packet by End.DT6. The packet received

by PE4 is (T, Mirror SID A4:1::3) (A1:1::, PE3's VPN SID A3:1::B100)Pkt0.

PE4 obtains Mirror SID A4:1::3 in the outer IPv6 header of the packet, removes this outer IPv6 header, and then processes the inner IPv6 packet (A1:1::, A3:1::B100)Pkt0. It finds the forwarding table for PE3 using Mirror SID A4:1::3 as the context ID, gets the forwarding entry for PE3's VPN SID A3:1::B100 from the table, and forwards the packet to CE2 using the entry.

#### 4. Extensions to IGP for Egress Protection

This section describes extensions to IS-IS and OSPF for advertising the information about SRv6 path egress protection.

##### 4.1. Extensions to IS-IS

A new sub-TLV, called IS-IS SRv6 Mirror SID sub-TLV, is defined. It is used in the SRv6 Locator TLV defined in [I-D.ietf-lsr-isis-srv6-extensions] to advertise SRv6 Mirror SID and the ID of the node to be protected. The SRv6 Mirror SID inherit the topology/algorithm from the parent locator. The format of the sub-TLV is illustrated below.

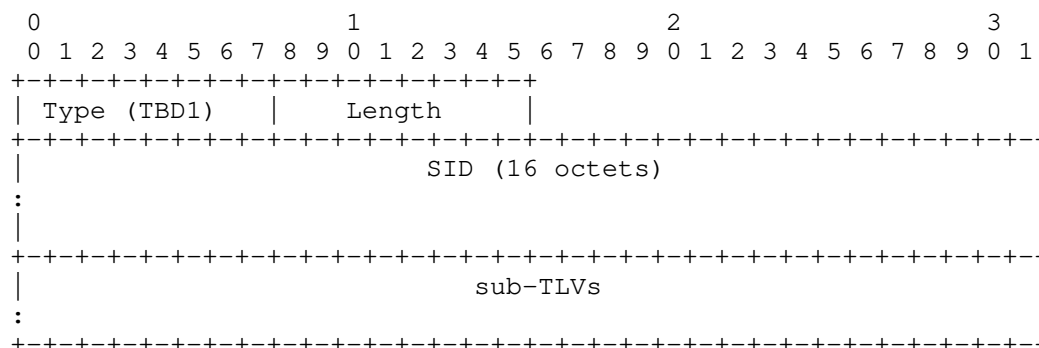


Figure 3: IS-IS SRv6 Mirror SID sub-TLV

Type: TBD1 (suggested value 8) is to be assigned by IANA.

Length: variable.

SID: 16 octets. This field contains the SRv6 Mirror SID to be advertised.

Two sub-TLVs are defined. One is the protected node sub-TLV, and the other is the protected SIDs sub-TLV.

A protected node sub-TLV is used to carry the ID of the node to be protected by the SRv6 Mirror SID. It has the following format.

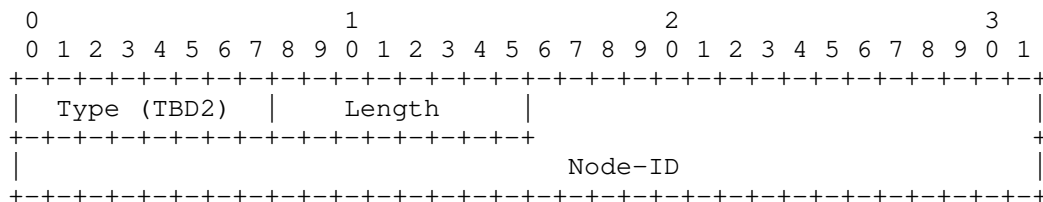


Figure 4: IS-IS Protected Node sub-TLV

Type: TBD2 (suggested value 1) is to be assigned by IANA.

Length: 1 octet. Its value is 6.

Node-ID: 6 octets. It contains a 6-octet ISO Node-ID (ISO system-ID).

A protected SIDs sub-TLV is used to carry the SIDs to be protected by the SRv6 Mirror SID. It has the following format.

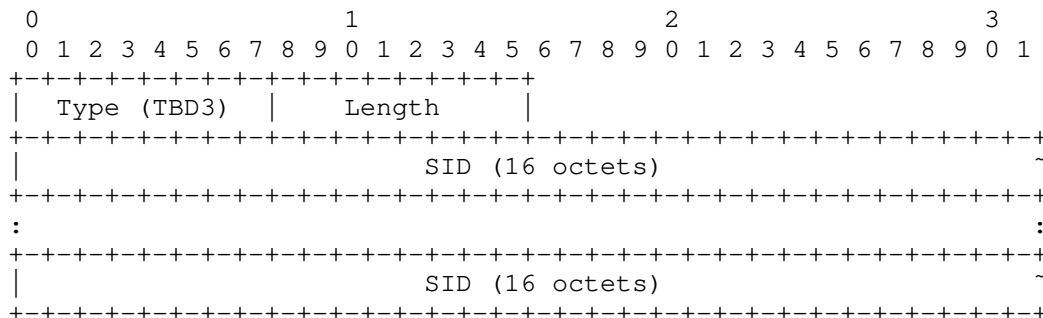


Figure 5: IS-IS Protected SIDs sub-TLV

Type: TBD3 (suggested value 2) is to be assigned by IANA.

Length: variable.

SID: 16 octets. This field encodes an SRv6 SID to be protected.

When node B advertises that B wants to protect node A with a Mirror SID through an LSP, the LSP contains an IS-IS SRv6 Mirror SID sub-TLV, which includes the Mirror SID and the node A's ID in an IS-IS Protected Node sub-TLV. If B wants to protect just a specific set of SIDs of node A, the Mirror SID sub-TLV includes these SIDs in an IS-

IS Protected SIDs sub-TLV; otherwise (i.e., B wants to protect all the SIDs of A) it does not contain any IS-IS Protected SIDs sub-TLV.

Note: the IS-IS extensions for SR MPLS is described in [RFC8667]. It says that the SID/Label Binding TLV may also be used to advertise a Mirror SID. For B to protect egress A of SR MPLS path, B may also use this TLV to advertise the node A's ID and a specific set of SIDs of A to be protected. An IS-IS SR MPLS Mirror SID sub-TLV may be obtained from an IS-IS SRv6 Mirror SID sub-TLV by replacing each SID field in the latter with an SID/Label sub-TLV. B may advertise a SID/Label Binding TLV including this IS-IS SR MPLS Mirror SID sub-TLV.

Alternatively, an IS-IS SR MPLS Mirror Supplement sub-TLV is defined from an IS-IS SRv6 Mirror SID sub-TLV by removing the SID field in the top level and replacing each other SID field with an SID/Label sub-TLV. That is that an IS-IS SR MPLS Mirror Supplement sub-TLV just contains a Protected Node sub-TLV and a Protected SIDs sub-TLV, which includes SID/Label sub-TLVs. When the SID/Label Binding TLV contains an SID/Label sub-TLV for the Mirror SID, it includes an IS-IS SR MPLS Mirror Supplement sub-TLV.

#### 4.2. Extensions to OSPF

Similarly, a new sub-TLV, called OSPF Mirror SID sub-TLV, is defined. It is used to advertise SRv6 Mirror SID and the ID of the node to be protected. Its format is illustrated below.

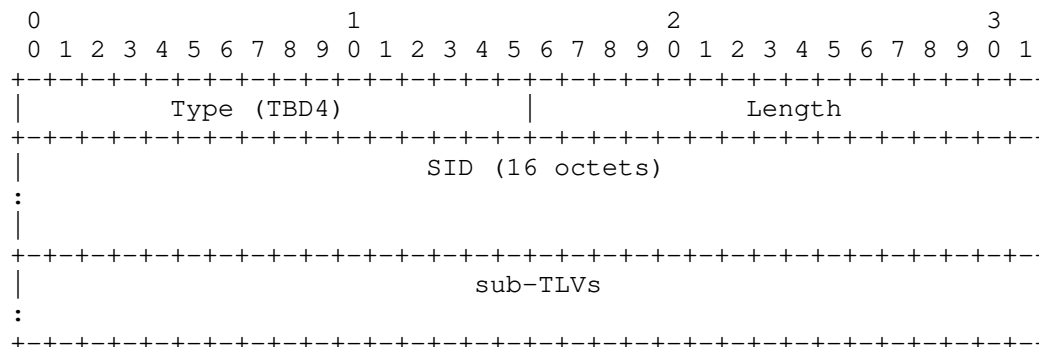


Figure 6: OSPF SRv6 Mirror SID sub-TLV

Type: TBD4 (suggested value 8) is to be assigned by IANA.

Length: variable.

SID: 16 octets. This field contains the SRv6 Mirror SID to be advertised.

Two sub-TLVs are defined. One is the protected node sub-TLV, and the other is the protected SIDs sub-TLV.

A protected node sub-TLV is used to carry the ID of the node to be protected by the SRv6 Mirror SID. It has the following format.

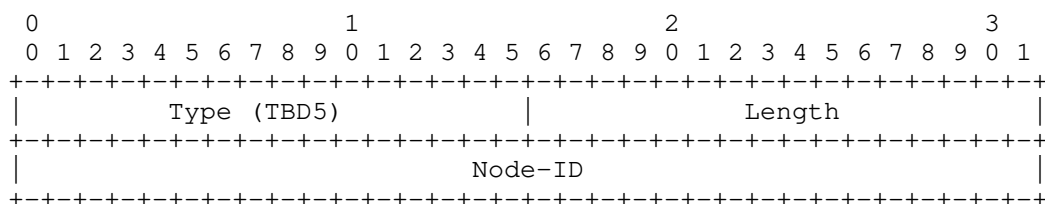


Figure 7: OSPF Protected Node sub-TLV

Type: TBD5 (suggested value 1) is to be assigned by IANA.

Length: 2 octets. Its value is 4.

Node-ID: 4 octets. It contains the ID of the OSPF node or router to be protected.

A protected SIDs sub-TLV is used to carry the SIDs to be protected by the SRv6 Mirror SID. It has the following format.

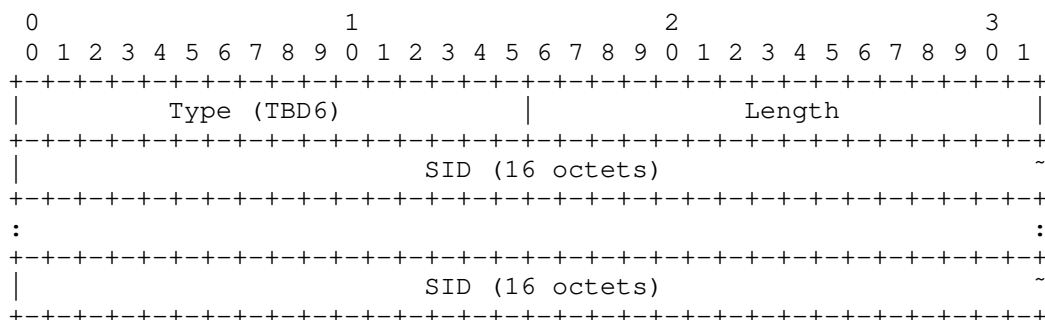


Figure 8: OSPF Protected SIDs sub-TLV

Type: TBD6 (suggested value 2) is to be assigned by IANA.

Length: variable.

SID: 16 octets. This field encodes an SRv6 SID to be protected.

## 5. Security Considerations

The security about the egress protection is described in in details in [RFC8679]. The extensions to OSPF and IS-IS described in this document for SRv6 path egress protection should not cause extra security issues.

## 6. IANA Considerations

### 6.1. IS-IS

Under "Sub-TLVs for TLVs 27, 135, 235, 236 and 237 registry" [I-D.ietf-lsr-isis-srv6-extensions], IANA is requested to add the following new Sub-TLV:

Sub-TLV Type	Sub-TLV Name	Reference
8	SRv6 Mirror SID Sub-TLV	This document

IANA is requested to create and maintain a new registry for sub-sub-TLVs of the SRv6 Mirror SID Sub-TLV. The suggested registry name is

- o Sub-Sub-TLVs for SRv6 Mirror SID Sub-TLV

Initial values for the registry are given below. The future assignments are to be made through IETF Review [RFC5226].

Value	Sub-Sub-TLV Name	Definition
0	Reserved	
1	Protected Node Sub-Sub-TLV	This Document
2	Protected SIDs Sub-Sub-TLV	
3-255	Unassigned	

### 6.2. OSPFv3

Under registry "OSPFv3 Locator LSA Sub-TLVs" [I-D.li-ospf-ospfv3-srv6-extensions], IANA is requested to assign the following new Sub-TLV:

Sub-TLV Type	Sub-TLV Name	Reference
8	SRv6 Mirror SID Sub-TLV	This document



IANA is requested to create and maintain a new registry for sub-sub-TLVs of the SRv6 Mirror SID Sub-TLV. The suggested registry name is

- o Sub-Sub-TLVs for SRv6 Mirror SID Sub-TLV

Initial values for the registry are given below. The future assignments are to be made through IETF Review [RFC5226].

Value	Sub-Sub-TLV Name	Definition
-----	-----	-----
0	Reserved	
1	Protected Node Sub-Sub-TLV	This Document
2	Protected SIDs Sub-Sub-TLV	
3-65535	Unassigned	

## 7. Acknowledgements

The authors would like to thank Peter Psenak, Yimin Shen, Zhenqiang Li, Alexander Vainshtein, Greg Mirsky, Bruno Decraene and Jeff Tantsura for their comments to this work.

## 8. References

### 8.1. Normative References

- [I-D.ietf-lsr-isis-srv6-extensions]  
Psenak, P., Filsfils, C., Bashandy, A., Decraene, B., and Z. Hu, "IS-IS Extension to Support Segment Routing over IPv6 Dataplane", draft-ietf-lsr-isis-srv6-extensions-08 (work in progress), April 2020.
- [I-D.li-ospf-ospfv3-srv6-extensions]  
Li, Z., Hu, Z., Cheng, D., Talaulikar, K., and P. Psenak, "OSPFv3 Extensions for SRv6", draft-li-ospf-ospfv3-srv6-extensions-07 (work in progress), November 2019.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC7356] Ginsberg, L., Previdi, S., and Y. Yang, "IS-IS Flooding Scope Link State PDUs (LSPs)", RFC 7356, DOI 10.17487/RFC7356, September 2014, <<https://www.rfc-editor.org/info/rfc7356>>.

- [RFC7490] Bryant, S., Filts, C., Previdi, S., Shand, M., and N. So, "Remote Loop-Free Alternate (LFA) Fast Reroute (FRR)", RFC 7490, DOI 10.17487/RFC7490, April 2015, <<https://www.rfc-editor.org/info/rfc7490>>.
- [RFC8400] Chen, H., Liu, A., Saad, T., Xu, F., and L. Huang, "Extensions to RSVP-TE for Label Switched Path (LSP) Egress Protection", RFC 8400, DOI 10.17487/RFC8400, June 2018, <<https://www.rfc-editor.org/info/rfc8400>>.
- [RFC8402] Filts, 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>>.
- [RFC8665] Psenak, P., Ed., Previdi, S., Ed., Filts, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPF Extensions for Segment Routing", RFC 8665, DOI 10.17487/RFC8665, December 2019, <<https://www.rfc-editor.org/info/rfc8665>>.
- [RFC8667] Previdi, S., Ed., Ginsberg, L., Ed., Filts, C., Bashandy, A., Gredler, H., and B. Decraene, "IS-IS Extensions for Segment Routing", RFC 8667, DOI 10.17487/RFC8667, December 2019, <<https://www.rfc-editor.org/info/rfc8667>>.
- [RFC8679] Shen, Y., Jeganathan, M., Decraene, B., Gredler, H., Michel, C., and H. Chen, "MPLS Egress Protection Framework", RFC 8679, DOI 10.17487/RFC8679, December 2019, <<https://www.rfc-editor.org/info/rfc8679>>.

## 8.2. Informative References

- [I-D.hegde-spring-node-protection-for-sr-te-paths] Hegde, S., Bowers, C., Litkowski, S., Xu, X., and F. Xu, "Node Protection for SR-TE Paths", draft-hegde-spring-node-protection-for-sr-te-paths-06 (work in progress), July 2020.
- [I-D.hu-spring-segment-routing-proxy-forwarding] Hu, Z., Chen, H., Yao, J., Bowers, C., and Y. Zhu, "SR-TE Path Midpoint Protection", draft-hu-spring-segment-routing-proxy-forwarding-09 (work in progress), July 2020.

- [I-D.ietf-rtgwg-segment-routing-ti-lfa]  
Litkowski, S., Bashandy, A., Filsfils, C., Decraene, B.,  
Francois, P., Voyer, D., Clad, F., and P. Camarillo,  
"Topology Independent Fast Reroute using Segment Routing",  
draft-ietf-rtgwg-segment-routing-ti-lfa-03 (work in  
progress), March 2020.
- [I-D.ietf-spring-segment-routing-policy]  
Filsfils, C., Talaulikar, K., Voyer, D., Bogdanov, A., and  
P. Mattes, "Segment Routing Policy Architecture", draft-  
ietf-spring-segment-routing-policy-08 (work in progress),  
July 2020.
- [I-D.sivabalan-pce-binding-label-sid]  
Sivabalan, S., Filsfils, C., Tantsura, J., Hardwick, J.,  
Previdi, S., and C. Li, "Carrying Binding Label/Segment-ID  
in PCE-based Networks.", draft-sivabalan-pce-binding-  
label-sid-07 (work in progress), July 2019.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an  
IANA Considerations Section in RFCs", RFC 5226,  
DOI 10.17487/RFC5226, May 2008,  
<<https://www.rfc-editor.org/info/rfc5226>>.
- [RFC5462] Andersson, L. and R. Asati, "Multiprotocol Label Switching  
(MPLS) Label Stack Entry: "EXP" Field Renamed to "Traffic  
Class" Field", RFC 5462, DOI 10.17487/RFC5462, February  
2009, <<https://www.rfc-editor.org/info/rfc5462>>.

## Authors' Addresses

Zhibo Hu  
Huawei  
Huawei Bld., No.156 Beiqing Rd.  
Beijing 100095  
China

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

Huaimo Chen  
Futurewei  
Boston, MA  
USA

Email: [Huaimo.chen@futurewei.com](mailto:Huaimo.chen@futurewei.com)

Huanan Chen  
China Telecom  
109, West Zhongshan Road, Tianhe District  
Guangzhou 510000  
China

Email: [chenhuan6@chinatelecom.cn](mailto:chenhuan6@chinatelecom.cn)

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

Email: [baggio.wupeng@huawei.com](mailto:baggio.wupeng@huawei.com)

Mehmet Toy  
Verizon  
USA

Email: [mehmet.toy@verizon.com](mailto:mehmet.toy@verizon.com)

Chang Cao  
China Unicom  
Beijing China

Email: [caoc15@chinaunicom.cn](mailto:caoc15@chinaunicom.cn)

Tao He  
China Unicom  
Beijing China

Email: [het21@chinaunicom.cn](mailto:het21@chinaunicom.cn)

Lei Liu  
Fujitsu  
USA

Email: [liulei.kddi@gmail.com](mailto:liulei.kddi@gmail.com)

Xufeng Liu  
Volta Networks  
McLean, VA  
USA

Email: xufeng.liu.ietf@gmail.com

Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: October 20, 2022

Z. Hu  
Huawei  
H. Chen  
Futurewei  
H. Chen  
China Telecom  
P. Wu  
Huawei  
M. Toy  
Verizon  
C. Cao  
T. He  
China Unicom  
L. Liu  
Fujitsu  
X. Liu  
Volta Networks  
April 18, 2022

SRv6 Path Egress Protection  
draft-ietf-rtgwg-srv6-egress-protection-05

Abstract

This document describes protocol extensions for protecting the egress node of a Segment Routing for IPv6 (SRv6) path or tunnel.

Requirements Language

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

Status of This Memo

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

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

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

This Internet-Draft will expire on October 20, 2022.

## Copyright Notice

Copyright (c) 2022 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. Terminologies . . . . .	3
3. SR Path Egress Protection . . . . .	4
3.1. Mechanism . . . . .	4
3.2. Example . . . . .	6
4. Extensions to IGP for Egress Protection . . . . .	8
4.1. Extensions to IS-IS . . . . .	9
4.2. Extensions to OSPF . . . . .	11
5. Security Considerations . . . . .	13
6. IANA Considerations . . . . .	13
6.1. SRv6 Endpoint Behaviors . . . . .	13
6.2. IS-IS . . . . .	14
6.3. OSPFv3 . . . . .	14
7. Acknowledgements . . . . .	15
8. References . . . . .	15
8.1. Normative References . . . . .	15
8.2. Informative References . . . . .	16
Authors' Addresses . . . . .	17

## 1. Introduction

The fast protection of a transit node of a Segment Routing (SR) path or tunnel is described in [I-D.ietf-rtgwg-segment-routing-ti-lfa] and [I-D.hu-spring-segment-routing-proxy-forwarding]. [RFC8400] specifies the fast protection of egress node(s) of an MPLS TE LSP tunnel including P2P TE LSP tunnel and P2MP TE LSP tunnel in details. However, these documents do not discuss the fast protection of the egress node of a Segment Routing for IPv6 (SRv6) path or tunnel.

This document fills that void and presents protocol extensions for the fast protection of the egress node of an SRv6 path or tunnel. Egress node and egress, fast protection and protection as well as SRv6 path and SRv6 tunnel will be used exchangeably below.

There are a number of topics related to the egress protection, which include the detection of egress node failure, the relation between egress protection and global repair, and so on. These are discussed in details in [RFC8679].

## 2. Terminologies

The following terminologies are used in this document.

SR: Segment Routing

SRv6: SR for IPv6

SRH: Segment Routing Header

SID: Segment Identifier

LSA: Link State Advertisement in OSPF

LSP: Label Switched Path in MPLS or Link State Protocol PDU in IS-IS

PDU: Protocol Data Unit

LS: Link State, which is LSA in OSPF or LSP in IS-IS

TE: Traffic Engineering

SA: Source Address

DA: Destination Address

P2MP: Point-to-MultiPoint

P2P: Point-to-Point

CE: Customer Edge

PE: Provider Edge

LFA: Loop-Free Alternate

TI-LFA: Topology Independent LFA



BFD: Bidirectional Forwarding Detection

VPN: Virtual Private Network

L3VPN: Layer 3 VPN

VRF: Virtual Routing and Forwarding

FIB: Forwarding Information Base

PLR: Point of Local Repair

BGP: Border Gateway Protocol

IGP: Interior Gateway Protocol

OSPF: Open Shortest Path First

IS-IS: Intermediate System to Intermediate System

### 3. SR Path Egress Protection

This section describes the mechanism of SR path egress protection and illustrates it through an example.

#### 3.1. Mechanism

Figure 1 is used to explain the mechanism of SR path egress node protection.

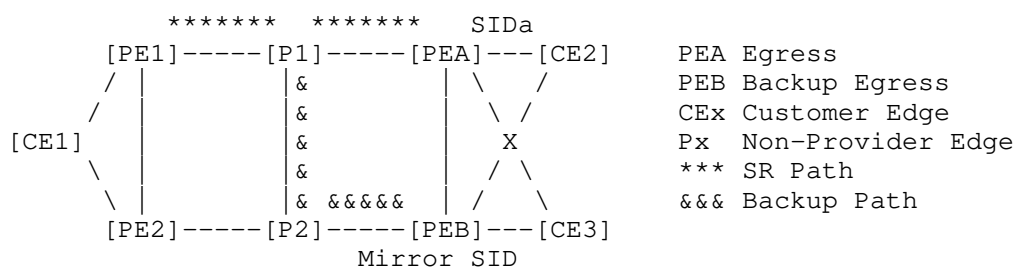


Figure 1: PEB Protects Egress PEA of SR Path

Where node PEA is the egress of the SR path from PE1 to PEA, and has SIDA which is the active segment in the packet from the SR path at PEA. Node PEB is the backup egress (or say protector) to provide the protection for egress (or say primary egress) PEA. Node P1 is the direct previous hop of egress PEA and acts as PLR to support the protection for PEA.

When PEB is selected as a backup egress to protect the egress PEA, a Mirror SID (refer to Section 5.1 of [RFC8402]) is configured on PEB to protect PEA. PEB advertises this information through IGP, which includes the Mirror SID and the egress PEA. The information is represented by <PEB, PEA, Mirror SID>, which indicates that PEB protects PEA with Mirror SID.

After PEA receives the information <PEB, PEA, Mirror SID>, it may send the forwarding behavior of the SIDA at PEA to PEB with the Mirror SID using some protocols such as BGP if PEB can not obtain this behavior from other approaches if PEB wants to protect SIDA of PEA. How to send the forwarding behavior of the SIDA to PEB is out scope of this document.

When PEB gets the forwarding behavior of the SIDA of PEA from PEA or other means, it adds a forwarding entry for the SIDA according to the behavior into the forwarding table for node PEA. This table is identified by the Mirror SID, which indicates node PEA's context. Using the forwarding entry for SIDA in this table, a packet with SIDA will be transmitted by PEB to the same destination as it is transmitted by PEA. For example, assume that the packet with SIDA is transmitted by PEA to CE2 through the forwarding behavior of the SIDA in PEA. The packet will be transmitted by PEB to the same CE2 through looking up the table identified by the Mirror SID.

After P1 as PLR receives the information <PEB, PEA, Mirror SID> and knows that PEB wants to protect SIDA of PEA, it computes a shortest path to PEB. A Repair List RL is obtained based on the path. It is one of the followings:

- o RL = <Mirror SID> if the path does not go through PEA; or
- o RL = <S1, ..., Sn, Mirror SID> if the path goes through PEA, where <S1, ..., Sn> is the TI-LFA Repair List to PEB computed by P1.

When PEA fails, P1 as PLR sends the packet with SIDA carried by the SR path to PEB, but encapsulates the packet before sending it by executing H.Encaps with the Repair List RL and a Source Address T.

Suppose that the packet received by P1 is represented by Pkt = (S, SIDA)Pkt0, where SA = S and DA = SIDA, and Pkt0 is the rest of the packet.

The execution of H.Encaps pushes an IPv6 header to Pkt and sets some fields in the outer and inner IPv6 header to produce an encapsulated packet Pkt'. Pkt' will be one of the followings:

- o Pkt' = (T, Mirror SID) (S, SIDA)Pkt0 if RL = <Mirror SID>; or

- o  $Pkt' = (T, S1)(Mirror\ SID, Sn, \dots, S1; SL=n) (S, SIDA)Pkt0$  if  $RL = \langle S1, \dots, Sn, Mirror\ SID \rangle$ .

When PEB receives the re-routed packet, which is  $(T, Mirror\ SID) (S, SIDA)Pkt0$ , it decapsulates the packet and forwards the decapsulated packet using the FIB table  $Tm$  identified by the Mirror SID as a variant of End.DT6 SID. The Mirror SID is called End.M.

It obtains the Mirror SID in the outer IPv6 header of the packet, removes this outer IPv6 header with all its extension headers, and then processes the inner IPv6 packet (i.e.,  $(S, SIDA)Pkt0$ , the packet without the outer IPv6 header). PEB finds the FIB table  $Tm$  for node PEA using the Mirror SID as the context ID, and submits the packet to this FIB table lookup and transmission to the same destination as PEA does.

The behavior of Mirror SID (End.M for short) is a variant of the End.DT6 behavior (refer to Section 4.6 of [RFC8986]). The End.M SID MUST be the last segment in an SR path, and a SID instance is associated with an IPv6 FIB table  $Tm$ .

When processing the Upper-Layer header of a packet matching a FIB entry locally instantiated as an End.M SID, N does the following:

```

S01. If (Upper-Layer header type == 41(IPv6) ) {
S02.   Remove the outer IPv6 header with all its extension headers
S03.   Set the packet's associated FIB table to  $Tm$ 
S04.   Submit the packet to the egress IPv6 FIB lookup for
       transmission to the new destination
S05. } Else {
S06.   Process as per Section 4.1.1 of RFC8986
S07. }
```

### 3.2. Example

Figure 2 shows an example of protecting egress PE3 of a SR path, which is from ingress PE1 to egress PE3.

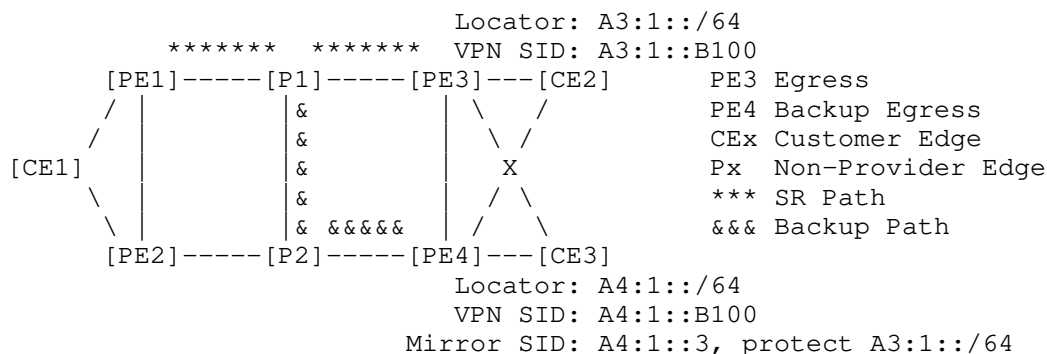


Figure 2: PE4 Protects Egress PE3 of SR Path

Where node P1's pre-computed backup path for PE3 is from P1 to PE4 via P2. In normal operations, after receiving a packet with destination PE3, P1 forwards the packet to PE3 according to its FIB. When PE3 receives the packet, it sends the packet to CE2.

When PE3 fails, P1 as PLR detects the failure through using a failure detection mechanism such as BFD and forwards the packet to PE4 via the backup path. When PE4 receives the packet, it sends the packet to the same CE2.

In Figure 2, Both CE2 and CE3 are dual home to PE3 and PE4. PE3 has a locator A3:1::/64 and a VPN SID A3:1::B100. PE4 has a locator A4:1::/64 and VPN SID A4:1::B100. A Mirror SID A4:1::3 is configured on PE4 for protecting PE3 with locator A3:1::/64.

After the configuration, PE4 advertises this information through an IGP LS (i.e., LSA in OSPF or LSP in IS-IS), which includes PE3's locator and Mirror SID A4:1::3. Every node in the SR domain will receive this IGP LS, which indicates that PE4 wants to protect PE3's locator with Mirror SID A4:1::3.

When PE4 (e.g., BGP on PE4) receives a prefix whose VPN SID belongs to PE3 that is protected by PE4 through Mirror SID A4:1::3, it finds PE4's VPN SID corresponding to PE3's VPN SID. For example, local PE4 has Prefix 1.1.1.1 with VPN SID A4:1::B100, when PE4 receives prefix 1.1.1.1 with remote PE3's VPN SID A3:1::B100, it knows that they are for the same VPN.

The forwarding behaviors for these two VPN SIDs are the same from function's point of view. If the behavior for PE3's VPN SID in PE3 forwards the packet with it to CE2, then the behavior for PE4's VPN SID in PE4 forwards the packet to the same CE2; and vice versa. PE4 creates a forwarding entry for PE3's VPN SID A3:1::B100 in the FIB

table identified by Mirror SID A4:1::3 according to the forwarding behavior for PE4's VPN SID A4:1::B100.

Node P1's pre-computed backup path for destination PE3's locator is from P1 to PE4 having mirror SID A4:1::3. When P1 receives a packet destined to PE3's VPN SID A3:1::B100, in normal operations, it forwards the packet with source A1:1:: and destination PE3's VPN SID A3:1::B100 according to the FIB using the destination PE3's VPN SID A3:1::B100.

When PE3 fails, P1 as PLR sends the packet to PE4 via the backup path pre-computed. P1 encapsulates the packet using H.Encaps before sending it to PE4.

Suppose that the packet received by P1 is represented by Pkt = (SA = A1:1::, DA = A3:1::B100)Pkt0, where DA = A3:1::B100 is PE3's VPN SID, and Pkt0 is the rest of the packet. The encapsulated packet Pkt' will be one of the followings:

- o Pkt' = (T, Mirror SID A4:1::3) (A1:1::, A3:1::B100)Pkt0 if backup path not via PE3; or (otherwise)
- o Pkt' = (T, S1) (Mirror SID A4:1::3, Sn, ..., S1; SL=n) (A1:1::, A3:1::B100)Pkt0.

where T is a Source Address, <S1, ..., Sn> is the TI-LFA Repair List to PE4 computed by P1 when the backup path to PE4 goes through PE3.

When PE4 receives the re-routed packet, it decapsulates the packet and forwards the decapsulated packet by executing End.DT6 behavior for an End.DT6 SID instance. The SID instance is End.M, the Mirror SID that is associated with the IPv6 FIB table for PE3. The packet received by PE4 is (T, Mirror SID A4:1::3) (A1:1::, PE3's VPN SID A3:1::B100)Pkt0.

PE4 obtains Mirror SID A4:1::3 in the outer IPv6 header of the packet, removes this outer IPv6 header, and then processes the inner IPv6 packet (A1:1::, A3:1::B100)Pkt0. It finds the FIB table for PE3 using Mirror SID A4:1::3 as the context ID, gets the forwarding entry for PE3's VPN SID A3:1::B100 from the table, and forwards the packet to CE2 using the entry.

#### 4. Extensions to IGP for Egress Protection

This section describes extensions to IS-IS and OSPF for advertising the information about SRv6 path egress protection.

#### 4.1. Extensions to IS-IS

A new sub-TLV, called IS-IS SRv6 Mirror SID sub-TLV, is defined. It is used in the SRv6 Locator TLV defined in [I-D.ietf-lsr-isis-srv6-extensions] to advertise SRv6 Mirror SID and the locators of the node to be protected. The SRv6 Mirror SID inherit the topology/algorithm from the parent locator. The format of the sub-TLV is illustrated below.

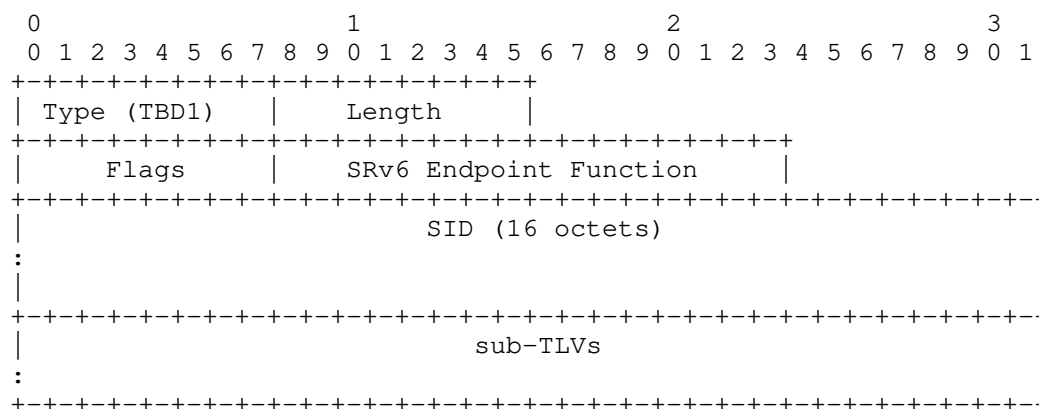


Figure 3: IS-IS SRv6 Mirror SID sub-TLV

Type: TBD1 (suggested value 8) is to be assigned by IANA.

Length: variable.

Flags: 1 octet. No flags are currently defined.

SRv6 Endpoint Function: 2 octets. It contains the endpoint function 74 for Mirror SID.

SID: 16 octets. This field contains the SRv6 Mirror SID to be advertised.

Two sub-TLVs are defined. One is the protected locators sub-TLV, and the other is the protected SIDs sub-TLV.

A protected locators sub-TLV is used to carry the Locators to be protected by the SRv6 mirror SID. It has the following format.

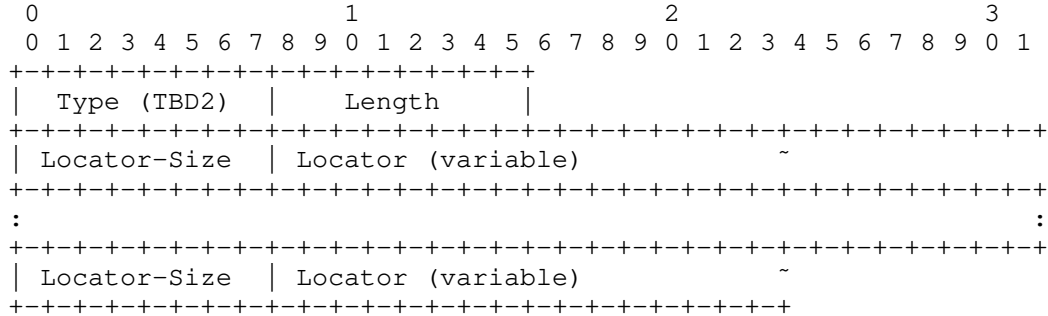


Figure 4: IS-IS Protected Locators sub-TLV

Type: TBD2 (suggested value 1) is to be assigned by IANA.

Length: variable.

Locator-Size: 1 octet. Number of bits (1 - 128) in the Locator field.

Locator: 1-16 octets. This field encodes an SRv6 Locator to be protected by the SRv6 mirror SID. The Locator is encoded in the minimal number of octets for the given number of bits.

A protected SIDs sub-TLV is used to carry the SIDs to be protected by the SRv6 Mirror SID. It has the following format.

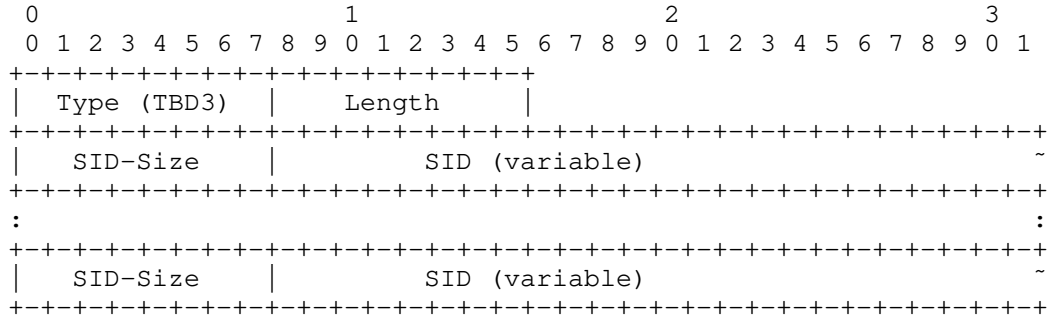


Figure 5: IS-IS Protected SIDs sub-TLV

Type: TBD3 (suggested value 2) is to be assigned by IANA.

Length: variable.

**SID-Size:** 1 octet. Number of bits in the SID field. It is from 1 to 128. When it is less than 128, the SID field is a locator. When it is 128, the SID field is an SRv6 SID.

**SID:** 1-16 octets. This field encodes an SRv6 SID or locator to be protected. The SID/locator is encoded in the minimal number of octets for the given number of bits. Trailing bits MUST be set to zero and ignored when received.

When node B advertises that B wants to protect node A's locators with a Mirror SID through an LSP, the LSP contains an IS-IS SRv6 Mirror SID sub-TLV, which includes the Mirror SID and the node A's locators in an IS-IS Protected locators sub-TLV. If B wants to protect just a specific set of SIDs of node A, the Mirror SID sub-TLV includes these SIDs in an IS-IS Protected SIDs sub-TLV.

#### 4.2. Extensions to OSPF

Similarly, a new sub-TLV, called OSPF Mirror SID sub-TLV, is defined. It is used to advertise SRv6 Mirror SID and the locators of the node to be protected. Its format is illustrated below.

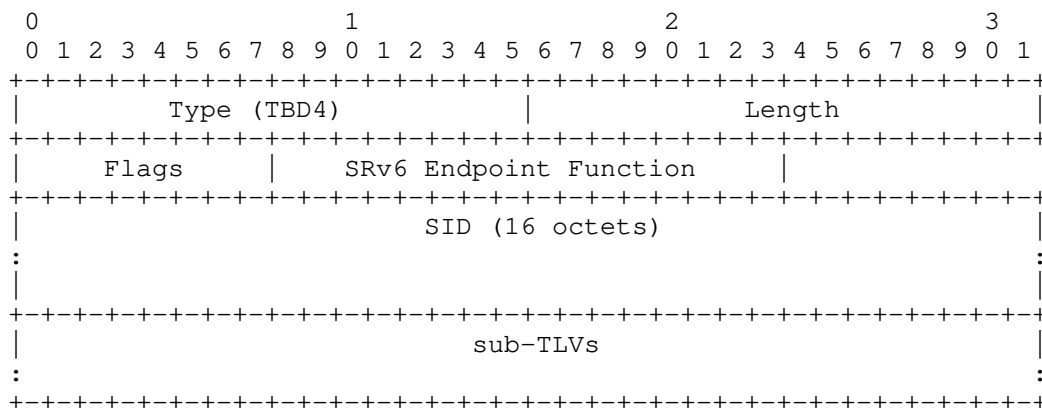


Figure 6: OSPF SRv6 Mirror SID sub-TLV

**Type:** TBD4 (suggested value 8) is to be assigned by IANA.

**Length:** variable.

**Flags:** 1 octet. No flags are currently defined.

**SRv6 Endpoint Function:** 2 octets. It contains the endpoint function 74 for End.M SID.



SID: 16 octets. This field contains the SRv6 Mirror SID to be advertised.

Two sub-TLVs are defined. One is the protected locators sub-TLV, and the other is the protected SIDs sub-TLV.

A protected locators sub-TLV is used to carry the locators of the node to be protected by the SRv6 Mirror SID. It has the following format.

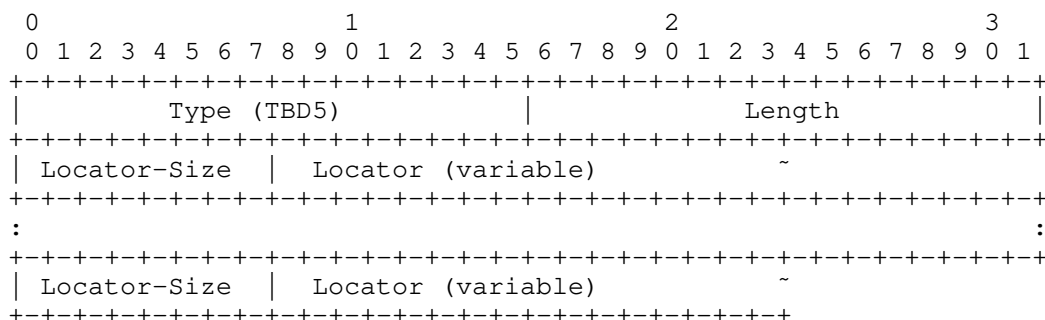


Figure 7: OSPF Protected Locators sub-TLV

Type: TBD5 (suggested value 1) is to be assigned by IANA.

Length: variable.

Locator-Size: 1 octet. Number of bits (1 - 128) in the Locator field.

Locator: 1-16 octets. This field encodes an SRv6 Locator to be protected by the SRv6 mirror SID. The Locator is encoded in the minimal number of octets for the given number of bits.

A protected SIDs sub-TLV is used to carry the SIDs to be protected by the SRv6 Mirror SID. It has the following format.

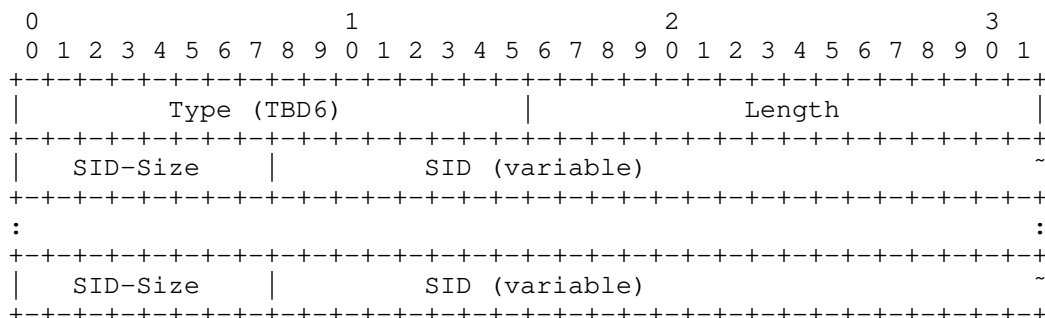


Figure 8: OSPF Protected SIDs sub-TLV

Type: TBD6 (suggested value 2) is to be assigned by IANA.

Length: variable.

SID-Size: 1 octet. Number of bits in the SID field. It is from 1 to 128. When it is less than 128, the SID field is a locator. When it is 128, the SID field is an SRv6 SID.

SID: 1-16 octets. This field encodes an SRv6 SID or locator to be protected. The SID/locator is encoded in the minimal number of octets for the given number of bits. Trailing bits MUST be set to zero and ignored when received.

## 5. Security Considerations

The security about the egress protection is described in details in [RFC8679]. The extensions to OSPF and IS-IS described in this document for SRv6 path egress protection should not cause extra security issues.

## 6. IANA Considerations

### 6.1. SRv6 Endpoint Behaviors

Under sub-registry "SRv6 Endpoint Behaviors" [RFC8986], IANA has assigned the following for End.M Endpoint Behavior:

Value	Hex	Endpoint behavior	Reference
74	0x004A	End.M (Mirror SID)	This document

## 6.2. IS-IS

Under "Sub-TLVs for TLVs 27, 135, 235, 236 and 237 registry" [I-D.ietf-lsr-isis-srv6-extensions], IANA is requested to add the following new Sub-TLV:

Sub-TLV Type	Sub-TLV Name	Reference
8	SRv6 Mirror SID Sub-TLV	This document

IANA is requested to create and maintain a new registry for sub-sub-TLVs of the SRv6 Mirror SID Sub-TLV. The suggested registry name is

- o Sub-Sub-TLVs for SRv6 Mirror SID Sub-TLV

Initial values for the registry are given below. The future assignments are to be made through IETF Review [RFC5226].

