Stream: Internet Engineering Task Force (IETF)
RFC: 9256
Updates: 8402
Category: Standards Track
Published: July 2022
ISSN: 2070-1721
Authors: C. Filsfils, *Cisco Systems, Inc.*
K. Talaulikar, Ed., *Cisco Systems, Inc.*
D. Voyer, *Bell Canada*
A. Bogdanov, *British Telecom*
P. Mattes, *Microsoft*

RFC 9256
Segment Routing Policy Architecture

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 an 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 RFC 8402 as it details the concepts of SR Policy and steering into an SR Policy.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9256.

Copyright Notice

Table of Contents

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. It 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 an 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 Service Level Agreement (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 the Path Computation Element Communication Protocol (PCEP) [RFC8664] [PCEP-SR-POLICY-CP] or BGP SR Policy [BGP-SR-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:

- The endpoints of the constituent SR Policies and the parent SR Policy **MUST** be identical.

- The colors of each of the constituent SR Policies and the parent SR Policy **MUST** be different.
- 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.

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] [PCEP-SR-POLICY-CP], or BGP [BGP-SR-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 headend assigns different Protocol-Origin values to each source of SR Policy information. The Protocol-Origin value is used as a tiebreaker 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

The Originator identifies the node