Value	Sub-Sub-TLV Name	Definition
0	Reserved	
1	Protected Locators Sub-Sub-TLV	This Document
2	Protected SIDs Sub-Sub-TLV	
3-255	Unassigned	

## 6.3. OSPFv3

Under registry "OSPFv3 Locator LSA Sub-TLVs" [I-D.ietf-lsr-ospfv3-srv6-extensions], IANA is requested to assign the following new Sub-TLV:

Sub-TLV Type	Sub-TLV Name	Reference
8	SRv6 Mirror SID Sub-TLV	This document

IANA is requested to create and maintain a new registry for sub-sub-TLVs of the SRv6 Mirror SID Sub-TLV. The suggested registry name is

- o Sub-Sub-TLVs for SRv6 Mirror SID Sub-TLV

Initial values for the registry are given below. The future assignments are to be made through IETF Review [RFC5226].

Value	Sub-Sub-TLV Name	Definition
-----	-----	-----
0	Reserved	
1	Protected Locators Sub-Sub-TLV	This Document
2	Protected SIDs Sub-Sub-TLV	
3-65535	Unassigned	

## 7. Acknowledgements

The authors would like to thank Peter Psenak, Yimin Shen, Zhenqiang Li, Alexander Vainshtein, Greg Mirsky, Bruno Decraene, Jeff Tantsura, Chris Bowers and Ketan Talaulikar for their comments to this work.

## 8. References

### 8.1. Normative References

- [I-D.ietf-lsr-isis-srv6-extensions]  
Psenak, P., Filsfils, C., Bashandy, A., Decraene, B., and Z. Hu, "IS-IS Extensions to Support Segment Routing over IPv6 Dataplane", draft-ietf-lsr-isis-srv6-extensions-18 (work in progress), October 2021.
- [I-D.ietf-lsr-ospfv3-srv6-extensions]  
Li, Z., Hu, Z., Cheng, D., Talaulikar, K., and P. Psenak, "OSPFv3 Extensions for SRv6", draft-ietf-lsr-ospfv3-srv6-extensions-03 (work in progress), November 2021.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC7356] Ginsberg, L., Previdi, S., and Y. Yang, "IS-IS Flooding Scope Link State PDUs (LSPs)", RFC 7356, DOI 10.17487/RFC7356, September 2014, <<https://www.rfc-editor.org/info/rfc7356>>.
- [RFC7490] Bryant, S., Filsfils, C., Previdi, S., Shand, M., and N. So, "Remote Loop-Free Alternate (LFA) Fast Reroute (FRR)", RFC 7490, DOI 10.17487/RFC7490, April 2015, <<https://www.rfc-editor.org/info/rfc7490>>.
- [RFC8400] Chen, H., Liu, A., Saad, T., Xu, F., and L. Huang, "Extensions to RSVP-TE for Label Switched Path (LSP) Egress Protection", RFC 8400, DOI 10.17487/RFC8400, June 2018, <<https://www.rfc-editor.org/info/rfc8400>>.

- [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>>.
- [RFC8665] Psenak, P., Ed., Previdi, S., Ed., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPF Extensions for Segment Routing", RFC 8665, DOI 10.17487/RFC8665, December 2019, <<https://www.rfc-editor.org/info/rfc8665>>.
- [RFC8667] Previdi, S., Ed., Ginsberg, L., Ed., Filsfils, C., Bashandy, A., Gredler, H., and B. Decraene, "IS-IS Extensions for Segment Routing", RFC 8667, DOI 10.17487/RFC8667, December 2019, <<https://www.rfc-editor.org/info/rfc8667>>.
- [RFC8679] Shen, Y., Jeganathan, M., Decraene, B., Gredler, H., Michel, C., and H. Chen, "MPLS Egress Protection Framework", RFC 8679, DOI 10.17487/RFC8679, December 2019, <<https://www.rfc-editor.org/info/rfc8679>>.
- [RFC8986] Filsfils, C., Ed., Camarillo, P., Ed., Leddy, J., Voyer, D., Matsushima, S., and Z. Li, "Segment Routing over IPv6 (SRv6) Network Programming", RFC 8986, DOI 10.17487/RFC8986, February 2021, <<https://www.rfc-editor.org/info/rfc8986>>.

## 8.2. Informative References

- [I-D.hu-spring-segment-routing-proxy-forwarding]  
Hu, Z., Chen, H., Yao, J., Bowers, C., Yongqing, and Yisong, "SR-TE Path Midpoint Restoration", draft-hu-spring-segment-routing-proxy-forwarding-19 (work in progress), April 2022.
- [I-D.ietf-rtgwg-segment-routing-ti-lfa]  
Litkowski, S., Bashandy, A., Filsfils, C., Francois, P., Decraene, B., and D. Voyer, "Topology Independent Fast Reroute using Segment Routing", draft-ietf-rtgwg-segment-routing-ti-lfa-08 (work in progress), January 2022.
- [I-D.ietf-spring-segment-routing-policy]  
Filsfils, C., Talaulikar, K., Voyer, D., Bogdanov, A., and P. Mattes, "Segment Routing Policy Architecture", draft-ietf-spring-segment-routing-policy-22 (work in progress), March 2022.

- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 5226, DOI 10.17487/RFC5226, May 2008, <<https://www.rfc-editor.org/info/rfc5226>>.
- [RFC5462] Andersson, L. and R. Asati, "Multiprotocol Label Switching (MPLS) Label Stack Entry: "EXP" Field Renamed to "Traffic Class" Field", RFC 5462, DOI 10.17487/RFC5462, February 2009, <<https://www.rfc-editor.org/info/rfc5462>>.

## Authors' Addresses

Zhibo Hu  
Huawei  
Huawei Bld., No.156 Beiqing Rd.  
Beijing 100095  
China

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

Huaimo Chen  
Futurewei  
Boston, MA  
USA

Email: [Huaimo.chen@futurewei.com](mailto:Huaimo.chen@futurewei.com)

Huanan Chen  
China Telecom  
109, West Zhongshan Road, Tianhe District  
Guangzhou 510000  
China

Email: [chenhn8.gd@chinatelecom.cn](mailto:chenhn8.gd@chinatelecom.cn)

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

Email: [baggio.wupeng@huawei.com](mailto:baggio.wupeng@huawei.com)

Mehmet Toy  
Verizon  
USA

Email: mehmet.toy@verizon.com

Chang Cao  
China Unicom  
Beijing China

Email: caoc15@chinaunicom.cn

Tao He  
China Unicom  
Beijing China

Email: het21@chinaunicom.cn

Lei Liu  
Fujitsu  
USA

Email: liulei.kddi@gmail.com

Xufeng Liu  
Volta Networks  
McLean, VA  
USA

Email: xufeng.liu.ietf@gmail.com

SPRING Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: May 5, 2021

C. Filsfils  
K. Talaulikar, Ed.  
Cisco Systems, Inc.  
D. Voyer  
Bell Canada  
A. Bogdanov  
Google, Inc.  
P. Mattes  
Microsoft  
November 1, 2020

Segment Routing Policy Architecture  
draft-ietf-spring-segment-routing-policy-09

Abstract

Segment Routing (SR) allows a headend node to steer a packet flow along any path. Intermediate per-flow states are eliminated thanks to source routing. The headend node steers a flow into an SR Policy. The header of a packet steered in an SR Policy is augmented with an ordered list of segments associated with that SR Policy. This document details the concepts of SR Policy and steering into an SR Policy.

Status of This Memo

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

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

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

This Internet-Draft will expire on May 5, 2021.

Copyright Notice

Copyright (c) 2020 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 . . . . .	3
1.1. Requirements Language . . . . .	3
2. SR Policy . . . . .	4
2.1. Identification of an SR Policy . . . . .	4
2.2. Candidate Path and Segment List . . . . .	5
2.3. Protocol-Origin of a Candidate Path . . . . .	6
2.4. Originator of a Candidate Path . . . . .	6
2.5. Discriminator of a Candidate Path . . . . .	7
2.6. Identification of a Candidate Path . . . . .	7
2.7. Preference of a Candidate Path . . . . .	8
2.8. Validity of a Candidate Path . . . . .	8
2.9. Active Candidate Path . . . . .	8
2.10. Validity of an SR Policy . . . . .	9
2.11. Instantiation of an SR Policy in the Forwarding Plane . .	9
2.12. Priority of an SR Policy . . . . .	10
2.13. Summary . . . . .	10
3. Segment Routing Database . . . . .	11
4. Segment Types . . . . .	12
4.1. Explicit Null . . . . .	16
5. Validity of a Candidate Path . . . . .	16
5.1. Explicit Candidate Path . . . . .	16
5.2. Dynamic Candidate Path . . . . .	18
5.3. Composite Candidate Path . . . . .	18
6. Binding SID . . . . .	19
6.1. BSID of a candidate path . . . . .	19
6.2. BSID of an SR Policy . . . . .	19
6.3. Forwarding Plane . . . . .	20
6.4. Non-SR usage of Binding SID . . . . .	21
7. SR Policy State . . . . .	21
8. Steering into an SR Policy . . . . .	21
8.1. Validity of an SR Policy . . . . .	22
8.2. Drop upon invalid SR Policy . . . . .	22
8.3. Incoming Active SID is a BSID . . . . .	22
8.4. Per-Destination Steering . . . . .	23
8.5. Recursion on an on-demand dynamic BSID . . . . .	24
8.6. Per-Flow Steering . . . . .	25

8.7. Policy-based Routing . . . . .	26
8.8. Optional Steering Modes for BGP Destinations . . . . .	26
9. Protection . . . . .	28
9.1. Leveraging TI-LFA local protection of the constituent IGP segments . . . . .	29
9.2. Using an SR Policy to locally protect a link . . . . .	29
9.3. Using a Candidate Path for Path Protection . . . . .	29
10. Security Considerations . . . . .	30
11. IANA Considerations . . . . .	30
11.1. Guidance for Designated Experts . . . . .	31
12. Acknowledgement . . . . .	31
13. Contributors . . . . .	32
14. References . . . . .	33
14.1. Normative References . . . . .	33
14.2. Informative References . . . . .	33
Authors' Addresses . . . . .	36

## 1. Introduction

Segment Routing (SR) allows a headend node to steer a packet flow along any path. Intermediate per-flow states are eliminated thanks to source routing [RFC8402].

The headend node is said to steer a flow into an Segment Routing Policy (SR Policy).

The header of a packet steered into an SR Policy is augmented with an ordered list of segments associated with that SR Policy.

This document details the concepts of SR Policy and steering packets into an SR Policy. These apply equally to the MPLS and SRv6 instantiations of segment routing.

For reading simplicity, the illustrations are provided for the MPLS instantiations.

### 1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

## 2. SR Policy

An SR Policy is a framework that enables instantiation of an ordered list of segments on a node for implementing a source routing policy with a specific intent for traffic steering from that node.

The Segment Routing architecture [RFC8402] specifies that any instruction can be bound to a segment. Thus, an SR Policy can be built using any type of Segment Identifier (SID) including those associated with topological or service instructions.

This section defines the key aspects and constituents of an SR Policy.

### 2.1. Identification of an SR Policy

An SR Policy is identified through the tuple <headend, color, endpoint>. In the context of a specific headend, one may identify an SR policy by the <color, endpoint> tuple.

The headend is the node where the policy is instantiated/implemented. The headend is specified as an IPv4 or IPv6 address and is expected to be unique in the domain.

The endpoint indicates the destination of the policy. The endpoint is specified as an IPv4 or IPv6 address and is expected to be unique in the domain. In a specific case (refer to Section 8.8.1), the endpoint can be the null address (0.0.0.0 for IPv4, ::0 for IPv6).

The color is a 32-bit numerical value that associates the SR Policy with an intent (e.g. low-latency).

The endpoint and the color are used to automate the steering of service or transport routes on SR Policies (refer to Section 8).

An implementation MAY allow assignment of a symbolic name comprising of printable ASCII characters to an SR Policy to serve as a user-friendly attribute for debug and troubleshooting purposes. Such symbolic names may identify an SR Policy when the naming scheme ensures uniqueness. The SR Policy name may also be signaled along with a candidate path of the SR Policy (refer Section 2.2). An SR Policy may have multiple names associated with it in the scenario where the headend receives different SR Policy names along with candidate paths for the same SR Policy.

## 2.2. Candidate Path and Segment List

An SR Policy is associated with one or more candidate paths. A candidate path is the unit for signaling of an SR Policy to a headend via protocols like Path Computation Element (PCE) Communication Protocol (PCEP) [RFC8664] or BGP SR Policy [I-D.ietf-idr-segment-routing-te-policy].

A Segment-List represents a specific source-routed path to send traffic from the headend to the endpoint of the corresponding SR policy.

A candidate path is either dynamic, explicit or composite.

An explicit candidate path is expressed as a Segment-List or a set of Segment-Lists.

A dynamic candidate path expresses an optimization objective and a set of constraints. The headend (potentially with the help of a PCE) computes the solution Segment-List (or set of Segment-Lists) that solves the optimization problem.

If a candidate path is associated with a set of Segment-Lists, each Segment-List is associated with a weight for weighted load balancing (refer Section 2.11 for details). The default weight is 1.

A composite candidate path acts as a container for grouping of SR Policies. The composite candidate path construct enables combination of SR Policies, each with explicit candidate paths and/or dynamic candidate paths with potentially different optimization objectives and constraints, for a load-balanced steering of packet flows over its constituent SR Policies. The following criteria apply for inclusion of constituent SR Policies using a composite candidate path under a parent SR Policy:

- o the endpoints of the constituent SR Policies and the parent SR Policy MUST be identical
- o The colors of each of the constituent SR Policies and the parent SR Policy MUST be different
- o the constituent SR Policies MUST NOT use composite candidate paths

Each constituent SR Policy of a composite candidate path is associated with a weight for load-balancing purposes (refer Section 2.11 for details). The default weight is 1.

### 2.3. Protocol-Origin of a Candidate Path

A headend may be informed about a candidate path for an SR Policy <color, endpoint> by various means including: via configuration, PCEP [RFC8664] or BGP [I-D.ietf-idr-segment-routing-te-policy].

Protocol-Origin of a candidate path is an 8-bit value which identifies the component or protocol that originates or signals the candidate path.

The head-end assigns different Protocol-Origin Priority values to each Protocol-Origin. The Protocol-Origin Priority is used as a tie-breaker between candidate-paths of equal preference, as described in Section 2.9. The table below specifies the RECOMMENDED default values of Protocol-Origin Priority:

Value	Protocol-Origin
10	PCEP
20	BGP SR Policy
30	Via Configuration

Table 1: Protocol-Origin Priority default values

Implementations MAY allow modifications of these default values assigned to protocols on the headend along similar lines as a routing administrative distance. Its application in the candidate path selection is described in Section 2.9.

### 2.4. Originator of a Candidate Path

Originator identifies the node which provisioned or signalled the candidate path on the headend. The originator is expressed in the form of a 160 bit numerical value formed by the concatenation of the fields of the tuple <ASN, node-address> as below:

- o ASN : represented as a 4 byte number.
- o Node Address : represented as a 128 bit value. IPv4 addresses are encoded in the lowest 32 bits.

Its application in the candidate path selection is described in Section 2.9.

When Protocol-Origin is Via Configuration, the ASN and node address MAY be set to either the headend or the provisioning controller/node ASN and address. Default value is 0 for both AS and node address.

When Protocol-Origin is PCEP, it is the IPv4 or IPv6 address of the PCE and the AS number SHOULD be set to 0 by default when not available or known.

When Protocol-Origin is BGP SR Policy, the ASN and Node Address are provided by BGP (refer [I-D.ietf-idr-segment-routing-te-policy]) on the headend.

## 2.5. Discriminator of a Candidate Path

The Discriminator is a 32 bit value associated with a candidate path that uniquely identifies it within the context of an SR Policy from a specific Protocol-Origin as specified below:

When Protocol-Origin is Via Configuration, this is an implementation's configuration model specific unique identifier for a candidate path. Default value is 0.

When PCEP is the Protocol-Origin, the method to uniquely identify signalled path will be specified in a future PCEP document. Default value is 0.

When BGP SR Policy is the Protocol-Origin, it is the distinguisher (refer Section 2.1 of [I-D.ietf-idr-segment-routing-te-policy]).

Its application in the candidate path selection is described in Section 2.9.

## 2.6. Identification of a Candidate Path

A candidate path is identified in the context of a single SR Policy.

A candidate path is not shared across SR Policies.

A candidate path is not identified by its Segment-List(s).

If CP1 is a candidate path of SR Policy Pol1 and CP2 is a candidate path of SR Policy Pol2, then these two candidate paths are independent, even if they happen to have the same Segment-List. The Segment-List does not identify a candidate path. The Segment-List is an attribute of a candidate path.

The identity of a candidate path MUST be uniquely established in the context of an SR Policy <headend, color, endpoint> in order to handle

add, delete or modify operations on them in an unambiguous manner regardless of their source(s).

The tuple <Protocol-Origin, originator, discriminator> uniquely identifies a candidate path.

Candidate paths MAY also be assigned or signaled with a symbolic name comprising printable ASCII characters to serve as a user-friendly attribute for debug and troubleshooting purposes. Such symbolic names MUST NOT be considered as identifiers for a candidate path.

## 2.7. Preference of a Candidate Path

The preference of the candidate path is used to select the best candidate path for an SR Policy. The default preference is 100.

It is RECOMMENDED that each candidate path of a given SR policy has a different preference.

## 2.8. Validity of a Candidate Path

A candidate path is usable when it valid. A common path validity criterion is the reachability of its constituent SIDs. The validation rules are specified in Section 5.

## 2.9. Active Candidate Path

A candidate path is selected when it is valid and it is determined to be the best path of the SR Policy. The selected path is referred to as the "active path" of the SR policy in this document.

Whenever a new path is learned or an active path is deleted, the validity of an existing path changes or an existing path is changed, the selection process MUST be re-executed.

The candidate path selection process operates on the candidate path Preference. A candidate path is selected when it is valid and it has the highest preference value among all the candidate paths of the SR Policy.

In the case of multiple valid candidate paths of the same preference, the tie-breaking rules are evaluated on the identification tuple in the following order until only one valid best path is selected:

1. Higher value of Protocol-Origin Priority is selected.
2. If specified by configuration, prefer the existing installed path.

3. Lower value of originator is selected.
4. Finally, the higher value of discriminator is selected.

The rules are framed with multiple protocols and sources in mind and hence may not follow the logic of a single protocol (e.g. BGP best path selection). The motivation behind these rules are as follows:

- o The Protocol-Origin allows an operator to setup a default selection mechanism across protocol sources, e.g., to prefer configured over paths signalled via BGP SR Policy or PCEP.
- o The preference, being the first tiebreaker, allows an operator to influence selection across paths thus allowing provisioning of multiple path options, e.g., CP1 is preferred and if it becomes invalid then fall-back to CP2 and so on. Since preference works across protocol sources it also enables (where necessary) selective override of the default protocol-origin preference, e.g., to prefer a path signalled via BGP SR Policy over what is configured.
- o The originator allows an operator to have multiple redundant controllers and still maintain a deterministic behaviour over which of them are preferred even if they are providing the same candidate paths for the same SR policies to the headend.
- o The discriminator performs the final tiebreaking step to ensure a deterministic outcome of selection regardless of the order in which candidate paths are signalled across multiple transport channels or sessions.

[I-D.filsfils-spring-sr-policy-considerations] provides a set of examples to illustrate the active candidate path selection rules.

## 2.10. Validity of an SR Policy

An SR Policy is valid when it has at least one valid candidate path.

## 2.11. Instantiation of an SR Policy in the Forwarding Plane

A valid SR Policy is instantiated in the forwarding plane.

Only the active candidate path SHOULD be used for forwarding traffic that is being steered onto that policy.

If a set of Segment-Lists is associated with the active path of the policy, then the steering is per flow and W-ECMP based according to the relative weight of each Segment-List.



The fraction of the flows associated with a given Segment-List is  $w/S_w$  where  $w$  is the weight of the Segment-List and  $S_w$  is the sum of the weights of the Segment-Lists of the selected path of the SR Policy.

When a composite candidate path is active, the fraction of flows steered into each constituent SR Policy is equal to the relative weight of each constituent SR Policy. Further load balancing of flows steered into a constituent SR Policy is performed based on the weights of the Segment-List of the active candidate path of that constituent SR Policy.

The accuracy of the weighted load-balancing depends on the platform implementation.

## 2.12. Priority of an SR Policy

Upon topological change, many policies could be recomputed or revalidated. An implementation MAY provide a per-policy priority configuration. The operator MAY set this field to indicate order in which the policies should be re-computed. Such a priority is represented by an integer in the range (0, 255) where the lowest value is the highest priority. The default value of priority is 128.

An SR Policy may comprise multiple Candidate Paths received from the same or different sources. A candidate path MAY be signaled with a priority value. When an SR Policy has multiple candidate paths with distinct signaled non-default priority values, the SR Policy as a whole takes the lowest value (i.e. the highest priority) amongst these signaled priority values.

## 2.13. Summary

In summary, the information model is the following:

```
SR policy POL1 <headend = H1, color = 1, endpoint = E1>
  Candidate-path CP1 <protocol-origin = 20, originator =
100:1.1.1.1, discriminator = 1>
    Preference 200
    Weight W1, SID-List1 <SID11...SID1i>
    Weight W2, SID-List2 <SID21...SID2j>
  Candidate-path CP2 <protocol-origin = 20, originator =
100:2.2.2.2, discriminator = 2>
    Preference 100
    Weight W3, SID-List3 <SID31...SID3i>
    Weight W4, SID-List4 <SID41...SID4j>
```

The SR Policy POL1 is identified by the tuple <headend, color, endpoint>. It has two candidate paths CP1 and CP2. Each is

identified by a tuple <protocol-origin, originator, discriminator>. CP1 is the active candidate path (it is valid and it has the highest preference). The two Segment-Lists of CP1 are installed as the forwarding instantiation of SR policy POL1. Traffic steered on POL1 is flow-based hashed on Segment-List <SID11...SID1i> with a ratio  $W1/(W1+W2)$ .

The information model of SR Policy POL100 having a composite candidate path is the following:

```
SR policy POL100 <headend = H1, color = 100, endpoint = E1>
  Candidate-path CP1 <protocol-origin = 20, originator =
100:1.1.1.1, discriminator = 1>
    Preference 200
    Weight W1, SR policy <color = 1>
    Weight W2, SR policy <color = 2>
```

The constituent SR Policies POL1 and POL2 have information model as described at the start of this section. They are referenced only by color in the composite candidate path since their headend and endpoint are identical to the POL100. The valid Segment-Lists of the active candidate path of POL1 and POL2 are installed in the forwarding. Traffic steered on POL100 is flow-based hashed on POL1 with a ratio  $W1/(W1+W2)$ . Within the POL1, the flow-based hashing over its Segment-Lists are performed as described earlier in this section.

### 3. Segment Routing Database

An SR headend maintains the Segment Routing Database (SR-DB). The SR-DB is a conceptual database to illustrate the various pieces of information and their sources that may help in SR Policy computation and validation. There is no specific requirement for an implementation to create a new database as such.

An SR headend leverages the SR-DB to validate explicit candidate paths and compute dynamic candidate paths.

The information in the SR-DB MAY include:

- o IGP information (topology, IGP metrics based on ISIS [RFC1195] and OSPF [RFC2328] [RFC5340])
- o Segment Routing information (such as SRGB, SRLB, Prefix-SIDs, Adj-SIDs, BGP Peering SID, SRv6 SIDs) [RFC8402] [I-D.ietf-spring-srv6-network-programming]
- o TE Link Attributes (such as TE metric, SRLG, attribute-flag, extended admin group) [RFC5305] [RFC3630].

- o Extended TE Link attributes (such as latency, loss) [RFC8570] [RFC7471]
- o Inter-AS Topology information [I-D.ietf-idr-bgppls-segment-routing-epe].

The attached domain topology MAY be learned via IGP, BGP-LS or NETCONF.

A non-attached (remote) domain topology MAY be learned via BGP-LS or NETCONF.

In some use-cases, the SR-DB may only contain the attached domain topology while in others, the SR-DB may contain the topology of multiple domains and in this case it is multi-domain capable.

The SR-DB MAY also contain the SR Policies instantiated in the network. This can be collected via BGP-LS [I-D.ietf-idr-te-lsp-distribution] or PCEP [RFC8231] and [I-D.ietf-pce-binding-label-sid]. This information allows to build an end-to-end policy on the basis of intermediate SR policies (see Section 6 for further details).

The SR-DB MAY also contain the Maximum SID Depth (MSD) capability of nodes in the topology. This can be collected via ISIS [RFC8491], OSPF [RFC8476], BGP-LS [RFC8814] or PCEP [RFC8664].

The use of the SR-DB for computation and validation of SR Policies is outside the scope of this document. Some implementation aspects related to this are covered in [I-D.filsfils-spring-sr-policy-considerations].

#### 4. Segment Types

A Segment-List is an ordered set of segments represented as <S1, S2, ... Sn> where S1 is the first segment.

Based on the desired dataplane, either the MPLS label stack or the SRv6 SRH is built from the Segment-List. However, the Segment-List itself can be specified using different segment-descriptor types and the following are currently defined:

##### Type A: SR-MPLS Label:

A MPLS label corresponding to any of the segment types defined for SR-MPLS (as defined in [RFC8402] or other SR-MPLS specifications) can be used. Additionally, reserved labels like explicit-null or in general any MPLS label may also be used. E.g. this type can be used to specify a label representation which maps to an optical transport path on a

packet transport node. This type does not require the headend to perform SID resolution.

Type B: SRv6 SID:

An IPv6 address corresponding to any of the SID behaviors for SRv6 (as defined in [I-D.ietf-spring-srv6-network-programming] or other SRv6 specifications) can be used. This type does not require the headend to perform SID resolution. Optionally, the SRv6 SID behavior (as defined in [I-D.ietf-spring-srv6-network-programming] or other SRv6 specifications) and structure (as defined in [I-D.ietf-spring-srv6-network-programming]) MAY also be provided for the headend to perform validation of the SID when using it for building the Segment List.

Type C: IPv4 Prefix with optional SR Algorithm:

The headend is required to resolve the specified IPv4 Prefix Address to the SR-MPLS label corresponding to a Prefix SID segment (as defined in [RFC8402]). The SR algorithm (refer to Section 3.1.1 of [RFC8402]) to be used MAY also be provided.

Type D: IPv6 Global Prefix with optional SR Algorithm for SR-MPLS:

In this case the headend is required to resolve the specified IPv6 Global Prefix Address to the SR-MPLS label corresponding to its Prefix SID segment (as defined in [RFC8402]). The SR Algorithm (refer to Section 3.1.1 of [RFC8402]) to be used MAY also be provided.

Type E: IPv4 Prefix with Local Interface ID:

This type allows identification of Adjacency SID or BGP Peer Adjacency SID (as defined in [RFC8402]) label for point-to-point links including IP unnumbered links. The headend is required to resolve the specified IPv4 Prefix Address to the Node originating it and then use the Local Interface ID to identify the point-to-point link whose adjacency is being referred to. The Local Interface ID link descriptor follows semantics as specified in [RFC7752]. This type can also be used to indicate indirection into a layer 2 interface (i.e. without IP address) like a representation of an optical transport path or a layer 2 Ethernet port or circuit at the specified node.

Type F: IPv4 Addresses for link endpoints as Local, Remote pair:

This type allows identification of Adjacency SID or BGP Peer Adjacency SID (as defined in [RFC8402]) label for links. The headend is required to resolve the specified IPv4 Local Address to the Node originating it and then use the IPv4 Remote Address to identify the link adjacency being referred to. The Local

and Remote Address pair link descriptors follows semantics as specified in [RFC7752].

Type G: IPv6 Prefix and Interface ID for link endpoints as Local, Remote pair for SR-MPLS:

This type allows identification of Adjacency SID or BGP Peer Adjacency SID (as defined in [RFC8402]) label for links including those with only Link Local IPv6 addresses. The headend is required to resolve the specified IPv6 Prefix Address to the Node originating it and then use the Local Interface ID to identify the point-to-point link whose adjacency is being referred to. For other than point-to-point links, additionally the specific adjacency over the link needs to be resolved using the Remote Prefix and Interface ID. The Local and Remote pair of Prefix and Interface ID link descriptor follows semantics as specified in [RFC7752]. This type can also be used to indicate indirection into a layer 2 interface (i.e. without IP address) like a representation of an optical transport path or a layer 2 Ethernet port or circuit at the specified node.

Type H: IPv6 Addresses for link endpoints as Local, Remote pair for SR-MPLS:

This type allows identification of Adjacency SID or BGP Peer Adjacency SID (as defined in [RFC8402]) label for links with Global IPv6 addresses. The headend is required to resolve the specified Local IPv6 Address to the Node originating it and then use the Remote IPv6 Address to identify the link adjacency being referred to. The Local and Remote Address pair link descriptors follows semantics as specified in [RFC7752].

Type I: IPv6 Global Prefix with optional SR Algorithm for SRv6:

The headend is required to resolve the specified IPv6 Global Prefix Address to an SRv6 SID corresponding to a Prefix SID segment (as defined in [RFC8402]), such as a SID associated with the End behavior (as defined in [I-D.ietf-spring-srv6-network-programming]), of the node which is originating the prefix. The SR Algorithm (refer to Section 3.1.1 of [RFC8402]), the SRv6 SID behavior (as defined in [I-D.ietf-spring-srv6-network-programming] or other SRv6 specifications) and structure (as defined in [I-D.ietf-spring-srv6-network-programming]) MAY also be provided.

Type J: IPv6 Prefix and Interface ID for link endpoints as Local, Remote pair for SRv6:

This type allows identification of an SRv6 SID corresponding to an Adjacency SID or BGP Peer Adjacency SID (as defined in

[RFC8402]), such as a SID associated with the End.X behavior (as defined in [I-D.ietf-spring-srv6-network-programming]), associated with link or adjacency with only Link Local IPv6 addresses. The headend is required to resolve the specified IPv6 Prefix Address to the Node originating it and then use the Local Interface ID to identify the point-to-point link whose adjacency is being referred to. For other than point-to-point links, additionally the specific adjacency needs to be resolved using the Remote Prefix and Interface ID. The Local and Remote pair of Prefix and Interface ID link descriptor follows semantics as specified in [RFC7752]. The SR Algorithm (refer to Section 3.1.1 of [RFC8402]), the SRv6 SID behavior (as defined in [I-D.ietf-spring-srv6-network-programming] or other SRv6 specifications) and structure (as defined in [I-D.ietf-spring-srv6-network-programming]) MAY also be provided.

Type K: IPv6 Addresses for link endpoints as Local, Remote pair for SRv6:

This type allows identification of an SRv6 SID corresponding to an Adjacency SID or BGP Peer Adjacency SID (as defined in [RFC8402]), such as a SID associated with the End.X behavior (as defined in [I-D.ietf-spring-srv6-network-programming]), associated with link or adjacency with Global IPv6 addresses. The headend is required to resolve the specified Local IPv6 Address to the Node originating it and then use the Remote IPv6 Address to identify the link adjacency being referred to. The Local and Remote Address pair link descriptors follows semantics as specified in [RFC7752]. The SR Algorithm (refer to Section 3.1.1 of [RFC8402]), the SRv6 SID behavior (as defined in [I-D.ietf-spring-srv6-network-programming] or other SRv6 specifications) and structure (as defined in [I-D.ietf-spring-srv6-network-programming]) MAY also be provided.

When the algorithm is not specified for the SID types above which optionally allow for it, the headend SHOULD use the Strict Shortest Path algorithm if available; otherwise it SHOULD use the default Shortest Path algorithm. The specification of algorithm enables the use of IGP Flex Algorithm [I-D.ietf-lsr-flex-algo] specific SIDs in SR Policy.

For SID types C-through-K, a SID value may also be optionally provided to the headend for verification purposes. Section 5.1. describes the resolution and verification of the SIDs and Segment Lists on the headend.

When building the MPLS label stack or the IPv6 Segment list from the Segment List, the node instantiating the policy MUST interpret the set of Segments as follows:

- o The first Segment represents the topmost label or the first IPv6 segment. It identifies the active segment the traffic will be directed toward along the explicit SR path.
- o The last Segment represents the bottommost label or the last IPv6 segment the traffic will be directed toward along the explicit SR path.

#### 4.1. Explicit Null

A Type A SID may be any MPLS label, including reserved labels.

For example, assuming that the desired traffic-engineered path from a headend 1 to an endpoint 4 can be expressed by the Segment-List <16002, 16003, 16004> where 16002, 16003 and 16004 respectively refer to the IPv4 Prefix SIDs bound to node 2, 3 and 4, then IPv6 traffic can be traffic-engineered from nodes 1 to 4 via the previously described path using an SR Policy with Segment-List <16002, 16003, 16004, 2> where mpls label value of 2 represents the "IPv6 Explicit NULL Label".

The penultimate node before node 4 will pop 16004 and will forward the frame on its directly connected interface to node 4.

The endpoint receives the traffic with top label "2" which indicates that the payload is an IPv6 packet.

When steering unlabeled IPv6 BGP destination traffic using an SR policy composed of Segment-List(s) based on IPv4 SIDs, the Explicit Null Label Policy is processed as specified in [I-D.ietf-idr-segment-routing-te-policy) Section 2.4.4. When an "IPv6 Explicit NULL label" is not present as the bottom label, the headend SHOULD automatically impose one. Refer to Section 8 for more details.

### 5. Validity of a Candidate Path

#### 5.1. Explicit Candidate Path

An explicit candidate path is associated with a Segment-List or a set of Segment-Lists.

An explicit candidate path is provisioned by the operator directly or via a controller.

The computation/logic that leads to the choice of the Segment-List is external to the SR Policy headend. The SR Policy headend does not compute the Segment-List. The SR Policy headend only confirms its validity.

An explicit candidate path MAY consist of a single explicit Segment-List containing only an implicit-null label to indicate pop-and-forward behavior. The BSID is popped and the traffic is forwarded based on the inner label or an IP lookup in the case of unlabeled IP packets. Such an explicit path can serve as a fallback or path of last resort for traffic being steered into an SR Policy using its BSID (refer to Section 8.3).

A Segment-List of an explicit candidate path MUST be declared invalid when:

- o It is empty.
- o Its weight is 0.
- o The headend is unable to perform path resolution for the first SID into one or more outgoing interface(s) and next-hop(s).
- o The headend is unable to perform SID resolution for any non-first SID of type C-through-K into an MPLS label or an SRv6 SID.
- o The headend verification fails for any SID for which verification has been explicitly requested.

"Unable to perform path resolution" means that the headend has no path to the SID in its SR database.

SID verification is performed when the headend is explicitly requested to verify SID(s) by the controller via the signaling protocol used. Implementations MAY provide a local configuration option to enable verification on a global or per policy or per candidate path basis.

"Verification fails" for a SID means any of the following:

- o The headend is unable to find the SID in its SR DB
- o The headend detects mis-match between the SID value and its context provided for SIDs of type C-through-L in its SR DB.
- o The headend is unable to perform SID resolution for any non-first SID of type C-through-K into an MPLS label or an SRv6 SID.

In multi-domain deployments, it is expected that the headend be unable to verify the reachability of the SIDs in remote domains. Types A or B MUST be used for the SIDs for which the reachability cannot be verified. Note that the first SID MUST always be reachable regardless of its type.



In addition, a Segment-List MAY be declared invalid when:

- o Its last segment is not a Prefix SID (including BGP Peer Node-SID) advertised by the node specified as the endpoint of the corresponding SR policy.
- o Its last segment is not an Adjacency SID (including BGP Peer Adjacency SID) of any of the links present on neighbor nodes and that terminate on the node specified as the endpoint of the corresponding SR policy.

An explicit candidate path is invalid as soon as it has no valid Segment-List.

## 5.2. Dynamic Candidate Path

A dynamic candidate path is specified as an optimization objective and constraints.

The headend of the policy leverages its SR database to compute a Segment-List ("solution Segment-List") that solves this optimization problem.

The headend re-computes the solution Segment-List any time the inputs to the problem change (e.g., topology changes).

When local computation is not possible (e.g., a policy's tailend is outside the topology known to the headend) or not desired, the headend MAY send path computation request to a PCE supporting PCEP extension specified in [RFC8664].

If no solution is found to the optimization objective and constraints, then the dynamic candidate path MUST be declared invalid.

[I-D.filsfils-spring-sr-policy-considerations] discusses some of the optimization objectives and constraints that may be considered by a dynamic candidate path. It illustrates some of the desirable properties of the computation of the solution Segment-List.

## 5.3. Composite Candidate Path

A composite candidate path is specified as a group of its constituent SR Policies.

A composite candidate path is valid when it has at least one valid constituent SR Policy.

## 6. Binding SID

The Binding SID (BSID) is fundamental to Segment Routing [RFC8402]. It provides scaling, network opacity and service independence. [I-D.filsfils-spring-sr-policy-considerations] illustrates some of these benefits. This section describes the association of BSID with an SR Policy.

### 6.1. BSID of a candidate path

Each candidate path MAY be defined with a BSID.

Candidate Paths of the same SR policy SHOULD have the same BSID.

Candidate Paths of different SR policies MUST NOT have the same BSID.

### 6.2. BSID of an SR Policy

The BSID of an SR Policy is the BSID of its active candidate path.

When the active candidate path has a specified BSID, the SR Policy uses that BSID if this value (label in MPLS, IPv6 address in SRv6) is available (i.e., not associated with any other usage: e.g. to another MPLS client, to another SID, to another SR Policy). In the case of SR-MPLS, an SRv6 BSID (e.g. with the behavior End.BM [I-D.ietf-spring-srv6-network-programming]) MAY be associated with the SR Policy in addition to the MPLS BSID. In the case of SRv6, multiple SRv6 BSIDs (e.g. with different behaviors like End.B6.Encap and End.B6.Encap.Red [I-D.ietf-spring-srv6-network-programming]) MAY be associated with the SR Policy.

Optionally, instead of only checking that the BSID of the active path is available, a headend MAY check that it is available within a given SID range i.e., Segment Routing Local Block (SRLB) as specified in [RFC8402].

When the specified BSID is not available (optionally is not in the SRLB), an alert message MUST be generated.

In the cases (as described above) where SR Policy does not have a BSID available, then the SR Policy MAY dynamically bind a BSID to itself. Dynamically bound BSID SHOULD use an available SID outside the SRLB.

Assuming that at time  $t$  the BSID of the SR Policy is  $B1$ , if at time  $t+dt$  a different candidate path becomes active and this new active path does not have a specified BSID or its BSID is specified but is

not available (e.g. it is in use by something else), then the SR Policy keeps the previous BSID B1.

The association of an SR Policy with a BSID thus MAY change over the life of the SR Policy (e.g., upon active path change). Hence, the BSID SHOULD NOT be used as an identification of an SR Policy.

#### 6.2.1. Frequent use-case : unspecified BSID

All the candidate paths of the same SR Policy can have an unspecified BSID.

In such a case, a BSID MAY be dynamically bound to the SR Policy as soon as the first valid candidate path is received. That BSID is kept along all the life of the SR Policy and across changes of active candidate path.

#### 6.2.2. Frequent use-case: all specified to the same BSID

All the paths of the SR Policy can have the same specified BSID.

#### 6.2.3. Specified-BSID-only

An implementation MAY support the configuration of the Specified-BSID-only restrictive behavior on the headend for all SR Policies or individual SR Policies. Further, this restrictive behavior MAY also be signaled on a per SR Policy basis to the headend.

When this restrictive behavior is enabled, if the candidate path has an unspecified BSID or if the specified BSID is not available when the candidate path becomes active then no BSID is bound to it and it is considered invalid. An alert MUST be triggered for this error. Other candidate paths MUST then be evaluated for becoming the active candidate path.

### 6.3. Forwarding Plane

A valid SR Policy installs a BSID-keyed entry in the forwarding plane with the action of steering the packets matching this entry to the selected path of the SR Policy.

If the Specified-BSID-only restrictive behavior is enabled and the BSID of the active path is not available (optionally not in the SRLB), then the SR Policy does not install any entry indexed by a BSID in the forwarding plane.

#### 6.4. Non-SR usage of Binding SID

An implementation MAY choose to associate a Binding SID with any type of interface (e.g. a layer 3 termination of an Optical Circuit) or a tunnel (e.g. IP tunnel, GRE tunnel, IP/UDP tunnel, MPLS RSVP-TE tunnel, etc). This enables the use of other non-SR enabled interfaces and tunnels as segments in an SR Policy Segment-List without the need of forming routing protocol adjacencies over them.

The details of this kind of usage are beyond the scope of this document. A specific packet optical integration use case is described in [I-D.anand-spring-poi-sr].

#### 7. SR Policy State

The SR Policy State is maintained on the headend to represent the state of the policy and its candidate paths. This is to provide an accurate representation of whether the SR Policy is being instantiated in the forwarding plane and which of its candidate paths and segment-list(s) are active. The SR Policy state MUST also reflect the reason when a policy and/or its candidate path is not active due to validation errors or not being preferred.

The SR Policy state can be reported by the headend node via BGP-LS [I-D.ietf-idr-te-lsp-distribution] or PCEP [RFC8231] and [I-D.ietf-pce-binding-label-sid].

SR Policy state on the headend also includes traffic accounting information for the flows being steered via the policies. The details of the SR Policy accounting are beyond the scope of this document. The aspects related to the SR traffic counters and their usage in the broader context of traffic accounting in a SR network are covered in [I-D.filsfils-spring-sr-traffic-counters] and [I-D.ali-spring-sr-traffic-accounting] respectively.

Implementations MAY support an administrative state to control locally provisioned policies via mechanisms like CLI or NETCONF.

#### 8. Steering into an SR Policy

A headend can steer a packet flow into a valid SR Policy in various ways:

- o Incoming packets have an active SID matching a local BSID at the headend.
- o Per-destination Steering: incoming packets match a BGP/Service route which recurses on an SR policy.

- o Per-flow Steering: incoming packets match or recurse on a forwarding array of where some of the entries are SR Policies.
- o Policy-based Steering: incoming packets match a routing policy which directs them on an SR policy.

For simplicity of illustration, this document uses the SR-MPLS example.

### 8.1. Validity of an SR Policy

An SR Policy is invalid when all its candidate paths are invalid as described in Section 5 and Section 2.10.

By default, upon transitioning to the invalid state,

- o an SR Policy and its BSID are removed from the forwarding plane.
- o any steering of a service (PW), destination (BGP-VPN), flow or packet on the related SR policy is disabled and the related service, destination, flow or packet is routed per the classic forwarding table (e.g. longest-match to the destination or the recursing next-hop).

### 8.2. Drop upon invalid SR Policy

An SR Policy MAY be enabled for the Drop-Upon-Invalid behavior:

- o an invalid SR Policy and its BSID is kept in the forwarding plane with an action to drop.
- o any steering of a service (PW), destination (BGP-VPN), flow or packet on the related SR policy is maintained with the action to drop all of this traffic.

The drop-upon-invalid behavior has been deployed in use-cases where the operator wants some PW to only be transported on a path with specific constraints. When these constraints are no longer met, the operator wants the PW traffic to be dropped. Specifically, the operator does not want the PW to be routed according to the IGP shortest-path to the PW endpoint.

### 8.3. Incoming Active SID is a BSID

Let us assume that headend H has a valid SR Policy P of Segment-List <S1, S2, S3> and BSID B.

When H receives a packet K with label stack <B, L2, L3>, H pops B and pushes <S1, S2, S3> and forwards the resulting packet according to SID S1.

"Forwarding the resulting packet according to S1" means: If S1 is an Adj SID or a PHP-enabled prefix SID advertised by a neighbor, H sends the resulting packet with label stack <S2, S3, L2, L3> on the outgoing interface associated with S1; Else H sends the resulting packet with label stack <S1, S2, S3, L2, L3> along the path of S1.

H has steered the packet into the SR policy P.

H did not have to classify the packet. The classification was done by a node upstream of H (e.g., the source of the packet or an intermediate ingress edge node of the SR domain) and the result of this classification was efficiently encoded in the packet header as a BSID.

This is another key benefit of the segment routing in general and the binding SID in particular: the ability to encode a classification and the resulting steering in the packet header to better scale and simplify intermediate aggregation nodes.

If the SR Policy P is invalid, the BSID B is not in the forwarding plane and hence the packet K is dropped by H.

#### 8.4. Per-Destination Steering

Let us assume that headend H:

- o learns a BGP route R/r via next-hop N, extended-color community C and VPN label V.
- o has a valid SR Policy P to (color = C, endpoint = N) of Segment-List <S1, S2, S3> and BSID B.
- o has a BGP policy which matches on the extended-color community C and allows its usage as SLA steering information.

If all these conditions are met, H installs R/r in RIB/FIB with next-hop = SR Policy P of BSID B instead of via N.

Indeed, H's local BGP policy and the received BGP route indicate that the headend should associate R/r with an SR Policy path to endpoint N with the SLA associated with color C. The headend therefore installs the BGP route on that policy.

This can be implemented by using the BSID as a generalized next-hop and installing the BGP route on that generalized next-hop.

When H receives a packet K with a destination matching R/r, H pushes the label stack <S1, S2, S3, V> and sends the resulting packet along the path to S1.

Note that any SID associated with the BGP route is inserted after the Segment-List of the SR Policy (i.e., <S1, S2, S3, V>).

The same behavior is applicable to any type of service route: any AFI/SAFI of BGP [RFC4760] any AFI/SAFI of LISP [RFC6830].

In a BGP multi-path scenario, the BGP route may be resolved over a mix of paths that include those that are steered over SR Policies and others resolved via the normal BGP nexthop resolution. Implementations MAY provide options to prefer one type over the other or other forms of local policy to determine the paths that are selected.

#### 8.4.1. Multiple Colors

When a BGP route has multiple extended-color communities each with a valid SR Policy, the BGP process installs the route on the SR Policy giving preference to the color with the highest numerical value.

Let us assume that headend H:

- o learns a BGP route R/r via next-hop N, extended-color communities C1 and C2 and VPN label V.
- o has a valid SR Policy P1 to (color = C1, endpoint = N) of Segment-List <S1, S2, S3> and BSID B1.
- o has a valid SR Policy P2 to (color = C2, endpoint = N) of Segment-List <S4, S5, S6> and BSID B2.
- o has a BGP policy which matches on the extended-color communities C1 and C2 and allows their usage as SLA steering information

If all these conditions are met, H installs R/r in RIB/FIB with next-hop = SR Policy P2 of BSID=B2 (instead of N) because C2 > C1.

When the SR Policy with a specific color is not instantiated or in the down/inactive state, the SR Policy with the next highest numerical value of color is considered.

#### 8.5. Recursion on an on-demand dynamic BSID

In the previous section, it was assumed that H had a pre-established "explicit" SR Policy (color C, endpoint N).

In this section, independently to the a-priori existence of any explicit candidate path of the SR policy (C, N), it is to be noted that the BGP process at headend node H triggers the instantiation of a dynamic candidate path for the SR policy (C, N) as soon as:

- o the BGP process learns of a route R/r via N and with color C.

- o a local policy at node H authorizes the on-demand SR Policy path instantiation and maps the color to a dynamic SR Policy path optimization template.

#### 8.5.1. Multiple Colors

When a BGP route R/r via N has multiple extended-color communities Ci (with i=1 ... n), an individual on-demand SR Policy dynamic path request (color Ci, endpoint N) is triggered for each color Ci. The SR Policy that is used for steering is then determined as described in Section 8.4.1.

#### 8.6. Per-Flow Steering

Let us assume that headend H:

- o has a valid SR Policy P1 to (color = C1, endpoint = N) of Segment-List <S1, S2, S3> and BSID B1.
- o has a valid SR Policy P2 to (color = C2, endpoint = N) of Segment-List <S4, S5, S6> and BSID B2.
- o is configured to instantiate an array of paths to N where the entry 0 is the IGP path to N, color C1 is the first entry and Color C2 is the second entry. The index into the array is called a Forwarding Class (FC). The index can have values 0 to 7.
- o is configured to match flows in its ingress interfaces (upon any field such as Ethernet destination/source/vlan/tos or IP destination/source/DSCP or transport ports etc.) and color them with an internal per-packet forwarding-class variable (0, 1 or 2 in this example).

If all these conditions are met, H installs in RIB/FIB:

- o N via a recursion on an array A (instead of the immediate outgoing link associated with the IGP shortest-path to N).
- o Entry A(0) set to the immediate outgoing link of the IGP shortest-path to N.
- o Entry A(1) set to SR Policy P1 of BSID=B1.
- o Entry A(2) set to SR Policy P2 of BSID=B2.

H receives three packets K, K1 and K2 on its incoming interface. These three packets either longest-match on N or more likely on a BGP/service route which recurses on N. H colors these 3 packets respectively with forwarding-class 0, 1 and 2. As a result:

- o H forwards K along the shortest-path to N (which in SR-MPLS results in the pushing of the prefix-SID of N).
- o H pushes <S1, S2, S3> on packet K1 and forwards the resulting frame along the shortest-path to S1.



- o H pushes <S4, S5, S6> on packet K2 and forwards the resulting frame along the shortest-path to S4.

If the local configuration does not specify any explicit forwarding information for an entry of the array, then this entry is filled with the same information as entry 0 (i.e. the IGP shortest-path).

If the SR Policy mapped to an entry of the array becomes invalid, then this entry is filled with the same information as entry 0. When all the array entries have the same information as entry0, the forwarding entry for N is updated to bypass the array and point directly to its outgoing interface and next-hop.

The array index values (e.g. 0, 1 and 2) and the notion of forwarding-class are implementation specific and only meant to describe the desired behavior. The same can be realized by other mechanisms.

This realizes per-flow steering: different flows bound to the same BGP endpoint are steered on different IGP or SR Policy paths.

A headend MAY support options to apply per-flow steering only for traffic matching specific prefixes (e.g. specific IGP or BGP prefixes).

## 8.7. Policy-based Routing

Finally, headend H may be configured with a local routing policy which overrides any BGP/IGP path and steer a specified packet on an SR Policy. This includes the use of mechanisms like IGP Shortcut for automatic routing of IGP prefixes over SR Policies intended for such purpose.

## 8.8. Optional Steering Modes for BGP Destinations

### 8.8.1. Color-Only BGP Destination Steering

In the previous section, it is seen that the steering on an SR Policy is governed by the matching of the BGP route's next-hop N and the authorized color C with an SR Policy defined by the tuple (N, C).

This is the most likely form of BGP destination steering and the one recommended for most use-cases.

This section defines an alternative steering mechanism based only on the color.

This color-only steering variation is governed by two new "CO" bits defined in the color extended community in section 3 of [I-D.ietf-idr-segment-routing-te-policy].

The Color-Only flags "CO" are set to 00 by default.

When 00, the BGP destination is steered as follows:

```
IF there is a valid SR Policy (N, C) where N is the IPv4 or IPv6
    endpoint address and C is a color;
    Steer into SR Policy (N, C);
ELSE;
    Steer on the IGP path to the next-hop N.
```

This is the classic case described in this document previously and what is recommended in most scenarios.

When 01, the BGP destination is steered as follows:

```
IF there is a valid SR Policy (N, C) where N is the IPv4 or IPv6
    endpoint address and C is a color;
    Steer into SR Policy (N, C);
ELSE IF there is a valid SR Policy (null endpoint, C) of the
    same address-family of N;
    Steer into SR Policy (null endpoint, C);
ELSE IF there is any valid SR Policy
    (any address-family null endpoint, C);
    Steer into SR Policy (any null endpoint, C);
ELSE;
    Steer on the IGP path to the next-hop N.
```

When 10, the BGP destination is steered as follows:

```
IF there is a valid SR Policy (N, C) where N is an IPv4 or IPv6
    endpoint address and C is a color;
    Steer into SR Policy (N, C);
ELSE IF there is a valid SR Policy (null endpoint, C)
    of the same address-family of N;
    Steer into SR Policy (null endpoint, C);
ELSE IF there is any valid SR Policy
    (any address-family null endpoint, C);
    Steer into SR Policy (any null endpoint, C);
ELSE IF there is any valid SR Policy (any endpoint, C)
    of the same address-family of N;
    Steer into SR Policy (any endpoint, C);
ELSE IF there is any valid SR Policy
```

```
        (any address-family endpoint, C);  
        Steer into SR Policy (any address-family endpoint, C);  
    ELSE;  
        Steer on the IGP path to the next-hop N.
```

The null endpoint is 0.0.0.0 for IPv4 and ::0 for IPv6 (all bits set to the 0 value).

The value 11 is reserved for future use and SHOULD NOT be used. Upon reception, an implementations MUST treat it like 00.

#### 8.8.2. Multiple Colors and CO flags

The steering preference is first based on highest color value and then CO-dependent for the color. Assuming a Prefix via (NH, C1(CO=01), C2(CO=01)); C1>C2 The steering preference order is:

- o SR policy (NH, C1).
- o SR policy (null, C1).
- o SR policy (NH, C2).
- o SR policy (null, C2).
- o IGP to NH.

#### 8.8.3. Drop upon Invalid

This document defined earlier that when all the following conditions are met, H installs R/r in RIB/FIB with next-hop = SR Policy P of BSID B instead of via N.

- o H learns a BGP route R/r via next-hop N, extended-color community C and VPN label V.
- o H has a valid SR Policy P to (color = C, endpoint = N) of Segment-List <S1, S2, S3> and BSID B.
- o H has a BGP policy which matches on the extended-color community C and allows its usage as SLA steering information.

This behavior is extended by noting that the BGP policy may require the BGP steering to always stay on the SR policy whatever its validity.

This is the "drop upon invalid" option described in Section 8.2 applied to BGP-based steering.

### 9. Protection

### 9.1. Leveraging TI-LFA local protection of the constituent IGP segments

In any topology, Topology-Independent Loop Free Alternate (TI-LFA) [I-D.ietf-rtgwg-segment-routing-ti-lfa] provides a 50msec local protection technique for IGP SIDs. The backup path is computed on a per IGP SID basis along the post-convergence path.

In a network that has deployed TI-LFA, an SR Policy built on the basis of TI-LFA protected IGP segments leverages the local protection of the constituent segments. Since TI-LFA protection is based on IGP computation, there are cases where the path used during the fast-reroute time window may not meet the exact constraints of the SR Policy.

In a network that has deployed TI-LFA, an SR Policy instantiated only with non-protected Adj SIDs does not benefit from any local protection.

### 9.2. Using an SR Policy to locally protect a link

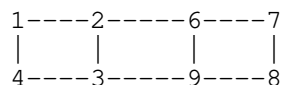


Figure 1: Local protection using SR Policy

An SR Policy can be instantiated at node 2 to protect the link 2to6. A typical explicit Segment-List would be <3, 9, 6>.

A typical use-case occurs for links outside an IGP domain: e.g. 1, 2, 3 and 4 are part of IGP/SR sub-domain 1 while 6, 7, 8 and 9 are part of IGP/SR sub-domain 2. In such a case, links 2to6 and 3to9 cannot benefit from TI-LFA automated local protection. The SR Policy with Segment-List <3, 9, 6> on node 2 can be locally configured to be a fast-reroute backup path for the link 2to6.

### 9.3. Using a Candidate Path for Path Protection

An SR Policy allows for multiple candidate paths, of which at any point in time there is a single active candidate path that is provisioned in the forwarding plane and used for traffic steering. However, another (lower preference) candidate path MAY be designated as the backup for a specific or all (active) candidate path(s). The following options are possible:

- o A pair of disjoint candidate paths are provisioned with one of them as primary and the other is identified as its backup.
- o A specific candidate path is provisioned as the backup for any (active) candidate path.
- o The headend picks the next (lower) preference valid candidate path as the backup for the active candidate path.

The headend MAY compute a-priori and validate such backup candidate paths as well as provision them into forwarding plane as backup for the active path. A fast re-route mechanism MAY then be used to trigger sub 50msec switchover from the active to the backup candidate path in the forwarding plane. Mechanisms like BFD MAY be used for fast detection of such failures.

#### 10. Security Considerations

This document specifies in detail the SR Policy construct introduced in [RFC8402] and its instantiation on a router supporting SR along with descriptions of mechanisms for steering of traffic flows over it. Therefore, the security considerations of [RFC8402] apply. This document does not define any new protocol extensions and does not introduce any further security considerations.

#### 11. IANA Considerations

This document requests IANA to create a new top-level registry called "Segment Routing Parameters". This registry is being defined to serve as a top-level registry for keeping all other Segment Routing sub-registries.

The document also requests creation of a new sub-registry called "Segment Types" to be defined under the top-level "Segment Routing Parameters" registry. This sub-registry maintains the alphabetic identifiers for the segment types (as specified in section 4) that may be used within a Segment List of an SR Policy. This sub-registry would follow the Specification Required allocation policy as specified in [RFC8126].

The initial registrations for this sub-registry are as follows:

Value	Description	Reference
A	SR-MPLS Label	[This.ID]
B	SRv6 SID	[This.ID]
C	IPv4 Prefix with optional SR Algorithm	[This.ID]
D	IPv6 Global Prefix with optional SR Algorithm for SR-MPLS	[This.ID]
E	IPv4 Prefix with Local Interface ID	[This.ID]
F	IPv4 Addresses for link endpoints as Local, Remote pair	[This.ID]
G	IPv6 Prefix and Interface ID for link endpoints as Local, Remote pair for SR-MPLS	[This.ID]
H	IPv6 Addresses for link endpoints as Local, Remote pair for SR-MPLS	[This.ID]
I	IPv6 Global Prefix with optional SR Algorithm for SRv6	[This.ID]
J	IPv6 Prefix and Interface ID for link endpoints as Local, Remote pair for SRv6	[This.ID]
K	IPv6 Addresses for link endpoints as Local, Remote pair for SRv6	[This.ID]

Table 2: Initial IANA Registration

### 11.1. Guidance for Designated Experts

The Designated Expert (DE) is expected to ascertain the existence of suitable documentation (a specification) as described in [RFC8126] and to verify that the document is permanently and publicly available. The DE is also expected to check the clarity of purpose and use of the requested assignment. Additionally, the DE must verify that any request for one of these assignments has been made available for review and comment within the IETF: the DE will post the request to the SPRING Working Group mailing list (or a successor mailing list designated by the IESG). If the request comes from within the IETF, it should be documented in an Internet-Draft. Lastly, the DE must ensure that any other request for a code point does not conflict with work that is active or already published within the IETF.

## 12. Acknowledgement

The authors would like to thank Tarek Saad, Dhanendra Jain, Ruediger Geib, Rob Shakir, Cheng Li and Dhruv Dhody for their review comments and suggestions.

### 13. Contributors

The following people have contributed to this document:

Siva Sivabalan  
Cisco Systems  
Email: msiva@cisco.com

Zafar Ali  
Cisco Systems  
Email: zali@cisco.com

Jose Liste  
Cisco Systems  
Email: jliste@cisco.com

Francois Clad  
Cisco Systems  
Email: fclad@cisco.com

Kamran Raza  
Cisco Systems  
Email: skraza@cisco.com

Mike Koldychev  
Cisco Systems  
Email: mkoldych@cisco.com

Shraddha Hegde  
Juniper Networks  
Email: shraddha@juniper.net

Steven Lin  
Google, Inc.  
Email: stevenlin@google.com

Przemyslaw Krol  
Google, Inc.  
Email: pkrol@google.com

Martin Horneffer  
Deutsche Telekom  
Email: martin.horneffer@telekom.de

Dirk Steinberg  
Steinberg Consulting  
Email: dws@steinbergnet.net

Bruno Decraene  
Orange Business Services  
Email: bruno.decraene@orange.com

Stephane Litkowski  
Orange Business Services  
Email: stephane.litkowski@orange.com

Luay Jalil  
Verizon  
Email: luay.jalil@verizon.com

## 14. References

### 14.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <<https://www.rfc-editor.org/info/rfc8402>>.

### 14.2. Informative References

- [I-D.ali-spring-sr-traffic-accounting] Filsfils, C., Talaulikar, K., Sivabalan, S., Horneffer, M., Raszuk, R., Litkowski, S., Voyer, D., and R. Morton, "Traffic Accounting in Segment Routing Networks", draft-ali-spring-sr-traffic-accounting-04 (work in progress), February 2020.



[I-D.anand-spring-poi-sr]

Anand, M., Bardhan, S., Subrahmaniam, R., Tantsura, J., Mukhopadhyaya, U., and C. Filsfils, "Packet-Optical Integration in Segment Routing", draft-anand-spring-poi-sr-08 (work in progress), July 2019.

[I-D.filsfils-spring-sr-policy-considerations]

Filsfils, C., Talaulikar, K., Krol, P., Horneffer, M., and P. Mattes, "SR Policy Implementation and Deployment Considerations", draft-filsfils-spring-sr-policy-considerations-06 (work in progress), October 2020.

[I-D.filsfils-spring-sr-traffic-counters]

Filsfils, C., Ali, Z., Horneffer, M., daniel.voyer@bell.ca, d., Durrani, M., and R. Raszuk, "Segment Routing Traffic Accounting Counters", draft-filsfils-spring-sr-traffic-counters-00 (work in progress), June 2018.

[I-D.ietf-idr-bgppls-segment-routing-epe]

Previdi, S., Talaulikar, K., Filsfils, C., Patel, K., Ray, S., and J. Dong, "BGP-LS extensions for Segment Routing BGP Egress Peer Engineering", draft-ietf-idr-bgppls-segment-routing-epe-19 (work in progress), May 2019.

[I-D.ietf-idr-segment-routing-te-policy]

Previdi, S., Filsfils, C., Talaulikar, K., Mattes, P., Rosen, E., Jain, D., and S. Lin, "Advertising Segment Routing Policies in BGP", draft-ietf-idr-segment-routing-te-policy-09 (work in progress), May 2020.

[I-D.ietf-idr-te-lsp-distribution]

Previdi, S., Talaulikar, K., Dong, J., Chen, M., Gredler, H., and J. Tantsura, "Distribution of Traffic Engineering (TE) Policies and State using BGP-LS", draft-ietf-idr-te-lsp-distribution-14 (work in progress), October 2020.

[I-D.ietf-lsr-flex-algo]

Psenak, P., Hegde, S., Filsfils, C., Talaulikar, K., and A. Gulko, "IGP Flexible Algorithm", draft-ietf-lsr-flex-algo-13 (work in progress), October 2020.

[I-D.ietf-pce-binding-label-sid]

Sivabalan, S., Filsfils, C., Tantsura, J., Hardwick, J., Prevdi, S., and C. Li, "Carrying Binding Label/Segment-ID in PCE-based Networks.", draft-ietf-pce-binding-label-sid-05 (work in progress), October 2020.

- [I-D.ietf-rtgwg-segment-routing-ti-lfa]  
Litkowski, S., Bashandy, A., Filsfils, C., Decraene, B.,  
Francois, P., Voyer, D., Clad, F., and P. Camarillo,  
"Topology Independent Fast Reroute using Segment Routing",  
draft-ietf-rtgwg-segment-routing-ti-lfa-04 (work in  
progress), August 2020.
- [I-D.ietf-spring-srv6-network-programming]  
Filsfils, C., Camarillo, P., Leddy, J., Voyer, D.,  
Matsushima, S., and Z. Li, "SRv6 Network Programming",  
draft-ietf-spring-srv6-network-programming-24 (work in  
progress), October 2020.
- [RFC1195] Callon, R., "Use of OSI IS-IS for routing in TCP/IP and  
dual environments", RFC 1195, DOI 10.17487/RFC1195,  
December 1990, <<https://www.rfc-editor.org/info/rfc1195>>.
- [RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328,  
DOI 10.17487/RFC2328, April 1998,  
<<https://www.rfc-editor.org/info/rfc2328>>.
- [RFC3630] Katz, D., Kompella, K., and D. Yeung, "Traffic Engineering  
(TE) Extensions to OSPF Version 2", RFC 3630,  
DOI 10.17487/RFC3630, September 2003,  
<<https://www.rfc-editor.org/info/rfc3630>>.
- [RFC4760] Bates, T., Chandra, R., Katz, D., and Y. Rekhter,  
"Multiprotocol Extensions for BGP-4", RFC 4760,  
DOI 10.17487/RFC4760, January 2007,  
<<https://www.rfc-editor.org/info/rfc4760>>.
- [RFC5305] Li, T. and H. Smit, "IS-IS Extensions for Traffic  
Engineering", RFC 5305, DOI 10.17487/RFC5305, October  
2008, <<https://www.rfc-editor.org/info/rfc5305>>.
- [RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF  
for IPv6", RFC 5340, DOI 10.17487/RFC5340, July 2008,  
<<https://www.rfc-editor.org/info/rfc5340>>.
- [RFC6830] Farinacci, D., Fuller, V., Meyer, D., and D. Lewis, "The  
Locator/ID Separation Protocol (LISP)", RFC 6830,  
DOI 10.17487/RFC6830, January 2013,  
<<https://www.rfc-editor.org/info/rfc6830>>.
- [RFC7471] Giacalone, S., Ward, D., Drake, J., Atlas, A., and S.  
 Previdi, "OSPF Traffic Engineering (TE) Metric  
Extensions", RFC 7471, DOI 10.17487/RFC7471, March 2015,  
<<https://www.rfc-editor.org/info/rfc7471>>.

- [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>>.
- [RFC8231] Crabbe, E., Minei, I., Medved, J., and R. Varga, "Path Computation Element Communication Protocol (PCEP) Extensions for Stateful PCE", RFC 8231, DOI 10.17487/RFC8231, September 2017, <<https://www.rfc-editor.org/info/rfc8231>>.
- [RFC8476] Tantsura, J., Chunduri, U., Aldrin, S., and P. Psenak, "Signaling Maximum SID Depth (MSD) Using OSPF", RFC 8476, DOI 10.17487/RFC8476, December 2018, <<https://www.rfc-editor.org/info/rfc8476>>.
- [RFC8491] Tantsura, J., Chunduri, U., Aldrin, S., and L. Ginsberg, "Signaling Maximum SID Depth (MSD) Using IS-IS", RFC 8491, DOI 10.17487/RFC8491, November 2018, <<https://www.rfc-editor.org/info/rfc8491>>.
- [RFC8570] Ginsberg, L., Ed., Previdi, S., Ed., Giacalone, S., Ward, D., Drake, J., and Q. Wu, "IS-IS Traffic Engineering (TE) Metric Extensions", RFC 8570, DOI 10.17487/RFC8570, March 2019, <<https://www.rfc-editor.org/info/rfc8570>>.
- [RFC8664] Sivabalan, S., Filsfils, C., Tantsura, J., Henderickx, W., and J. Hardwick, "Path Computation Element Communication Protocol (PCEP) Extensions for Segment Routing", RFC 8664, DOI 10.17487/RFC8664, December 2019, <<https://www.rfc-editor.org/info/rfc8664>>.
- [RFC8814] Tantsura, J., Chunduri, U., Talaulikar, K., Mirsky, G., and N. Triantafyllis, "Signaling Maximum SID Depth (MSD) Using the Border Gateway Protocol - Link State", RFC 8814, DOI 10.17487/RFC8814, August 2020, <<https://www.rfc-editor.org/info/rfc8814>>.

#### Authors' Addresses

Clarence Filsfils  
Cisco Systems, Inc.  
Pegasus Parc  
De kleetlaan 6a, DIEGEM BRABANT 1831  
BELGIUM

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

Ketan Talaulikar (editor)  
Cisco Systems, Inc.  
India

Email: ketant@cisco.com

Daniel Voyer  
Bell Canada  
671 de la gauchetiere W  
Montreal, Quebec H3B 2M8  
Canada

Email: daniel.voyer@bell.ca

Alex Bogdanov  
Google, Inc.

Email: bogdanov@google.com

Paul Mattes  
Microsoft  
One Microsoft Way  
Redmond, WA 98052-6399  
USA

Email: pamattes@microsoft.com

SPRING Working Group  
Internet-Draft  
Updates: 8402 (if approved)  
Intended status: Standards Track  
Expires: September 23, 2022

C. Filsfils  
K. Talaulikar, Ed.  
Cisco Systems, Inc.  
D. Voyer  
Bell Canada  
A. Bogdanov  
British Telecom  
P. Mattes  
Microsoft  
March 22, 2022

Segment Routing Policy Architecture  
draft-ietf-spring-segment-routing-policy-22

Abstract

Segment Routing (SR) allows a node to steer a packet flow along any path. Intermediate per-path states are eliminated thanks to source routing. SR Policy is an ordered list of segments (i.e., instructions) that represent a source-routed policy. Packet flows are steered into a SR Policy on a node where it is instantiated called a headend node. The packets steered into an SR Policy carry an ordered list of segments associated with that SR Policy.

This document updates RFC8402 as it details the concepts of SR Policy and steering into an SR Policy.

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 23, 2022.

## Copyright Notice

Copyright (c) 2022 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 . . . . .	3
1.1. Requirements Language . . . . .	4
2. SR Policy . . . . .	4
2.1. Identification of an SR Policy . . . . .	4
2.2. Candidate Path and Segment List . . . . .	5
2.3. Protocol-Origin of a Candidate Path . . . . .	6
2.4. Originator of a Candidate Path . . . . .	7
2.5. Discriminator of a Candidate Path . . . . .	7
2.6. Identification of a Candidate Path . . . . .	8
2.7. Preference of a Candidate Path . . . . .	8
2.8. Validity of a Candidate Path . . . . .	9
2.9. Active Candidate Path . . . . .	9
2.10. Validity of an SR Policy . . . . .	10
2.11. Instantiation of an SR Policy in the Forwarding Plane . .	10
2.12. Priority of an SR Policy . . . . .	11
2.13. Summary . . . . .	11
3. Segment Routing Database . . . . .	12
4. Segment Types . . . . .	13
4.1. Explicit Null . . . . .	17
5. Validity of a Candidate Path . . . . .	17
5.1. Explicit Candidate Path . . . . .	17
5.2. Dynamic Candidate Path . . . . .	19
5.3. Composite Candidate Path . . . . .	19
6. Binding SID . . . . .	20
6.1. BSID of a candidate path . . . . .	20
6.2. BSID of an SR Policy . . . . .	20
6.3. Forwarding Plane . . . . .	21
6.4. Non-SR usage of Binding SID . . . . .	22
7. SR Policy State . . . . .	22
8. Steering into an SR Policy . . . . .	22
8.1. Validity of an SR Policy . . . . .	23

8.2.	Drop upon invalid SR Policy . . . . .	23
8.3.	Incoming Active SID is a BSID . . . . .	23
8.4.	Per-Destination Steering . . . . .	24
8.5.	Recursion on an on-demand dynamic BSID . . . . .	26
8.6.	Per-Flow Steering . . . . .	26
8.7.	Policy-based Routing . . . . .	28
8.8.	Optional Steering Modes for BGP Destinations . . . . .	28
9.	Recovering from Network Failures . . . . .	30
9.1.	Leveraging TI-LFA local protection of the constituent IGP segments . . . . .	30
9.2.	Using an SR Policy to locally protect a link . . . . .	30
9.3.	Using a Candidate Path for Path Protection . . . . .	31
10.	Security Considerations . . . . .	31
11.	Manageability Considerations . . . . .	32
12.	IANA Considerations . . . . .	33
12.1.	Guidance for Designated Experts . . . . .	33
13.	Acknowledgement . . . . .	34
14.	Contributors . . . . .	34
15.	References . . . . .	35
15.1.	Normative References . . . . .	35
15.2.	Informative References . . . . .	36
	Authors' Addresses . . . . .	40

## 1. Introduction

Segment Routing (SR) [RFC8402] allows a node to steer a packet flow along any path. The headend is a node where the instructions for source routing (i.e., segments) are written into the packet and hence becomes the starting node for a specific segment routing path. Intermediate per-path states are eliminated thanks to source routing.

A Segment Routing Policy (SR Policy) [RFC8402] is an ordered list of segments (i.e., instructions) that represent a source-routed policy. The headend node is said to steer a flow into a SR Policy. The packets steered into an SR Policy have an ordered list of segments associated with that SR Policy written into them. [RFC8660] describes the representation and processing of this ordered list of segments as an MPLS label stack for SR-MPLS, while [RFC8754] and [RFC8986] describe the same for Segment Routing over IPv6 (SRv6) with the use of the Segment Routing Header (SRH).

[RFC8402] introduces the SR Policy construct and provides an overview of how it is leveraged for Segment Routing use-cases. This document updates [RFC8402] to specify detailed concepts of SR Policy and steering packets into an SR Policy. It applies equally to the SR-MPLS and SRv6 instantiations of segment routing.

### 1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

## 2. SR Policy

The general concept of SR Policy provides a framework that enables the instantiation of an ordered list of segments on a node for implementing a source routing policy for the steering of traffic for a specific purpose (e.g. for a specific SLA) from that node.

The Segment Routing architecture [RFC8402] specifies that any instruction can be bound to a segment. Thus, an SR Policy can be built using any type of Segment Identifier (SID) including those associated with topological or service instructions.

This section defines the key aspects and constituents of an SR Policy.

### 2.1. Identification of an SR Policy

An SR Policy MUST be identified through the tuple <headend, color, endpoint>. In the context of a specific headend, an SR policy MUST be identified by the <color, endpoint> tuple.

The headend is the node where the policy is instantiated/implemented. The headend is specified as an IPv4 or IPv6 address and MUST resolve to a unique node in the SR domain [RFC8402].

The endpoint indicates the destination of the policy. The endpoint is specified as an IPv4 or IPv6 address and SHOULD resolve to a unique node in the domain. In a specific case (refer to Section 8.8.1), the endpoint can be the unspecified address (0.0.0.0 for IPv4, :: for IPv6) and in this case, the destination of the policy is indicated by the last segment in the segment list(s).

The color is an unsigned non-zero 32-bit integer value that associates the SR Policy with an intent or objective (e.g. low-latency).

The endpoint and the color are used to automate the steering of service or transport routes on SR Policies (refer to Section 8).



An implementation MAY allow the assignment of a symbolic name comprising printable ASCII [RFC0020] characters (i.e., 0x20 to 0x7E) to an SR Policy to serve as a user-friendly attribute for debugging and troubleshooting purposes. Such symbolic names may identify an SR Policy when the naming scheme ensures uniqueness. The SR Policy name MAY also be signaled along with a candidate path of the SR Policy (refer to Section 2.2). An SR Policy MAY have multiple names associated with it in the scenario where the headend receives different SR Policy names along with different candidate paths for the same SR Policy via the same or different sources.

## 2.2. Candidate Path and Segment List

An SR Policy is associated with one or more candidate paths. A candidate path is the unit for signaling of an SR Policy to a headend via protocol extensions like Path Computation Element (PCE) Communication Protocol (PCEP) [RFC8664] [I-D.ietf-pce-segment-routing-policy-cp] or BGP SR Policy [I-D.ietf-idr-segment-routing-te-policy].

A Segment-List represents a specific source-routed path to send traffic from the headend to the endpoint of the corresponding SR policy.

A candidate path is either dynamic, explicit, or composite.

An explicit candidate path is expressed as a Segment-List or a set of Segment-Lists.

A dynamic candidate path expresses an optimization objective and a set of constraints for a specific data plane (i.e., SR-MPLS or SRv6). The headend (potentially with the help of a PCE) computes a solution Segment-List (or set of Segment-Lists) that solves the optimization problem.

If a candidate path is associated with a set of Segment-Lists, each Segment-List is associated with weight for weighted load balancing (refer to Section 2.11 for details). The default weight is 1.

A composite candidate path acts as a container for grouping SR Policies. The composite candidate path construct enables the combination of SR Policies, each with explicit candidate paths and/or dynamic candidate paths with potentially different optimization objectives and constraints, for load-balanced steering of packet flows over its constituent SR Policies. The following criteria apply for inclusion of constituent SR Policies using a composite candidate path under a parent SR Policy:

- o the endpoints of the constituent SR Policies and the parent SR Policy MUST be identical
- o The colors of each of the constituent SR Policies and the parent SR Policy MUST be different
- o the constituent SR Policies MUST NOT use composite candidate paths

Each constituent SR Policy of a composite candidate path is associated with weight for load-balancing purposes (refer to Section 2.11 for details). The default weight is 1.

The Section 2.13 illustrates an information model for hierarchical relationships between the SR Policy constructs described in this section.

### 2.3. Protocol-Origin of a Candidate Path

A headend may be informed about a candidate path for an SR Policy <color, endpoint> by various means including: via configuration, PCEP [RFC8664] [I-D.ietf-pce-segment-routing-policy-cp] or BGP [I-D.ietf-idr-segment-routing-te-policy].

Protocol-Origin of a candidate path is an 8-bit value associated with the mechanism or protocol used for signaling/provisioning the SR Policy. It helps identify the protocol/mechanism that provides or signals the candidate path and indicates its preference relative to other protocols/mechanisms.

The head-end assigns different Protocol-Origin values to each source of SR Policy information. The Protocol-Origin value is used as a tie-breaker between candidate paths of equal preference, as described in Section 2.9. The table below specifies the RECOMMENDED default values of Protocol-Origin:

Protocol-Origin	Description
10	PCEP
20	BGP SR Policy
30	Via Configuration

Table 1: Protocol-Origin default values

Note that the above order is to satisfy the need for having a clear ordering and implementations MAY allow modifications of these default values assigned to protocols on the headend along similar lines as a

routing administrative distance. Its application in the candidate path selection is described in Section 2.9.

#### 2.4. Originator of a Candidate Path

Originator identifies the node which provisioned or signaled the candidate path on the headend. The originator is expressed in the form of a 160-bit numerical value formed by the concatenation of the fields of the tuple <Autonomous System Number (ASN), node-address> as below:

- o Autonomous System Number (ASN) : represented as a 4-byte number. If 2-byte ASNs are in use, the low-order 16 bits MUST be used, and the high-order bits MUST be set to zero.
- o Node Address : represented as a 128-bit value. IPv4 addresses MUST be encoded in the lowest 32 bits, and the high-order bits MUST be set to zero.

Its application in the candidate path selection is described in Section 2.9.

When provisioning is via configuration, the ASN and node address MAY be set to either the headend or the provisioning controller/node ASN and address. The default value is 0 for both AS and node address.

When signaling is via PCEP, it is the IPv4 or IPv6 address of the PCE and the AS number is expected to be set to 0 by default when not available or known.

When signaling is via BGP SR Policy, the ASN and Node Address are provided by BGP (refer to [I-D.ietf-idr-segment-routing-te-policy]) on the headend.

#### 2.5. Discriminator of a Candidate Path

The Discriminator is a 32-bit value associated with a candidate path that uniquely identifies it within the context of an SR Policy from a specific Protocol-Origin as specified below:

- o When provisioning is via configuration, this is an implementation's configuration-model-specific unique identifier for a candidate path. The default value is 0.
- o When signaling is via PCEP, the method to uniquely signal an individual candidate path along with its discriminator is described in [I-D.ietf-pce-segment-routing-policy-cp]. The default value is 0.

- o When signaling is via BGP SR Policy, the BGP process receiving the route provides the distinguisher (refer to Section 2.1 of [I-D.ietf-idr-segment-routing-te-policy]) as the discriminator. Note that the BGP best path selection is applied before the route is supplied as a candidate path, so only a single candidate path for a given SR Policy will be seen for a given discriminator.

Its application in the candidate path selection is described in Section 2.9.

## 2.6. Identification of a Candidate Path

A candidate path is identified in the context of a single SR Policy.

A candidate path is not shared across SR Policies.

A candidate path is not identified by its Segment-List(s).

If CP1 is a candidate path of SR Policy Pol1 and CP2 is a candidate path of SR Policy Pol2, then these two candidate paths are independent, even if they happen to have the same Segment-List. The Segment-List does not identify a candidate path. The Segment-List is an attribute of a candidate path.

The identity of a candidate path MUST be uniquely established in the context of an SR Policy <headend, color, endpoint> to handle add, delete or modify operations on them in an unambiguous manner regardless of their source(s).

The tuple <Protocol-Origin, originator, discriminator> uniquely identifies a candidate path.

Candidate paths MAY also be assigned or signaled with a symbolic name comprising printable ASCII [RFC0020] characters (i.e., 0x20 to 0x7E) to serve as a user-friendly attribute for debugging and troubleshooting purposes. Such symbolic names MUST NOT be considered as identifiers for a candidate path. The signaling of the candidate path name via BGP and PCEP is described in [I-D.ietf-pce-segment-routing-policy-cp] and [I-D.ietf-idr-segment-routing-te-policy] respectively.

## 2.7. Preference of a Candidate Path

The preference of the candidate path is used to select the best candidate path for an SR Policy. It is a 32-bit value where a higher value indicates higher preference and the default preference value is 100.

It is RECOMMENDED that each candidate path of a given SR policy has a different preference.

The signaling of the candidate path preference via BGP and PCEP is described in [I-D.ietf-pce-segment-routing-policy-cp] and [I-D.ietf-idr-segment-routing-te-policy] respectively.

## 2.8. Validity of a Candidate Path

A candidate path is usable when it is valid. The RECOMMENDED candidate path validity criterion is the validity of at least one of its constituent Segment-Lists. The validation rules are specified in Section 5.

## 2.9. Active Candidate Path

A candidate path is selected when it is valid and it is determined to be the best path of the SR Policy. The selected path is referred to as the "active path" of the SR policy in this document.

Whenever a new path is learned or an active path is deleted, the validity of an existing path changes or an existing path is changed, the selection process MUST be re-executed.

The candidate path selection process operates primarily on the candidate path Preference. A candidate path is selected when it is valid and it has the highest preference value among all the valid candidate paths of the SR Policy.

In the case of multiple valid candidate paths of the same preference, the tie-breaking rules are evaluated on the identification tuple in the following order until only one valid best path is selected:

1. Higher value of Protocol-Origin is selected.
2. If specified by configuration, prefer the existing installed path.
3. Lower value of originator is selected.
4. Finally, the higher value of discriminator is selected.

The rules are framed with multiple protocols and sources in mind and hence may not follow the logic of a single protocol (e.g. BGP best path selection). The motivation behind these rules are as follows:

- o The preference, being the primary criterion, allows an operator to influence selection across paths thus allowing provisioning of

multiple path options, e.g., CP1 is preferred and if it becomes invalid then fallback to CP2 and so on. Since preference works across protocol sources, it also enables (where necessary) selective override of the default Protocol-Origin preference, e.g., to prefer a path signaled via BGP SR Policy over what is configured.

- o The Protocol-Origin allows an operator to set up a default selection mechanism across protocol sources, e.g., to prefer configured over paths signaled via BGP SR Policy or PCEP.
- o The originator allows an operator to have multiple redundant controllers and still maintain a deterministic behavior over which of them are preferred even if they are providing the same candidate paths for the same SR policies to the headend.
- o The discriminator performs the final tiebreaking step to ensure a deterministic outcome of selection regardless of the order in which candidate paths are signaled across multiple transport channels or sessions.

Section 4 of [I-D.filsfils-spring-sr-policy-considerations] provides a set of examples to illustrate the active candidate path selection rules.

#### 2.10. Validity of an SR Policy

An SR Policy is valid when it has at least one valid candidate path.

#### 2.11. Instantiation of an SR Policy in the Forwarding Plane

Generally, only valid SR policies are instantiated in the forwarding plane.

Only the active candidate path MUST be used for forwarding traffic that is being steered onto that policy except for certain scenarios such as fast-reroute where a backup candidate path may be used as described in Section 9.3.

If a set of Segment-Lists is associated with the active path of the policy, then the steering is per-flow and weighted-ECMP (W-ECMP) based according to the relative weight of each Segment-List.

The fraction of the flows associated with a given Segment-List is  $w/S_w$ , where  $w$  is the weight of the Segment-List and  $S_w$  is the sum of the weights of the Segment-Lists of the selected path of the SR Policy.

When a composite candidate path is active, the fraction of flows steered into each constituent SR Policy is equal to the relative weight of each constituent SR Policy. Further load balancing of flows steered into a constituent SR Policy is performed based on the weights of the Segment-List of the active candidate path of that constituent SR Policy.

The accuracy of the weighted load-balancing depends on the platform implementation.

## 2.12. Priority of an SR Policy

Upon topological change, many policies could be recomputed or revalidated. An implementation MAY provide a per-policy priority configuration. The operator may set this field to indicate the order in which the policies should be re-computed. Such a priority is represented by an integer in the range (0, 255) where the lowest value is the highest priority. The default value of priority is 128.

An SR Policy may comprise multiple Candidate Paths received from the same or different sources. A candidate path MAY be signaled with a priority value. When an SR Policy has multiple candidate paths with distinct signaled non-default priority values and the SR Policy itself does not have a priority value configured, the SR Policy as a whole takes the lowest value (i.e., the highest priority) amongst these signaled priority values.

## 2.13. Summary

In summary, the information model is the following:

```
SR policy POL1 <headend = H1, color = 1, endpoint = E1>
  Candidate-path CP1 <protocol-origin = 20, originator =
64511:192.0.2.1, discriminator = 1>
    Preference 200
    Priority 10
    Segment List 1 <SID11...SID1i>, Weight W1
    Segment List 2 <SID21...SID2j>, Weight W2
  Candidate-path CP2 <protocol-origin = 20, originator =
64511:192.0.2.2, discriminator = 2>
    Preference 100
    Priority 10
    Segment List 3 <SID31...SID3i>, Weight W3
    Segment List 4 <SID41...SID4j>, Weight W4
```

The SR Policy POL1 is identified by the tuple <headend, color, endpoint>. It has two candidate paths CP1 and CP2. Each is identified by a tuple <protocol-origin, originator, discriminator>

within the scope of POL1. CP1 is the active candidate path (it is valid and has the highest preference). The two Segment-Lists of CP1 are installed as the forwarding instantiation of SR policy POL1. Traffic steered on POL1 is flow-based hashed on Segment-List <SID11...SID1i> with a ratio  $W1/(W1+W2)$ .

The information model of SR Policy POL100 having a composite candidate path is the following:

```
SR policy POL100 <headend = H1, color = 100, endpoint = E1>
  Candidate-path CP1 <protocol-origin = 20, originator =
64511:192.0.2.1, discriminator = 1>
    Preference 200
    SR policy <color = 1>, Weight W1
    SR policy <color = 2>, Weight W2
```

The constituent SR Policies POL1 and POL2 have an information model as described at the start of this section. They are referenced only by color in the composite candidate path since their headend and endpoint are identical to the POL100. The valid Segment-Lists of the active candidate path of POL1 and POL2 are installed in the forwarding. Traffic steered on POL100 is flow-based hashed on POL1 with a proportion  $W1/(W1+W2)$ . Within the POL1, the flow-based hashing over its Segment-Lists are performed as described earlier in this section.

### 3. Segment Routing Database

An SR Policy computation node (e.g. headend or controller) maintains the Segment Routing Database (SR-DB). The SR-DB is a conceptual database to illustrate the various pieces of information and their sources that may help in SR Policy computation and validation. There is no specific requirement for an implementation to create a new database as such.

An SR headend leverages the SR-DB to validate explicit candidate paths and compute dynamic candidate paths.

The information in the SR-DB may include:

- o IGP information (topology, IGP metrics based on IS-IS [RFC1195] and OSPF [RFC2328] [RFC5340])
- o Segment Routing information (such as Segment Routing Global Block, Segment Routing Local Block, Prefix-SIDs, Adj-SIDs, BGP Peering SID, SRv6 SIDs) [RFC8402] [RFC8986]
- o TE Link Attributes (such as TE metric, Shared Risk Link Groups, attribute-flag, extended admin group) [RFC5305] [RFC3630] [RFC5329].



- o Extended TE Link attributes (such as latency, loss) [RFC8570]  
[RFC7471]
- o Inter-AS Topology information [RFC9086].

The attached domain topology may be learned via protocol/mechanisms such as IGP, BGP-LS or NETCONF.

A non-attached (remote) domain topology may be learned via protocol/mechanisms such as BGP-LS or NETCONF.

In some use-cases, the SR-DB may only contain the attached domain topology while in others, the SR-DB may contain the topology of multiple domains and in this case, it is multi-domain capable.

The SR-DB may also contain the SR Policies instantiated in the network. This can be collected via BGP-LS [I-D.ietf-idr-te-lsp-distribution] or PCEP [RFC8231], [I-D.ietf-pce-segment-routing-policy-cp], and [I-D.ietf-pce-binding-label-sid]. This information allows to build an end-to-end policy on the basis of intermediate SR policies (see Section 6 for further details).

The SR-DB may also contain the Maximum SID Depth (MSD) capability of nodes in the topology. This can be collected via IS-IS [RFC8491], OSPF [RFC8476], BGP-LS [RFC8814] or PCEP [RFC8664].

The use of the SR-DB for path computation and for the validation of optimization objective and constraints of paths is outside the scope of this document. Some implementation aspects related to path computation are covered in [I-D.filsfils-spring-sr-policy-considerations].

#### 4. Segment Types

A Segment-List is an ordered set of segments represented as <S1, S2, ... Sn> where S1 is the first segment.

Based on the desired dataplane, either the MPLS label stack or the SRv6 Segment Routing Header [RFC8754] is built from the Segment-List. However, the Segment-List itself can be specified using different segment-descriptor types and the following are currently defined:

##### Type A: SR-MPLS Label:

An MPLS label corresponding to any of the segment types defined for SR-MPLS (as defined in [RFC8402] or other SR-MPLS specifications) can be used. Additionally, special purpose labels like explicit-null or in general any MPLS label MAY also be used. E.g. this type can be used to specify a label

representation that maps to an optical transport path on a packet transport node.

Type B: SRv6 SID:

An IPv6 address corresponding to any of the SID behaviors for SRv6 (as defined in [RFC8986] or other SRv6 specifications) can be used. Optionally, the SRv6 SID behavior (as defined in [RFC8986] or other SRv6 specifications) and structure (as defined in [RFC8986]) MAY also be provided for the headend to perform validation of the SID when using it for building the Segment List.

Type C: IPv4 Prefix with optional SR Algorithm:

In this case, the headend is required to resolve the specified IPv4 Prefix Address to the SR-MPLS label corresponding to its Prefix SID segment (as defined in [RFC8402]). The SR algorithm (refer to Section 3.1.1 of [RFC8402]) to be used MAY also be provided.

Type D: IPv6 Global Prefix with optional SR Algorithm for SR-MPLS:

In this case, the headend is required to resolve the specified IPv6 Global Prefix Address to the SR-MPLS label corresponding to its Prefix SID segment (as defined in [RFC8402]). The SR Algorithm (refer to Section 3.1.1 of [RFC8402]) to be used MAY also be provided.

Type E: IPv4 Prefix with Local Interface ID:

This type allows identification of Adjacency SID or BGP Peer Adjacency SID (as defined in [RFC8402]) SR-MPLS label for point-to-point links including IP unnumbered links. The headend is required to resolve the specified IPv4 Prefix Address to the Node originating it and then use the Local Interface ID to identify the point-to-point link whose adjacency is being referred to. The Local Interface ID link descriptor follows semantics as specified in [RFC5307]. This type can also be used to indicate indirection into a layer 2 interface (i.e., without IP address) like a representation of an optical transport path or a layer 2 Ethernet port or circuit at the specified node.

Type F: IPv4 Addresses for link endpoints as Local, Remote pair:

This type allows identification of Adjacency SID or BGP Peer Adjacency SID (as defined in [RFC8402]) SR-MPLS label for links. The headend is required to resolve the specified IPv4 Local Address to the Node originating it and then use the IPv4 Remote Address to identify the link adjacency being referred to. The Local and Remote Address pair link descriptors follow semantics as specified in [RFC7752].

Type G: IPv6 Prefix and Interface ID for link endpoints as Local, Remote pair for SR-MPLS:

This type allows identification of Adjacency SID or BGP Peer Adjacency SID (as defined in [RFC8402]) label for links including those with only Link-Local IPv6 addresses. The headend is required to resolve the specified IPv6 Prefix Address to the Node originating it and then use the Local Interface ID to identify the point-to-point link whose adjacency is being referred to. For other than point-to-point links, additionally the specific adjacency over the link needs to be resolved using the Remote Prefix and Interface ID. The Local and Remote pair of Prefix and Interface ID link descriptor follows semantics as specified in [RFC7752]. This type can also be used to indicate indirection into a layer 2 interface (i.e., without IP address) like a representation of an optical transport path or a layer 2 Ethernet port or circuit at the specified node.

Type H: IPv6 Addresses for link endpoints as Local, Remote pair for SR-MPLS:

This type allows identification of Adjacency SID or BGP Peer Adjacency SID (as defined in [RFC8402]) label for links with Global IPv6 addresses. The headend is required to resolve the specified Local IPv6 Address to the Node originating it and then use the Remote IPv6 Address to identify the link adjacency being referred to. The Local and Remote Address pair link descriptors follow semantics as specified in [RFC7752].

Type I: IPv6 Global Prefix with optional SR Algorithm for SRv6:

The headend is required to resolve the specified IPv6 Global Prefix Address to an SRv6 SID corresponding to a Prefix SID segment (as defined in [RFC8402]), such as a SID associated with the End behavior (as defined in [RFC8986]) of the node which is originating the prefix. The SR Algorithm (refer to Section 3.1.1 of [RFC8402]), the SRv6 SID behavior (as defined in [RFC8986] or other SRv6 specifications) and structure (as defined in [RFC8986]) MAY also be provided.

Type J: IPv6 Prefix and Interface ID for link endpoints as Local, Remote pair for SRv6:

This type allows identification of an SRv6 SID corresponding to an Adjacency SID or BGP Peer Adjacency SID (as defined in [RFC8402]), such as a SID associated with the End.X behavior (as defined in [RFC8986]) associated with link or adjacency with only Link-Local IPv6 addresses. The headend is required to resolve the specified IPv6 Prefix Address to the Node originating it and then use the Local Interface ID to identify the point-to-point link whose adjacency is being referred to.

For other than point-to-point links, additionally the specific adjacency needs to be resolved using the Remote Prefix and Interface ID. The Local and Remote pair of Prefix and Interface ID link descriptor follows semantics as specified in [RFC7752]. The SR Algorithm (refer to Section 3.1.1 of [RFC8402]), the SRv6 SID behavior (as defined in [RFC8986] or other SRv6 specifications) and structure (as defined in [RFC8986]) MAY also be provided.

Type K: IPv6 Addresses for link endpoints as Local, Remote pair for SRv6:

This type allows identification of an SRv6 SID corresponding to an Adjacency SID or BGP Peer Adjacency SID (as defined in [RFC8402]), such as a SID associated with the End.X behavior (as defined in [RFC8986]) associated with link or adjacency with Global IPv6 addresses. The headend is required to resolve the specified Local IPv6 Address to the Node originating it and then use the Remote IPv6 Address to identify the link adjacency being referred to. The Local and Remote Address pair link descriptors follow semantics as specified in [RFC7752]. The SR Algorithm (refer to Section 3.1.1 of [RFC8402]), the SRv6 SID behavior (as defined in [RFC8986] or other SRv6 specifications) and structure (as defined in [RFC8986]) MAY also be provided.

When the algorithm is not specified for the SID types above which optionally allow for it, the headend SHOULD use the Strict Shortest Path algorithm if available and otherwise, it SHOULD use the default Shortest Path algorithm. The specification of the algorithm enables the use of the IGP Flex Algorithm [I-D.ietf-lsr-flex-algo] specific SIDs in SR Policy.

For SID types C-through-K, a SID value MAY also be optionally provided to the headend for verification purposes. Section 5.1. describes the resolution and verification of the SIDs and Segment Lists on the headend.

When building the MPLS label stack or the IPv6 Segment list from the Segment List, the node instantiating the policy MUST interpret the set of Segments as follows:

- o The first Segment represents the topmost label or the first IPv6 segment. It identifies the active segment the traffic will be directed toward along the explicit SR path.
- o The last Segment represents the bottommost label or the last IPv6 segment the traffic will be directed toward along the explicit SR path.

#### 4.1. Explicit Null

A Type A SID MAY be any MPLS label, including special purpose labels.

For example, assuming that the desired traffic-engineered path from a headend 1 to an endpoint 4 can be expressed by the Segment-List <16002, 16003, 16004> where 16002, 16003 and 16004 respectively refer to the IPv4 Prefix SIDs bound to nodes 2, 3, and 4, then IPv6 traffic can be traffic-engineered from nodes 1 to 4 via the previously described path using an SR Policy with Segment-List <16002, 16003, 16004, 2> where the MPLS label value of 2 represents the "IPv6 Explicit NULL Label".

The penultimate node before node 4 will pop 16004 and will forward the frame on its directly connected interface to node 4.

The endpoint receives the traffic with the top label "2" which indicates that the payload is an IPv6 packet.

When steering unlabeled IPv6 BGP destination traffic using an SR policy composed of Segment-List(s) based on IPv4 SIDs, the Explicit Null Label Policy is processed as specified in [I-D.ietf-idr-segment-routing-te-policy] Section 2.4.5. When an "IPv6 Explicit NULL label" is not present as the bottom label, the headend SHOULD automatically impose one. Refer to Section 8 for more details.

#### 5. Validity of a Candidate Path

##### 5.1. Explicit Candidate Path

An explicit candidate path is associated with a Segment-List or a set of Segment-Lists.

An explicit candidate path is provisioned by the operator directly or via a controller.

The computation/logic that leads to the choice of the Segment-List is external to the SR Policy headend. The SR Policy headend does not compute the Segment-List. The SR Policy headend only confirms its validity.

An explicit candidate path MAY consist of a single explicit Segment-List containing only an implicit-null label to indicate pop-and-forward behavior. The Binding SID (BSID) is popped and the traffic is forwarded based on the inner label or an IP lookup in the case of unlabeled IP packets. Such an explicit path can serve as a fallback

or path of last resort for traffic being steered into an SR Policy using its BSID (refer to Section 8.3).

A Segment-List of an explicit candidate path MUST be declared invalid when any of the following is true:

- o It is empty.
- o Its weight is 0.
- o It comprises a mix of SR-MPLS and SRv6 segment types.
- o The headend is unable to perform path resolution for the first SID into one or more outgoing interface(s) and next-hop(s).
- o The headend is unable to perform SID resolution for any non-first SID of type C-through-K into an MPLS label or an SRv6 SID.
- o The headend verification fails for any SID for which verification has been explicitly requested.

"Unable to perform path resolution" means that the headend has no path to the SID in its SR database.

SID verification is performed when the headend is explicitly requested to verify SID(s) by the controller via the signaling protocol used. Implementations MAY provide a local configuration option to enable verification on a global or per policy or per candidate path basis.

"Verification fails" for a SID means any of the following:

- o The headend is unable to find the SID in its SR-DB
- o The headend detects a mismatch between the SID value provided and the SID value resolved by context provided for SIDs of type C-through-K in its SR-DB.
- o The headend is unable to perform SID resolution for any non-first SID of type C-through-K into an MPLS label or an SRv6 SID.

In multi-domain deployments, it is expected that the headend may be unable to verify the reachability of the SIDs in remote domains. Types A or B MUST be used for the SIDs for which the reachability cannot be verified. Note that the first SID MUST always be reachable regardless of its type.

Additionally, a Segment-List MAY be declared invalid when both of the conditions below are met :

- o Its last segment is not a Prefix SID (including BGP Peer Node-SID) advertised by the node specified as the endpoint of the corresponding SR policy.
- o Its last segment is not an Adjacency SID (including BGP Peer Adjacency SID) of any of the links present on neighbor nodes and

that terminate on the node specified as the endpoint of the corresponding SR policy.

An explicit candidate path is invalid as soon as it has no valid Segment-List.

Additionally, an explicit candidate path MAY be declared invalid when its constituent segment lists (valid or invalid) are using segment types of different SR data planes.

## 5.2. Dynamic Candidate Path

A dynamic candidate path is specified as an optimization objective and constraints.

The headend of the policy leverages its SR database to compute a Segment-List ("solution Segment-List") that solves this optimization problem for either the SR-MPLS or the SRv6 data-plane as specified.

The headend re-computes the solution Segment-List any time the inputs to the problem change (e.g., topology changes).

When the local computation is not possible (e.g., a policy's tail-end is outside the topology known to the headend) or not desired, the headend may rely on an external entity. For example, a path computation request may be sent to a PCE supporting PCEP extensions specified in [RFC8664].

If no solution is found to the optimization objective and constraints, then the dynamic candidate path MUST be declared invalid.

Section 3 of [I-D.filsfils-spring-sr-policy-considerations] discusses some of the optimization objectives and constraints that may be considered by a dynamic candidate path. It illustrates some of the desirable properties of the computation of the solution Segment-List.

## 5.3. Composite Candidate Path

A composite candidate path is specified as a group of its constituent SR Policies.

A composite candidate path is valid when it has at least one valid constituent SR Policy.

## 6. Binding SID

The Binding SID (BSID) is fundamental to Segment Routing [RFC8402]. It provides scaling, network opacity, and service independence. Section 6 of [I-D.filsfils-spring-sr-policy-considerations] illustrates some of these benefits. This section describes the association of BSID with an SR Policy.

### 6.1. BSID of a candidate path

Each candidate path MAY be defined with a BSID.

Candidate Paths of the same SR policy SHOULD have the same BSID.

Candidate Paths of different SR policies MUST NOT have the same BSID.

### 6.2. BSID of an SR Policy

The BSID of an SR Policy is the BSID of its active candidate path.

When the active candidate path has a specified BSID, the SR Policy uses that BSID if this value (label in MPLS, IPv6 address in SRv6) is available (i.e., not associated with any other usage: e.g. label used by some other MPLS forwarding entry, SRv6 SID used in some other context, to another SID, to another SR Policy, outside the range of SRv6 Locators).

In the case of SR-MPLS, SRv6 BSIDs (e.g. with the behavior End.BM [RFC8986]) MAY be associated with the SR Policy in addition to the MPLS BSID. In the case of SRv6, multiple SRv6 BSIDs (e.g. with different behaviors like End.B6.Encap and End.B6.Encap.Red [RFC8986]) MAY be associated with the SR Policy.

Optionally, instead of only checking that the BSID of the active path is available, a headend MAY check that it is available within the given SID range i.e., Segment Routing Local Block (SRLB) as specified in [RFC8402].

When the specified BSID is not available (optionally is not in the SRLB), an alert message MUST be generated via mechanisms like syslog.

In the cases (as described above) where SR Policy does not have a BSID available, then the SR Policy MAY dynamically bind a BSID to itself. Dynamically bound BSID SHOULD use an available SID outside the SRLB.

Assuming that at time  $t$  the BSID of the SR Policy is  $B_1$ , if at time  $t+dt$  a different candidate path becomes active and this new active



path does not have a specified BSID or its BSID is specified but is not available (e.g. it is in use by something else), then the SR Policy MAY keep the previous BSID B1.

The association of an SR Policy with a BSID thus MAY change over the life of the SR Policy (e.g., upon active path change). Hence, the BSID SHOULD NOT be used as an identification of an SR Policy.

#### 6.2.1. Frequent use-case : unspecified BSID

All the candidate paths of the same SR Policy can have an unspecified BSID.

In such a case, a BSID MAY be dynamically bound to the SR Policy as soon as the first valid candidate path is received. That BSID is kept through the life of the SR Policy and across changes of active candidate path.

#### 6.2.2. Frequent use-case: all specified to the same BSID

All the paths of the SR Policy can have the same specified BSID.

#### 6.2.3. Specified-BSID-only

An implementation MAY support the configuration of the Specified-BSID-only restrictive behavior on the headend for all SR Policies or individual SR Policies. Further, this restrictive behavior MAY also be signaled on a per SR Policy basis to the headend.

When this restrictive behavior is enabled, if the candidate path has an unspecified BSID or if the specified BSID is not available when the candidate path becomes active then no BSID is bound to it and the candidate path is considered invalid. An alert MUST be triggered for this error via mechanisms like syslog. Other candidate paths MUST then be evaluated for becoming the active candidate path.

### 6.3. Forwarding Plane

A valid SR Policy results in the installation of a BSID-keyed entry in the forwarding plane with the action of steering the packets matching this entry to the selected path of the SR Policy.

If the Specified-BSID-only restrictive behavior is enabled and the BSID of the active path is not available (optionally not in the SRLB), then the SR Policy does not install any entry indexed by a BSID in the forwarding plane.

#### 6.4. Non-SR usage of Binding SID

An implementation MAY choose to associate a Binding SID with any type of interface (e.g. a layer 3 termination of an Optical Circuit) or a tunnel (e.g. IP tunnel, GRE tunnel, IP/UDP tunnel, MPLS RSVP-TE tunnel, etc). This enables the use of other non-SR enabled interfaces and tunnels as segments in an SR Policy Segment-List without the need of forming routing protocol adjacencies over them.

The details of this kind of usage are beyond the scope of this document. A specific packet-optical integration use case is described in [I-D.anand-spring-poi-sr].

#### 7. SR Policy State

The SR Policy State is maintained on the headend to represent the state of the policy and its candidate paths. This is to provide an accurate representation of whether the SR Policy is being instantiated in the forwarding plane and which of its candidate paths and segment-list(s) are active. The SR Policy state MUST also reflect the reason when a policy and/or its candidate path is not active due to validation errors or not being preferred. The operational state information reported for SR Policies are specified in [I-D.ietf-spring-sr-policy-yang].

The SR Policy state can be reported by the headend node via BGP-LS [I-D.ietf-idr-te-lsp-distribution] or PCEP [RFC8231] and [I-D.ietf-pce-binding-label-sid].

SR Policy state on the headend also includes traffic accounting information for the flows being steered via the policies. The details of the SR Policy accounting are beyond the scope of this document. The aspects related to the SR traffic counters and their usage in the broader context of traffic accounting in an SR network are covered in [I-D.filsfils-spring-sr-traffic-counters] and [I-D.ali-spring-sr-traffic-accounting] respectively.

Implementations MAY support an administrative state to control locally provisioned policies via mechanisms like CLI or NETCONF.

#### 8. Steering into an SR Policy

A headend can steer a packet flow into a valid SR Policy in various ways:

- o Incoming packets have an active SID matching a local BSID at the headend.

- o Per-destination Steering: incoming packets match a BGP/Service route which recurses on an SR policy.
- o Per-flow Steering: incoming packets match or recurse on a forwarding array of which some of the entries are SR Policies.
- o Policy-based Steering: incoming packets match a routing policy that directs them on an SR policy.

### 8.1. Validity of an SR Policy

An SR Policy is invalid when all its candidate paths are invalid as described in Section 5 and Section 2.10.

By default, upon transitioning to the invalid state,

- o an SR Policy and its BSID are removed from the forwarding plane.
- o any steering of a service (Pseudowire (PW)), destination (BGP-VPN), flow or packet on the related SR policy is disabled and the related service, destination, flow, or packet is routed per the classic forwarding table (e.g. longest-match to the destination or the recursing next-hop).

### 8.2. Drop upon invalid SR Policy

An SR Policy MAY be enabled for the Drop-Upon-Invalid behavior:

- o an invalid SR Policy and its BSID is kept in the forwarding plane with an action to drop.
- o any steering of a service (PW), destination (BGP-VPN), flow or packet on the related SR policy is maintained with the action to drop all of this traffic.

The drop-upon-invalid behavior has been deployed in use-cases where the operator wants some PW to only be transported on a path with specific constraints. When these constraints are no longer met, the operator wants the PW traffic to be dropped. Specifically, the operator does not want the PW to be routed according to the IGP shortest path to the PW endpoint.

### 8.3. Incoming Active SID is a BSID

Let us assume that headend H has a valid SR Policy P of Segment-List <S1, S2, S3> and BSID B.

In the case of SR-MPLS, when H receives a packet K with label stack <B, L2, L3>, H pops B and pushes <S1, S2, S3> and forwards the resulting packet according to SID S1.

"Forwarding the resulting packet according to S1" means: If S1 is an Adj-SID or a PHP-enabled prefix SID advertised by a neighbor, H sends the resulting packet with label stack <S2, S3, L2, L3> on the outgoing interface associated with S1; Else H sends the resulting packet with label stack <S1, S2, S3, L2, L3> along the path of S1.

In the case of SRv6, the processing is similar and follows the SR Policy headend behaviors as specified in section 5 of [RFC8986].

H has steered the packet into the SR policy P.

H did not have to classify the packet. The classification was done by a node upstream of H (e.g., the source of the packet or an intermediate ingress edge node of the SR domain) and the result of this classification was efficiently encoded in the packet header as a BSID.

This is another key benefit of the segment routing in general and the binding SID in particular: the ability to encode a classification and the resulting steering in the packet header to better scale and simplify intermediate aggregation nodes.

When Drop-Upon-Invalid (refer Section 8.2) is not in use, for an invalid SR Policy P, its BSID B is not in the forwarding plane and hence the packet K is dropped by H.

#### 8.4. Per-Destination Steering

This section describes how a headend applies steering of flows corresponding to BGP routes over SR Policy using the Color Extended community [RFC9012].

In the case of SR-MPLS, let us assume that headend H:

- o learns a BGP route R/r via next-hop N, Color Extended community C and VPN label V.
- o has a valid SR Policy P to (color = C, endpoint = N) of Segment-List <S1, S2, S3> and BSID B.
- o has a BGP policy that matches on the Color Extended community C and allows its usage as SLA steering information.

If all these conditions are met, H installs R/r in RIB/FIB with next-hop = SR Policy P of BSID B instead of via N.

Indeed, H's local BGP policy and the received BGP route indicate that the headend should associate R/r with an SR Policy path to endpoint N

with the SLA associated with color C. The headend, therefore, installs the BGP route on that policy.

This can be implemented by using the BSID as a generalized next-hop and installing the BGP route on that generalized next-hop.

When H receives a packet K with a destination matching R/r, H pushes the label stack <S1, S2, S3, V> and sends the resulting packet along the path to S1.

Note that any SID associated with the BGP route is inserted after the Segment-List of the SR Policy (i.e., <S1, S2, S3, V>).

In the case of SRv6, the processing is similar and follows the SR Policy headend behaviors as specified in section 5 of [RFC8986].

The same behavior applies to any type of service route: any AFI/SAFI of BGP [RFC4760] or LISP [RFC6830] for both IPv4/IPv6.

In a BGP multi-path scenario, the BGP route MAY be resolved over a mix of paths that include those that are steered over SR Policies and others resolved via the normal BGP nexthop resolution. Implementations MAY provide options to prefer one type over the other or other forms of local policy to determine the paths that are selected.

#### 8.4.1. Multiple Colors

When a BGP route has multiple Color Extended communities each with a valid SR Policy, the BGP process installs the route on the SR Policy giving preference to the Color Extended community with the highest numerical value.

Let us assume that headend H:

- o learns a BGP route R/r via next-hop N, Color Extended communities C1 and C2.
- o has a valid SR Policy P1 to (color = C1, endpoint = N) of Segment-List <S1, S2, S3> and BSID B1.
- o has a valid SR Policy P2 to (color = C2, endpoint = N) of Segment-List <S4, S5, S6> and BSID B2.
- o has a BGP policy that matches the Color Extended communities C1 and C2 and allows their usage as SLA steering information

If all these conditions are met, H installs R/r in RIB/FIB with next-hop = SR Policy P2 of BSID=B2 (instead of N) because C2 > C1.

When the SR Policy with a specific color is not instantiated or in the down/inactive state, the SR Policy with the next highest numerical value of color is considered.

#### 8.5. Recursion on an on-demand dynamic BSID

In the previous section, it was assumed that H had a pre-established "explicit" SR Policy (color C, endpoint N).

In this section, independent of the a-priori existence of any explicit candidate path of the SR policy (C, N), it is to be noted that the BGP process at headend node H triggers the instantiation of a dynamic candidate path for the SR policy (C, N) as soon as:

- o the BGP process learns of a route R/r via N and with Color Extended community C.
- o a local policy at node H authorizes the on-demand SR Policy path instantiation and maps the color to a dynamic SR Policy path optimization template.

##### 8.5.1. Multiple Colors

When a BGP route R/r via N has multiple Color Extended communities  $C_i$  (with  $i=1 \dots n$ ), an individual on-demand SR Policy dynamic path request (color  $C_i$ , endpoint N) is triggered for each color  $C_i$ . The SR Policy that is used for steering is then determined as described in Section 8.4.1.

#### 8.6. Per-Flow Steering

This section provides an example of how a headend might apply per-flow steering in practice.

Let us assume that headend H:

- o has a valid SR Policy P1 to (color = C1, endpoint = N) of Segment-List <S1, S2, S3> and BSID B1.
- o has a valid SR Policy P2 to (color = C2, endpoint = N) of Segment-List <S4, S5, S6> and BSID B2.
- o is configured to instantiate an array of paths to N where the entry 0 is the IGP path to N, color C1 is the first entry and color C2 is the second entry. The index into the array is called a Forwarding Class (FC). The index can have values 0 to 7, especially when derived from the MPLS TC bits [RFC5462].
- o is configured to match flows in its ingress interfaces (upon any field such as Ethernet destination/source/VLAN/TOS or IP destination/source/DSCP or transport ports etc.) and color them

with an internal per-packet forwarding-class variable (0, 1 or 2 in this example).

If all these conditions are met, H installs in RIB/FIB:

- o N via recursion on an array A (instead of the immediate outgoing link associated with the IGP shortest-path to N).
- o Entry A(0) set to the immediate outgoing link of the IGP shortest-path to N.
- o Entry A(1) set to SR Policy P1 of BSID=B1.
- o Entry A(2) set to SR Policy P2 of BSID=B2.

H receives three packets K, K1, and K2 on its incoming interface. These three packets either longest-match on N or more likely on a BGP/service route which recurses on N. H colors these 3 packets respectively with forwarding-class 0, 1, and 2.

As a result, for SR-MPLS:

- o H forwards K along the shortest path to N (i.e., pushes the Prefix-SID of N).
- o H pushes <S1, S2, S3> on packet K1 and forwards the resulting frame along the shortest path to S1.
- o H pushes <S4, S5, S6> on packet K2 and forwards the resulting frame along the shortest path to S4.

For SRv6, the processing is similar and the segment lists of the individual SR Policies P1 and P2 are enforced for packets K1 and K2 using the SR Policy headend behaviors as specified in section 5 of [RFC8986].

If the local configuration does not specify any explicit forwarding information for an entry of the array, then this entry is filled with the same information as entry 0 (i.e., the IGP shortest path).

If the SR Policy mapped to an entry of the array becomes invalid, then this entry is filled with the same information as entry 0. When all the array entries have the same information as entry0, the forwarding entry for N is updated to bypass the array and point directly to its outgoing interface and next-hop.

The array index values (e.g. 0, 1, and 2) and the notion of forwarding-class are implementation-specific and only meant to describe the desired behavior. The same can be realized by other mechanisms.

This realizes per-flow steering: different flows bound to the same BGP endpoint are steered on different IGP or SR Policy paths.

A headend MAY support options to apply per-flow steering only for traffic matching specific prefixes (e.g. specific IGP or BGP prefixes).

### 8.7. Policy-based Routing

Finally, headend H MAY be configured with a local routing policy which overrides any BGP/IGP path and steers a specified packet on an SR Policy. This includes the use of mechanisms like IGP Shortcut for automatic routing of IGP prefixes over SR Policies intended for such purpose.

### 8.8. Optional Steering Modes for BGP Destinations

#### 8.8.1. Color-Only BGP Destination Steering

In the previous section, it is seen that the steering on an SR Policy is governed by the matching of the BGP route's next-hop N and the authorized Color Extended community C with an SR Policy defined by the tuple (N, C).

This is the most likely form of BGP destination steering and the one recommended for most use-cases.

This section defines an alternative steering mechanism based only on the Color Extended community.

Three types of steering modes are defined.

For the default, Type 0, the BGP destination is steered as follows:

```
IF there is a valid SR Policy (N, C) where N is the IPv4 or IPv6
    endpoint address and C is a color;
    Steer into SR Policy (N, C);
ELSE;
    Steer on the IGP path to the next-hop N.
```

This is the classic case described in this document previously and what is recommended in most scenarios.

For Type 1, the BGP destination is steered as follows:

```
IF there is a valid SR Policy (N, C) where N is the IPv4 or IPv6
    endpoint address and C is a color;
    Steer into SR Policy (N, C);
ELSE IF there is a valid SR Policy (null endpoint, C) of the
```



```
        same address-family of N;
        Steer into SR Policy (null endpoint, C);
    ELSE IF there is any valid SR Policy
        (any address-family null endpoint, C);
        Steer into SR Policy (any null endpoint, C);
    ELSE;
        Steer on the IGP path to the next-hop N.
```

For Type 2, the BGP destination is steered as follows:

```
    IF there is a valid SR Policy (N, C) where N is an IPv4 or IPv6
        endpoint address and C is a color;
        Steer into SR Policy (N, C);
    ELSE IF there is a valid SR Policy (null endpoint, C)
        of the same address-family of N;
        Steer into SR Policy (null endpoint, C);
    ELSE IF there is any valid SR Policy
        (any address-family null endpoint, C);
        Steer into SR Policy (any null endpoint, C);
    ELSE IF there is any valid SR Policy (any endpoint, C)
        of the same address-family of N;
        Steer into SR Policy (any endpoint, C);
    ELSE IF there is any valid SR Policy
        (any address-family endpoint, C);
        Steer into SR Policy (any address-family endpoint, C);
    ELSE;
        Steer on the IGP path to the next-hop N.
```

The null endpoint is 0.0.0.0 for IPv4 and :: for IPv6 (all bits set to the 0 value).

Please refer to [I-D.ietf-idr-segment-routing-te-policy] for the updates to the BGP Color Extended community for the implementation of these mechanisms.

#### 8.8.2. Multiple Colors and CO flags

The steering preference is first based on the highest Color Extended community value and then Color-Only steering type for the color. Assuming a Prefix via (NH, C1(CO=01), C2(CO=01)); C1>C2 The steering preference order is:

- o SR policy (NH, C1).
- o SR policy (null, C1).
- o SR policy (NH, C2).
- o SR policy (null, C2).
- o IGP to NH.

### 8.8.3. Drop upon Invalid

This document defined earlier that when all the following conditions are met, H installs R/r in RIB/FIB with next-hop = SR Policy P of BSID B instead of via N.

- o H learns a BGP route R/r via next-hop N, Color Extended community C.
- o H has a valid SR Policy P to (color = C, endpoint = N) of Segment-List <S1, S2, S3> and BSID B.
- o H has a BGP policy that matches the Color Extended community C and allows its usage as SLA steering information.

This behavior is extended by noting that the BGP policy may require the BGP steering to always stay on the SR policy whatever its validity.

This is the "drop upon invalid" option described in Section 8.2 applied to BGP-based steering.

## 9. Recovering from Network Failures

### 9.1. Leveraging TI-LFA local protection of the constituent IGP segments

In any topology, Topology-Independent Loop-Free Alternate (TI-LFA) [I-D.ietf-rtgwg-segment-routing-ti-lfa] provides a 50msec local protection technique for IGP SIDs. The backup path is computed on a per IGP SID basis along the post-convergence path.

In a network that has deployed TI-LFA, an SR Policy built on the basis of TI-LFA protected IGP segments leverages the local protection of the constituent segments. Since TI-LFA protection is based on IGP computation, there are cases where the path used during the fast-reroute time window may not meet the exact constraints of the SR Policy.

In a network that has deployed TI-LFA, an SR Policy instantiated only with non-protected Adj SIDs does not benefit from any local protection.

### 9.2. Using an SR Policy to locally protect a link

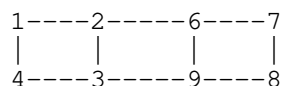


Figure 1: Local protection using SR Policy

An SR Policy can be instantiated at node 2 to protect link 2to6. A typical explicit Segment-List would be <3, 9, 6>.

A typical use-case occurs for links outside an IGP domain: e.g. 1, 2, 3, and 4 are part of IGP/SR sub-domain 1 while 6, 7, 8, and 9 are part of IGP/SR sub-domain 2. In such a case, links 2to6 and 3to9 cannot benefit from TI-LFA automated local protection. The SR Policy with Segment-List <3, 9, 6> on node 2 can be locally configured to be a fast-reroute backup path for the link 2to6.

### 9.3. Using a Candidate Path for Path Protection

An SR Policy allows for multiple candidate paths, of which at any point in time there is a single active candidate path that is provisioned in the forwarding plane and used for traffic steering. However, another (lower preference) candidate path MAY be designated as the backup for a specific or all (active) candidate path(s). The following options are possible:

- o A pair of disjoint candidate paths are provisioned with one of them as primary and the other is identified as its backup.
- o A specific candidate path is provisioned as the backup for any (active) candidate path.
- o The headend picks the next (lower) preference valid candidate path as the backup for the active candidate path.

The headend MAY compute a-priori and validate such backup candidate paths as well as provision them into the forwarding plane as a backup for the active path. The backup candidate path may be dynamically computed or explicitly provisioned in such a way that they provide the most appropriate alternative for the active candidate path. A fast re-route mechanism MAY then be used to trigger sub 50msec switchover from the active to the backup candidate path in the forwarding plane. Mechanisms like Bidirectional Forwarding Detection (BFD) MAY be used for fast detection of such failures.

## 10. Security Considerations

This document specifies in detail the SR Policy construct introduced in [RFC8402] and its instantiation on a router supporting SR along with descriptions of mechanisms for steering of traffic flows over

it. Therefore, the security considerations of [RFC8402] apply. The security consideration related to SR-MPLS [RFC8660] and SRv6 [RFC8754] [RFC8986] also apply.

The endpoint of the SR Policy, other than in the case of a null endpoint, uniquely identifies the tail-end node of the segment routed path. If an address that is used as an endpoint for an SR Policy is advertised by more than one node due to a misconfiguration or spoofing and the same is advertised via an IGP, the traffic steered over the SR Policy may end up getting diverted to an undesired node resulting in misrouting. Mechanisms for detection of duplicate prefix advertisement can be used to identify and correct such scenarios. The details of these mechanisms are outside the scope of this document.

The Section 8 specifies mechanism for steering of traffic flows corresponding to BGP routes over SR Policies matching the color value signaled via the BGP Color Extended Community attached with the BGP routes. Misconfiguration or error in setting of the Color Extended Community with the BGP routes can result in forwarding of packets for those routes along undesired paths.

In Section 2.1 and Section 2.6, the document mentions that a symbolic name MAY be signaled along with a candidate path for the SR Policy and for the SR Policy Candidate Path respectively. While the value of symbolic names for display clarity is indisputable, as with any unrestricted freeform text received from external parties, there can be no absolute assurance that the information the text purports to show is accurate or even truthful. For this reason, users of implementations that display such information would be well-advised not to rely on it without question and to use the specific identifiers of the SR Policy and SR Policy Candidate Path for validation. Furthermore, implementations that display such information might wish to display it in such a fashion as to differentiate it from known-good information. (Such display conventions are inherently implementation-specific; one example might be use of a distinguished text color or style for information that should be treated with caution.)

This document does not define any new protocol extensions and does not introduce any further security considerations.

## 11. Manageability Considerations

This document specifies in detail the SR Policy construct introduced in [RFC8402] and its instantiation on a router supporting SR along with descriptions of mechanisms for steering of traffic flows over it. Therefore, the manageability considerations of [RFC8402] apply.

A YANG model for the configuration and operation of SR Policy has been defined in [I-D.ietf-spring-sr-policy-yang].

## 12. IANA Considerations

The document requests IANA to create a new sub-registry called "Segment Types" under the top-level "Segment Routing" registry that was created by [RFC8986]. This sub-registry maintains the alphabetic identifiers for the segment types (as specified in section 4) that may be used within a Segment List of an SR Policy. The alphabetical identifiers run from A to Z and may be extended on exhaustion with the identifiers AA to AZ, BA to BZ, and so on through till ZZ. This sub-registry would follow the Specification Required allocation policy as specified in [RFC8126].

The initial registrations for this sub-registry are as follows:

Value	Description	Reference
A	SR-MPLS Label	[This.ID]
B	SRv6 SID	[This.ID]
C	IPv4 Prefix with optional SR Algorithm	[This.ID]
D	IPv6 Global Prefix with optional SR Algorithm for SR-MPLS	[This.ID]
E	IPv4 Prefix with Local Interface ID	[This.ID]
F	IPv4 Addresses for link endpoints as Local, Remote pair	[This.ID]
G	IPv6 Prefix and Interface ID for link endpoints as Local, Remote pair for SR-MPLS	[This.ID]
H	IPv6 Addresses for link endpoints as Local, Remote pair for SR-MPLS	[This.ID]
I	IPv6 Global Prefix with optional SR Algorithm for SRv6	[This.ID]
J	IPv6 Prefix and Interface ID for link endpoints as Local, Remote pair for SRv6	[This.ID]
K	IPv6 Addresses for link endpoints as Local, Remote pair for SRv6	[This.ID]

Table 2: Initial IANA Registration

### 12.1. Guidance for Designated Experts

The Designated Expert (DE) is expected to ascertain the existence of suitable documentation (a specification) as described in [RFC8126] and to verify that the document is permanently and publicly available. The DE is also expected to check the clarity of purpose

and use of the requested assignment. Additionally, the DE must verify that any request for one of these assignments has been made available for review and comment within the IETF: the DE will post the request to the SPRING Working Group mailing list (or a successor mailing list designated by the IESG). If the request comes from within the IETF, it should be documented in an Internet-Draft. Lastly, the DE must ensure that any other request for a code point does not conflict with work that is active or already published within the IETF.

### 13. Acknowledgement

The authors would like to thank Tarek Saad, Dhanendra Jain, Ruediger Geib, Rob Shakir, Cheng Li, Dhruv Dhody, Gyan Mishra, Nandan Saha, Jim Guichard, Martin Vigoureux, Benjamin Schwartz, David Schinazi, Matthew Bocci, Cullen Jennings, and Carlos Bernardos for their review comments and suggestions.

### 14. Contributors

The following people have contributed to this document:

Siva Sivabalan  
Cisco Systems  
Email: msiva@cisco.com

Zafar Ali  
Cisco Systems  
Email: zali@cisco.com

Jose Liste  
Cisco Systems  
Email: jliste@cisco.com

Francois Clad  
Cisco Systems  
Email: fclad@cisco.com

Kamran Raza  
Cisco Systems  
Email: skraza@cisco.com

Mike Koldychev  
Cisco Systems  
Email: mkoldych@cisco.com

Shraddha Hegde  
Juniper Networks  
Email: shraddha@juniper.net

Steven Lin  
Google, Inc.  
Email: stevenlin@google.com

Przemyslaw Krol  
Google, Inc.  
Email: pkrol@google.com

Martin Horneffer  
Deutsche Telekom  
Email: martin.horneffer@telekom.de

Dirk Steinberg  
Steinberg Consulting  
Email: dws@steinbergnet.net

Bruno Decraene  
Orange Business Services  
Email: bruno.decraene@orange.com

Stephane Litkowski  
Orange Business Services  
Email: stephane.litkowski@orange.com

Luay Jalil  
Verizon  
Email: luay.jalil@verizon.com

## 15. References

### 15.1. Normative References

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

- [RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8402] Filtsfils, 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>>.
- [RFC8660] Bashandy, A., Ed., Filtsfils, C., Ed., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with the MPLS Data Plane", RFC 8660, DOI 10.17487/RFC8660, December 2019, <<https://www.rfc-editor.org/info/rfc8660>>.
- [RFC8754] Filtsfils, C., Ed., Dukes, D., Ed., Previdi, S., Leddy, J., Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header (SRH)", RFC 8754, DOI 10.17487/RFC8754, March 2020, <<https://www.rfc-editor.org/info/rfc8754>>.
- [RFC8986] Filtsfils, C., Ed., Camarillo, P., Ed., Leddy, J., Voyer, D., Matsushima, S., and Z. Li, "Segment Routing over IPv6 (SRv6) Network Programming", RFC 8986, DOI 10.17487/RFC8986, February 2021, <<https://www.rfc-editor.org/info/rfc8986>>.
- [RFC9012] Patel, K., Van de Velde, G., Sangli, S., and J. Scudder, "The BGP Tunnel Encapsulation Attribute", RFC 9012, DOI 10.17487/RFC9012, April 2021, <<https://www.rfc-editor.org/info/rfc9012>>.

## 15.2. Informative References

- [I-D.ali-spring-sr-traffic-accounting]  
Ali, Z., Filtsfils, C., Talaulikar, K., Sivabalan, S., Horneffer, M., Raszuk, R., Litkowski, S., Voyer, D., and R. Morton, "Traffic Accounting in Segment Routing Networks", draft-ali-spring-sr-traffic-accounting-06 (work in progress), November 2021.



[I-D.anand-spring-poi-sr]

Anand, M., Bardhan, S., Subrahmaniam, R., Tantsura, J., Mukhopadhyaya, U., and C. Filsfils, "Packet-Optical Integration in Segment Routing", draft-anand-spring-poi-sr-08 (work in progress), July 2019.

[I-D.filsfils-spring-sr-policy-considerations]

Filsfils, C., Talaulikar, K., Krol, P., Horneffer, M., and P. Mattes, "SR Policy Implementation and Deployment Considerations", draft-filsfils-spring-sr-policy-considerations-08 (work in progress), October 2021.

[I-D.filsfils-spring-sr-traffic-counters]

Filsfils, C., Ali, Z., Horneffer, M., Voyer, D., Durrani, M., and R. Raszuk, "Segment Routing Traffic Accounting Counters", draft-filsfils-spring-sr-traffic-counters-02 (work in progress), October 2021.

[I-D.ietf-idr-segment-routing-te-policy]

Previdi, S., Filsfils, C., Talaulikar, K., Mattes, P., Jain, D., and S. Lin, "Advertising Segment Routing Policies in BGP", draft-ietf-idr-segment-routing-te-policy-16 (work in progress), March 2022.

[I-D.ietf-idr-te-lsp-distribution]

Previdi, S., Talaulikar, K., Dong, J., Chen, M., Gredler, H., and J. Tantsura, "Distribution of Traffic Engineering (TE) Policies and State using BGP-LS", draft-ietf-idr-te-lsp-distribution-16 (work in progress), October 2021.

[I-D.ietf-lsr-flex-algo]

Psenak, P., Hegde, S., Filsfils, C., Talaulikar, K., and A. Gulko, "IGP Flexible Algorithm", draft-ietf-lsr-flex-algo-18 (work in progress), October 2021.

[I-D.ietf-pce-binding-label-sid]

Sivabalan, S., Filsfils, C., Tantsura, J., Previdi, S., and C. L. (editor), "Carrying Binding Label/Segment Identifier (SID) in PCE-based Networks.", draft-ietf-pce-binding-label-sid-15 (work in progress), March 2022.

[I-D.ietf-pce-segment-routing-policy-cp]

Koldychev, M., Sivabalan, S., Barth, C., Peng, S., and H. Bidgoli, "PCEP extension to support Segment Routing Policy Candidate Paths", draft-ietf-pce-segment-routing-policy-cp-06 (work in progress), October 2021.

- [I-D.ietf-rtgwg-segment-routing-ti-lfa]  
Litkowski, S., Bashandy, A., Filsfils, C., Francois, P., Decraene, B., and D. Voyer, "Topology Independent Fast Reroute using Segment Routing", draft-ietf-rtgwg-segment-routing-ti-lfa-08 (work in progress), January 2022.
- [I-D.ietf-spring-sr-policy-yang]  
Raza, K., Sawaya, R., Shunwan, Z., Voyer, D., Durrani, M., Matsushima, S., and V. P. Beeram, "YANG Data Model for Segment Routing Policy", draft-ietf-spring-sr-policy-yang-01 (work in progress), April 2021.
- [RFC0020] Cerf, V., "ASCII format for network interchange", STD 80, RFC 20, DOI 10.17487/RFC0020, October 1969, <<https://www.rfc-editor.org/info/rfc20>>.
- [RFC1195] Callon, R., "Use of OSI IS-IS for routing in TCP/IP and dual environments", RFC 1195, DOI 10.17487/RFC1195, December 1990, <<https://www.rfc-editor.org/info/rfc1195>>.
- [RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, DOI 10.17487/RFC2328, April 1998, <<https://www.rfc-editor.org/info/rfc2328>>.
- [RFC3630] Katz, D., Kompella, K., and D. Yeung, "Traffic Engineering (TE) Extensions to OSPF Version 2", RFC 3630, DOI 10.17487/RFC3630, September 2003, <<https://www.rfc-editor.org/info/rfc3630>>.
- [RFC4760] Bates, T., Chandra, R., Katz, D., and Y. Rekhter, "Multiprotocol Extensions for BGP-4", RFC 4760, DOI 10.17487/RFC4760, January 2007, <<https://www.rfc-editor.org/info/rfc4760>>.
- [RFC5305] Li, T. and H. Smit, "IS-IS Extensions for Traffic Engineering", RFC 5305, DOI 10.17487/RFC5305, October 2008, <<https://www.rfc-editor.org/info/rfc5305>>.
- [RFC5307] Kompella, K., Ed. and Y. Rekhter, Ed., "IS-IS Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)", RFC 5307, DOI 10.17487/RFC5307, October 2008, <<https://www.rfc-editor.org/info/rfc5307>>.
- [RFC5329] Ishiguro, K., Manral, V., Davey, A., and A. Lindem, Ed., "Traffic Engineering Extensions to OSPF Version 3", RFC 5329, DOI 10.17487/RFC5329, September 2008, <<https://www.rfc-editor.org/info/rfc5329>>.

- [RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, DOI 10.17487/RFC5340, July 2008, <<https://www.rfc-editor.org/info/rfc5340>>.
- [RFC5462] Andersson, L. and R. Asati, "Multiprotocol Label Switching (MPLS) Label Stack Entry: "EXP" Field Renamed to "Traffic Class" Field", RFC 5462, DOI 10.17487/RFC5462, February 2009, <<https://www.rfc-editor.org/info/rfc5462>>.
- [RFC6830] Farinacci, D., Fuller, V., Meyer, D., and D. Lewis, "The Locator/ID Separation Protocol (LISP)", RFC 6830, DOI 10.17487/RFC6830, January 2013, <<https://www.rfc-editor.org/info/rfc6830>>.
- [RFC7471] Giacalone, S., Ward, D., Drake, J., Atlas, A., and S. Previdi, "OSPF Traffic Engineering (TE) Metric Extensions", RFC 7471, DOI 10.17487/RFC7471, March 2015, <<https://www.rfc-editor.org/info/rfc7471>>.
- [RFC8231] Crabbe, E., Minei, I., Medved, J., and R. Varga, "Path Computation Element Communication Protocol (PCEP) Extensions for Stateful PCE", RFC 8231, DOI 10.17487/RFC8231, September 2017, <<https://www.rfc-editor.org/info/rfc8231>>.
- [RFC8476] Tantsura, J., Chunduri, U., Aldrin, S., and P. Psenak, "Signaling Maximum SID Depth (MSD) Using OSPF", RFC 8476, DOI 10.17487/RFC8476, December 2018, <<https://www.rfc-editor.org/info/rfc8476>>.
- [RFC8491] Tantsura, J., Chunduri, U., Aldrin, S., and L. Ginsberg, "Signaling Maximum SID Depth (MSD) Using IS-IS", RFC 8491, DOI 10.17487/RFC8491, November 2018, <<https://www.rfc-editor.org/info/rfc8491>>.
- [RFC8570] Ginsberg, L., Ed., Previdi, S., Ed., Giacalone, S., Ward, D., Drake, J., and Q. Wu, "IS-IS Traffic Engineering (TE) Metric Extensions", RFC 8570, DOI 10.17487/RFC8570, March 2019, <<https://www.rfc-editor.org/info/rfc8570>>.
- [RFC8664] Sivabalan, S., Filsfils, C., Tantsura, J., Henderickx, W., and J. Hardwick, "Path Computation Element Communication Protocol (PCEP) Extensions for Segment Routing", RFC 8664, DOI 10.17487/RFC8664, December 2019, <<https://www.rfc-editor.org/info/rfc8664>>.

- [RFC8814] Tantsura, J., Chunduri, U., Talaulikar, K., Mirsky, G., and N. Triantafyllis, "Signaling Maximum SID Depth (MSD) Using the Border Gateway Protocol - Link State", RFC 8814, DOI 10.17487/RFC8814, August 2020, <<https://www.rfc-editor.org/info/rfc8814>>.
- [RFC9086] Previdi, S., Talaulikar, K., Ed., Filsfils, C., Patel, K., Ray, S., and J. Dong, "Border Gateway Protocol - Link State (BGP-LS) Extensions for Segment Routing BGP Egress Peer Engineering", RFC 9086, DOI 10.17487/RFC9086, August 2021, <<https://www.rfc-editor.org/info/rfc9086>>.

## Authors' Addresses

Clarence Filsfils  
Cisco Systems, Inc.  
Pegasus Parc  
De kleetlaan 6a, DIEGEM BRABANT 1831  
BELGIUM

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

Ketan Talaulikar (editor)  
Cisco Systems, Inc.  
India

Email: [ketant.ietf@gmail.com](mailto:ketant.ietf@gmail.com)

Daniel Voyer  
Bell Canada  
671 de la gauchetiere W  
Montreal, Quebec H3B 2M8  
Canada

Email: [daniel.voyer@bell.ca](mailto:daniel.voyer@bell.ca)

Alex Bogdanov  
British Telecom

Email: [alex.bogdanov@bt.com](mailto:alex.bogdanov@bt.com)

Paul Mattes  
Microsoft  
One Microsoft Way  
Redmond, WA 98052-6399  
USA

Email: [pamattes@microsoft.com](mailto:pamattes@microsoft.com)

SPRING Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: May 6, 2021

J. Rajamanickam  
K. Raza  
Cisco Systems

D. Bernier  
Bell Canada

November 2, 2020

YANG Data Model for SR Service Programming  
draft-jags-spring-sr-service-programming-yang-00

Abstract

This document describes a YANG data model for Segment Routing (SR) Service Programming. The model serves as a base framework for configuring and managing an SR based service programming. Additionally, this document specifies the model for a Service Proxy for SR-unaware services.

The YANG modules in this document conform to the Network Management Datastore Architecture (NMDA).

Status of This Memo

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

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

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

This Internet-Draft will expire on May 6, 2021.

Copyright Notice

Copyright (c) 2020 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. Specification of Requirements . . . . .	3
3. YANG Model . . . . .	4
3.1. Overview . . . . .	4
3.2. Service Function Types . . . . .	4
3.3. SR Service Programming Types . . . . .	5
3.4. SR Service Programming Base . . . . .	5
3.4.1. Configuration . . . . .	5
3.4.2. Operational State . . . . .	6
3.4.3. Notification . . . . .	7
3.5. SR Service Proxy . . . . .	7
3.5.1. Static Proxy . . . . .	8
3.5.2. Dynamic Proxy . . . . .	9
3.5.3. Masquerading Proxy . . . . .	10
4. YANG Specification . . . . .	11
4.1. Service Types . . . . .	11
4.2. SR Service Programming Types . . . . .	13
4.3. SR Service Programming Base . . . . .	17
4.4. SR Service Proxy . . . . .	23
5. Security Considerations . . . . .	28
6. IANA Considerations . . . . .	29
7. Acknowledgments . . . . .	30
8. Normative References . . . . .	30
Authors' Addresses . . . . .	32

## 1. Introduction

The Network Configuration Protocol (NETCONF) [RFC6241] is one of the network management protocols that defines mechanisms to manage network devices. YANG [RFC6020] is a modular language that represents data structures in an XML tree format, and is used as a data modeling language for the NETCONF.

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

path, but also steer them through Virtual Network Function (VNF) or physical service appliances available in the network.

In an SR network, each of these services, running either on a physical appliance or in a virtual environment, are associated with a segment identifier (SID). These service SIDs are then leveraged as part of a SID-list to steer packets through the desired services in the service chain. Service SIDs may be combined together in a SID-list to achieve the service programming, but also with other types of segments as defined in [RFC8402]. SR thus provides a fully integrated solution for overlay, underlay and service programming. Furthermore, the IPv6 instantiation of SR (SRv6) supports metadata transportation in the Segment Routing header [RFC8754], either natively in the tag field or with extensions such as TLVs.

This document describes how a service can be associated with a SID, including legacy services with no SR capabilities, and how these service SIDs are integrated within an SR policy. The definition of an SR Policy and the traffic steering mechanisms are covered in [I-D.ietf-spring-segment-routing-policy] and hence outside the scope of this document.

This document introduces a YANG data model for the SR based service programming configuration and management. Furthermore, this document also covers the basic SR unaware behaviours as defined in [I-D.ietf-spring-sr-service-programming].

This document does not cover the following:

- o SR-aware service specific management parameters

The model currently defines the following constructs that are used for managing SR based service programming:

- o Configuration
- o Operational State
- o Notifications

## 2. Specification of Requirements

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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.



### 3. YANG Model

#### 3.1. Overview

This document defines the following four new YANG modules:

- o `ietf-service-function-types`: Defines common service function types
- o `ietf-sr-service-programming-types`: Defines common type definitions used for SR based service programming YANG model
- o `ietf-sr-service-programming`: Defines management model for SR based service programming framework. This is a base and common framework for both SR-aware and SR-unaware services.
- o `ietf-sr-service-programming-proxy`: Defines management model for SR service proxy for SR unaware services

The modelling in this document complies with the Network Management Datastore Architecture (NMDA) defined in [RFC8342]. The operational state data is combined with the associated configuration data in the same hierarchy [RFC8407]. When protocol states are retrieved from the NMDA operational state datastore, the returned states cover all "config true" (rw) and "config false" (ro) nodes defined in the schema.

In this document, when a simplified graphical representation of YANG model is presented in a tree diagram, the meaning of the symbols in these tree diagrams is defined in [RFC8340].

#### 3.2. Service Function Types

A service is identified by (type, instance). The type represents the type of service functions (such as Firewall, DPI IPS etc.), whereas instance is used to refer to a specific instance of the same service.

We define a new YANG module `ietf-service-function-types` to specify common definitions and types for service and service function. The types and definitions are generic and hence can be used in any (SR based or non-SR) YANG models.

The main definitions and types defined in `ietf-service-function-types` module include:

- o `service-function-type`: A new identity type to specify service function types, such as firewall, dpi etc. Other identities can be define by other modules in future.

### 3.3. SR Service Programming Types

The types required to model SR based service programming are defined in a new module `ietf-sr-service-programming-types`.

The main types defined in this module includes:

- o `service-program-behaviour-type`: Defines SR service program behaviours like `sr-aware`, `static-proxy` etc...
- o `service-program-oper-status-type`: Defines SR service programming operational status. This includes the reason for down status as well
- o `service-proxy-inner-pkt-type`: Defines SR service proxy inner packet types

### 3.4. SR Service Programming Base

The base model and framework for SR based service programming is defined in a new module `ietf-sr-service-programming`. This module provides a common base for both the `SR-aware` and `SR-unaware` service programming in terms of configuration, operation state and notifications. The `ietf-sr-service-programming` module hangs off main SR parent by augmenting `"/rt:routing/sr:segment-routing"`.

#### 3.4.1. Configuration

This module defines some fundamental items required to configure SR based service programming. In particular, it defines service program provisioning as follows:

- o `service program behaviour`: Defining a service program behaviour
- o `service offered`: Defining a specific service (type, instance) offered this service programming
- o `Assigning a SR service SID`: Defining SID data plane, method to allocate the SID etc..
- o `service program enablement`: Administratively Enable/Disable a service program
- o `SR services`: Defining a base container which could be augmented to define `SR-aware` or `SR-unaware` (via `service-proxy`) service specific parameters

Following is a simplified graphical tree representation of the data model for SR service programming base configuration only

```

module: ietf-sr-service-programming
  augment /rt:routing/sr:segment-routing:
    +--rw service-programming
      +--rw service-program* [name]
        +--rw name          string
        +--rw behaviour     identityref
        +--rw service-type  identityref
        +--rw service-instance uint32
        +--rw dataplane     sr-svc-pgm-types:dataplane-type
        +--rw admin-status? sr-svc-pgm-types:admin-status-type
        +--rw sid-binding
          +--rw alloc-mode  sr-svc-pgm-types:sid-alloc-mode-type
          +--rw mpls
            +--rw sid?      rt-types:mpls-label
          +--rw srv6
            +--rw sid?      srv6-types:srv6-sid
            +--rw locator?  -> /rt:routing/sr:segment-routing/
                               srv6:srv6/locators/locator/name
        +--rw sr-services

```

Figure 1: SR Service Programming Config Tree

### 3.4.2. Operational State

As per NMDA model, the state related to configuration items specified in above section Section 3.4.1 can be retrieved from the same tree. This section defines other operational state items related to SR based service programming.

The operational state corresponding to an SR based service program includes:

- o Operational status: Provides detail information on the operational state of the SR service program.
- o statistics: Provides the statistics details such as number of packets/bytes received, processed and dropped corresponding to a SR service program.

Following is a simplified graphical tree representation of the data model for the SR service programming base operational state (for read-only items):

```

module: ietf-sr-service-programming
augment /rt:routing/sr:segment-routing:
  +--rw service-programming
    +--rw service-program* [name]
      +--ro oper-status?      identityref
      +--ro statistics
        +--ro in-packet-count?      yang:counter64
        +--ro in-bytes-count?       yang:counter64
        +--ro out-packet-count?     yang:counter64
        +--ro out-bytes-count?      yang:counter64
        +--ro in-drop-packet-count? yang:counter64
        +--ro out-drop-packet-count? yang:counter64

```

Figure 2: SR Service Programming Operational State Tree

### 3.4.3. Notification

This model defines a list of notifications to inform an operator of important events detected during the SR service programming operation. These events are:

- o SR service program operational state changes: This would also give the reason for the state change when it is down

Following is a simplified graphical tree representation of the data model for the SR service programming notification:

```

module: ietf-sr-service-programming
notifications:
  +---n service-program-oper-status
    +--ro name      -> /rt:routing/sr:segment-routing/
                      sr-svc-pgm:service-programming/
                      service-program/name
    +--ro oper-status -> /rt:routing/sr:segment-routing/
                      sr-svc-pgm:service-programming/
                      service-program/oper-status

```

Figure 3: SR Service Programming Notification Tree

### 3.5. SR Service Proxy

This document also defines a separate and new YANG data model for Service Proxy for SR unaware services. The model defines the configuration and operational state related to different proxy behaviours defined earlier in ietf-sr-service-programming-types. The

model is defined in a new module `ietf-sr-service-programming proxy`. This module augments the SR service program tree (`/rt:routing/sr:segment-routing/sr-svc-pgm:service-programming/ sr-svc-pgm:service-program/sr-svc-pgm:sr-services`) as defined earlier in `ietf-sr-service-programming` module.

The following sections describe different types of proxy behaviours and associated YANG modelling constructs.

#### 3.5.1. Static Proxy

The static proxy is an SR endpoint behaviour for processing SR-MPLS or SRv6 encapsulated traffic on behalf of an SR-unaware services.

The following parameters are required to provision the SR static proxy:

- o `inner-packet-type`: Inner packet type
- o `next-hop`: Next hop Ethernet address (only for the inner type is IPv4 or IPv6)
- o `out-interface-name`: Local interface for sending traffic towards the service Endpoint
- o `in-interface-name`: Local interface receiving traffic coming back from the service Endpoint
- o `packet-cache-info`: SR information to be attached on the traffic coming back from the service. This could be list of MPLS Label stack or SRv6 SIDs

Following is a simplified graphical tree representation of the data model for the SR static proxy:

```

module: ietf-sr-service-programming-proxy
augment /rt:routing/sr:segment-routing/
  sr-svc-pgm:service-programming/
  sr-svc-pgm:service-program/
  sr-svc-pgm:sr-services:
  +--rw service-proxy
    +--rw (proxy-type)
      +--:(static)
        +--rw static-proxy
          +--rw inner-packet-type      identityref
          +--rw next-hop?              yang:mac-address
          +--rw out-interface-name     string
          +--rw in-interface-name      string
          +--rw packet-cache-info
            +--rw (cache-type)
              +--:(mpls)
                +--rw mpls-sids* [index]
                  +--rw index          uint8
                  +--rw mpls-label      rt-types:mpls-label
              +--:(srv6)
                +--rw ipv6-source-address?  inet:ipv6-address
                +--rw srv6-sids* [index]
                  +--rw index          uint8
                  +--rw srv6-sid        srv6-types:srv6-sid

```

Figure 4: SR Static Proxy Tree

### 3.5.2. Dynamic Proxy

The dynamic proxy is an improvement over the static proxy that dynamically learns the SR information before removing it from the incoming traffic. The same information can be re-attached to the traffic returning from the service Endpoints. The dynamic proxy relies on the local caching.

The following parameters are required to provision the SR dynamic proxy:

- o out-interface-name: Local interface for sending traffic towards the service Endpoint
- o in-interface-name: Local interface receiving traffic coming back from the service Endpoint

Following is a simplified graphical tree representation of the data model for the SR static proxy:

```
module: ietf-sr-service-programming-proxy
augment /rt:routing/sr:segment-routing/
  sr-svc-pgm:service-programming/
  sr-svc-pgm:service-program/
  sr-svc-pgm:sr-services:
  +--rw service-proxy
    +--rw (proxy-type)
      +--:(dynamic)
        +--rw dynamic-proxy
          +--rw out-interface-name    string
          +--rw in-interface-name     string
```

Figure 5: SR Dynamic Proxy Tree

### 3.5.3. Masquerading Proxy

The masquerading proxy is an SR endpoint behaviour for processing SRv6 traffic on behalf of an SR-unaware service. This masquerading behaviour is independent from the inner payload type.

The following parameters are required to provision the SR masquerading proxy

- o next-hop: Next hop Ethernet address
- o out-interface-name: Local interface for sending traffic towards the service Endpoint
- o in-interface-name: Local interface receiving traffic coming back from the service Endpoint

Following is a simplified graphical tree representation of the data model for the SR masquerading proxy:

```
module: ietf-sr-service-programming-proxy
  augment /rt:routing/sr:segment-routing/
    sr-svc-pgm:service-programming/
    sr-svc-pgm:service-program/
    sr-svc-pgm:sr-services:
  +--rw service-proxy
    +--rw (proxy-type)
      +--:(masquerading)
        +--rw masquerading-proxy
          +--rw next-hop?          yang:mac-address
          +--rw out-interface-name string
          +--rw in-interface-name  string
```

Figure 6: SR masquerading Proxy Tree

#### 4. YANG Specification

Following are actual YANG definition for SR service programming modules defined earlier in the document.

##### 4.1. Service Types

Following are the Service Types definitions.

```
<CODE BEGINS> file "ietf-service-function-types.yang" -->

module ietf-service-function-types {
  yang-version 1.1;

  namespace "urn:ietf:params:xml:ns:yang:ietf-service-function-types";
  prefix "service-types";

  organization "IETF SPRING Working Group";

  contact
    "WG Web:    <http://tools.ietf.org/wg/spring/>
    WG List:    <mailto:spring@ietf.org>

    Editor:     Jaganbabu Rajamanickam
                <mailto:jrajaman@cisco.com>

    Editor:     Kamran Raza
                <mailto:skraza@cisco.com>

    Editor:     Daniel Bernier
                <mailto:daniel.bernier@bell.ca>";
```



```
/*
 * Below are the definition for the service types
 * Any new service type could added by extending
 * this identity
 */
identity service-function-type {
    description
        "Base identity from which specific service function
        types are derived.";
}

identity firewall {
    base service-function-type;
    description
        "Firewall Service type";
}

identity dpi {
    base service-function-type;
    description
        "Deep Packet Inspection Service type";
}

identity napt44 {
    base service-function-type;
    description
        "Network Address and Port Translation 44
        Service type";
}

identity classifier {
    base service-function-type;
    description
        "classifier Service type";
}

identity load-balancer {
    base service-function-type;
    description
        "load-balancer Service type";
}

identity ips {
    base service-function-type;
    description
        "Intrusion Prevention System Service type (Ex: Snort)";
}
```

```
}
```

```
<CODE ENDS>
```

Figure 7: ietf-service-function-types.yang

#### 4.2. SR Service Programming Types

Following are the SR service programming specific types definitions.

```
<CODE BEGINS> file "ietf-sr-service-programming-types.yang" -->
module ietf-sr-service-programming-types {
  yang-version 1.1;

  namespace "urn:ietf:params:xml:ns:yang:ietf-sr-service-programming-types";
  prefix "sr-service-types";

  organization "IETF SPRING Working Group";

  contact
    "WG Web:  <http://tools.ietf.org/wg/spring/>
    WG List:  <mailto:spring@ietf.org>

    Editor:    Jaganbabu Rajamanickam
               <mailto:jrajaman@cisco.com>

    Editor:    Kamran Raza
               <mailto:skraza@cisco.com>

    Editor:    Daniel Bernier
               <mailto:daniel.bernier@bell.ca>";

  /*
   * SR Service programming behaviour
   */
  identity service-program-behaviour-type {
    description
      "Base identity for SR service programming behaviour";
  }

  identity sr-aware {
    base service-program-behaviour-type;
    description
      "SR aware native applications.";
  }
}
```

```
identity static-proxy {
    base service-program-behaviour-type;
    description
        "Static Proxy";
}

identity dynamic-proxy {
    base service-program-behaviour-type;
    description
        "Dynamic Proxy";
}

identity Masquerading-proxy {
    base service-program-behaviour-type;
    description
        "Masquerading Proxy";
}

identity Masquerading-NAT-proxy {
    base service-program-behaviour-type;
    description
        "Masquerading Proxy with NAT flavor";
}

identity Masquerading-caching-proxy {
    base service-program-behaviour-type;
    description
        "Masquerading Proxy with caching flavor";
}

identity Masquerading-NAT-caching-proxy {
    base service-program-behaviour-type;
    description
        "Masquerading Proxy with caching flavor";
}

/*
 * Below are the definition for the service proxy inner packet types
 * Any new service proxy inner packet type could added by extending
 * this identity
 */
identity service-proxy-inner-pkt-type {
    description
        "Base identity from which SR service proxy types are derived.";
}

identity Ethernet {
```

```
    base service-proxy-inner-pkt-type;
    description
        "Expected inner packet type as Ethernet - derived from
        service-proxy-inner-pkt-type";
}

identity IPv4 {
    base service-proxy-inner-pkt-type;
    description
        "Expected inner packet type as IPv4 - derived from
        service-proxy-inner-pkt-type";
}

identity IPv6 {
    base service-proxy-inner-pkt-type;
    description
        "Expected inner packet type as IPv6 - derived from
        service-proxy-inner-pkt-type";
}

/*
 * SR Service SID operational status
 */
identity service-program-oper-status-type {
    description
        "Base identity from which SR service program operational
        status types are derived.";
}

identity up {
    base service-program-oper-status-type;
    description
        "Service program status is operational";
}

identity down-unknown {
    base service-program-oper-status-type;
    description
        "Service program status is down because of unknown reason";
}

identity sid-allocation-pending {
    base service-program-oper-status-type;
    description
        "Service program status is down because of SID allocation is pending";
}
```

```
identity sid-allocation-conflict {
    base service-program-oper-status-type;
    description
        "Service program status is down because of SID conflict";
}

identity sid-out-of-bound {
    base service-program-oper-status-type;
    description
        "Service program status is down because of SID is out of bound";
}

identity interface-down {
    base service-program-oper-status-type;
    description
        "Service program status is down because of out/in interface is down";
}

identity admin-forced-down {
    base service-program-oper-status-type;
    description
        "Service program status is administratively forced down";
}

/*
 * Typedefs
 */
typedef admin-status-type {
    type enumeration {
        enum up {
            description "Admin Up";
        }
        enum down {
            description "Admin Down";
        }
    }
}

typedef dataplane-type {
    type enumeration {
        enum mpls {
            description "MPLS dataplane";
        }
        enum srv6 {
            description "SRv6 dataplane";
        }
    }
}
```

```
typedef sid-alloc-mode-type {  
  type enumeration {  
    enum static {  
      description "Static SID allocation";  
    }  
    enum dynamic {  
      description "Dynamic SID allocation";  
    }  
  }  
}
```

<CODE ENDS>

Figure 8: ietf-sr-service-programming-types.yang

#### 4.3. SR Service Programming Base

Following are the SR service programming base model definition.

<CODE BEGINS> file "ietf-sr-service-programming.yang" -->

```
module ietf-sr-service-programming {  
  yang-version 1.1;  
  
  namespace "urn:ietf:params:xml:ns:yang:ietf-sr-service-programming";  
  prefix "sr-svc-pgm";  
  
  import ietf-yang-types {  
    prefix "yang";  
  }  
  
  import ietf-srv6-base {  
    prefix "srv6";  
  }  
  
  import ietf-routing {  
    prefix rt;  
    reference "RFC 8349: A YANG Data Model for Routing  
      Management (NMDA Version)";  
  }  
  
  import ietf-service-function-types {  
    prefix "service-types";  
  }
```

```
import ietf-segment-routing {
    prefix sr;
}

import ietf-sr-service-programming-types {
    prefix "sr-svc-pgm-types";
}

import ietf-routing-types {
    prefix "rt-types";
}

import ietf-srv6-types {
    prefix "srv6-types";
}

organization "IETF SPRING Working Group";

contact
    "WG Web:    <http://tools.ietf.org/wg/spring/>
    WG List:    <mailto:spring@ietf.org>

    Editor:     Jaganbabu Rajamanickam
                <mailto:jrajaman@cisco.com>

    Editor:     Kamran Raza
                <mailto:skraza@cisco.com>

    Editor:     Daniel Bernier
                <mailto:daniel.bernier@bell.ca>";

grouping service-statistics {
    container statistics {
        config false;
        description "Service statistics";

        leaf in-packet-count {
            type yang:counter64;
            description
                "Total number of packets processed by this service";
        }

        leaf in-bytes-count {
            type yang:counter64;
            description
                "Total number of bytes processed by this service";
        }
    }
}
```

```
    }

    leaf out-packet-count {
      type yang:counter64;
      description
        "Total number of packets end out after processing by this service";
    }

    leaf out-bytes-count {
      type yang:counter64;
      description
        "Total number of bytes end out after processing by this service";
    }

    leaf in-drop-packet-count {
      type yang:counter64;
      description
        "Total number of packets dropped while processing by this service";
    }

    leaf out-drop-packet-count {
      type yang:counter64;
      description
        "Total number of packets dropped while this service try to
        forward to its destination";
    }
  }
}

grouping service-mpls-sid-binding {
  container mpls {
    description
      "MPLS Service SID binding Container";

    when "../../dataplane = 'mpls'";

    leaf sid {
      type rt-types:mpls-label;
      description
        "MPLS SID value.";
    }
  }
}

grouping service-srv6-sid-binding {
  container srv6 {
    description
      "SRv6 Service SID binding Container";
  }
}
```



```

    when "../../../dataplane = 'srv6'";

    leaf sid {
        type srv6-types:srv6-sid;
        description
            "SRv6 SID value.";
    }

    leaf locator {
        type leafref {
            path "/rt:routing/sr:segment-routing"
                + "/srv6:srv6/srv6:locators/srv6:locator/srv6:name";
        }
        description
            "Reference to a SRv6 locator. This is valid only when
            the SID allocation mode is dynamic";
    }
}

grouping service-sid-binding {
    container sid-binding {
        description
            "Service SID binding Container";

        leaf alloc-mode {
            mandatory true;
            type sr-svc-pgm-types:sid-alloc-mode-type;
            description
                "Service SID allocation mode";
        }

        uses service-mpls-sid-binding;
        uses service-srv6-sid-binding;
    }
}

grouping service-programming {
    container service-programming {
        description
            "service programming container.
            Any new services programming added could augment
            this container to support that specific services.
            Currently in this model, only service proxy
            is defined. (i.e) For example if
            a Firewall services needs to be added then
            they could augment this container and
            extend this model";
    }
}

```

```
list service-program {
  key "name";
  description
    "Service program is keyed by the service program name";

  leaf name {
    type string;
    description
      "Service program name to identify a specific program.";
  }

  leaf behaviour {
    mandatory true;
    type identityref {
      base sr-svc-pgm-types:service-program-behaviour-type;
    }
    description
      "SR program behaviour";
  }

  leaf service-type {
    mandatory true;
    type identityref {
      base service-types:service-function-type;
    }
    description
      "Service-Type defined by IANA (STT). This is either the SR-aware
      service of SR-unaware service offered by an SR proxy";
  }

  leaf service-instance {
    mandatory true;
    type uint32;
    description
      "Service instance which differentiates the same service -- e.g.
      same Firewall service could have several instances available.
      The type and the instance would
      describe a specific instance which the application would
      like to choose";
  }

  leaf dataplane {
    mandatory true;
    type sr-svc-pgm-types:dataplane-type;
    description
      "Service SID dataplane.";
  }
}
```

```
    leaf admin-status {
      type sr-svc-pgm-types:admin-status-type;
      default down;
      description
        "Admin Status";
    }

    leaf oper-status {
      config false;
      type identityref {
        base sr-svc-pgm-types:service-program-oper-status-type;
      }
      description
        "Service SID operational mode.";
    }

    uses service-sid-binding;
    uses service-statistics;

    container sr-services {

      description
        "Any SR-aware or AR-unaware services could augment this container";
      reference "Segment Routing Service Programming Architecture.";
    }
  }
}

augment "/rt:routing/sr:segment-routing" {
  description
    "Augmenting the segment-routing bindings to add SR service programming";

  uses service-programming;
}

notification service-program-oper-status {
  description
    "This notification is sent when there is a change in the service
    program oper status.";
  leaf name {
    mandatory true;
    type leafref {
      path "/rt:routing/sr:segment-routing/"
        + "sr-svc-pgm:service-programming/"
        + "sr-svc-pgm:service-program/"
        + "sr-svc-pgm:name";
    }
  }
}
```

```

        description
            "Service program name to identify a specific programming.";
    }

    leaf oper-status {
        mandatory true;
        type leafref {
            path "/rt:routing/sr:segment-routing/"
                + "sr-svc-pgm:service-programming/"
                + "sr-svc-pgm:service-program/"
                + "sr-svc-pgm:oper-status";
        }
        description
            "Service program operational status.";
    }
}
}
}

<CODE ENDS>

```

Figure 9: ietf-sr-service-programming.yang

#### 4.4. SR Service Proxy

Following are the SR service programming service proxy model definition.

```

<CODE BEGINS> file "ietf-sr-service-programming-proxy.yang" -->
module ietf-sr-service-programming-proxy {
    yang-version 1.1;

    namespace "urn:ietf:params:xml:ns:yang:ietf-sr-service-programming-proxy";
    prefix "sr-svc-proxy";

    import ietf-yang-types {
        prefix yang;
    }

    import ietf-routing {
        prefix rt;
        reference "RFC 8349: A YANG Data Model for Routing
            Management (NMDA Version)";
    }

    import ietf-inet-types {

```

```
    prefix "inet";
}

import ietf-segment-routing {
    prefix sr;
}

import ietf-sr-service-programming {
    prefix "sr-svc-pgm";
}

import ietf-sr-service-programming-types {
    prefix "sr-svc-pgm-types";
}

import ietf-routing-types {
    prefix "rt-types";
}

import ietf-srv6-types {
    prefix "srv6-types";
}

organization "IETF SPRING Working Group";

contact
    "WG Web:    <http://tools.ietf.org/wg/spring/>
     WG List:   <mailto:spring@ietf.org>

     Editor:    Jaganbabu Rajamanickam
                <mailto:jrajaman@cisco.com>

     Editor:    Kamran Raza
                <mailto:skraza@cisco.com>

     Editor:    Daniel Bernier
                <mailto:daniel.bernier@bell.ca>";

grouping service-proxy-parameters {

    leaf out-interface-name {
        mandatory true;
        type string;
        description
            "Interface name on which the packet sent to the service endpoint";
    }

    leaf in-interface-name {
```

```
        mandatory true;
        type string;
        description
            "Interface name on which the packet received from the service endpoint";
    }
}

grouping mpls-packet-cache-info {
    description
        "MPLS Label stack";

    list mpls-sids {
        key "index";

        leaf index {
            type uint8 {
                range "1..16";
            }
            description
                "cache index - MPLS Label stack index";
        }

        leaf mpls-label {
            mandatory true;
            type rt-types:mpls-label;
            description
                "MPLS Label value.";
        }
    }
}

grouping srv6-packet-cache-info {
    description
        "SRv6 SID stack";

    leaf ipv6-source-address {
        type inet:ipv6-address;
        description
            "IPv6 source address that needs in the case if SRv6.";
    }
    list srv6-sids {
        key "index";

        leaf index {
            type uint8 {
                range "1..16";
            }
            description

```

```

        "cache index - SRv6 SID index";
    }

    leaf srv6-sid {
        mandatory true;
        type srv6-types:srv6-sid;
        description
            "SRv6 SID.";
    }
}

grouping service-proxy-packet-cache-info {
    description
        "SRv6 Proxy header cache";

    container packet-cache-info {

        choice cache-type {
            mandatory true;
            case mpls {

                when "/rt:routing/sr:segment-routing/sr-svc-pgm:service-programming
                    /sr-svc-pgm:service-program
                    /sr-svc-pgm:dataplane = 'mpls'";

                uses mpls-packet-cache-info;
            }
            case srv6 {

                when "/rt:routing/sr:segment-routing/sr-svc-pgm:service-programming
                    /sr-svc-pgm:service-program
                    /sr-svc-pgm:dataplane = 'srv6'";

                uses srv6-packet-cache-info;
            }
        }
        // uses mpls-packet-cache-info;
        // uses srv6-packet-cache-info;
    }
}

grouping static-service-proxy {
    container static-proxy {
        description
            "Parameters related to static service proxy";

        leaf inner-packet-type {

```

```

        mandatory true;
        type identityref {
            base sr-svc-pgm-types:service-proxy-inner-pkt-type;
        }
        description
            "Defines the expected inner packet type";
    }

    leaf next-hop {
        when "(../inner-packet-type = 'IPv4' or ../inner-packet-type = 'IPv6')";
        type yang:mac-address;
        description
            "Nexthop Ethernet address for inner packet type IPv4/IPv6";
    }
    uses service-proxy-parameters;
    uses service-proxy-packet-cache-info;
}

grouping dynamic-service-proxy {
    container dynamic-proxy {
        description
            "Parameters related to dynamic service proxy";
        uses service-proxy-parameters;
    }
}

grouping masquerading-service-parameters {

    leaf next-hop {
        mandatory true;
        type yang:mac-address;
        description
            "Nexthop Ethernet address";
    }
    uses service-proxy-parameters;
}

grouping masquerading-service-proxy {
    container masquerading-proxy {
        description
            "Parameters related to masquerading service proxy";

        when "/rt:routing/sr:segment-routing/sr-svc-pgm:service-programming
            /sr-svc-pgm:service-program
            /sr-svc-pgm:dataplane = 'srv6'";

        uses masquerading-service-parameters;
    }
}

```



```

    }
}

grouping service-proxy-programming {
    container service-proxy {
        choice proxy-type {
            mandatory true;
            case static {
                when "/rt:routing/sr:segment-routing/
                    sr-svc-pgm:service-programming
                    /sr-svc-pgm:service-program
                    /sr-svc-pgm:dataplane = 'srv6'";
                uses static-service-proxy;
            }
            case dynamic {
                uses dynamic-service-proxy;
            }
            case masquerading {
                uses masquerading-service-proxy;
            }
        }
        //uses dynamic-service-proxy;
    }
}

augment "/rt:routing/sr:segment-routing/sr-svc-pgm:service-programming/sr-svc-p
gm:service-program/sr-svc-pgm:sr-services" {
    description
        "Augmenting the segment-routing bindings to add SR-unaware
        service programming";

    uses service-proxy-programming;
}
}

<CODE ENDS>

```

Figure 10: ietf-sr-service-programming-proxy.yang

## 5. Security Considerations

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure

transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8446].

The Network Configuration Access Control Model (NACM) [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

There are a number of data nodes defined in this YANG module that are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) to these data nodes without proper protection can have a negative effect on network operations.

Some of the readable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes.

It goes without saying that this specification also inherits the security considerations captured in the SRv6 specification document [I-D.ietf-spring-sr-service-programming].

## 6. IANA Considerations

This document requests the registration of the following URIs in the IETF "XML registry" [RFC3688]:

URI	Registration	XML
urn:ietf:params:xml:ns:yang:ietf-service-function-types	The IESG	N/A
urn:ietf:params:xml:ns:yang:ietf-sr-service-programming-types	The IESG	N/A
urn:ietf:params:xml:ns:yang:ietf-sr-service-programming	The IESG	N/A
urn:ietf:params:xml:ns:yang:ietf-sr-service-programming-proxy	The IESG	N/A

This document requests the registration of the following YANG modules in the "YANG Module Names" registry [RFC6020]:

Name	Namespace	Prefix	Reference
ietf-service-function-types	urn:ietf:params:xml:ns:yang:ietf-service-function-types	service-function-types	This document
ietf-sr-service-programming-types	urn:ietf:params:xml:ns:yang:ietf-sr-service-programming-types	ietf-sr-service-programming-types	This document
ietf-sr-service-programming	urn:ietf:params:xml:ns:yang:ietf-sr-service-programming	ietf-sr-service-programming	This document
ietf-sr-service-programming-proxy	urn:ietf:params:xml:ns:yang:ietf-sr-service-programming-proxy	ietf-sr-service-programming-proxy	This document

-- RFC Editor: Replace "This document" with the document RFC number at time of publication, and remove this note.

## 7. Acknowledgments

The authors would like to acknowledge Francois Clad, Ketan Talaulikar, and Darren Dukes for their review of some of the contents in this document.

## 8. Normative References

[I-D.ietf-spring-segment-routing-policy]  
 Filsfils, C., Talaulikar, K., Voyer, D., Bogdanov, A., and P. Mattes, "Segment Routing Policy Architecture", draft-ietf-spring-segment-routing-policy-09 (work in progress), November 2020.

[I-D.ietf-spring-sr-service-programming]  
 Clad, F., Xu, X., Filsfils, C., daniel.bernier@bell.ca, d., Li, C., Decraene, B., Ma, S., Yadlapalli, C., Henderickx, W., and S. Salsano, "Service Programming with Segment Routing", draft-ietf-spring-sr-service-programming-03 (work in progress), September 2020.

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <<https://www.rfc-editor.org/info/rfc3688>>.
- [RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <<https://www.rfc-editor.org/info/rfc6020>>.
- [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <<https://www.rfc-editor.org/info/rfc6241>>.
- [RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <<https://www.rfc-editor.org/info/rfc6242>>.
- [RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <<https://www.rfc-editor.org/info/rfc8040>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <<https://www.rfc-editor.org/info/rfc8340>>.
- [RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <<https://www.rfc-editor.org/info/rfc8341>>.
- [RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <<https://www.rfc-editor.org/info/rfc8342>>.

- [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>>.
- [RFC8407] Bierman, A., "Guidelines for Authors and Reviewers of Documents Containing YANG Data Models", BCP 216, RFC 8407, DOI 10.17487/RFC8407, October 2018, <<https://www.rfc-editor.org/info/rfc8407>>.
- [RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <<https://www.rfc-editor.org/info/rfc8446>>.
- [RFC8754] Filsfils, C., Ed., Dukes, D., Ed., Previdi, S., Leddy, J., Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header (SRH)", RFC 8754, DOI 10.17487/RFC8754, March 2020, <<https://www.rfc-editor.org/info/rfc8754>>.

#### Authors' Addresses

Jaganbabu Rajamanickam  
Cisco Systems

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

Kamran Raza  
Cisco Systems

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

Daniel Bernier  
Bell Canada

Email: [daniel.bernier@bell.ca](mailto:daniel.bernier@bell.ca)

SPRING Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: 29 July 2022

J. Rajamanickam  
K. Raza  
Cisco Systems  
D. Bernier  
Bell Canada  
G. Dawra  
LinkedIn  
C. Li  
Huawei  
25 January 2022

YANG Data Model for SR Service Programming  
draft-jags-spring-sr-service-programming-yang-03

Abstract

This document describes a YANG data model for Segment Routing (SR) Service Programming. The model serves as a base framework for configuring and managing an SR based service programming. Additionally, this document specifies the model for a Service Proxy for SR-unaware services.

The YANG modules in this document conform to the Network Management Datastore Architecture (NMDA).

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 29 July 2022.

Copyright Notice

Copyright (c) 2022 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 Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

## Table of Contents

1. Introduction . . . . .	2
2. Specification of Requirements . . . . .	4
3. YANG Model . . . . .	4
3.1. Overview . . . . .	4
3.2. Service Function Types . . . . .	5
3.3. SR Service Programming Types . . . . .	5
3.4. SR Service Programming Base . . . . .	5
3.4.1. Configuration . . . . .	6
3.4.2. Operational State . . . . .	8
3.4.3. Notification . . . . .	10
3.5. SR Service Proxy . . . . .	10
3.5.1. Static Proxy . . . . .	11
3.5.2. Dynamic Proxy . . . . .	13
3.5.3. Masquerading Proxy . . . . .	14
4. YANG Specification . . . . .	15
4.1. Service Types . . . . .	15
4.2. SR Service Programming Types . . . . .	17
4.3. SR Service Programming Base . . . . .	22
4.4. SR Service Proxy . . . . .	32
5. Security Considerations . . . . .	39
6. IANA Considerations . . . . .	39
7. Acknowledgments . . . . .	41
8. Normative References . . . . .	41
Authors' Addresses . . . . .	43

## 1. Introduction

The Network Configuration Protocol (NETCONF) [RFC6241] is one of the network management protocols that defines mechanisms to manage network devices. YANG [RFC6020] is a modular language that represents data structures in an XML tree format, and is used as a data modeling language for the NETCONF.

Segment Routing is an architecture based on the source routing paradigm that seeks the right balance between distributed intelligence and centralized programmability. SR can be used with an MPLS or an IPv6 data plane to steer packets through an ordered list

of instructions, called segments. These segments may encode simple routing instructions for forwarding packets along a specific network path, but also steer them through Virtual Network Function (VNF) or physical service appliances available in the network.

In an SR network, each of these services, running either on a physical appliance or in a virtual environment, are associated with a segment identifier (SID). These service SIDs are then leveraged as part of a SID-list to steer packets through the desired services in the service chain. Service SIDs may be combined together in a SID-list to achieve the service programming, but also with other types of segments as defined in [RFC8402]. SR thus provides a fully integrated solution for overlay, underlay and service programming. Furthermore, the IPv6 instantiation of SR (SRv6) supports metadata transportation in the Segment Routing header [RFC8754], either natively in the tag field or with extensions such as TLVs.

This document describes how a service can be associated with a SID, including legacy services with no SR capabilities, and how these service SIDs are integrated within an SR policy. The definition of an SR Policy and the traffic steering mechanisms are covered in [I-D.ietf-spring-segment-routing-policy] and hence outside the scope of this document.

This document introduces a YANG data model for the SR based service programming configuration and management. Furthermore, this document also covers the basic SR unaware behaviours as defined in [I-D.ietf-spring-sr-service-programming].

This document does not cover the following:

- \* SR-aware service specific management parameters

The model currently defines the following constructs that are used for managing SR based service programming:

- \* Configuration
- \* Operational State
- \* Notifications



## 2. Specification of Requirements

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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

## 3. YANG Model

### 3.1. Overview

This document defines the following four new YANG modules:

- \* ietf-service-function-types: Defines common service function types
- \* ietf-sr-service-programming-types: Defines common type definitions used for SR based service programming YANG model
- \* ietf-sr-service-programming: Defines management model for SR based service programming framework. This is a base and common framework for both SR-aware and SR-unaware services.
- \* ietf-sr-service-programming-proxy: Defines management model for SR service proxy for SR unaware services

The modelling in this document complies with the Network Management Datastore Architecture (NMDA) defined in [RFC8342]. The operational state data is combined with the associated configuration data in the same hierarchy [RFC8407]. When protocol states are retrieved from the NMDA operational state datastore, the returned states cover all "config true" (rw) and "config false" (ro) nodes defined in the schema.

In this document, when a simplified graphical representation of YANG model is presented in a tree diagram, the meaning of the symbols in these tree diagrams is defined in [RFC8340].

In this document, the SR service programming YANG model is split based on dynamic SID allocation and static SID allocation. In the case of dynamic SID allocation, new SR service programming tree would be used. In the case of static MPLS SID allocation for the SR service programming, the existing SR MPLS YANG model [RFC9020] would be augmented with the SR MPLS service programming specific parameters. Similarly the static SRv6 base YANG model (TBD) would be augmented with the SRv6 service programming specific parameters.

### 3.2. Service Function Types

A service is identified by (type, variant, instance). The type represents the type of service functions (such as Firewall, DPI IPS etc.), The variant value is a unique identifier which could identify the vendor and its product informations, The instance is used to refer to a specific instance of the same (service, variant).

We define a new YANG module `ietf-service-function-types` to specify common definitions and types for service and service function. The types and definitions are generic and hence can be used in any (SR based or non-SR) YANG models.

The main definitions and types defined in `ietf-service-function-types` module include:

- \* `service-function-type`: A new identity type to specify service function types, such as firewall, dpi etc. Other identities can be define by other modules in future.

### 3.3. SR Service Programming Types

The types required to model SR based service programming are defined in a new module `ietf-sr-service-programming-types`.

The main types defined in this module includes:

- \* `service-program-behaviour-type`: Defines SR service program behaviours like sr-aware, static-proxy etc...
- \* `service-program-oper-status-type`: Defines SR service programming operational status. This includes the reason for down status as well
- \* `service-proxy-inner-pkt-type`: Defines SR service proxy inner packet types

### 3.4. SR Service Programming Base

The base model and framework for SR based service programming using dynamic SID allocation is defined in a new module `ietf-sr-service-programming`.

In the case of static MPLS SID allocation for the SR service programming, the existing SR MPLS YANG model [RFC9020] would be augmented with the SR MPLS service programming specific parameters.

In the case of static SRv6 based YANG model (TBD) would be augmented with the SRv6 service programming specific parameters.

This module provides a common base for both the SR-aware and SR-unaware service programming in terms of configuration, operation state and notifications.

The ietf-sr-service-programming module hangs off main SR parent by augmenting "/rt:routing/sr:segment-routing".

#### 3.4.1. Configuration

This module defines some fundamental items required to configure SR based service programming. In particular, it defines service program provisioning as follows:

- \* service program behaviour: Defining a service program behaviour
- \* service offered: Defining a specific service (type, variant, instance) offered this service programming
- \* Assigning a SR service SID: Defining SID data plane, method to allocate the SID etc..
- \* service program enablement: Administratively Enable/Disable a service program
- \* SR services: Defining a base container which could be augmented to define SR-aware or SR-unaware (via service-proxy) service specific parameters

Following is a simplified graphical tree representation of the data model for SR service programming (Dynamic SID allocation) base configuration only

```

module: ietf-sr-service-programming
augment /rt:routing/sr:segment-routing:
  +--rw service-programming
    +--rw service-program* [name]
      +--rw name
        -> /rt:routing/
          sr:segment-routing/
            sr-svc-pgm:service-programming/
              service-program/
                service-programming-info/
                  service-name

    +--rw sid-binding
      +--ro alloc-mode? sr-svc-pgm-types:sid-alloc-mode-type
      +--rw mpls
        | +--ro sid? rt-types:mpls-label
        +--rw srv6
          +--ro sid? srv6-types:srv6-sid
          +--rw locator? -> /rt:routing/sr:segment-routing/
            srv6:srv6/locators/locator/name

    +--rw service-programming-info
      +--rw behaviour identityref
      +--rw dataplane sr-svc-pgm-types:dataplane-type
      +--rw service-name string
      +--rw service-type identityref
      +--rw service-variant string
      +--rw service-instance uint32
      +--rw admin-status? sr-svc-pgm-types:admin-status-type
      +--rw sr-services

```

Figure 1: SR Service Programming Config Tree - Dynamic SID allocation

Following is a simplified graphical tree representation of the data model for SR service programming (Static SR MPLS SID allocation) base configuration only. In this case SR MPLS base YANG model has been augmented to support SR service programming using static SR MPLS SID allocation. This has been done for the user convince to program all the SR service programming parameters from the based SR MPLS YANG itself

```

module: ietf-sr-service-programming
augment /rt:routing/sr:segment-routing/sr-mpls:sr-mpls/sr-mpls:bindings:
  +--rw mpls-static-service-programming
    +--rw service-program* [name]
      +--rw name -> /rt:routing/
                    sr:segment-routing/
                    sr-svc-pgm:service-programming/
                    service-program/
                    service-programming-info/
                    service-name
      +--rw sid rt-types:mpls-label
    +--rw service-programming-info
      +--rw behaviour identityref
      +--ro dataplane? sr-svc-pgm-types:dataplane-type
      +--rw service-name string
      +--rw service-type identityref
      +--rw service-variant string
      +--rw service-instance uint32
      +--rw admin-status? sr-svc-pgm-types:admin-status-type
      +--rw sr-services

```

Figure 2: SR Service Programming Config Tree - Static SR MPLS SID allocation

Following is a simplified graphical tree representation of the data model for SR service programming (Static SRv6 SID allocation) base configuration only. TBD (Once the based SRv6 static model is available, this section will be filled)

### 3.4.2. Operational State

As per NMDA model, the state related to configuration items specified in above section Section 3.4.1 can be retrieved from the same tree. This section defines other operational state items related to SR based service programming.

The operational state corresponding to an SR based service program includes:

- \* Operational status: Provides detail information on the operational state of the SR service program.
- \* statistics: Provides the statistics details such as number of packets/bytes received, processed and dropped corresponding to a SR service program.

Following is a simplified graphical tree representation of the data model for the SR service programming base operational state (for read-only items):

Dynamic SID allocation case:

```
module: ietf-sr-service-programming
  augment /rt:routing/sr:segment-routing:
    +--rw service-programming
      +--rw service-program* [name]
        +--rw service-programming-info
          +--ro oper-status?          identityref
          +--ro statistics
            +--ro in-packet-count?      yang:counter64
            +--ro in-bytes-count?       yang:counter64
            +--ro out-packet-count?     yang:counter64
            +--ro out-bytes-count?      yang:counter64
            +--ro in-drop-packet-count? yang:counter64
            +--ro out-drop-packet-count? yang:counter64
```

Static SR MPLS SID allocation case:

```
module: ietf-sr-service-programming
  augment /rt:routing/sr:segment-routing/sr-mpls:sr-mpls/sr-mpls:bindings:
    +--rw mpls-static-service-programming
      +--rw service-program* [name]
        +--rw service-programming-info
          +--ro oper-status?          identityref
          +--ro statistics
            +--ro in-packet-count?      yang:counter64
            +--ro in-bytes-count?       yang:counter64
            +--ro out-packet-count?     yang:counter64
            +--ro out-bytes-count?      yang:counter64
            +--ro in-drop-packet-count? yang:counter64
            +--ro out-drop-packet-count? yang:counter64
```

Static SRv6 SID allocation case:

TBD

Figure 3: SR Service Programming Operational State Tree

### 3.4.3. Notification

This model defines a list of notifications to inform an operator of important events detected during the SR service programming operation. These events are:

- \* SR service program operational state changes: This would also give the reason for the state change when it is down

Following is a simplified graphical tree representation of the data model for the SR service programming notification:

```
module: ietf-sr-service-programming
  notifications:
    +---n service-program-oper-status
      +--ro name          -> /rt:routing/sr:segment-routing/
                           sr-svc-pgm:service-programming/
                           service-program/name
      +--ro oper-status   -> /rt:routing/sr:segment-routing/
                           sr-svc-pgm:service-programming/
                           service-program/oper-status
```

Figure 4: SR Service Programming Notification Tree

### 3.5. SR Service Proxy

This document also defines a separate and new YANG data model for Service Proxy for SR unaware services. The model defines the configuration and operational state related to different proxy behaviours defined earlier in ietf-sr-service-programming-types. The model is defined in a new module ietf-sr-service-programming proxy.

To support SR service programming proxy for dynamic SID allocation, this module augments the SR service program tree (/rt:routing/sr:segment-routing/sr-svc-pgm:service-programming/ sr-svc-pgm:service-program/sr-svc-pgm:sr-services) as defined earlier in ietf-sr-service-programming module.

To support SR service programming proxy for static SR MPLS SID allocation, this module augments the base SR MPLS YANG mode defined in the RFC [RFC9020] (/rt:routing/sr:segment-routing/sr-mpls:sr-mpls/sr-mpls:bindings/ sr-svc-pgm:mpls-static-service-programming/ sr-svc-pgm:service-program/sr-svc-pgm:service-programming-info/ sr-svc-pgm:sr-services:)

To support SR service programming proxy for static SRv6 SID allocation, this module augments the base static SRv6 model - TBD

The following sections describe different types of proxy behaviours and associated YANG modelling constructs.

### 3.5.1. Static Proxy

The static proxy is an SR endpoint behaviour for processing SR-MPLS or SRv6 encapsulated traffic on behalf of an SR-unaware services.

The following parameters are required to provision the SR static proxy:

- \* inner-packet-type: Inner packet type
- \* next-hop: Next hop Ethernet address (only for the inner type is IPv4 or IPv6)
- \* out-interface-name: Local interface for sending traffic towards the service Endpoint
- \* in-interface-name: Local interface receiving traffic coming back from the service Endpoint
- \* packet-cache-info: SR information to be attached on the traffic coming back from the service. This could be list of MPLS Label stack or SRv6 SIDs

Following is a simplified graphical tree representation of the data model for the SR static proxy:

Dynamic SID allocation case:

```

module: ietf-sr-service-programming-proxy
  augment /rt:routing/sr:segment-routing/
    sr-svc-pgm:service-programming/
      sr-svc-pgm:service-program/
        sr-svc-pgm:service-programming-info/
          sr-svc-pgm:sr-services:
            +--rw service-proxy
              +--rw (proxy-type)
                +--:(static)
                  +--rw static-proxy
                    +--rw inner-packet-type      identityref
                    +--rw next-hop?               yang:mac-address
                    +--rw out-interface-name      string
                    +--rw in-interface-name       string
                    +--rw packet-cache-info
                      +--rw (cache-type)
                        +--:(mpls)

```



```

    |   +--rw mpls-sids* [index]
    |   |   +--rw index          uint8
    |   |   +--rw mpls-label     rt-types:mpls-label
    +--:(srv6)
    |   +--rw ipv6-source-address? inet:ipv6-address
    |   +--rw srv6-sids* [index]
    |   |   +--rw index          uint8
    |   |   +--rw srv6-sid      srv6-types:srv6-sid

```

Static SR MPLS SID allocation case:

```

module: ietf-sr-service-programming-proxy
  augment /rt:routing/sr:segment-routing/
    sr-mpls:sr-mpls/sr-mpls:bindings/
    sr-svc-pgm:mpls-static-service-programming/
    sr-svc-pgm:service-program/
    sr-svc-pgm:service-programming-info/
    sr-svc-pgm:sr-services:
    +--rw static-mpls-service-proxy
    |   +--rw (proxy-type)
    |   |   +--:(static)
    |   |   |   +--rw static-proxy
    |   |   |   |   +--rw inner-packet-type      identityref
    |   |   |   |   +--rw next-hop?              yang:mac-address
    |   |   |   |   +--rw out-interface-name     string
    |   |   |   |   +--rw in-interface-name      string
    |   |   |   |   +--rw packet-cache-info
    |   |   |   |   |   +--rw (cache-type)
    |   |   |   |   |   |   +--:(mpls)
    |   |   |   |   |   |   |   +--rw mpls-sids* [index]
    |   |   |   |   |   |   |   |   +--rw index          uint8
    |   |   |   |   |   |   |   |   +--rw mpls-label     rt-types:mpls-label
    |   |   |   |   |   |   +--:(srv6)
    |   |   |   |   |   |   |   +--rw ipv6-source-address? inet:ipv6-address
    |   |   |   |   |   |   |   +--rw srv6-sids* [index]
    |   |   |   |   |   |   |   |   +--rw index          uint8
    |   |   |   |   |   |   |   |   +--rw srv6-sid      srv6-types:srv6-sid

```

Static SRv6 SID allocation case:  
TDB

Figure 5: SR Static Proxy Tree

### 3.5.2. Dynamic Proxy

The dynamic proxy is an improvement over the static proxy that dynamically learns the SR information before removing it from the incoming traffic. The same information can be re-attached to the traffic returning from the service Endpoints. The dynamic proxy relies on the local caching.

The following parameters are required to provision the SR dynamic proxy:

- \* out-interface-name: Local interface for sending traffic towards the service Endpoint
- \* in-interface-name: Local interface receiving traffic coming back from the service Endpoint

Following is a simplified graphical tree representation of the data model for the SR static proxy:

Dynamic SID allocation case:

```

module: ietf-sr-service-programming-proxy
  augment /rt:routing/sr:segment-routing/
    sr-svc-pgm:service-programming/
    sr-svc-pgm:service-program/
    sr-svc-pgm:service-programming-info/
    sr-svc-pgm:sr-services:
  +--rw service-proxy
    +--rw (proxy-type)
      +--:(dynamic)
        +--rw dynamic-proxy
          +--rw out-interface-name    string
          +--rw in-interface-name     string

```

Static SR MPLS SID allocation case:

```

module: ietf-sr-service-programming-proxy
  augment /rt:routing/sr:segment-routing/
    sr-mpls:sr-mpls/sr-mpls:bindings/
    sr-svc-pgm:mpls-static-service-programming/
    sr-svc-pgm:service-program/
    sr-svc-pgm:service-programming-info/
    sr-svc-pgm:sr-services:
  +--rw static-mpls-service-proxy
    +--rw (proxy-type)
      +--:(dynamic)
        +--rw dynamic-proxy
          +--rw out-interface-name    string
          +--rw in-interface-name     string

```

Static SRv6 SID allocation case:

TBD

Figure 6: SR Dynamic Proxy Tree

### 3.5.3. Masquerading Proxy

The masquerading proxy is an SR endpoint behaviour for processing SRv6 traffic on behalf of an SR-unaware service. This masquerading behaviour is independent from the inner payload type.

The following parameters are required to provision the SR masquerading proxy

- \* next-hop: Next hop Ethernet address
- \* out-interface-name: Local interface for sending traffic towards the service Endpoint
- \* in-interface-name: Local interface receiving traffic coming back from the service Endpoint

Following is a simplified graphical tree representation of the data model for the SR masquerading proxy:

Dynamic SID allocation case:

```

module: ietf-sr-service-programming-proxy
  augment /rt:routing/sr:segment-routing/
    sr-svc-pgm:service-programming/
    sr-svc-pgm:service-program/
    sr-svc-pgm:service-programming-info/
    sr-svc-pgm:sr-services:
  +--rw service-proxy
    +--rw (proxy-type)
      +--:(masquerading)
        +--rw masquerading-proxy
          +--rw next-hop?                yang:mac-address
          +--rw out-interface-name       string
          +--rw in-interface-name        string

```

Static SRv6 SID allocation case:

TBD

Figure 7: SR masquerading Proxy Tree

#### 4. YANG Specification

Following are actual YANG definition for SR service programming modules defined earlier in the document.

##### 4.1. Service Types

Following are the Service Types definitions.

```
<CODE BEGINS> file "ietf-service-function-types.yang"
-->

module ietf-service-function-types {
  yang-version 1.1;

  namespace "urn:ietf:params:xml:ns:yang:ietf-service-function-types";
  prefix "service-types";

  organization "IETF SPRING Working Group";

  contact
    "WG Web:  <http://tools.ietf.org/wg/spring/>
    WG List:  <mailto:spring@ietf.org>

    Editor:   Jaganbabu Rajamanickam
              <mailto:jrajaman@cisco.com>

    Editor:   Kamran Raza
              <mailto:skraza@cisco.com>

    Editor:   Daniel Bernier
              <mailto:daniel.bernier@bell.ca>

    Editor:   Gaurav Dawra
              <mailto:gdawra.ietf@gmail.com>

    Editor:   Cheng Li
              <mailto:c.l@huawei.com>";

  /*
   * Below are the definition for the service types
   * Any new service type could added by extending
   * this identity
   */
  identity service-function-type {
    description
      "Base identity from which specific service function
       types are derived.";
  }

  identity firewall {
    base service-function-type;
    description
      "Firewall Service type";
  }

  identity dpi {
```

```
        base service-function-type;
        description
            "Deep Packet Inspection Service type";
    }

    identity napt44 {
        base service-function-type;
        description
            "Network Address and Port Translation 44
             Service type";
    }

    identity classifier {
        base service-function-type;
        description
            "classifier Service type";
    }

    identity load-balancer {
        base service-function-type;
        description
            "load-balancer Service type";
    }

    identity ips {
        base service-function-type;
        description
            "Intrusion Prevention System Service type (Ex: Snort)";
    }
}
<CODE ENDS>
```

Figure 8: ietf-service-function-types.yang

#### 4.2. SR Service Programming Types

Following are the SR service programming specific types definitions.

```
<CODE BEGINS> file "ietf-sr-service-programming-types.yang"
-->
```

```
module ietf-sr-service-programming-types {
    yang-version 1.1;

    namespace "urn:ietf:params:xml:ns:yang:ietf-sr-service-programming-types";
    prefix "sr-service-types";
```

```
organization "IETF SPRING Working Group";

contact
  "WG Web:  <http://tools.ietf.org/wg/spring/>
   WG List: <mailto:spring@ietf.org>

   Editor:  Jaganbabu Rajamanickam
            <mailto:jrajaman@cisco.com>

   Editor:  Kamran Raza
            <mailto:skraza@cisco.com>

   Editor:  Daniel Bernier
            <mailto:daniel.bernier@bell.ca>

   Editor:  Gaurav Dawra
            <mailto:gdawra.ietf@gmail.com>

   Editor:  Cheng Li
            <mailto:c.l@huawei.com>";

/*
 * SR Service programming behaviour
 */
identity service-program-behaviour-type {
  description
    "Base identity for SR service programming behaviour";
}

identity sr-aware {
  base service-program-behaviour-type;
  description
    "SR aware native applications.";
}

identity static-proxy {
  base service-program-behaviour-type;
  description
    "Static Proxy";
}

identity dynamic-proxy {
  base service-program-behaviour-type;
  description
    "Dynamic Proxy";
}

identity Masquerading-proxy {
```

```
    base service-program-behaviour-type;
    description
        "Masquerading Proxy";
}

identity Masquerading-NAT-proxy {
    base service-program-behaviour-type;
    description
        "Masquerading Proxy with NAT flavor";
}

identity Masquerading-caching-proxy {
    base service-program-behaviour-type;
    description
        "Masquerading Proxy with caching flavor";
}

identity Masquerading-NAT-caching-proxy {
    base service-program-behaviour-type;
    description
        "Masquerading Proxy with caching flavor";
}

/*
 * Below are the definition for the service proxy inner packet types
 * Any new service proxy inner packet type could added by extending
 * this identity
 */
identity service-proxy-inner-pkt-type {
    description
        "Base identity from which SR service proxy types are derived.";
}

identity Ethernet {
    base service-proxy-inner-pkt-type;
    description
        "Expected inner packet type as Ethernet - derived from
        service-proxy-inner-pkt-type";
}

identity IPv4 {
    base service-proxy-inner-pkt-type;
    description
        "Expected inner packet type as IPv4 - derived from
        service-proxy-inner-pkt-type";
}
```



```
identity IPv6 {
    base service-proxy-inner-pkt-type;
    description
        "Expected inner packet type as IPv6 - derived from
        service-proxy-inner-pkt-type";
}

/*
 * SR Service SID operational status
 */
identity service-program-oper-status-type {
    description
        "Base identity from which SR service program operational
        status types are derived.";
}

identity up {
    base service-program-oper-status-type;
    description
        "Service program status is operational";
}

identity down-unknown {
    base service-program-oper-status-type;
    description
        "Service program status is down because of unknown reason";
}

identity sid-allocation-pending {
    base service-program-oper-status-type;
    description
        "Service program status is down because of SID allocation is pending";
}

identity sid-allocation-conflict {
    base service-program-oper-status-type;
    description
        "Service program status is down because of SID conflict";
}

identity sid-out-of-bound {
    base service-program-oper-status-type;
    description
        "Service program status is down because of SID is out of bound";
}

identity interface-down {
```

```
    base service-program-oper-status-type;
    description
        "Service program status is down because of out/in interface is down";
}

identity admin-forced-down {
    base service-program-oper-status-type;
    description
        "Service program status is administratively forced down";
}

/*
 * Typedefs
 */
typedef admin-status-type {
    type enumeration {
        enum up {
            description "Admin Up";
        }
        enum down {
            description "Admin Down";
        }
    }
}

typedef dataplane-type {
    type enumeration {
        enum mpls {
            description "MPLS dataplane";
        }
        enum srv6 {
            description "SRv6 dataplane";
        }
    }
}

typedef sid-alloc-mode-type {
    type enumeration {
        enum static {
            description "Static SID allocation";
        }
        enum dynamic {
            description "Dynamic SID allocation";
        }
    }
}
}
<CODE ENDS>
```

Figure 9: ietf-sr-service-programming-types.yang

#### 4.3. SR Service Programming Base

Following are the SR service programming base model definition.

```
<CODE BEGINS> file "ietf-sr-service-programming.yang"
-->

module ietf-sr-service-programming {
  yang-version 1.1;

  namespace "urn:ietf:params:xml:ns:yang:ietf-sr-service-programming";
  prefix "sr-svc-pgm";

  import ietf-yang-types {
    prefix "yang";
  }

  import ietf-srv6-base {
    prefix "srv6";
  }

  import ietf-routing {
    prefix rt;
    reference "RFC 8349: A YANG Data Model for Routing
              Management (NMDA Version)";
  }

  import ietf-service-function-types {
    prefix "service-types";
  }

  import ietf-segment-routing {
    prefix sr;
  }

  import ietf-segment-routing-mpls {
    prefix srmppls;
  }

  import ietf-sr-service-programming-types {
    prefix "sr-svc-pgm-types";
  }

  import ietf-routing-types {
    prefix "rt-types";
  }
}
```

```
import ietf-srv6-types {
  prefix "srv6-types";
}

organization "IETF SPRING Working Group";

contact
  "WG Web: <http://tools.ietf.org/wg/spring/>
  WG List: <mailto:spring@ietf.org>

  Editor: Jaganbabu Rajamanickam
          <mailto:jrajaman@cisco.com>

  Editor: Kamran Raza
          <mailto:skraza@cisco.com>

  Editor: Daniel Bernier
          <mailto:daniel.bernier@bell.ca>

  Editor: Gaurav Dawra
          <mailto:gdawra.ietf@gmail.com>

  Editor: Cheng Li
          <mailto:c.l@huawei.com>";

grouping service-statistics {
  container statistics {
    config false;
    description "Service statistics";

    leaf in-packet-count {
      type yang:counter64;
      description
        "Total number of packets processed by this service";
    }

    leaf in-bytes-count {
      type yang:counter64;
      description
        "Total number of bytes processed by this service";
    }

    leaf out-packet-count {
      type yang:counter64;
      description
        "Total number of packets end out after processing by this service";
```

```
    }

    leaf out-bytes-count {
      type yang:counter64;
      description
        "Total number of bytes end out after processing by this service";
    }

    leaf in-drop-packet-count {
      type yang:counter64;
      description
        "Total number of packets dropped while processing by this service";
    }

    leaf out-drop-packet-count {
      type yang:counter64;
      description
        "Total number of packets dropped while this service try to
        forward to its destination";
    }
  }
}

grouping service-mpls-sid-binding {
  container mpls {
    description
      "MPLS Service SID binding Container";

    when "../../service-programming-info/dataplane = 'mpls'";

    leaf sid {
      config false;
      type rt-types:mpls-label;
      description
        "MPLS SID value.";
    }
  }
}

grouping service-srv6-sid-binding {
  container srv6 {
    description
      "SRv6 Service SID binding Container";

    when "../../service-programming-info/dataplane = 'srv6'";

    leaf sid {
      config false;
```

```
    type srv6-types:srv6-sid;
    description
      "SRv6 SID value.";
  }

  leaf locator {
    type leafref {
      path "/rt:routing/sr:segment-routing"
        + "/srv6:srv6/srv6:locators/srv6:locator/srv6:name";
    }
    description
      "Reference to a SRv6 locator. This is valid only when
       the SID allocation mode is dynamic";
  }
}

grouping service-sid-binding {
  container sid-binding {
    description
      "Service SID binding Container";

    leaf alloc-mode {
      config false;
      default dynamic;
      type sr-svc-pgm-types:sid-alloc-mode-type;
      description
        "Service SID allocation mode";
    }

    uses service-mpls-sid-binding;
    uses service-srv6-sid-binding;
  }
}

grouping service-programming-infos {
  container service-programming-info {

    leaf behaviour {
      mandatory true;
      type identityref {
        base sr-svc-pgm-types:service-program-behaviour-type;
      }
      description
        "SR program behaviour";
    }

    leaf dataplane {
```

```
        mandatory true;
        type sr-svc-pgm-types:dataplane-type;
        description
            "Service SID dataplane.";
    }

    leaf service-name {
        mandatory true;
        type string;
        description
            "Service program name to identify a specific program.";
    }

    leaf service-type {
        mandatory true;
        type identityref {
            base service-types:service-function-type;
        }
        description
            "Service-Type defined by IANA Service Type Table (STT). Like
            Firewall, DPI etc...";
    }

    leaf service-variant {
        mandatory true;
        type string;
        description
            "This identifies the variant of the service. This value should
            be unique in the given network. Example Format:
            <vendor>-<vendor-sub-variant>-<product-version>.";
    }

    leaf service-instance {
        mandatory true;
        type uint32;
        description
            "Service instance which differentiates the same service -- e.g.
            same vendors Firewall service could have several instances
            available. This could be used to differentiate the VPN
            customers or for load sharing purposes.";
    }

    leaf admin-status {
        type sr-svc-pgm-types:admin-status-type;
        default down;
        description
            "Admin Status";
    }
}
```

```

    leaf oper-status {
        config false;
        type identityref {
            base sr-svc-pgm-types:service-program-oper-status-type;
        }
        description
            "Service SID operational mode.";
    }

    uses service-statistics;

    container sr-services {

        description
            "Any SR-aware or AR-unaware services could augment this containe
r";
        reference "Segment Routing Service Programming Architecture.";
    }
}

grouping service-programmings {
    container service-programming {
        description
            "service programming container.
            Any new services programming added could augment
            this container to support that specific services.
            Currently in this model, only service proxy
            is defined. (i.e) For example if
            a Firewall services needs to be added then
            they could augment this container and
            extend this model";

        list service-program {
            key "name";
            description
                "Service program is keyed by the service program name";

            leaf name {
                mandatory true;
                type leafref {
                    path "/rt:routing/sr:segment-routing/"
                        + "sr-svc-pgm:service-programming/"
                        + "sr-svc-pgm:service-program/"
                        + "sr-svc-pgm:service-programming-info/"
                        + "sr-svc-pgm:service-name";
                }
            }
        }
    }
}

```



```
        uses service-sid-binding;
        uses service-programming-infos;
    }
}

/*
 * MPLS/SRv6 SR service programming using dynamic SID allocation
 */
augment "/rt:routing/sr:segment-routing" {
    description
        "Augmenting the segment-routing to add SR service programming";

    uses service-programmings;
}

/*
 * MPLS SR service programming using static MPLS binding SID
 */
augment "/rt:routing/sr:segment-routing/srmppls:sr-mpls/srmppls:bindings" {
    description
        "Augmenting the segment-routing MPLS static binding to add static
        MPLS SR service programming";

    container mpls-static-service-programming {
        description
            "Augmenting the MPLS segment-routing bindings with the SR service
            programming";
        list service-program {
            key "name";
            description
                "Service program is keyed by the service program name";

            leaf name {
                mandatory true;
                type leafref {
                    path "/rt:routing/sr:segment-routing/"
                        + "sr-svc-pgm:service-programming/"
                        + "sr-svc-pgm:service-program/"
                        + "sr-svc-pgm:service-programming-info/"
                        + "sr-svc-pgm:service-name";
                }
            }

            leaf sid {
                mandatory true;
                type rt-types:mpls-label;
                description

```

```

        "MPLS SID value.";
    }

    uses service-programming-infos {
        /*
         * In the case of MPLs static binding configuration
         * the dataplane is set to mpls and not allowed to
         * configure
         */
        refine service-programming-info/dataplane {
            mandatory false;
            default mpls;
            config false;
        }
    }
}

/*
 * SRv6 SR service programming using static SRv6 binding SID
 */
augment "/rt:routing/sr:segment-routing/srv6:srv6/srv6:locators/srv6:locator
" {
    description
        "Augmenting the segment-routing SRv6 static to add static binding to
        SRv6 SR service programming";

    container end-AS {
        description
            "End-AS - Static Proxy SID behaviour";
        list service-program {
            key "name";
            description
                "Service program is keyed by the service program name";

            leaf name {
                mandatory true;
                type leafref {
                    path "/rt:routing/sr:segment-routing/"
                        + "sr-svc-pgm:service-programming/"
                        + "sr-svc-pgm:service-program/"
                        + "sr-svc-pgm:service-programming-info/"
                        + "sr-svc-pgm:service-name";
                }
            }
        }

        uses service-programming-infos {

```

```
    /*
    * In the case of SRv6 static binding configuration
    * the dataplane is set to mpls and not allowed to
    * configure
    */
    refine service-programming-info/dataplane {
        config false;
        mandatory false;
        default srv6;
    }
    refine service-programming-info/behaviour {
        config false;
        //when "service-programming-info/dataplane = 'srv6'";
        mandatory false;
        default sr-svc-pgm-types:static-proxy;
    }
}
}

container end-AD {
    description
        "End.AD - Dynamic Proxy SID behaviour";
    list service-program {
        key "name";
        description
            "Service program is keyed by the service program name";

        leaf name {
            mandatory true;
            type leafref {
                path "/rt:routing/sr:segment-routing/"
                    + "sr-svc-pgm:service-programming/"
                    + "sr-svc-pgm:service-program/"
                    + "sr-svc-pgm:service-programming-info/"
                    + "sr-svc-pgm:service-name";
            }
        }
    }

    uses service-programming-infos {

        refine service-programming-info/dataplane {
            config false;
            mandatory false;
            default srv6;
        }
        refine service-programming-info/behaviour {
```

```
        //when "service-programming-info/dataplane = 'srv6'";
        config false;
        mandatory false;
        default sr-svc-pgm-types:dynamic-proxy;
    }
}
}

container end-AM {
    description
        "End.AD - Masquerading Proxy SID behaviour";
    list service-program {
        key "name";
        description
            "Service program is keyed by the service program name";

        leaf name {
            mandatory true;
            type leafref {
                path "/rt:routing/sr:segment-routing/"
                    + "sr-svc-pgm:service-programming/"
                    + "sr-svc-pgm:service-program/"
                    + "sr-svc-pgm:service-programming-info/"
                    + "sr-svc-pgm:service-name";
            }
        }
    }

    uses service-programming-infos {

        refine service-programming-info/dataplane {
            config false;
            mandatory false;
            default srv6;
        }
        refine service-programming-info/behaviour {
            //when "service-programming-info/dataplane = 'srv6'";
            mandatory false;
            default sr-svc-pgm-types:Masquerading-proxy;
        }
    }
}
}
```

```

notification service-program-oper-status {
  description
    "This notification is sent when there is a change in the service
    program oper status.";
  leaf name {
    mandatory true;
    type leafref {
      path "/rt:routing/sr:segment-routing/"
        + "sr-svc-pgm:service-programming/"
        + "sr-svc-pgm:service-program/"
        + "sr-svc-pgm:name";
    }
    description
      "Service program name to identify a specific programming.";
  }

  leaf oper-status {
    mandatory true;
    type leafref {
      path "/rt:routing/sr:segment-routing/"
        + "sr-svc-pgm:service-programming/"
        + "sr-svc-pgm:service-program/"
        + "sr-svc-pgm:service-programming-info/"
        + "sr-svc-pgm:oper-status";
    }
    description
      "Service program operational status.";
  }
}
}
}
<CODE ENDS>

```

Figure 10: ietf-sr-service-programming.yang

#### 4.4. SR Service Proxy

Following are the SR service programming service proxy model definition.

```

<CODE BEGINS> file "ietf-sr-service-programming-proxy.yang"
-->
module ietf-sr-service-programming-proxy {
  yang-version 1.1;

  namespace "urn:ietf:params:xml:ns:yang:ietf-sr-service-programming-proxy";
  prefix "sr-svc-proxy";

```

```
import ietf-yang-types {
  prefix yang;
}

import ietf-routing {
  prefix rt;
  reference "RFC 8349: A YANG Data Model for Routing
            Management (NMDA Version)";
}

import ietf-inet-types {
  prefix "inet";
}

import ietf-segment-routing {
  prefix sr;
}

import ietf-sr-service-programming {
  prefix "sr-svc-pgm";
}

import ietf-sr-service-programming-types {
  prefix "sr-svc-pgm-types";
}

import ietf-routing-types {
  prefix "rt-types";
}

import ietf-srv6-types {
  prefix "srv6-types";
}

import ietf-segment-routing-mpls {
  prefix sr-mpls;
}

organization "IETF SPRING Working Group";

contact
  "WG Web:  <http://tools.ietf.org/wg/spring/>
  WG List:  <mailto:spring@ietf.org>

  Editor:   Jaganbabu Rajamanickam
            <mailto:jrajan@cs.cisco.com>

  Editor:   Kamran Raza
```

```
<mailto:skraza@cisco.com>

Editor: Daniel Bernier
<mailto:daniel.bernier@bell.ca>

Editor: Gaurav Dawra
<mailto:gdawra.ietf@gmail.com>

Editor: Cheng Li
<mailto:c.l@huawei.com>";

grouping service-proxy-parameters {

    leaf out-interface-name {
        mandatory true;
        type string;
        description
            "Interface name on which the packet sent to the service endpoint";
    }

    leaf in-interface-name {
        mandatory true;
        type string;
        description
            "Interface name on which the packet received from the service endpoint
";
    }
}

grouping mpls-packet-cache-info {
    description
        "MPLS Label stack";

    list mpls-sids {
        key "index";

        leaf index {
            type uint8 {
                range "1..16";
            }
            description
                "cache index - MPLS Label stack index";
        }

        leaf mpls-label {
            mandatory true;
            type rt-types:mpls-label;
            description
                "MPLS Label value.";
        }
    }
}
```

```
    }
  }
}

grouping srv6-packet-cache-info {
  description
    "SRv6 SID stack";

  leaf ipv6-source-address {
    type inet:ipv6-address;
    description
      "IPv6 source address that needs in the case if SRv6.";
  }
  list srv6-sids {
    key "index";

    leaf index {
      type uint8 {
        range "1..16";
      }
      description
        "cache index - SRv6 SID index";
    }

    leaf srv6-sid {
      mandatory true;
      type srv6-types:srv6-sid;
      description
        "SRv6 SID.";
    }
  }
}

grouping service-proxy-packet-cache-info {
  description
    "SRv6 Proxy header cache";

  container packet-cache-info {

    choice cache-type {
      mandatory true;
      case mpls {

        when "/rt:routing/sr:segment-routing
/sr-svc-pgm:service-programming
/sr-svc-pgm:service-program
/sr-svc-pgm:service-programming-info
/sr-svc-pgm:dataplane = 'mpls'";
```



```
        uses mpls-packet-cache-info;
    }
    case srv6 {

        when "/rt:routing/sr:segment-routing/sr-svc-pgm:service-programming
            /sr-svc-pgm:service-program
            /sr-svc-pgm:service-programming-info
            /sr-svc-pgm:dataplane = 'srv6'";

        uses srv6-packet-cache-info;
    }
}

grouping static-service-proxy {
    container static-proxy {
        when "/rt:routing/sr:segment-routing/sr-svc-pgm:service-programming
            /sr-svc-pgm:service-program
            /sr-svc-pgm:service-programming-info
            /sr-svc-pgm:behaviour = 'static-proxy'";
        description
            "Parameters related to static service proxy";

        leaf inner-packet-type {
            mandatory true;
            type identityref {
                base sr-svc-pgm-types:service-proxy-inner-pkt-type;
            }
            description
                "Defines the expected inner packet type";
        }

        leaf next-hop {
            when "(../inner-packet-type = 'IPv4' or ../inner-packet-type = 'IPv6')";
            type yang:mac-address;
            description
                "Nexthop Ethernet address for inner packet type IPv4/IPv6";
        }
        uses service-proxy-parameters;
        uses service-proxy-packet-cache-info;
    }
}

grouping dynamic-service-proxy {
    container dynamic-proxy {
        when "/rt:routing/sr:segment-routing/sr-svc-pgm:service-programming
            /sr-svc-pgm:service-program
```

```
        /sr-svc-pgm:service-programming-info
        /sr-svc-pgm:behaviour = 'dynamic-proxy'";
    description
        "Parameters related to dynamic service proxy";
    uses service-proxy-parameters;
}
}

grouping masquerading-service-parameters {

    leaf next-hop {
        type yang:mac-address;
        description
            "Nexthop Ethernet address";
    }
    uses service-proxy-parameters;
}

grouping masquerading-service-proxy {
    container masquerading-proxy {
        description
            "Parameters related to masquerading service proxy";

        when "/rt:routing/sr:segment-routing
            /sr-svc-pgm:service-programming
            /sr-svc-pgm:service-program
            /sr-svc-pgm:service-programming-info
            /sr-svc-pgm:dataplane = 'srv6' and /rt:routing
            /sr:segment-routing/sr-svc-pgm:service-programming
            /sr-svc-pgm:service-program
            /sr-svc-pgm:service-programming-info
            /sr-svc-pgm:behaviour = 'Masquerading-proxy'";

        uses masquerading-service-parameters;
    }
}

grouping service-proxy-programming {
    container service-proxy {

        choice proxy-type {
            mandatory true;
            case static {
                uses static-service-proxy;
            }
            case dynamic {
                uses dynamic-service-proxy;
            }
        }
    }
}
```

```
        case masquerading {
            uses masquerading-service-proxy;
        }
    }
}

augment "/rt:routing/sr:segment-routing/
    sr-svc-pgm:service-programming/
    sr-svc-pgm:service-program/
    sr-svc-pgm:service-programming-info/
    sr-svc-pgm:sr-services" {
    description
        "Augmenting the segment-routing bindings to add SR-unaware
        service programming";

    uses service-proxy-programming;
}

grouping static-mpls-service-proxy-programming {
    container static-mpls-service-proxy {

        choice proxy-type {
            mandatory true;
            case static {
                uses static-service-proxy;
            }
            case dynamic {
                uses dynamic-service-proxy;
            }
        }
    }
}

augment "/rt:routing/sr:segment-routing/
    sr-mpls:sr-mpls/sr-mpls:bindings/
    sr-svc-pgm:mpls-static-service-programming/
    sr-svc-pgm:service-program/
    sr-svc-pgm:service-programming-info/
    sr-svc-pgm:sr-services" {
    uses static-mpls-service-proxy-programming;
}

}
<CODE ENDS>
```

Figure 11: ietf-sr-service-programming-proxy.yang

## 5. Security Considerations

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8446].

The Network Configuration Access Control Model (NACM) [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

There are a number of data nodes defined in this YANG module that are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) to these data nodes without proper protection can have a negative effect on network operations.

Some of the readable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes.

It goes without saying that this specification also inherits the security considerations captured in the SRv6 specification document [I-D.ietf-spring-sr-service-programming].

## 6. IANA Considerations

This document requests the registration of the following URIs in the IETF "XML registry" [RFC3688]:

URI	Registrant	XML
urn:ietf:params:xml:ns:yang:ietf-service-function-types	The IESG	N/A
urn:ietf:params:xml:ns:yang:ietf-sr-service-programming-types	The IESG	N/A
urn:ietf:params:xml:ns:yang:ietf-sr-service-programming	The IESG	N/A
urn:ietf:params:xml:ns:yang:ietf-sr-service-programming-proxy	The IESG	N/A

Table 1

This document requests the registration of the following YANG modules in the "YANG Module Names" registry [RFC6020]:

Name	Namespace	Prefix	Reference
ietf-service-function-types	urn:ietf:params:xml:ns:yang:ietf-service-function-types	service-function-types	This document
ietf-sr-service-programming-types	urn:ietf:params:xml:ns:yang:ietf-sr-service-programming-types	ietf-sr-service-programming-types	This document
ietf-sr-service-programming	urn:ietf:params:xml:ns:yang:ietf-sr-service-programming	ietf-sr-service-programming	This document
ietf-sr-service-programming-proxy	urn:ietf:params:xml:ns:yang:ietf-sr-service-programming-proxy	ietf-sr-service-programming-proxy	This document

Table 2

-- RFC Editor: Replace "This document" with the document RFC number at time of publication, and remove this note.

## 7. Acknowledgments

The authors would like to acknowledge Francois Clad, Ketan Talaulikar, and Darren Dukes for their review of some of the contents in this document.

## 8. Normative References

[I-D.ietf-spring-segment-routing-policy]  
 Filsfils, C., Talaulikar, K., Voyer, D., Bogdanov, A., and P. Mattes, "Segment Routing Policy Architecture", Work in Progress, Internet-Draft, draft-ietf-spring-segment-routing-policy-14, 25 October 2021,  
<https://www.ietf.org/archive/id/draft-ietf-spring-segment-routing-policy-14.txt>.

- [I-D.ietf-spring-sr-service-programming]  
Clad, F., Xu, X., Filsfils, C., Bernier, D., Li, C.,  
Decraene, B., Ma, S., Yadlapalli, C., Henderickx, W., and  
S. Salsano, "Service Programming with Segment Routing",  
Work in Progress, Internet-Draft, draft-ietf-spring-sr-  
service-programming-05, 10 September 2021,  
<[https://www.ietf.org/archive/id/draft-ietf-spring-sr-  
service-programming-05.txt](https://www.ietf.org/archive/id/draft-ietf-spring-sr-service-programming-05.txt)>.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate  
Requirement Levels", BCP 14, RFC 2119,  
DOI 10.17487/RFC2119, March 1997,  
<<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688,  
DOI 10.17487/RFC3688, January 2004,  
<<https://www.rfc-editor.org/info/rfc3688>>.
- [RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for  
the Network Configuration Protocol (NETCONF)", RFC 6020,  
DOI 10.17487/RFC6020, October 2010,  
<<https://www.rfc-editor.org/info/rfc6020>>.
- [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed.,  
and A. Bierman, Ed., "Network Configuration Protocol  
(NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011,  
<<https://www.rfc-editor.org/info/rfc6241>>.
- [RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure  
Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011,  
<<https://www.rfc-editor.org/info/rfc6242>>.
- [RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF  
Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017,  
<<https://www.rfc-editor.org/info/rfc8040>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC  
2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174,  
May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams",  
BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018,  
<<https://www.rfc-editor.org/info/rfc8340>>.
- [RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration  
Access Control Model", STD 91, RFC 8341,  
DOI 10.17487/RFC8341, March 2018,  
<<https://www.rfc-editor.org/info/rfc8341>>.

- [RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <<https://www.rfc-editor.org/info/rfc8342>>.
- [RFC8402] Filtsfils, 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>>.
- [RFC8407] Bierman, A., "Guidelines for Authors and Reviewers of Documents Containing YANG Data Models", BCP 216, RFC 8407, DOI 10.17487/RFC8407, October 2018, <<https://www.rfc-editor.org/info/rfc8407>>.
- [RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <<https://www.rfc-editor.org/info/rfc8446>>.
- [RFC8754] Filtsfils, C., Ed., Dukes, D., Ed., Previdi, S., Leddy, J., Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header (SRH)", RFC 8754, DOI 10.17487/RFC8754, March 2020, <<https://www.rfc-editor.org/info/rfc8754>>.
- [RFC9020] Litkowski, S., Qu, Y., Lindem, A., Sarkar, P., and J. Tantsura, "YANG Data Model for Segment Routing", RFC 9020, DOI 10.17487/RFC9020, May 2021, <<https://www.rfc-editor.org/info/rfc9020>>.

Authors' Addresses

Jaganbabu Rajamanickam  
Cisco Systems

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

Kamran Raza  
Cisco Systems

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

Daniel Bernier  
Bell Canada

Email: [daniel.bernier@bell.ca](mailto:daniel.bernier@bell.ca)



Gaurav Dawra  
LinkedIn

Email: gdawra.ietf@gmail.com

Cheng Li  
Huawei

Email: c.l@huawei.com

Network Work group  
Internet-Draft  
Intended status: Standards Track  
Expires: April 30, 2021

N. Nainar, Ed.  
C. Pignataro, Ed.  
Z. Ali  
C. Filsfils  
Cisco  
T. Saad  
Juniper  
October 27, 2020

Segment Routing Generic TLV for MPLS Label Switched Path (LSP) Ping/  
Traceroute  
draft-nainar-mpls-spring-lsp-ping-sr-generic-sid-04

## Abstract

RFC8402 introduces Segment Routing architecture that leverages source routing and tunneling paradigms and can be directly applied to the Multi Protocol Label Switching (MPLS) data plane. A node steers a packet through a controlled set of instructions called segments, by prepending the packet with Segment Routing header. SR architecture defines different types of segments with different forwarding semantics associated. SR can be applied to the MPLS directly and to IPv6 dataplane using a new routing header.

RFC8287 defines the extensions to MPLS LSP Ping and Traceroute for Segment Routing IGP-Prefix and IGP-Adjacency Segment Identifier (SIDs) with an MPLS data plane. Various SIDs are proposed as part of SR architecture with different associated instructions that raises a need to come up with new Target FEC Stack Sub-TLV for each such SIDs.

This document defines a new Target FEC Stack Sub-TLV that is used to validate the instruction associated with any SID.

## Status of This Memo

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

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

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

This Internet-Draft will expire on April 30, 2021.

## Copyright Notice

Copyright (c) 2020 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
1.1. Challenges with Existing Mechanism . . . . .	3
2. Requirements notation . . . . .	3
3. Terminology . . . . .	4
4. Target FEC Stack sub-TLV for Segment Routing SID . . . . .	4
4.1. Segment Routing Generic Label . . . . .	4
4.2. FEC for Path validation . . . . .	4
5. Procedures . . . . .	5
5.1. SID to Interface Mapping . . . . .	5
5.2. Initiator behavior . . . . .	6
5.2.1. SRGL in Target FEC Stack TLV . . . . .	6
5.3. Responder behavior . . . . .	7
5.4. PHP flag behavior . . . . .	7
6. IANA Considerations . . . . .	8
6.1. New Target FEC Stack Sub-TLVs . . . . .	8
6.2. Security Considerations . . . . .	8
7. Acknowledgement . . . . .	8
8. Contributors . . . . .	8
9. References . . . . .	8
9.1. Normative References . . . . .	8
9.2. Informative References . . . . .	9
Authors' Addresses . . . . .	10

## 1. Introduction

[RFC8402] introduces and describes a Segment Routing architecture that leverages the source routing and tunneling paradigms. A node steers a packet through a controlled set of instructions called segments, by prepending the packet with Segment Routing header. A

detailed definition of the Segment Routing architecture is available in [RFC8402]

As described in [RFC8402] and [I-D.ietf-spring-segment-routing-mpls], the Segment Routing architecture can be directly applied to an MPLS data plane, the Segment identifier (Segment ID) will be of 20-bits size and the Segment Routing header is the label stack.

### 1.1. Challenges with Existing Mechanism

[RFC8287] defines the mechanism to perform LSP Ping and Traceroute for Segment Routing with MPLS data plane. [RFC8287] defines the Target FEC Stack Sub-TLVs for IGP-Prefix Segment ID and IGP-Adjacency Segment ID.

There are various other Segment IDs proposed by different documents that are applicable for SR architecture. [I-D.ietf-idr-bgp-prefix-sid] defines BGP Prefix Segment ID, [I-D.ietf-idr-bgppls-segment-routing-epe] defines BGP Peering Segment ID such as Peer Node SID, Peer Adj SID and Peer Set SID. [I-D.sivabalan-pce-binding-label-sid] defines Path Binding Segment ID. As SR evolves for different usecases, we may see more types of SIDs defined in the future. This raises a need to propose new Target FEC Stack Sub-TLV for each such Segment ID that may need specific or network wide software upgrade to support such new Target FEC Stack Sub-TLVs.

So instead of proposing different Target FEC Stack Sub-TLV for each SID, this document attempt to propose a SR Generic Label Sub-TLV for Target FEC Stack TLV with the procedure to validate the associated instruction.

This document describes the new Target FEC Stack Sub-TLV that carries the SID and the procedure to use LSP Ping and Traceroute using the new sub-tlv to support path validation and fault isolation for any SR Segment IDs. This document neither deprecates any existing Target FEC Stack Sub-TLVs nor precludes defining new Sub-TLVs in the future.

### 2. Requirements notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119] RFC 8174 [RFC8174] when and only when, they appear in all capitals, as shown here.

### 3. Terminology

This document uses the terminologies defined in [RFC8402], [RFC8029], readers are expected to be familiar with it.

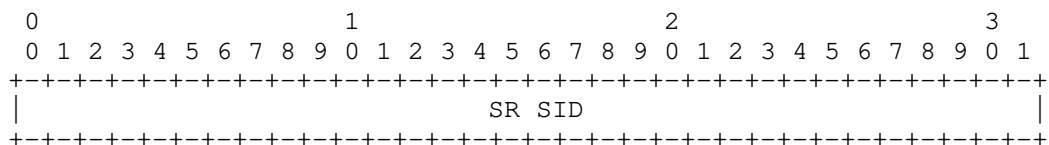
### 4. Target FEC Stack sub-TLV for Segment Routing SID

Following the procedure defined in [RFC8029], below defined Target FEC Stack Sub-TLV will be included for each labels in the stack. The below Sub-TLV is defined for Target FEC Stack TLV (Type 1), the Reverse-Path Target FEC Stack TLV (Type 16), and the Reply Path TLV (Type 21).

sub-Type	Value Field
TBD1	Segment Routing Generic Label (SRGL)

#### 4.1. Segment Routing Generic Label

The format of the Sub-TLV is as specified below:



SR SID

Carries 20 bits of Segment ID that is used for validating the instruction.

#### 4.2. FEC for Path validation

In SR architecture, any SID is associated with topology or service instruction. While the topology instruction steers the packet over best path or specific path, the service instruction instructs the type of service to be applied on the packet.

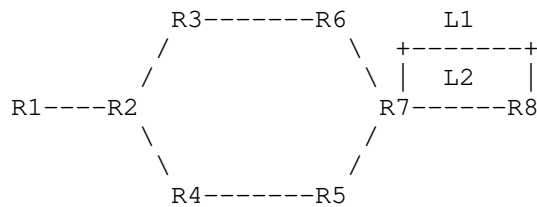


Figure 1: Segment Routing network

The Node Segment IDs for Rx for Algo 0 is 16000x. (Ex: For R1, it is 160001)  
 The Node Segment IDs for Rx for Algo 128 is 16128x. (Ex: For R1, it is 161281)

9178 --> Adjacency Segment ID from R7 to R8 over link L1.  
 9278 --> Adjacency Segment ID from R7 to R8 over link L2.  
 9378 --> Parallel Adjacency Segment ID from R7 to R8 over Link L1 or L2.  
 9187 --> Adjacency Segment ID from R8 to R7 over link L1.  
 9287 --> Adjacency Segment ID from R8 to R7 over link L2.  
 9387 --> Parallel Adjacency Segment ID from R8 to R7 over Link L1 or L2.

The instruction associated with any SID can be validated by verifying if the segment is terminated on the correct node and optionally received over the correct incoming interface. In Figure 1, in order to validate the SID 9178, R1 can use {(SID=9178)} as FEC in Target FEC Stack Sub-TLV.

## 5. Procedures

This section describes the procedure to validate SR Generic Label Sub-TLV.

### 5.1. SID to Interface Mapping

Any End point MAY maintain a SID to Interface mapping table that maintains the below:

- o All the local Prefix/Node SID with any SR enabled interface as incoming interface.
- o All the Adj-SIDs assigned by directly connected neighbor nodes with the relevant interface incoming interface.

In Figure 1, R8 maintains 160008 and 161288 with Incoming interface as any SR enabled interface. Similarly, R8 maintains 9178 with Link L1 as incoming interface, 9278 with Link L2 as incoming interface and 9378 with Link L1 or L2 as incoming interface.

How this mapping is populated and maintained is a local implementation matter. It can be populated based on the IGP database or can be based on a query to Path Computation Element (PCE) controller. The mapping can be persistent or on-demand triggered by receiving LSP Ping Request.

## 5.2. Initiator behavior

This section defines the Target FEC Stack TLV construction mechanism by an initiator when using SR Generic Label Sub-TLV.

### Ping

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

Initiator MAY include FECs corresponding to some or all of segments imposed in the label stack by the initiator to communicate the segments traversed.

### Traceroute

Initiator MUST initially include FECs corresponding to all of segments imposed in the label stack.

When a received echo reply contains FEC Stack Change TLV with one or more of original segment(s) being popped, initiator MAY remove corresponding FEC(s) from Target FEC Stack TLV in the next (TTL+1) traceroute request as defined in section 4.6 of [RFC8029].

When a received echo reply does not contain FEC Stack Change TLV, initiator MUST NOT attempt to remove FEC(s) from Target FEC Stack TLV in the next (TTL+1) traceroute request.

### 5.2.1. SRGL in Target FEC Stack TLV

When the last segment ID in the label stack is IGP Prefix SID, Adj-SID, Binding SID, BGP Prefix SID or BGP Peering SID, set the SR SID field to the Segment ID value advertised by the LSP End Point. When the SID is advertised as index, the Segment ID value MUST be derived based on the index and the SRGB advertised by the LSP End Point.

How the above values are derived is a local implementation matter. It can be manually defined using CLI knob while triggering the LSP Ping Request or can use other mechanisms like querying the local database.

### 5.3. Responder behavior

Step 4a defined in Section 7.4 of [RFC8287] is updated as below:

If the Label-stack-depth is 0 and Target FEC Stack Sub-TLV at FEC-stack-depth is TBD1 (SRGL) {

- \* Set the Best-return-code to 10 when the responding node is not the LSP End Point for SR SID.
  - \* Set the Best-return-code to 35, if Interface-I does not match the SID to Interface mapping for the received SR SID.
  - \* set FEC-Status to 1, and return.
- }

If the Label-stack-depth is greater than 0 and Target FEC Stack Sub-TLV at FEC-stack-depth is TBD1 (SRGL), {

- \* If the Label at Label-stack-depth is Imp-null {
  - + Set the Best-return-code to 10 when the responding node is not the LSP End Point for the SR SID.
  - + Set the Best-return-code to 35, if Interface-I does not match the SID to Interface mapping for the received SR SID.
  - + set FEC-Status to 1, and return.
- }
- \* Else:
  - + Set the Best-return-code to 10 when the index derived from the label at Label-stack-depth is not advertised by LSP End Point.
  - + set FEC-Status to 1, and return.
- }

### 5.4. PHP flag behavior

Section 7.2 of [RFC8287] explains the behavior for FEC stack change for Adjacency Segment ID. The same procedure is applicable for BGP Peering SID as well.



## 6. IANA Considerations

### 6.1. New Target FEC Stack Sub-TLVs

IANA is requested to assign three new Sub-TLVs from "Sub-TLVs for TLV Types 1, 16 and 21" sub-registry from the "Multi-Protocol Label Switching (MPLS) Label Switched Paths (LSPs) Ping Parameters" [IANA-MPLS-LSP-PING] registry.

Sub-Type	Sub-TLV Name	Reference
TBD1	Segment Routing Generic Label	Section 4.1 of this document

### 6.2. Security Considerations

This document defines additional MPLS LSP Ping Sub-TLVs and follows the mechanisms defined in [RFC8029]. All the security considerations defined in [RFC8029] will be applicable for this document, and in addition, they do not impose any additional security challenges to be considered.

## 7. Acknowledgement

TBD

## 8. Contributors

Danial Johari, Cisco Systems

## 9. References

### 9.1. Normative References

[I-D.ietf-idr-bgp-prefix-sid]  
 Previdi, S., Filsfils, C., Lindem, A., Sreekantiah, A.,  
 and H. Gredler, "Segment Routing Prefix SID extensions for  
 BGP", draft-ietf-idr-bgp-prefix-sid-27 (work in progress),  
 June 2018.

[I-D.ietf-idr-bgppls-segment-routing-epe]  
 Previdi, S., Talaulikar, K., Filsfils, C., Patel, K., Ray,  
 S., and J. Dong, "BGP-LS extensions for Segment Routing  
 BGP Egress Peer Engineering", draft-ietf-idr-bgppls-  
 segment-routing-epe-19 (work in progress), May 2019.

- [I-D.sivabalan-pce-binding-label-sid]  
Sivabalan, S., Filsfils, C., Tantsura, J., Hardwick, J.,  
Previdi, S., and C. Li, "Carrying Binding Label/Segment-ID  
in PCE-based Networks.", draft-sivabalan-pce-binding-  
label-sid-07 (work in progress), July 2019.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate  
Requirement Levels", BCP 14, RFC 2119,  
DOI 10.17487/RFC2119, March 1997,  
<<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC8029] Kompella, K., Swallow, G., Pignataro, C., Ed., Kumar, N.,  
Aldrin, S., and M. Chen, "Detecting Multiprotocol Label  
Switched (MPLS) Data-Plane Failures", RFC 8029,  
DOI 10.17487/RFC8029, March 2017,  
<<https://www.rfc-editor.org/info/rfc8029>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC  
2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174,  
May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8287] Kumar, N., Ed., Pignataro, C., Ed., Swallow, G., Akiya,  
N., Kini, S., and M. Chen, "Label Switched Path (LSP)  
Ping/Traceroute for Segment Routing (SR) IGP-Prefix and  
IGP-Adjacency Segment Identifiers (SIDs) with MPLS Data  
Planes", RFC 8287, DOI 10.17487/RFC8287, December 2017,  
<<https://www.rfc-editor.org/info/rfc8287>>.
- [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>>.

## 9.2. Informative References

- [I-D.ietf-spring-segment-routing-mpls]  
Bashandy, A., Filsfils, C., Previdi, S., Decraene, B.,  
Litkowski, S., and R. Shakir, "Segment Routing with MPLS  
data plane", draft-ietf-spring-segment-routing-mpls-22  
(work in progress), May 2019.
- [IANA-MPLS-LSP-PING]  
IANA, "Multi-Protocol Label Switching (MPLS) Label  
Switched Paths (LSPs) Ping Parameters",  
<[http://www.iana.org/assignments/mpls-lsp-ping-parameters/  
mpls-lsp-ping-parameters.xhtml](http://www.iana.org/assignments/mpls-lsp-ping-parameters/mpls-lsp-ping-parameters.xhtml)>.

Authors' Addresses

Nagendra Kumar Nainar (editor)  
Cisco Systems, Inc.  
7200-12 Kit Creek Road  
Research Triangle Park, NC 27709-4987  
US

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

Carlos Pignataro (editor)  
Cisco Systems, Inc.  
7200-11 Kit Creek Road  
Research Triangle Park, NC 27709-4987  
US

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

Zafar Ali  
Cisco Systems, Inc.

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

Clarence Filsfils  
Cisco Systems, Inc.

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

Tarek Saad  
Juniper Networks

Email: [tsaad@juniper.net](mailto:tsaad@juniper.net)

Network Work group  
Internet-Draft  
Intended status: Standards Track  
Expires: 28 May 2022

N. Nainar, Ed.  
C. Pignataro, Ed.  
Z. Ali  
C. Filsfils  
Cisco  
T. Saad  
Juniper  
24 November 2021

Segment Routing Generic TLV for MPLS Label Switched Path (LSP) Ping/  
Traceroute  
draft-nainar-mpls-spring-lsp-ping-sr-generic-sid-06

## Abstract

RFC8402 introduces Segment Routing architecture that leverages source routing and tunneling paradigms and can be directly applied to the Multi Protocol Label Switching (MPLS) data plane. A node steers a packet through a controlled set of instructions called segments, by prepending the packet with Segment Routing header. SR architecture defines different types of segments with different forwarding semantics associated. SR can be applied to the MPLS directly and to IPv6 dataplane using a new routing header.

RFC8287 defines the extensions to MPLS LSP Ping and Traceroute for Segment Routing IGP-Prefix and IGP-Adjacency Segment Identifier (SIDs) with an MPLS data plane. Various SIDs are proposed as part of SR architecture with different associated instructions that raises a need to come up with new Target FEC Stack Sub-TLV for each such SIDs.

This document defines a new Target FEC Stack Sub-TLV that is used to validate the instruction associated with any SID.

## 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 28 May 2022.

#### Copyright Notice

Copyright (c) 2021 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 Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

#### Table of Contents

1. Introduction . . . . .	3
1.1. Challenges with Existing Mechanism . . . . .	3
2. Requirements notation . . . . .	4
3. Terminology . . . . .	4
4. Target FEC Stack sub-TLV for Segment Routing SID . . . . .	4
4.1. Segment Routing Generic Label . . . . .	4
4.2. FEC for Path validation . . . . .	4
5. Procedures . . . . .	5
5.1. SID to Interface Mapping . . . . .	5
5.2. Initiator behavior . . . . .	6
5.2.1. SRGL in Target FEC Stack TLV . . . . .	6
5.3. Responder behavior . . . . .	7
5.4. PHP flag behavior . . . . .	7
6. IANA Considerations . . . . .	8
6.1. New Target FEC Stack Sub-TLVs . . . . .	8
6.2. Security Considerations . . . . .	8
7. Acknowledgement . . . . .	8
8. Contributors . . . . .	8
9. References . . . . .	8
9.1. Normative References . . . . .	8
9.2. Informative References . . . . .	9
Authors' Addresses . . . . .	10

## 1. Introduction

[RFC8402] introduces and describes a Segment Routing architecture that leverages the source routing and tunneling paradigms. A node steers a packet through a controlled set of instructions called segments, by prepending the packet with Segment Routing header. A detailed definition of the Segment Routing architecture is available in [RFC8402]

As described in [RFC8402] and [I-D.ietf-spring-segment-routing-mpls], the Segment Routing architecture can be directly applied to an MPLS data plane, the Segment identifier (Segment ID) will be of 20-bits size and the Segment Routing header is the label stack.

### 1.1. Challenges with Existing Mechanism

[RFC8287] defines the mechanism to perform LSP Ping and Traceroute for Segment Routing with MPLS data plane. [RFC8287] defines the Target FEC Stack Sub-TLVs for IGP-Prefix Segment ID and IGP-Adjacency Segment ID.

There are various other Segment IDs proposed by different documents that are applicable for SR architecture.

[I-D.ietf-idr-bgp-prefix-sid] defines BGP Prefix Segment ID,

[I-D.ietf-idr-bgppls-segment-routing-epe] defines BGP Peering Segment ID such as Peer Node SID, Peer Adj SID and Peer Set SID.

[I-D.sivabalan-pce-binding-label-sid] defines Path Binding Segment ID. As SR evolves for different usecases, we may see more types of SIDs defined in the future. This raises a need to propose new Target FEC Stack Sub-TLV for each such Segment ID that may need specific or network wide software upgrade to support such new Target FEC Stack Sub-TLVs.

So instead of proposing different Target FEC Stack Sub-TLV for each SID, this document attempt to propose a SR Generic Label Sub-TLV for Target FEC Stack TLV with the procedure to validate the associated instruction.

This document describes the new Target FEC Stack Sub-TLV that carries the SID and the procedure to use LSP Ping and Traceroute using the new sub-tlv to support path validation and fault isolation for any SR Segment IDs. This document neither deprecates any existing Target FEC Stack Sub-TLVs nor precludes defining new Sub-TLVs in the future.

## 2. Requirements notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119] RFC 8174 [RFC8174] when and only when, they appear in all capitals, as shown here.

## 3. Terminology

This document uses the terminologies defined in [RFC8402], [RFC8029], readers are expected to be familiar with it.

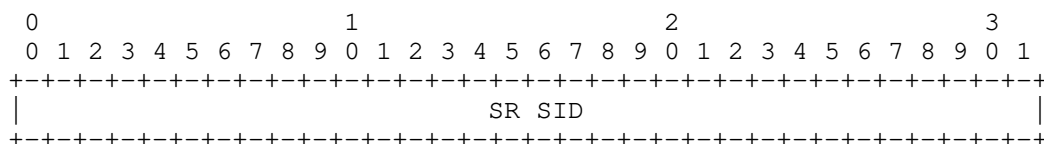
## 4. Target FEC Stack sub-TLV for Segment Routing SID

Following the procedure defined in [RFC8029], below defined Target FEC Stack Sub-TLV will be included for each labels in the stack. The below Sub-TLV is defined for Target FEC Stack TLV (Type 1), the Reverse-Path Target FEC Stack TLV (Type 16), and the Reply Path TLV (Type 21).

sub-Type	Value Field
TBD1	Segment Routing Generic Label (SRGL)

### 4.1. Segment Routing Generic Label

The format of the Sub-TLV is as specified below:



SR SID

Carries 20 bits of Segment ID that is used for validating the instruction.

### 4.2. FEC for Path validation

In SR architecture, any SID is associated with topology or service instruction. While the topology instruction steers the packet over best path or specific path, the service instruction instructs the type of service to be applied on the packet.

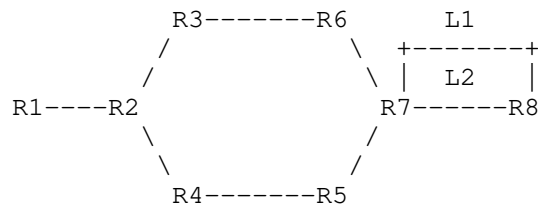


Figure 1: Segment Routing network

The Node Segment IDs for Rx for Algo 0 is 16000x. (Ex: For R1, it is 160001)  
 The Node Segment IDs for Rx for Algo 128 is 16128x. (Ex: For R1, it is 161281)

9178 --> Adjacency Segment ID from R7 to R8 over link L1.  
 9278 --> Adjacency Segment ID from R7 to R8 over link L2.  
 9378 --> Parallel Adjacency Segment ID from R7 to R8 over Link L1 or L2.  
 9187 --> Adjacency Segment ID from R8 to R7 over link L1.  
 9287 --> Adjacency Segment ID from R8 to R7 over link L2.  
 9387 --> Parallel Adjacency Segment ID from R8 to R7 over Link L1 or L2.

The instruction associated with any SID can be validated by verifying if the segment is terminated on the correct node and optionally received over the correct incoming interface. In Figure 1, in order to validate the SID 9178, R1 can use {(SID=9178)} as FEC in Target FEC Stack Sub-TLV.

## 5. Procedures

This section describes the procedure to validate SR Generic Label Sub-TLV.

### 5.1. SID to Interface Mapping

Any End point MAY maintain a SID to Interface mapping table that maintains the below:

- \* All the local Prefix/Node SID with any SR enabled interface as incoming interface.
- \* All the Adj-SIDs assigned by directly connected neighbor nodes with the relevant interface incoming interface.

In Figure 1, R8 maintains 160008 and 161288 with Incoming interface as any SR enabled interface. Similarly, R8 maintains 9178 with Link L1 as incoming interface, 9278 with Link L2 as incoming interface and 9378 with Link L1 or L2 as incoming interface.



How this mapping is populated and maintained is a local implementation matter. It can be populated based on the IGP database or can be based on a query to Path Computation Element (PCE) controller. The mapping can be persistent or on-demand triggered by receiving LSP Ping Request.

## 5.2. Initiator behavior

This section defines the Target FEC Stack TLV construction mechanism by an initiator when using SR Generic Label Sub-TLV.

### Ping

- Initiator MUST include FEC(s) corresponding to the destination segment.
- Initiator MAY include FECs corresponding to some or all of segments imposed in the label stack by the initiator to communicate the segments traversed.

### Traceroute

- Initiator MUST initially include FECs corresponding to all of segments imposed in the label stack.
- When a received echo reply contains FEC Stack Change TLV with one or more of original segment(s) being popped, initiator MAY remove corresponding FEC(s) from Target FEC Stack TLV in the next (TTL+1) traceroute request as defined in section 4.6 of [RFC8029].
- When a received echo reply does not contain FEC Stack Change TLV, initiator MUST NOT attempt to remove FEC(s) from Target FEC Stack TLV in the next (TTL+1) traceroute request.

### 5.2.1. SRGL in Target FEC Stack TLV

When the last segment ID in the label stack is IGP Prefix SID, Adj-SID, Binding SID, BGP Prefix SID or BGP Peering SID, set the SR SID field to the Segment ID value advertised by the LSP End Point. When the SID is advertised as index, the Segment ID value MUST be derived based on the index and the SRGB advertised by the LSP End Point.

How the above values are derived is a local implementation matter. It can be manually defined using CLI knob while triggering the LSP Ping Request or can use other mechanisms like querying the local database.

### 5.3. Responder behavior

Step 4a defined in Section 7.4 of [RFC8287] is updated as below:

If the Label-stack-depth is 0 and Target FEC Stack Sub-TLV at FEC-stack-depth is TBD1 (SRGL) {

- Set the Best-return-code to 10 when the responding node is not the LSP End Point for SR SID.
  - Set the Best-return-code to 35, if Interface-I does not match the SID to Interface mapping for the received SR SID.
  - set FEC-Status to 1, and return.
- }

If the Label-stack-depth is greater than 0 and Target FEC Stack Sub-TLV at FEC-stack-depth is TBD1 (SRGL), {

- If the Label at Label-stack-depth is Imp-null {
  - o Set the Best-return-code to 10 when the responding node is not the LSP End Point for the SR SID.
  - o Set the Best-return-code to 35, if Interface-I does not match the SID to Interface mapping for the received SR SID.
  - o set FEC-Status to 1, and return.
- }
- Else:
  - o Set the Best-return-code to 10 when the index derived from the label at Label-stack-depth is not advertised by LSP End Point.
  - o set FEC-Status to 1, and return.
- }

### 5.4. PHP flag behavior

Section 7.2 of [RFC8287] explains the behavior for FEC stack change for Adjacency Segment ID. The same procedure is applicable for BGP Peering SID as well.

## 6. IANA Considerations

### 6.1. New Target FEC Stack Sub-TLVs

IANA is requested to assign three new Sub-TLVs from "Sub-TLVs for TLV Types 1, 16 and 21" sub-registry from the "Multi-Protocol Label Switching (MPLS) Label Switched Paths (LSPs) Ping Parameters" [IANA-MPLS-LSP-PING] registry.

Sub-Type	Sub-TLV Name	Reference
TBD1	Segment Routing Generic Label	Section 4.1 of this document

### 6.2. Security Considerations

This document defines additional MPLS LSP Ping Sub-TLVs and follows the mechanisms defined in [RFC8029]. All the security considerations defined in [RFC8029] will be applicable for this document, and in addition, they do not impose any additional security challenges to be considered.

## 7. Acknowledgement

TBD

## 8. Contributors

Danial Johari, Cisco Systems

## 9. References

### 9.1. Normative References

[I-D.ietf-idr-bgp-prefix-sid]  
Previdi, S., Filsfils, C., Lindem, A., Sreekantiah, A.,  
and H. Gredler, "Segment Routing Prefix Segment Identifier  
Extensions for BGP", Work in Progress, Internet-Draft,  
draft-ietf-idr-bgp-prefix-sid-27, 26 June 2018,  
<[https://www.ietf.org/archive/id/draft-ietf-idr-bgp-  
prefix-sid-27.txt](https://www.ietf.org/archive/id/draft-ietf-idr-bgp-prefix-sid-27.txt)>.

- [I-D.ietf-idr-bgppls-segment-routing-epe]  
Previdi, S., Talaulikar, K., Filsfils, C., Patel, K., Ray, S., and J. Dong, "Border Gateway Protocol - Link State (BGP-LS) Extensions for Segment Routing BGP Egress Peer Engineering", Work in Progress, Internet-Draft, draft-ietf-idr-bgppls-segment-routing-epe-19, 16 May 2019, <<https://www.ietf.org/archive/id/draft-ietf-idr-bgppls-segment-routing-epe-19.txt>>.
- [I-D.sivabalan-pce-binding-label-sid]  
Sivabalan, S., Filsfils, C., Tantsura, J., Hardwick, J., Previdi, S., and C. Li, "Carrying Binding Label/Segment-ID in PCE-based Networks.", Work in Progress, Internet-Draft, draft-sivabalan-pce-binding-label-sid-07, 8 July 2019, <<https://www.ietf.org/archive/id/draft-sivabalan-pce-binding-label-sid-07.txt>>.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC8029] Kompella, K., Swallow, G., Pignataro, C., Ed., Kumar, N., Aldrin, S., and M. Chen, "Detecting Multiprotocol Label Switched (MPLS) Data-Plane Failures", RFC 8029, DOI 10.17487/RFC8029, March 2017, <<https://www.rfc-editor.org/info/rfc8029>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8287] Kumar, N., Ed., Pignataro, C., Ed., Swallow, G., Akiya, N., Kini, S., and M. Chen, "Label Switched Path (LSP) Ping/Traceroute for Segment Routing (SR) IGP-Prefix and IGP-Adjacency Segment Identifiers (SIDs) with MPLS Data Planes", RFC 8287, DOI 10.17487/RFC8287, December 2017, <<https://www.rfc-editor.org/info/rfc8287>>.
- [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>>.

## 9.2. Informative References

[I-D.ietf-spring-segment-routing-mpls]  
Bashandy, A., Filsfils, C., Previdi, S., Decraene, B.,  
Litkowski, S., and R. Shakir, "Segment Routing with the  
MPLS Data Plane", Work in Progress, Internet-Draft, draft-  
ietf-spring-segment-routing-mpls-22, 1 May 2019,  
<<https://www.ietf.org/archive/id/draft-ietf-spring-segment-routing-mpls-22.txt>>.

[IANA-MPLS-LSP-PING]  
IANA, "Multi-Protocol Label Switching (MPLS) Label  
Switched Paths (LSPs) Ping Parameters",  
<<http://www.iana.org/assignments/mpls-lsp-ping-parameters/mpls-lsp-ping-parameters.xhtml>>.

#### Authors' Addresses

Nagendra Kumar Nainar (editor)  
Cisco Systems, Inc.  
7200-12 Kit Creek Road  
Research Triangle Park, NC 27709-4987  
United States of America

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

Carlos Pignataro (editor)  
Cisco Systems, Inc.  
7200-11 Kit Creek Road  
Research Triangle Park, NC 27709-4987  
United States of America

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

Zafar Ali  
Cisco Systems, Inc.

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

Clarence Filsfils  
Cisco Systems, Inc.

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

Tarek Saad  
Juniper Networks

Email: tsaad@juniper.net

Internet Engineering Task Force  
Internet-Draft  
Intended status: Standards Track  
Expires: April 24, 2021

Yimin Shen  
Zhaohui Zhang  
Juniper Networks  
Rishabh Parekh  
Cisco Systems  
Hooman Bidgoli  
Nokia  
Yuji Kamite  
NTT Communications  
October 21, 2020

Point-to-Multipoint Transport Using Chain Replication in Segment Routing  
draft-shen-spring-p2mp-transport-chain-03

## Abstract

This document specifies a point-to-multipoint (P2MP) transport mechanism based on chain replication. It can be used in segment routing to achieve traffic optimization for multicast.

## Status of This Memo

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

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

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

This Internet-Draft will expire on April 24, 2021.

## Copyright Notice

Copyright (c) 2020 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. Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

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. Specification of Requirements . . . . .	3
3. Applicability . . . . .	3
4. P2MP Transport Using Chain Replication . . . . .	4
4.1. Bud Segment . . . . .	5
4.2. P2MP Chain . . . . .	6
4.3. Example . . . . .	7
5. Path Computation for P2MP Chains . . . . .	9
6. Protocol Extensions for Bud Segment . . . . .	10
7. Special Purpose Bud Segments . . . . .	10
8. IANA Considerations . . . . .	10
9. Security Considerations . . . . .	11
10. Acknowledgements . . . . .	11
11. Contributors . . . . .	11
12. References . . . . .	11
12.1. Normative References . . . . .	11
12.2. Informative References . . . . .	12
Authors' Addresses . . . . .	12

## 1. Introduction

The Segment Routing Architecture [RFC8402] describes segment routing (SR) and its instantiation in two data planes, i.e. MPLS and IPv6. In SR, point-to-multipoint (P2MP) transport is currently achieved by using ingress replication, where a point-to-point (P2P) SR tunnel is constructed from a root node to each leaf node, and every ingress packet is replicated and sent via a bundle of such P2P SR tunnels to all the leaf nodes. Although this approach provides P2MP reachability, it does not consider traffic optimization across the tunnels, as the path of each tunnel is computed or decided independently.

An alternative approach would be to use P2MP-tree based transport. Such approach can achieve maximum traffic optimization, but it relies a controller or path computation element (PCE) to provision and manage "replication segments" on branch nodes. The replication segments are essentially P2MP-tree state (i.e. transport tunnel state) on transit routers. Therefore, this approach is not fully aligned with SR's principles of single-point provisioning (at ingress routers and border routers) and stateless core network.



This document introduces a new solution for P2MP transport in SR, based on "chain replication". In this solution, P2MP transport is achieved by constructing a set of "P2MP chain tunnels" (or simply "P2MP chains") from a root node to leaf nodes. Each P2MP chain is a single-path tunnel, with a leaf node at tail end and some transit leaf nodes along the path, resembling a chain. The leaf node at the tail end behaves as a normal receiver. Each transit leaf node replicates a packet once for local processing off the chain, and also forwards the original packet down the chain. The root node replicates and sends packets via the set of P2MP chains to all the leaf nodes.

As a P2MP chain can reach multiple leaf nodes, it is considered more optimal than the multiple P2P tunnels which would be needed by ingress replication. Compared with ingress replication and the P2MP-tree based approach, this solution can achieve transport efficiency in general, while maintaining the simplicity of SR, including single-point provisioning and stateless core.

## 2. Specification of Requirements

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119] and [RFC8174].

## 3. Applicability

The P2MP transport mechanism in this document is generally applicable to all networks. However, it benefits more for certain types of topologies than others. These topologies include ring topologies, linear topologies, topologies with leaf nodes concentrated in geographical sites which can be modeled as leaf groups, etc.

The mechanism is stateless in the core of a network. It is transparent to all transit routers. Leaf nodes intended to take advantage of the mechanism will need to support the new forwarding behavior specified in this document. For other leaf nodes, the mechanism has a backward compatibility to allow them to be reached by P2P tunnels using ingress replication. Path computation and P2MP chain construction will need to be supported by a controller or root nodes, depending on where they are performed.

The mechanism is applicable to both SR-MPLS [RFC8660] and SRv6 [SRv6-SRH], [SRv6-Programming].

The mechanism does not create any state of P2MP tunnel or P2MP tree on routers. Therefore, if leaf nodes need to know the service level

context (e.g. source, VPN) of a P2MP stream, they must rely on the information contained in packet headers (or inner headers). In SR-MPLS, service labels may be allocated from a domain-wide common block (DCB) to serve as globally unique context indicators. In SRv6, a root node's IP address or an upstream-assigned context indicator may be encoded in the source address of IPv6 header, or a downstream-assigned context indicator may be encoded in the ARG portion of a service SID.

This document introduces a new type of segments, called bud segments (Section 4.1). The segments are generic in nature. They may be used in cases other than P2MP transport, such as traffic mirroring and monitoring, OAM, etc. These use cases are out of the scope of this document.

#### 4. P2MP Transport Using Chain Replication

In this document, a P2MP stream associated with a root node and a set of leaf nodes is denoted as {root node, leaf nodes}. It is achieved by using a bundle of P2MP chains covering all the leaf nodes. Each P2MP chain is a single-path tunnel starting from the root node and reaching one or multiple leaf nodes along the path. The tail-end node of the P2MP chain is a leaf node, called a "tail-end" leaf node. Each leaf node traversed by the P2MP chain is called a "transit" leaf node. As a special case, a P2MP chain may have no transit leaf node, but only a tail-end leaf node, essentially becoming a P2P tunnel of ingress replication.

R ----- R1 ----- R2 ----- L1 ----- R3 ----- L2 ----- L3

R : root node  
Li : leaf node  
Ri : transit router

Figure 1

A tail-end leaf node and a transit leaf nodes have different behaviors when processing a received packet. In particular, a tail-end leaf node processes the packet as a normal receiver. A transit leaf node not only processes the packet as a receiver, but also forwards it downstream along the P2MP chain, hence acting as a "bud node". To achieve this, the transit leaf node needs to replicate the packet, producing two packets, one for forwarding and the other for local processing. Such packet replication happens on every transit

leaf node along a P2MP chain. Therefore, it is called "chain replication".

This document introduces a new type of segments, called "bud segments", to facilitate the above packet processing on transit leaf nodes. The segment ID (SID) of a bud segment is a "bud-SID".

#### 4.1. Bud Segment

On a transit leaf node, a bud segment represents the following instructions for forwarding hardware to execute on a received packet P. They apply when the active SID of the packet P is the bud-SID of this bud segment.

[1] Replicate the packet P to generate a copy P1.

[2] For P, perform a NEXT operation on the bud-SID, make the next SID active, and forward the packet based on that SID.

[3] For P1, perform a sequence of NEXT operations on the bud-SID and all the subsequent SIDs of the P2MP chain, and process the packet locally. (The SIDs of the P2MP chain are not useful for processing P1 locally. Hence, they are removed before the processing.)

Bud segments are global segments of leaf nodes. They are routable segments via topological shortest-paths. Bud-SIDs are allocated from SRGB (SR global block). Only one bud segment is needed per leaf node, and per SR-MPLS or SRv6. It is used only when the leaf node is a transit leaf node on a P2MP chain.

In SR-MPLS, bud-SIDs are labels, and penultimate hop popping (PHP) MUST be disabled for bud-SID labels. In SRv6, bud-SIDs are IPv6 addresses explicitly associated with bud segments. Therefore, the above instructions [1] to [3] are achieved in different ways in SR-MPLS and SRv6:

(a) In SR-MPLS, the packet may have a service label(s) after P2MP chain labels in MPLS header, e.g. a VPN label, a bridge domain label, a source Ethernet segment label, etc. Therefore, the bud segment MUST have a way to identify the position of the last P2MP chain label, in order to execute [3] above. This document introduces an "end-of-chain" (EoC) label to facilitate the process. The EoC label is an extended special-purpose label (ESPL) [RFC 7274] with value TDB. When a root node constructs an MPLS header for a packet, if the packet has a service label(s), the root node MUST push the Extension Label (XL, value 15) and the EoC label, after pushing the service label(s) and before pushing

P2MP chain labels. Hence, [XL, EoC] serve as a recognizable pattern to indicate the end of the P2MP chain labels. If the packet does not have a service label(s), its MPLS header will contain P2MP chain labels only, and the root node SHOULD NOT push [XL, EoC] to the MPLS header. In any case, in [3] above, the bud segment MUST pop labels until [XL, EoC] are popped or all labels have been popped.

(b) In SRv6, the packet is encapsulated with an outer IPv6 header corresponding to the P2MP chain, optionally followed by a segment routing header (SRH) containing the SIDs of the P2MP chain, and followed by an inner header (of IPv4, IPv6, MPLS, layer-2, etc.) associated with a service. In [3] above, the bud segment SHOULD simply remove the outer IPv6 header and the SRH (if any), and leave the packet with the inner header to local processing.

Bud segments are shared by all P2MP streams, i.e. all combinations of {root node, leaf nodes}. A leaf node SHOULD advertise a bud segment for SR-MPLS, if its forwarding hardware supports the above SR-MPLS processing. Likewise, it SHOULD advertise a bud segment for SRv6, if its forwarding hardware supports the above SRv6 processing. The advertisement may be via a protocol, e.g. ISIS, OSPF, or BGP. The advertisement allows the leaf node to be considered as a transit leaf node on a P2MP chain. If a leaf node does not advertise a bud segment, it can only be considered as a tail-end leaf node on a P2MP chain, or reached via a P2P tunnel using ingress replication.

#### 4.2. P2MP Chain

Construction of P2MP chains for a P2MP stream is performed by a controller or the root node based on configuration or path computation (Section 5). This decides the number of P2MP chains to use, and the set of leaf nodes that each P2MP chain reaches. In general, if the leaf nodes of the P2MP stream cannot be covered by using a single P2MP chain, multiple P2MP chains MUST be used, and the root node MUST replicate ingress packets over the P2MP chains.

The path of a P2MP chain is a single path traversing one or multiple transit leaf nodes and terminating at a tail-end leaf node. Between the root node and the first transit leaf node, and between two consecutive leaf nodes, there may be none, one, or multiple transit routers.

The path is then translated to a SID list to be programmed on the root node. In the SID list, each transit leaf node has its bud-SID in a corresponding position. Given a P2MP chain to a set of leaf nodes in the order of L1, L2, ..., Ln, the SID list may be represented as:

<SID<sub>11</sub>, SID<sub>12</sub>, ...>, bud-SID of L<sub>1</sub>, ..., <SID<sub>i1</sub>, SID<sub>i2</sub>, ...>, bud-SID of L<sub>i</sub>, ..., <SID<sub>n1</sub>, SID<sub>n2</sub>, ...>

Where:

- o <SID<sub>11</sub>, SID<sub>12</sub>, ...> is the sub-path from the root node to L<sub>1</sub>.
- o <SID<sub>i1</sub>, SID<sub>i2</sub>, ...> is the sub-path from L<sub>i-1</sub> to L<sub>i</sub>.
- o <SID<sub>n1</sub>, SID<sub>n2</sub>, ...> is the sub-path from L<sub>n-1</sub> to L<sub>n</sub>. There is no need for L<sub>n</sub>'s bud-SID to be at the end of the SID list, because the tail-end leaf node does not perform a chain replication.

Each of the above sub-paths is a regular point-to-point path. The SIDs in the sub-path are regular SIDs, such as adjacency-SIDs, node-SIDs, binding-SIDs, etc. A sub-path from L<sub>i-1</sub> to L<sub>i</sub> may have an empty SID list, if the sub-path takes the shortest path indicated by the bud-SID of L<sub>i</sub>.

The root node then applies the SID list to packets, by using the encapsulation procedure of SR-MPLS or SRv6. Note that in an SR-MPLS case where a service label(s) applies, the service label(s) is pushed to an MPLS header first, then [XL, EoC] are pushed, and finally the labels of the SID list are pushed. This also places a requirement on the tail-end leaf node to handle [XL, EoC]. On the tail-end leaf node, a received MPLS header may have either [XL, EoC] at the top, or the node's node-SID label at the top, followed by [XL, EoC]. In the latter case, [XL, EoC] will be exposed to the top after the node-SID label is popped. In either case, the node MUST pop [XL, EoC] and continue to process the next label, i.e. the service label.

#### 4.3. Example

In the following example, P2MP transport is needed from the root node R, to leaf nodes L<sub>1</sub>, L<sub>2</sub>, L<sub>3</sub> and L<sub>4</sub>.

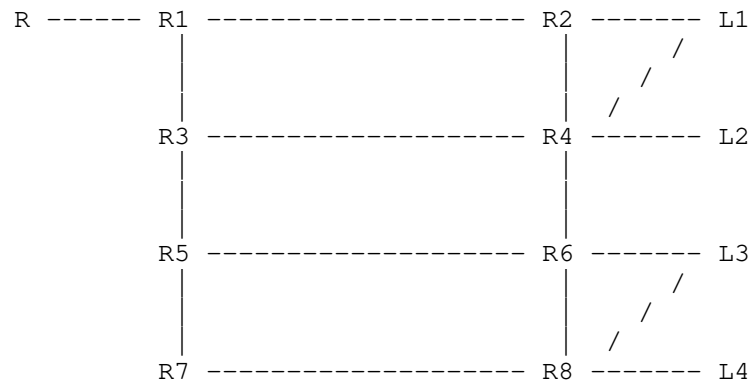


Figure 2

Path computation results in two P2MP chains:

P2MP chain 1:

Path: R -> R1 -> R2 -> L1 -> R4 -> L2, where L1 is a transit leaf node, and L2 is the tail-end leaf node.

Assuming that the sub-path R -> R1 -> R2 -> L1 is not the shortest path from R to L1, so that an explicit sub-path must be used. Also assuming that the sub-path L1 -> R4 -> L2 is the shortest path from L1 to L2, so that the node-SID of L2 can be used to represent this sub-path. The segment list applied to packets on R is:

```
adj-SID 100 - link from R to R1
adj-SID 200 - link from R1 to R2
adj-SID 300 - link from R2 to L1
bud-SID 1000 - L1
node-SID 2000 - L2
```

P2MP chain 2:

Path: R -> R1 -> R3 -> R5 -> R6 -> L3 -> R8 -> L4, where L3 is a transit leaf node, and L4 is the tail-end leaf node.

Assuming that the sub-path R -> R1 -> R3 -> R5 -> R6 -> L3 is the shortest path from R to L3, so that the bud-SID of L3 can

be used to represent this sub-path. Also assuming that the sub-path L3 -> R8 -> L4 is not the shortest path from L3 to L4, so that an explicit sub-path must be used. The segment list applied to packets on R is:

```
bud-SID 3000 - L3

adj-SID 600 - link from L3 to R8

adj-SID 700 - link from R8 to L4

node-SID 4000 - L4
```

## 5. Path Computation for P2MP Chains

P2MP chain path computation for a P2MP stream {root node, leaf nodes} may be performed by a controller or the root node. P2MP chains are single-path tunnels. In general, any P2P path computation algorithm may be extended to serve the purpose. This document does not enforce a particular algorithm.

The path computation may consider general metric for shortest paths, or traffic engineering (TE) constraints for TE paths. In addition, this document also considers the following constraints:

- Maximum hops per P2MP chain. This SHOULD be based on the maximum delay allowed for a packet to accumulate before reaching a tail-end leaf node.
- Maximum length of SID list. This SHOULD be based on the maximum header size which a root node may apply to a packet. This is typically a limit of forwarding hardware. Note that a SID list is translated from a computed path. Hence, the SID list's length and the path's hop count are not necessarily the same.
- Maximum leaf nodes per P2MP chain. This may be used to restrict the length of each P2MP chain.
- Maximum hops between two consecutive leaf nodes. This may be used to avoid a sparse chain, where an excessive distance between two consecutive leaf nodes will cause a P2MP chain's efficiency to degrade.
- Maximum number of times that a node or link may be traversed by a P2MP chain. This may be used to prevent a node or link from being congested by duplicate traffic.

The path computation is generally deterministic in a ring or linear topology. In an arbitrary topology, the path computation may be more controllable by dividing leaf nodes into groups based on geographic location or policies, and computing a separate path for each group. A leaf group may be defined as a sequence (i.e. ordered) or set (i.e. unordered) of leaf nodes, which are treated as loose hops in path computation.

## 6. Protocol Extensions for Bud Segment

The protocol extensions of ISIS, OSPF, and BGP for bud segment advertisement will be specified in the next version of this document.

## 7. Special Purpose Bud Segments

So far, the discussion in this document has been focusing on bud segments that are created on a per SR-MPLS or SRv6 basis on each leaf node. These bud segments indicate generic local processing which is completely based on the inner header of a packet, i.e. after all the SIDs of a P2MP chain are removed by the instruction [3] in Section 4.1. They are applicable to common P2MP transport cases, and hence are considered as the default and general purpose bud segments.

The concept of bud segment can also be extended to other cases, where a transit leaf node needs to perform a special kind of local processing for packets, but cannot derive the context from the inner headers of the packets. For example, the node may need to forward the packets over a particular interface or tunnel to some device(s), or to process the packets based on a particular forwarding table or policy, and so on. In such cases, a dedicated bud segment may be created for each special local processing, indicating the context. The bud segment is called a special purpose bud segment.

Note that the scaling of special purpose bud segments per leaf node SHOULD be a consideration in network design, as well as the requirement for a controller or ingress router to learn all the special purpose bud segments in a network and apply them in P2MP chain construction.

## 8. IANA Considerations

This document requires IANA to allocate a value from the "Extended Special-Purpose MPLS Label Values" registry for the EoC label.

The document also requires IANA registration and allocation for the ISIS, OSPF and BGP extensions for bud segment advertisement. The details will be provided in the next version of this document.



## 9. Security Considerations

This document introduces bud segments for leaf nodes to act as both packet receivers and transit routers. A security attack may target on a leaf node by constructing malicious packets with the node's bud-SID. Such kind of attacks can be defeated by restricting bud segment distribution and P2MP chain construction within the scope of a controller and a given network.

## 10. Acknowledgements

This document leverages work done by Alexander Arseniev, Ron Bonica, and G Sri Karthik Goud.

## 11. Contributors

Alexander Arseniev

Juniper networks

Email: aarseniev@juniper.net

Ron Bonica

Juniper networks

Virginia

USA

Email: rbonica@juniper.net

## 12. References

### 12.1. Normative References

- [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>>.
- [RFC8660] Bashandy, A., Ed., Filsfils, C., Ed., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with the MPLS Data Plane", RFC 8660, DOI 10.17487/RFC8660, December 2019, <<https://www.rfc-editor.org/info/rfc8660>>.

[RFC7274] Kompella, K., Andersson, L., and A. Farrel, "Allocating and Retiring Special-Purpose MPLS Labels", RFC 7274, DOI 10.17487/RFC7274, June 2014, <<https://www.rfc-editor.org/info/rfc7274>>.

[SRv6-SRH] Filsfils, C., Dukes, D., Previdi, S., Leddy, J., Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header", draft-ietf-6man-segment-routing-header (work in progress), 2019.

[SRv6-Programming] Filsfils, C., Garvia, P., Leddy, J., Voyer, D., Matsushima, S., and Z. Li, "SRv6 Network Programming", draft-ietf-spring-srv6-network-programming (work in progress), 2019.

## 12.2. 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, <<https://www.rfc-editor.org/info/rfc2119>>.

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

## Authors' Addresses

Yimin Shen  
Juniper Networks  
10 Technology Park Drive  
Westford, MA 01886  
USA

Email: [yshen@juniper.net](mailto:yshen@juniper.net)

Zhaohui Zhang  
Juniper Networks  
10 Technology Park Drive  
Westford, MA 01886  
USA

Email: [zzhang@juniper.net](mailto:zzhang@juniper.net)

Rishabh Parekh  
Cisco Systems  
San Jose, CA  
USA

Email: riparekh@cisco.com

Hooman Bidgoli  
Nokia  
Ottawa  
Canada

Email: hooman.bidgoli@nokia.com

Yuji Kamite  
NTT Communications  
Tokyo  
Japan

Email: y.kamite@ntt.com

Internet Engineering Task Force  
Internet-Draft  
Intended status: Standards Track  
Expires: December 16, 2021

Yimin Shen  
Zhaohui Zhang  
Juniper Networks  
Rishabh Parekh  
Cisco Systems  
Hooman Bidgoli  
Nokia  
Yuji Kamite  
NTT Communications  
June 14, 2021

Point-to-Multipoint Transport Using Chain Replication in Segment Routing  
draft-shen-spring-p2mp-transport-chain-04

## Abstract

This document specifies a point-to-multipoint (P2MP) transport mechanism based on chain replication. It can be used in segment routing to achieve traffic optimization for multicast.

## 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 December 16, 2021.

## Copyright Notice

Copyright (c) 2021 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. Specification of Requirements . . . . .	3
3. Applicability . . . . .	3
4. P2MP Transport Using Chain Replication . . . . .	4
4.1. P2MP Chain . . . . .	5
4.2. Bud Segment . . . . .	6
4.3. Forwarding Behaviors . . . . .	6
4.3.1. SR-MPLS . . . . .	6
4.3.2. SRv6 . . . . .	7
4.4. Example . . . . .	7
5. Path Computation for P2MP Chains . . . . .	9
6. Special Purpose Bud Segments . . . . .	10
7. IANA Considerations . . . . .	10
8. Security Considerations . . . . .	10
9. Acknowledgements . . . . .	11
10. Contributors . . . . .	11
11. References . . . . .	11
11.1. Normative References . . . . .	11
11.2. Informative References . . . . .	12
Authors' Addresses . . . . .	12

## 1. Introduction

The Segment Routing Architecture [RFC8402] describes segment routing (SR) and its instantiation in two data planes, i.e. MPLS and IPv6. In SR, point-to-multipoint (P2MP) transport is currently achieved by using two approaches. The first approach is ingress replication, where a dedicated point-to-point (P2P) SR tunnel is set up from a root node to each leaf node, and the root node replicates and sends packets via a bundle of such P2P SR tunnels to all the leaf nodes. Although this approach provides P2MP reachability, it does not consider traffic optimization across the tunnels.

The second approach is to use P2MP trees. This approach can achieve maximum traffic optimization, but it relies a controller or path computation element (PCE) to provision and manage "replication segments" on branch nodes. The replication segments are essentially P2MP-tree state (i.e. transport tunnel state) on transit routers. Therefore, this approach is not fully aligned with SR's principles of single-point provisioning (at ingress routers) and stateless core network.

This document introduces a new solution for P2MP transport in SR, based on "chain replication". In this solution, P2MP transport is achieved by constructing a set of "P2MP chain tunnels" (or simply "P2MP chains") from a root node to leaf nodes. Each P2MP chain is a single-path tunnel, with a leaf node at tail end and some transit leaf nodes along the path, resembling a chain. The leaf node at the tail end behaves as a normal receiver. Each transit leaf node behaves as a receiver and a transit router, by replicating incoming packets once for local processing off the chain, and forwarding the original packets down the chain. The root node sends packets via the set of P2MP chains to all the leaf nodes.

As each P2MP chain can reach multiple leaf nodes via a single flow of packets, this solution is considered to be more optimal than ingress replication. Compared to the P2MP-tree based approach, this solution can retain the simplicity of SR, including single-point provisioning and statelessness in the core of a network.

## 2. Specification of Requirements

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119] and [RFC8174].

## 3. Applicability

The P2MP transport mechanism in this document is generally applicable to all networks. However, it may benefit more for certain types of topologies than others. These topologies include ring topologies, linear topologies, topologies with leaf nodes concentrated in geographical sites which can be modeled as leaf groups (or clusters), etc.

The mechanism does not create any state of P2MP tunnel or P2MP tree on routers. It is transparent to all transit routers. Leaf nodes intended to take advantage of the mechanism will need to support the new forwarding behaviors specified in this document. For other leaf nodes, the mechanism has a backward compatibility to allow them to be reached by P2P tunnels of ingress replication. Path computation and P2MP chain construction will need to be supported by controllers or root nodes, depending on the location of the computation.

The mechanism is applicable to both SR-MPLS [RFC8660] and SRv6 [SRv6-SRH], [SRv6-Programming].

In this mechanism, if a leaf node needs to know the service level context (e.g. source, VPN) of a P2MP stream, it must rely on the

information contained in payload headers (i.e. inner headers) of packets. In SR-MPLS, service labels may be allocated from a domain-wide common block (DCB) to serve as globally unique context indicators. In SRv6, a root node's IP address or an upstream-assigned context indicator may be encoded in the source address of IPv6 header, or a downstream-assigned context indicator may be encoded in the ARG portion of a service SID.

This document introduces a new type of segments, called bud segments (Section 4.2). The segments are generic in nature. They may be used in cases other than P2MP transport, such as traffic mirroring and monitoring, OAM, etc. These use cases are out of the scope of this document.

#### 4. P2MP Transport Using Chain Replication

In this document, a P2MP stream associated with a root node and a set of leaf nodes is denoted as {root node, leaf nodes}. It is achieved by using a bundle of P2MP chains covering all the leaf nodes. Each P2MP chain is a single-path tunnel starting from the root node and reaching one or multiple leaf nodes along the path. The tail-end node of the P2MP chain is a leaf node, called a "tail-end" leaf node. Each leaf node traversed by the P2MP chain is called a "transit" leaf node. As a special case, a P2MP chain may have only a tail-end leaf node and no transit leaf node, essentially becoming a P2P tunnel, but it is not the focus of this document.

R ----- R1 ----- R2 ----- L1 ----- R3 ----- L2 ----- L3

R : root node  
Li : leaf node  
Ri : transit router

Figure 1

A tail-end leaf node and a transit leaf nodes have different behaviors when processing an incoming packet. In particular, a tail-end leaf node processes the packet as a normal receiver. A transit leaf node not only processes the packet as a receiver, but also forwards it downstream along the P2MP chain, hence acting as a "bud node". To achieve this, the transit leaf node needs to replicate the packet, producing two packets, one for forwarding and the other for local processing. Such packet replication happens on every transit

leaf node along a P2MP chain. Therefore, it is called "chain replication".

This document introduces a new type of segments, called "bud segments", to model the above packet replication and processing on transit leaf nodes. The segment ID (SID) of a bud segment is a "bud-SID".

#### 4.1. P2MP Chain

Construction of P2MP chains for a P2MP stream is performed by a controller or the root node based on configuration or path computation (Section 5). This generates the set of P2MP chains to use, and decides the set of leaf nodes that each P2MP chain reaches. In general, if not all leaf nodes can be covered by using a single P2MP chain, multiple P2MP chains MUST be used, and the root node MUST replicate ingress packets over the P2MP chains.

The path of a P2MP chain is a single path traversing one or multiple transit leaf nodes and terminating at a tail-end leaf node. Between the root node and the first transit leaf node, and between two consecutive leaf nodes, there may be none, one, or multiple transit routers.

The path is then translated to a SID list to be programmed on the root node. In the SID list, each transit leaf node has its bud-SID in a corresponding position. Given a P2MP chain to a set of leaf nodes in the order of L1, L2, ..., Ln, the SID list may be represented as below:

<SID\_11, SID\_12, ...>, L1's bud-SID, ..., <SID\_i1, SID\_i2, ...>, Li's bud-SID, ..., <SID\_n1, SID\_n2, ...>

Where:

- o <SID\_11, SID\_12, ...> is the sub-path from the root node to L1.
- o <SID\_i1, SID\_i2, ...> is the sub-path from Li-1 to Li.
- o <SID\_n1, SID\_n2, ...> is the sub-path from Ln-1 to Ln. There is no need for Ln's bud-SID to be at the end of the SID list, because the tail-end leaf node does not perform a chain replication.

Each of the above sub-paths is a regular point-to-point path, and its SIDs are regular SIDs, such as adjacency-SIDs, node-SIDs, binding-SIDs, etc. As a special case, a sub-path from Li-1 to Li may have an empty SID list, if the sub-path takes the shortest path represented by Li's bud-SID.



#### 4.2. Bud Segment

On a transit leaf node, a bud segment represents the forwarding instructions below. They are applied to an incoming packet P when the packet's active SID is the bud-SID of this bud segment.

- [1] Replicate the packet P to generate a copy P1.
- [2] For P, perform a NEXT operation on the bud-SID, make the next SID active, and forward the packet based on that SID.
- [3] For P1, perform a sequence of NEXT operations on the bud-SID and all the subsequent SIDs of the P2MP chain, and process the packet locally as an endpoint.

Bud segments are global segments of leaf nodes. They are routable segments via the shortest topological paths. Bud-SIDs are allocated from SRGB (SR global block). Only one bud segment is needed per leaf node, and per SR-MPLS or SRv6 dataplane. It is used only when the leaf node is a transit leaf node on a P2MP chain.

Bud segments are shared by all P2MP streams, i.e. all instances of {root node, leaf nodes}. A leaf node SHOULD advertise a bud segment for SR-MPLS if its forwarding hardware supports the SR-MPLS behavior (Section 4.3.1), and a bud segment for SRv6 if its forwarding hardware supports the above SRv6 behavior (Section 4.3.2). The advertisement may be via a protocol, e.g. ISIS, OSPF, or BGP. The advertisement allows the leaf node to be considered as a transit leaf node on a P2MP chain. If a leaf node does not advertise a bud segment, it can only be considered as a tail-end leaf node on a P2MP chain, or reached via a P2P tunnel using ingress replication. The extensions of the protocols are out of the scope of this document.

#### 4.3. Forwarding Behaviors

##### 4.3.1. SR-MPLS

In SR-MPLS, bud-SIDs are labels. A root node applies a stack of labels corresponding to a P2MP chain's SID list to ingress packets. These labels are called P2MP chain labels. A packet may have an inner service label(s), e.g. a VPN label, a bridge domain label, a source Ethernet segment label, etc. In this case, the root node must have a way to mark the end of P2MP chain labels in the MPLS header, in order for transit leaf nodes to process the packet as receivers. This document introduces an "end-of-chain" (EoC) label to facilitate this. The EoC label is an extended special-purpose label (ESPL) [RFC 7274] with value TDB. If an ingress packet has a service label(s), the root node MUST push the service label(s) first, then the

Extension Label (XL, value 15) and the EoC label, and finally the P2MP chain labels. Hence, the [XL, EoC] labels serve as a unique pattern to indicate the end of the P2MP chain labels. If a packet does not have a service label(s), the root node MUST NOT push the [XL, EoC] labels, but only the P2MP chain labels.

A transit leaf node will receive the packet (P) with the node's bud-SID label as top label. The node replicates P to generate a copy P1. For P, the node pops the bud-SID label and forwards the packet based on the next label in the MPLS header. For P1, the node removes all the P2MP chain labels, by popping labels until [XL, EoC] are popped or all labels are popped. The node then processes P1 locally as an SR-MPLS endpoint.

The tail-end leaf node will receives the packet with (1) no label; (2) the [XL, EoC] labels as top labels; or (3) the node's node-SID label as top label, followed by the [XL, EoC] labels. In case (3), the [XL, EoC] labels will be exposed to the top after the node-SID label is popped. In both cases (2) and (3), the node pops [XL, EoC] and continues to process the inner service label(s). This imposes a requirement on the node to be able to handle the [XL, EoC] labels as described. Ultimately, the node processes the packet with no label or an exposed service label(s), as an SR-MPLS endpoint.

#### 4.3.2. SRv6

In SRv6, bud-SIDs are IPv6 addresses specifically assigned to bud segments. A root node constructs a packet with an IPv6 header corresponding to the P2MP chain, followed by a segment routing header (SRH) containing the SIDs of the P2MP chain, and followed by an inner IP header or service header.

A transit leaf node will receive the packet (P) with the node's bud-SID as active SID. The node replicates P to generate a copy P1. For P, the node performs NEXT operation on the bud-SID by adjusting the IPv6 header and SRH, and forwards the packet based on the new active SID. For P1, the node removes the IPv6 header and SRH, and processes the packet (with the inner header exposed) as an SRv6 endpoint.

The tail-end leaf node will receives the packet with the IPv6 header and optionally the SRH. The node processes the packet as an SRv6 endpoint.

#### 4.4. Example

In the following example, P2MP transport is needed from the root node R, to leaf nodes L1, L2, L3 and L4.

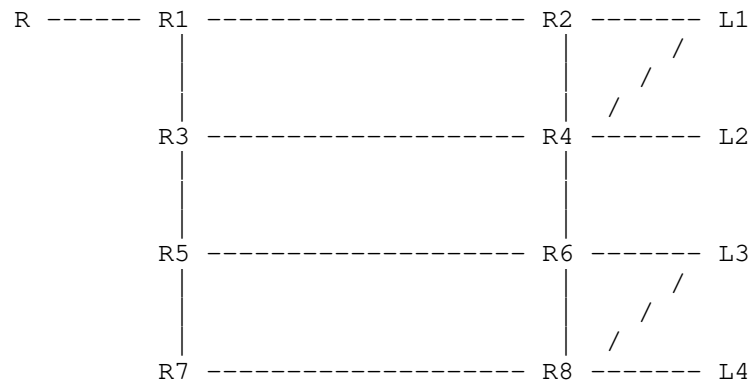


Figure 2

Path computation results in two P2MP chains:

P2MP chain 1:

Path: R -> R1 -> R2 -> L1 -> R4 -> L2, where L1 is a transit leaf node, and L2 is the tail-end leaf node.

Assuming that the sub-path R -> R1 -> R2 -> L1 is not the shortest path from R to L1, so that an explicit sub-path must be used. Also assuming that the sub-path L1 -> R4 -> L2 is the shortest path from L1 to L2, so that the node-SID of L2 can be used to represent this sub-path. The segment list applied to packets on R is:

adj-SID 100 - link from R to R1  
 adj-SID 200 - link from R1 to R2  
 adj-SID 300 - link from R2 to L1  
 bud-SID 1000 - L1  
 node-SID 2000 - L2

P2MP chain 2:

Path: R -> R1 -> R3 -> R5 -> R6 -> L3 -> R8 -> L4, where L3 is a transit leaf node, and L4 is the tail-end leaf node.

Assuming that the sub-path R -> R1 -> R3 -> R5 -> R6 -> L3 is the shortest path from R to L3, so that the bud-SID of L3 can

be used to represent this sub-path. Also assuming that the sub-path L3 -> R8 -> L4 is not the shortest path from L3 to L4, so that an explicit sub-path must be used. The segment list applied to packets on R is:

```
bud-SID 3000 - L3

adj-SID 600 - link from L3 to R8

adj-SID 700 - link from R8 to L4

node-SID 4000 - L4
```

## 5. Path Computation for P2MP Chains

P2MP chain path computation for a P2MP stream {root node, leaf nodes} may be performed by a controller or the root node. Each P2MP chain is a single-path tunnel. In general, any P2P path computation algorithm may be extended to serve the purpose. This document does not enforce a particular algorithm.

The path computation may consider topological metrics for shortest paths, or traffic engineering (TE) constraints for TE paths. In addition, this document also suggests the following constraints:

- Maximum hops per P2MP chain. This SHOULD be based on the maximum delay allowed for packets to accumulate before reaching a tail-end leaf node.
- Maximum length of SID list. This SHOULD be based on the maximum header size which may be applied to packets by a root node. This is typically a limit of forwarding hardware. Note that a SID list is translated from a computed path. Hence, the SID list's length and the path's hop count are not necessarily the same.
- Maximum leaf nodes per P2MP chain. This may be used to restrict the length of each P2MP chain.
- Maximum hops between two consecutive leaf nodes. This may be useful to avoid a sparse chain, where an excessive distance between two consecutive leaf nodes will cause a P2MP chain's efficiency to degrade.
- Maximum number of times that a node or link may be traversed by a P2MP chain. This may be useful to prevent a node or link from being congested by duplicate traffic.

The path computation tends to be deterministic in a ring or linear topology. In an arbitrary topology, the path computation can be made controllable by dividing leaf nodes into groups (or clusters) based on geographic locations or policies, and computing a separate path for each group. A leaf group may be defined as a sequence (i.e. ordered) or set (i.e. unordered) of leaf nodes, which are treated as loose hops in path computation.

## 6. Special Purpose Bud Segments

So far, the discussion has been focusing on nodal bud segments, which are per node and per SR-MPLS/SRv6. The local processing represented by these bud segments is completely based on a packet's inner header, i.e. after all the SIDs of a P2MP chain are removed by the instruction [3] in Section 4.2. These bud segments are applicable to common P2MP transport, and hence are considered as the default and general purpose bud segments.

The concept of bud segment is also applicable to other types of local processing on a transit leaf node, where the context of the local processing cannot be derived from a packet's inner header. For example, the node may want to forward packets over a particular interface or tunnel, or based on a particular forwarding table or policy. In such cases, a dedicated bud segment may be introduced to serve each distinct scenario and indicate a specific context. These bud segments are called special purpose bud segments.

The scale of special purpose bud segments per leaf node SHOULD be a consideration in network design, as well as the mechanisms for distributing these bud segments to controllers or all the root nodes.

## 7. IANA Considerations

This document requires IANA to allocate a value from the "Extended Special-Purpose MPLS Label Values" registry for the EoC label.

The document also requires IANA registration and allocation for the ISIS, OSPF and BGP extensions for bud segment advertisement. The details will be provided in the next version of this document.

## 8. Security Considerations

This document introduces bud segments for leaf nodes to act as both packet receivers and transit routers. A security attack may target on a leaf node by constructing malicious packets with the node's bud-SID. Such kind of attacks can be defeated by restricting bud segment distribution and P2MP chain construction within the scope of a controller and a given network.

## 9. Acknowledgements

This document leverages work done by Alexander Arseniev, Ron Bonica, and G Sri Karthik Goud.

## 10. Contributors

Alexander Arseniev

Juniper networks

Email: aarseniev@juniper.net

Ron Bonica

Juniper networks

Virginia

USA

Email: rbonica@juniper.net

## 11. References

### 11.1. Normative References

- [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>>.
- [RFC8660] Bashandy, A., Ed., Filsfils, C., Ed., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with the MPLS Data Plane", RFC 8660, DOI 10.17487/RFC8660, December 2019, <<https://www.rfc-editor.org/info/rfc8660>>.
- [RFC7274] Kompella, K., Andersson, L., and A. Farrel, "Allocating and Retiring Special-Purpose MPLS Labels", RFC 7274, DOI 10.17487/RFC7274, June 2014, <<https://www.rfc-editor.org/info/rfc7274>>.
- [SRv6-SRH] Filsfils, C., Dukes, D., Previdi, S., Leddy, J., Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header", draft-ietf-6man-segment-routing-header (work in progress), 2019.

[SRv6-Programming]

Filsfils, C., Garvia, P., Leddy, J., Voyer, D.,  
Matsushima, S., and Z. Li, "SRv6 Network Programming",  
draft-ietf-spring-srv6-network-programming (work in  
progress), 2019.

11.2. 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,  
<<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC  
2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174,  
May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

Authors' Addresses

Yimin Shen  
Juniper Networks  
10 Technology Park Drive  
Westford, MA 01886  
USA

Email: [yshen@juniper.net](mailto:yshen@juniper.net)

Zhaohui Zhang  
Juniper Networks  
10 Technology Park Drive  
Westford, MA 01886  
USA

Email: [zzhang@juniper.net](mailto:zzhang@juniper.net)

Rishabh Parekh  
Cisco Systems  
San Jose, CA  
USA

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

Hooman Bidgoli  
Nokia  
Ottawa  
Canada

Email: [hooman.bidgoli@nokia.com](mailto:hooman.bidgoli@nokia.com)

Yuji Kamite  
NTT Communications  
Tokyo  
Japan

Email: [y.kamite@ntt.com](mailto:y.kamite@ntt.com)



SPRING  
Internet-Draft  
Intended status: Informational  
Expires: May 3, 2021

W. Cheng  
China Mobile  
S. Steffann  
SJM Steffann Consultancy  
October 30, 2020

Compressed SRv6 SID List Requirements  
draft-srcompdt-spring-compression-requirement-00

Abstract

This document specifies requirements for solutions to compress SRv6 SID lists.

Status of This Memo

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

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

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

This Internet-Draft will expire on May 3, 2021.

Copyright Notice

Copyright (c) 2020 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. Conventions used in this document . . . . .	3
2.1. Requirements Language . . . . .	3
2.2. Terminology . . . . .	3
3. SRv6 SID List Compression Requirements . . . . .	4
3.1. Dataplane Efficiency and Performance Requirements . . . . .	4
3.1.1. Encapsulation Header Size . . . . .	4
3.1.2. Forwarding Efficiency . . . . .	4
3.1.3. State Efficiency . . . . .	5
4. SRv6 Specific Requirements . . . . .	5
4.1. Functional Requirements . . . . .	5
4.1.1. SRv6 Based . . . . .	5
4.1.2. SRv6 Functionality . . . . .	5
4.1.3. SID list length . . . . .	5
4.1.4. SID summarization . . . . .	6
4.1.5. Heterogeneous SID lists . . . . .	6
4.2. Operational Requirements . . . . .	6
4.2.1. Lossless Compression . . . . .	6
4.3. Scalability Requirements . . . . .	6
5. Protocol Design Requirements . . . . .	6
5.1. Ships in the Night Deployment . . . . .	7
6. IANA Considerations . . . . .	7
7. Security Considerations . . . . .	7
8. Contributors . . . . .	7
9. Acknowledgements . . . . .	8
10. Normative References . . . . .	8
Authors' Addresses . . . . .	9

## 1. Introduction

The SPRING working group defined SRv6, with [RFC8402] describing how the Segment Routing (SR) architecture is instantiated on two data-planes: SR over MPLS (SR-MPLS) and SR over IPv6 (SRv6). SRv6 uses a routing header called the SR Header (SRH) [RFC8754], and defines SRv6 SID behaviors and a registry for identifying them in [I-D.ietf-spring-srv6-network-programming]. SRv6 is a proposed standard and is deployed today.

The SPRING working group has observed that some use cases, such as strict path TE, may require long SRv6 SID lists. There are several proposed methods to reduce the resulting SRv6 encapsulation size by compressing the SID list.

The SPRING working group formed a design team to define requirements for, and analyze, proposals to compress SRv6 SID lists.

It is a goal of the design team to identify the solutions to SR over IPv6 list compression.

In each category, the requirements are described under several sub-categories including

- o Efficiency and performance
- o Functional requirements
- o Operational requirements
- o Scalability requirements
- o Convergence requirements
- o Security requirements

For each requirement, a description, rationale and metrics are described.

The design team will produce a separate document to analyze the proposals.

## 2. Conventions used in this document

### 2.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

### 2.2. Terminology

SR: Segment Routing

SRH: Segment Routing Header

MPLS: Multiprotocol Label Switching

SR-MPLS: Segment Routing over MPLS data plane

SID: Segment Identifier

SRv6: Segment Routing over IPv6

SRv6 SID List: A list of SRv6 SIDs

Compression proposal: A proposal to compress SRv6 SID lists

SRv6 base: SRv6 as defined in [RFC8402], [RFC8754],  
[I-D.ietf-spring-srv6-network-programming]

### 3. SRv6 SID List Compression Requirements

#### 3.1. Dataplane Efficiency and Performance Requirements

##### 3.1.1. Encapsulation Header Size

Description: The compression solution MUST reduce the size of the SRv6 encapsulation header.

Rationale: A smaller SRv6 encapsulation results in better MTU efficiency.

Metric: Compression is the ratio of the IPv6 encapsulation size of SRv6 as defined in [RFC8402], [RFC8754], [I-D.ietf-spring-srv6-network-programming] vs the IPv6 encapsulation size of a given proposal. The encapsulation savings of a compression proposal vs the SRv6 base is a useful measurement to compare proposals.

The encapsulation metric (E) records the number of bytes required for a proposal to encapsulate a packet given a specific segment list.

o  $E(\text{proposal}, \text{segment list})$ .

The encapsulation savings (ES) records the encapsulation savings for a proposal to encapsulate a packet given a specific segment list.

o  $ES(\text{proposal}, \text{segment list}) = 1 - E(\text{proposal}, \text{segment list})/E(\text{SRv6 base}, \text{segment list})$ .

##### 3.1.2. Forwarding Efficiency

Description: The compression solution SHOULD minimize the number of required hardware resources accessed to process a segment.

Rational: Efficiency in bits on the wire and processing efficiency are both important. Optimizing one at the expense of the other may lead to significant performance impact.

Metric: The data plane efficiency metric (D) records the data plane forwarding efficiency of the proposed solution. Two metrics are used and recorded at the each segment endpoint:

- o D.PRS(segment list): number of headers parsed during processing of the segment list.
- o D.LKU(segment list): number of FIB lookups during processing of the segment list. The type of lookup is also recorded as longest prefix match (LPM) or exact match (EM)

### 3.1.3. State Efficiency

Description: The compression solution SHOULD minimize the amount of additional forwarding state stored at a node

Rational: Additional state increases the complexity of the control plane and data plane. It can also result in an increase in memory usage.

Metric: The state efficiency metric (S) records the amount of additional forwarding state required by the proposed solution.

- o S(node parameters): the number of additional forwarding states that need to be stored at a node, given a set of node parameters consisting of number of nodes in the network, number of local interface, number of adjacencies. The forwarding state is counted as entries required in a Forwarding Information Base (FIB) at a node.

## 4. SRv6 Specific Requirements

### 4.1. Functional Requirements

#### 4.1.1. SRv6 Based

TBD

#### 4.1.2. SRv6 Functionality

TBD

#### 4.1.3. SID list length

Description: The compression mechanism must be able to represent SR paths that contain up to 16 segments.

**Rationale:** Strict TE paths require SID list lengths proportional to the diameter of the SR domain.

**Metric:** The compression mechanism must be able steer a packet through an SR path that contains up to sixteen segments.

#### 4.1.4. SID summarization

**Description:** The solution **MUST** be compatible with segment summarization.

**Rationale:** Summarization of segments is a key benefit of SRv6 vs SR MPLS. In interdomain deployments any node can reach any other node via a single prefix segment. Without summarization, border router SIDs must be leaked and an additional global prefix segment is required for each domain border to be traversed.

**Metric:** A solution supports summarization when segments can be summarized for advertisement into other IGP domains or levels.

#### 4.1.5. Heterogeneous SID lists

TBD

### 4.2. Operational Requirements

#### 4.2.1. Lossless Compression

**Description:** The segments of the compressed SID list **MUST** be equivalent to the original SID List. For example, a strict path TE SID List is not compressed to a loose path TE SID list.

**Rational:** In SRv6 we can represent a path to meet certain objectives. A compressed solution needs to support the objectives in the same way.

**Metric:** Information present in the pre-compression segment list **MUST** also be present in the post-compression SID list.

#### 4.3. Scalability Requirements

TBD

### 5. Protocol Design Requirements

### 5.1. Ships in the Night Deployment

Description: The compression solution MUST support deployment in existing SRv6 networks.

Rationale: SRv6 is deployed today. A compression solution that interoperates well with SRv6, as deployed, will reduce the overhead and simplify operations. For Network operators who would migrate to compressed SRv6 SID lists, the move is expected to gradually occur over a period time, as they upgrade networks, domains, device families and software instances.

Metric: A compliant compression solution provides the following

- o Supports simultaneous deployment at a node with current SRv6 SIDs.
- o Supports simultaneous deployment at a node with current SRv6 control plane.
- o Supports simultaneous operation of current SRv6 paths with compressed paths.
- o Supports the behaviors in [I-D.ietf-spring-srv6-network-programming].
- o Does not require removal of existing IPv6 planning.

### 6. IANA Considerations

This document has no requests to IANA.

### 7. Security Considerations

TBD

### 8. Contributors

The following individuals contributed to this draft

Chongfeng Xie, China Telecom, xiechf@chinatelecom.cn

Ron Bonica, Juniper Networks, rbonica@juniper.net

Darren Dukes, Cisco Systems, ddukes@cisco.com

Cheng Li, Huawei, c.l@huawei.com

Peng Shaofu, ZTE, peng.shaofu@zte.com.cn

Wim Henderickx, Nokia, wim.henderickx@nokia.com

## 9. Acknowledgements

TBD

## 10. Normative References

[I-D.ietf-6man-spring-srv6-oam]

Ali, Z., Filsfils, C., Matsushima, S., Voyer, D., and M. Chen, "Operations, Administration, and Maintenance (OAM) in Segment Routing Networks with IPv6 Data plane (SRv6)", draft-ietf-6man-spring-srv6-oam-07 (work in progress), July 2020.

[I-D.ietf-bess-srv6-services]

Dawra, G., Filsfils, C., Raszuk, R., Decraene, B., Zhuang, S., and J. Rabadan, "SRv6 BGP based Overlay services", draft-ietf-bess-srv6-services-04 (work in progress), July 2020.

[I-D.ietf-idr-bgppls-srv6-ext]

Dawra, G., Filsfils, C., Talaulikar, K., Chen, M., daniel.bernier@bell.ca, d., and B. Decraene, "BGP Link State Extensions for SRv6", draft-ietf-idr-bgppls-srv6-ext-03 (work in progress), July 2020.

[I-D.ietf-lsr-flex-algo]

Psenak, P., Hegde, S., Filsfils, C., Talaulikar, K., and A. Gulko, "IGP Flexible Algorithm", draft-ietf-lsr-flex-algo-13 (work in progress), October 2020.

[I-D.ietf-lsr-isis-srv6-extensions]

Psenak, P., Filsfils, C., Bashandy, A., Decraene, B., and Z. Hu, "IS-IS Extension to Support Segment Routing over IPv6 Dataplane", draft-ietf-lsr-isis-srv6-extensions-11 (work in progress), October 2020.

[I-D.ietf-rtgwg-segment-routing-ti-lfa]

Litkowski, S., Bashandy, A., Filsfils, C., Decraene, B., Francois, P., Voyer, D., Clad, F., and P. Camarillo, "Topology Independent Fast Reroute using Segment Routing", draft-ietf-rtgwg-segment-routing-ti-lfa-04 (work in progress), August 2020.



- [I-D.ietf-spring-segment-routing-policy]  
Filsfils, C., Talaulikar, K., Voyer, D., Bogdanov, A., and P. Mattes, "Segment Routing Policy Architecture", draft-ietf-spring-segment-routing-policy-08 (work in progress), July 2020.
- [I-D.ietf-spring-sr-service-programming]  
Clad, F., Xu, X., Filsfils, C., daniel.bernier@bell.ca, d., Li, C., Decraene, B., Ma, S., Yadlapalli, C., Henderickx, W., and S. Salsano, "Service Programming with Segment Routing", draft-ietf-spring-sr-service-programming-03 (work in progress), September 2020.
- [I-D.ietf-spring-srv6-network-programming]  
Filsfils, C., Camarillo, P., Leddy, J., Voyer, D., Matsushima, S., and Z. Li, "SRv6 Network Programming", draft-ietf-spring-srv6-network-programming-24 (work in progress), October 2020.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <<https://www.rfc-editor.org/info/rfc8402>>.
- [RFC8754] Filsfils, C., Ed., Dukes, D., Ed., Previdi, S., Leddy, J., Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header (SRH)", RFC 8754, DOI 10.17487/RFC8754, March 2020, <<https://www.rfc-editor.org/info/rfc8754>>.

#### Authors' Addresses

Weiqliang Cheng  
China Mobile

Email: [chengweiqliang@chinamobile.com](mailto:chengweiqliang@chinamobile.com)

Sanders Steffann  
SJM Steffann Consultancy  
Email: [sander@steffann.nl](mailto:sander@steffann.nl)

SPRING  
Internet-Draft  
Intended status: Informational  
Expires: January 10, 2022

W. Cheng  
China Mobile  
C. Xie  
China Telecom  
R. Bonica  
Juniper  
D. Dukes  
Cisco Systems  
C. Li  
Huawei  
P. Shaofu  
ZTE  
W. Henderickx  
Nokia  
July 09, 2021

Compressed SRv6 SID List Requirements  
draft-srcompdt-spring-compression-requirement-07

Abstract

This document specifies requirements for solutions to compress SRv6 SID lists.

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 January 10, 2022.

Copyright Notice

Copyright (c) 2021 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 . . . . .	3
2. Conventions used in this document . . . . .	4
2.1. Requirements Language . . . . .	4
2.2. Terminology . . . . .	4
3. SRv6 SID List Compression Requirements . . . . .	4
3.1. Dataplane Efficiency and Performance Requirements . . . . .	4
3.1.1. Encapsulation Header Size . . . . .	5
3.1.2. Forwarding Efficiency . . . . .	5
3.1.3. State Efficiency . . . . .	6
4. SRv6 Specific Requirements . . . . .	6
4.1. SRv6 Based . . . . .	6
4.2. Functional Requirements . . . . .	7
4.2.1. SRv6 Functionality . . . . .	7
4.2.2. Heterogeneous SID lists . . . . .	8
4.2.3. SID list length . . . . .	8
4.2.4. SID summarization . . . . .	8
4.3. Operational Requirements . . . . .	9
4.3.1. Lossless Compression . . . . .	9
4.3.2. Preservation of non-routing information . . . . .	9
4.3.3. Address Planning . . . . .	10
4.4. Scalability Requirements . . . . .	10
4.4.1. Adjacency segment scale . . . . .	10
4.4.2. Prefix segment scale . . . . .	11
4.4.3. Service Scale . . . . .	11
4.4.4. Compression Levels . . . . .	11
5. Protocol Design Requirements . . . . .	11
5.1. SRv6 Base Coexistence . . . . .	11
6. Security Requirements . . . . .	12
6.1. Security Mechanisms . . . . .	12
6.2. SR Domain Protection . . . . .	12
7. IANA Considerations . . . . .	12
8. Security Considerations . . . . .	13
9. Contributors . . . . .	13
10. Normative References . . . . .	13
Appendix A. Proposed Requirements . . . . .	14
A.1. IPv6 Based . . . . .	14

A.2. Point to Multipoint . . . . .	15
A.3. Parsability . . . . .	15
Authors' Addresses . . . . .	15

## 1. Introduction

The SPRING working group defined SRv6, with [RFC8402] describing how the Segment Routing (SR) architecture is instantiated on two data-planes: SR over MPLS (SR-MPLS) and SR over IPv6 (SRv6). SRv6 uses a routing header called the SR Header (SRH) [RFC8754] and defines SRv6 SID behaviors and a registry for identifying them in [RFC8986]. SRv6 is a proposed standard and is deployed today.

The SPRING working group has observed that some use cases, such as strict path TE, may require long SRv6 SID lists. There are several proposed methods to reduce the resulting SRv6 encapsulation size by compressing the SID list.

The SPRING working group formed a design team to define requirements for, and analyze proposals to, compress SRv6 SID lists.

It is a goal of the design team to identify solutions to SRv6 SID list compression that are based on the SRv6 standards. As such, this document provides requirements for SRv6 SID list compression solutions that utilize the existing SRv6 data plane and control plane.

It is also a goal of the design team to consider proposals that are not based on the SRv6 data plane and control plane. As such, this document includes requirements to evaluate whether a compression proposal provides all the functionality of SRv6 (section "SRv6 Functionality") in addition to satisfying compression specific requirements.

For each requirement, a description, rationale and metrics are described.

The design team will produce a separate document to analyze the proposals.

This document is a draft; additional requirements are under review, additional requirements will be added, and current requirements may change. Appendix A contains a subset of requirements without unanimous consensus. Additional requirements without unanimous consensus are not in the appendix.

## 2. Conventions used in this document

### 2.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

### 2.2. Terminology

SR: Segment Routing

SRH: Segment Routing Header

MPLS: Multiprotocol Label Switching

SR-MPLS: Segment Routing over MPLS data plane

SID: Segment Identifier

SRv6: Segment Routing over IPv6

SRv6 SID List: A list of SRv6 SIDs

Compression proposal: A proposal to compress SRv6 SID lists

SRv6 base: SRv6 as defined in [RFC8402], [RFC8754], [RFC8986]

SID numbering space: may be implemented as

- o a single IGP instance
- o a single IGP level or area
- o two or more autonomous systems that coordinate SID numbering space
- o two or more IGP instances that coordinate SID numbering space

SRv6 Encapsulation Header: The IPv6 header, and any extension headers preceding a payload, used to implement a SRv6 base or compression proposal.

## 3. SRv6 SID List Compression Requirements

### 3.1. Dataplane Efficiency and Performance Requirements

### 3.1.1. Encapsulation Header Size

Description: The compression proposal MUST reduce the size of the SRv6 encapsulation header.

Rationale: A smaller SRv6 encapsulation results in better MTU efficiency.

Metric: Compression is the ratio of the IPv6 encapsulation size of SRv6 as defined in [RFC8402], [RFC8754], [RFC8986] vs the IPv6 encapsulation size of a given proposal. The encapsulation savings of a compression proposal vs the SRv6 base is a useful measurement to compare proposals.

The encapsulation metric (E) records the number of bytes required for a proposal to encapsulate a packet given a specific segment list.

o  $E(\text{proposal}, \text{segment list})$ .

The encapsulation savings (ES) records the encapsulation savings for a proposal to encapsulate a packet given a specific segment list.

o  $ES(\text{proposal}, \text{segment list}) = 1 - E(\text{proposal}, \text{segment list})/E(\text{SRv6 base}, \text{segment list})$ .

### 3.1.2. Forwarding Efficiency

Description: The compression proposal SHOULD minimize the number of required hardware resources accessed to process a segment.

Rationale: Efficiency in bits on the wire and processing efficiency are both important. Optimizing one at the expense of the other may lead to significant performance impact.

Metric: The data plane efficiency metric (D) records the data plane forwarding efficiency of the proposed solution. Two metrics are used and recorded at each segment endpoint:

- o  $D.PRS(\text{segment list})$ : number of headers parsed during processing of the segment list, starting from and including the IPv6 header.
- o  $D.LKU(\text{segment list})$ : number of FIB lookups during processing of the segment list. The type of lookup is also recorded as longest prefix match (LPM) or exact match (EM)

### 3.1.3. State Efficiency

Description: The compression proposal SHOULD minimize the amount of additional forwarding state stored at a node.

Rationale: Additional state increases the complexity of the control plane and data plane. It can also result in an increase in memory usage.

Metric: The state efficiency metric (S) records the amount of additional forwarding state required by the proposed solution.

- o S(node parameters): the number of additional forwarding states that need to be stored at a node, given a set of node parameters consisting of the number of nodes in the network, number of local interfaces, number of adjacencies. The forwarding state is counted as entries required in a Forwarding Information Base (FIB) at a node.

## 4. SRv6 Specific Requirements

### 4.1. SRv6 Based

Description: A solution to compress SRv6 SID Lists SHOULD be based on the SRv6 architecture, control plane and data plane. The compression solution MAY be based on a different data plane and control plane, provided that it derives sufficient benefit.

Rationale: A compression proposal built on existing IETF standards is preferable to creating new standards with equivalent functionality and performance.

Metric: The utilization metric (U) records whether a proposal utilizes the SRv6 specifications.

Utilization is recorded in a table, with a column per proposal and rows consisting of the following metrics:

- o U.RFC8402: utilizes [RFC8402].
- o U.RFC8754: utilizes [RFC8754].
- o U.PGM: utilizes [RFC8986].
- o U.IGP: utilizes [I-D.ietf-lsr-isis-srv6-extensions].
- o U.BGP: utilizes [I-D.ietf-bess-srv6-services].
- o U.POL: utilizes [I-D.ietf-spring-segment-routing-policy].
- o U.BLS: utilizes [I-D.ietf-idr-bgppls-srv6-ext].
- o U.SVC: utilizes [I-D.ietf-spring-sr-service-programming].
- o U.OAM: utilizes [I-D.ietf-6man-spring-srv6-oam].
- o U.ALG: utilizes [I-D.ietf-lsr-flex-algo].



Each cell contains "yes" for utilizes, or "no" for does not utilize.

## 4.2. Functional Requirements

### 4.2.1. SRv6 Functionality

**Description:** A solution to compress an SRv6 SID list MUST support the functionality of SRv6. This requirement ensures no SRv6 functionality is lost. It is particularly important to understand how a proposal, as evaluated in section "SRv6 Based", provides this functionality.

**Rationale:** Operators require SRv6 functionality. Evaluating the extent to which a proposal supports SRv6 functionality is important for operators and implementors to understand the impact on network operations.

**Metric:** The Functionality metric (F) records whether a proposal supports SRv6 functionality and how the functionality is provided.

Functionality is recorded in a table with columns for each proposal and rows consisting of the following metrics:

- o F.SID: Supports SRv6 SID functionality as described in [RFC8402]
- o F.SCOPE: Supports globally and locally scoped SID functionality as described in [RFC8402]
- o F.PFX: Supports prefix SID functionality as described in [RFC8402] and [RFC8986]
- o F.ADJ: Supports adjacency SID functionality as described in [RFC8402] and [RFC8986]
- o F.BIND: Supports binding SID functionality as described in [RFC8402] and [RFC8986]
- o F.PEER: Supports BGP peering SID functionality as described in [RFC8402] and [RFC8986]
- o F.SVC: Supports L3 and L2 VPN service SID functionality as described in [RFC8986]
- o F.ALG: Supports flexible algorithms functionality as described in [I-D.ietf-lsr-flex-algo]
- o F.TILFA: Supports TI-LFA functionality as described in [I-D.ietf-rtgwg-segment-routing-ti-lfa]
- o F.SEC: Supports securing an SR domain with ingress filtering as functionally defined in [RFC8754]
- o F.IGP: Supports distributing topological SIDs and behaviors via ISIS as functionally described in [I-D.ietf-lsr-isis-srv6-extensions]
- o F.BGP: Supports BGP VPNs as functionally described in [I-D.ietf-bess-srv6-services]

- o F.POL: Supports SR policies and steering traffic over those policies as functionally described in [I-D.ietf-spring-segment-routing-policy]
- o F.BLS: Supports Link State distribution via BGP as functionally described in [I-D.ietf-idr-bgpls-srv6-ext]
- o F.SFC: Supports stateless service programming as functionally described in [I-D.ietf-spring-sr-service-programming]
- o F.PING: Supports pinging a SID to verify the SID is implemented as functionally described in [I-D.ietf-6man-spring-srv6-oam]

Each cell contains the specification name documenting the functionality.

#### 4.2.2. Heterogeneous SID lists

Description: The compression proposal SHOULD support a combination of compressed and non-compressed segments in a single path. As an example, a solution may satisfy this requirement without being SRv6 based by using a binding SID to impose an additional SRv6 header (IPv6 header plus optional SRH) with non-compressed SID.

Rationale: Support of SID lists with compressed and non-compressed SIDs reduces encapsulation size when not all SRv6 nodes deploy the compression proposal or 128-bit SIDs are required.

Metric: A compliant compression proposal supports both:

- o classic 128-bit SRv6 SIDs in the IPv6 Destination Address field
- o segment lists (i.e., paths) with both compressed and 128-bit SRv6 SIDs.

#### 4.2.3. SID list length

Description: The compression proposal MUST be able to represent SR paths that contain up to 16 segments.

Rationale: Strict TE paths require SID list lengths proportional to the diameter of the SR domain.

Metric: The compression proposal must be able to steer a packet through an SR path that contains up to sixteen segments.

#### 4.2.4. SID summarization

Description: The solution MUST be compatible with segment summarization.

**Rationale:** Summarization of segments is a key benefit of SRv6 vs SR MPLS. In interdomain deployments, any node can reach any other node via a single prefix segment. Without summarization, border router SIDs must be leaked, and an additional global prefix segment is required for each domain border to be traversed.

**Metric:** A solution supports summarization when segments can be summarized for advertisement into other IGP domains or levels.

#### 4.3. Operational Requirements

##### 4.3.1. Lossless Compression

**Description:** A path traversed using a compressed SID list MUST always be the same as the path traversed using the uncompressed SID list if no compression was applied.

**Rationale:** In SRv6, we can represent a path to meet certain objectives. A compression proposal needs to support the objectives with the same path.

**Metric:** Information present in the pre-compression segment list MUST also be present in the post-compression SID list.

##### 4.3.2. Preservation of non-routing information

**Description:** The compression mechanism MUST NOT cause the loss of non-routing information when delivering a packet from the SR ingress node to the egress/penultimate SR node

**Rationale:** SRv6 ingress nodes encode non-routing information in the IPv6 header chain. This information can be encoded in the following fields:

- o Hop Count
- o DSCP bits
- o ECN bits
- o Flow label
- o HBH Options Extension header
- o Fragment Extension header
- o Authentication Extension header
- o Encrypted Security Payload Extension header
- o Destination Options Extension header

Some of these fields are mutable (e.g., Hop Count) while others are immutable (e.g., Fragment Extension Header).

Some of these fields contain information that is required by every node along a packet's delivery path (e.g., Hop Count). Others contain information that is required only by the packet's ultimate destination (e.g., Fragment Extension Header).

Therefore, the compression mechanism MUST NOT prevent this information from being delivered, in an IPv6 header chain, to any node that needs it.

**Metric:** The SR source node encapsulates its payload (e.g., Ethernet, IP, TCP) in an IPv6 header. The SRv6 header contains both routing and non-routing information. The compression mechanism MUST NOT cause the loss of non-routing information when delivering a packet from the SR ingress node to the egress/penultimate SR node.

#### 4.3.3. Address Planning

**Description:** Network operators require addressing plan flexibility, The compression mechanism MUST support flexible IPv6 address planning, it MUST support deployment by using GUA from different address blocks.

**Rationale:** The address planning of the network may vary based on the addressing scheme of the operator, so the solution MUST support a flexible addressing scheme. Operators need to deploy the solution based on their own address planning.

**Metric:** The compression proposal supports locators drawn from different prefixes with the solutions analysis indicating efficiency.

#### 4.4. Scalability Requirements

##### 4.4.1. Adjacency segment scale

**Description:** The compression proposal MUST be capable of representing 65000 adjacency segments per node

**Rationale:** Typically, network operators deploy networks with tens or hundreds of adjacency segments per node, but some network operators may deploy networks that use more adjacency segments per node.

**Metric:** A proposal that allows 65000 adjacency segments per node satisfies this requirement.

#### 4.4.2. Prefix segment scale

Description: The compression proposal MUST be capable of representing 1 million prefix segments per SID numbering space.

Rationale: Typically, network operators deploy networks with thousands of prefix segments per SID numbering space, but some network operators may deploy networks that use more prefix segments per SID numbering space.

Metric: A proposal that allows 1 million prefix segments per SID numbering space satisfies this requirement.

#### 4.4.3. Service Scale

Description: The compression proposal MUST be capable of representing 1 million services per node.

Rationale: Typically, network operators deploy networks with tens to hundreds of thousands of services per node, but some network operators may deploy networks that use more services per node.

Metric: A proposal that allows 1 million services per node satisfies this requirement.

#### 4.4.4. Compression Levels

Description: The compression proposal SHOULD be able to support multiple levels of compression.

Rationale: The compression proposal will be deployed in networks of varying size with SID numbering spaces of varying size. Network and service scale can directly impact SID length and the ability of a proposal to compress the SID list.

Metric: A compression proposal that supports relatively better compression for smaller SID numbering spaces and service scale satisfies this requirement.

### 5. Protocol Design Requirements

#### 5.1. SRv6 Base Coexistence

Description: The compression proposal MUST support simultaneous deployment with SRv6 networks.

Rationale: SRv6 is deployed today. A compression proposal that interoperates well with SRv6, as deployed, will reduce the overhead

and simplify operations. For Network operators who would migrate to compressed SRv6 SID lists, the migration is expected to gradually occur over a period of time as they upgrade networks, domains, device families and software instances.

Metric: A compliant compression proposal provides the following

- o Supports simultaneous deployment at a node with current SRv6 SIDs.
- o Supports simultaneous deployment at a node with current SRv6 control plane.
- o Supports simultaneous operation of current SRv6 paths with compressed paths.
- o Supports the behaviors in [RFC8986].
- o Does not require removal of existing IPv6 address planning.

## 6. Security Requirements

### 6.1. Security Mechanisms

Description: The compression solution SHOULD be able to address security issues that it introduces, using existing security mechanisms.

Rationale: It is important to identify security issues and how to address them in any specification.

Metric: A compression solution that does not introduce unresolved security issues meets this requirement.

### 6.2. SR Domain Protection

Description: A compression solution must not require nodes outside the SR domain to know SID values within the SR domain, and it must provide the ability to block nodes outside an SR domain from accessing SIDs.

Rationale: The unauthorized use of SIDs within the SR domain by nodes outside the domain can disrupt an operators' network.

Metric: A compliant solution describes how access to SIDs within the SR domain is denied to nodes outside the SR domain.

## 7. IANA Considerations

This document has no requests to IANA.

## 8. Security Considerations

TBD

## 9. Contributors

The following individuals contributed to this draft

Sanders Steffann, SJM Steffann Consultancy, sander@steffann.nl

## 10. Normative References

[I-D.ietf-6man-spring-srv6-oam]

Ali, Z., Filsfils, C., Matsushima, S., Voyer, D., and M. Chen, "Operations, Administration, and Maintenance (OAM) in Segment Routing Networks with IPv6 Data plane (SRv6)", draft-ietf-6man-spring-srv6-oam-10 (work in progress), April 2021.

[I-D.ietf-bess-srv6-services]

Dawra, G., Filsfils, C., Talaulikar, K., Raszuk, R., Decraene, B., Zhuang, S., and J. Rabadan, "SRv6 BGP based Overlay Services", draft-ietf-bess-srv6-services-07 (work in progress), April 2021.

[I-D.ietf-idr-bgpls-srv6-ext]

Dawra, G., Filsfils, C., Talaulikar, K., Chen, M., Bernier, D., and B. Decraene, "BGP Link State Extensions for SRv6", draft-ietf-idr-bgpls-srv6-ext-07 (work in progress), March 2021.

[I-D.ietf-lsr-flex-algo]

Psenak, P., Hegde, S., Filsfils, C., Talaulikar, K., and A. Gulko, "IGP Flexible Algorithm", draft-ietf-lsr-flex-algo-15 (work in progress), April 2021.

[I-D.ietf-lsr-isis-srv6-extensions]

Psenak, P., Filsfils, C., Bashandy, A., Decraene, B., and Z. Hu, "IS-IS Extension to Support Segment Routing over IPv6 Dataplane", draft-ietf-lsr-isis-srv6-extensions-14 (work in progress), April 2021.

[I-D.ietf-rtgwg-segment-routing-ti-lfa]

Litkowski, S., Bashandy, A., Filsfils, C., Francois, P., Decraene, B., and D. Voyer, "Topology Independent Fast Reroute using Segment Routing", draft-ietf-rtgwg-segment-routing-ti-lfa-06 (work in progress), February 2021.

- [I-D.ietf-spring-segment-routing-policy]  
Filsfils, C., Talaulikar, K., Voyer, D., Bogdanov, A., and P. Mattes, "Segment Routing Policy Architecture", draft-ietf-spring-segment-routing-policy-11 (work in progress), April 2021.
- [I-D.ietf-spring-sr-service-programming]  
Clad, F., Xu, X., Filsfils, C., Bernier, D., Li, C., Decraene, B., Ma, S., Yadlapalli, C., Henderickx, W., and S. Salsano, "Service Programming with Segment Routing", draft-ietf-spring-sr-service-programming-04 (work in progress), March 2021.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <<https://www.rfc-editor.org/info/rfc8402>>.
- [RFC8754] Filsfils, C., Ed., Dukes, D., Ed., Previdi, S., Leddy, J., Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header (SRH)", RFC 8754, DOI 10.17487/RFC8754, March 2020, <<https://www.rfc-editor.org/info/rfc8754>>.
- [RFC8986] Filsfils, C., Ed., Camarillo, P., Ed., Leddy, J., Voyer, D., Matsushima, S., and Z. Li, "Segment Routing over IPv6 (SRv6) Network Programming", RFC 8986, DOI 10.17487/RFC8986, February 2021, <<https://www.rfc-editor.org/info/rfc8986>>.

## Appendix A. Proposed Requirements

This appendix contains requirements that the design team discussed but could not be agreed upon.

### A.1. IPv6 Based

Description: The compression mechanism requires every node along the packet's delivery path to be IPv6-capable. It MUST not require any



node along the packet's forwarding path to support any other forwarding plane (e.g., IPv4, MPLS)

Rational: According to RFC 8402, SRv6 is an instantiation of the SR Architecture over the IPv6 data plane.

Metric: A compliant solution requires every node along the packet's delivery path to be IPv6-capable. It does not require any node along the packet's forwarding path to support any other forwarding plane (e.g., IPv4, MPLS)

#### A.2. Point to Multipoint

Description: The compression mechanism SHOULD support point-to-multipoint SR paths.

Rationale: Many VPN services require point-to-multipoint SR paths.

Metric: A compliant proposal can encode a multicast address in the ultimate segment of the segment list.

#### A.3. Parsability

Description: The compression mechanism MUST be parsable. That is, the node that consumes the compressed SID list must be able to decode the active and next segment. Parsing information MAY be conveyed in either the forwarding or control plane.

Rationale: Failure to parse the compressed SID list leads to undesired behaviors.

Metric: In the nominal case the producer and consumer of the SID list agree on the active segment and next segment. In foreseeable failure modes it is possible to determine why the producer and consumer don't agree.

#### Authors' Addresses

Weiqiang Cheng  
China Mobile

Email: chengweiqiang@chinamobile.com

Chongfeng Xie  
China Telecom

Email: xiechf@chinatelecom.cn

Ron Bonica  
Juniper

Email: [rbonica@juniper.net](mailto:rbonica@juniper.net)

Darren Dukes  
Cisco Systems

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

Cheng Li  
Huawei

Email: [c.l@huawei.com](mailto:c.l@huawei.com)

Peng Shaofu  
ZTE

Email: [peng.shaofu@zte.com.cn](mailto:peng.shaofu@zte.com.cn)

Wim Henderickx  
Nokia

Email: [wim.henderickx@nokia.com](mailto:wim.henderickx@nokia.com)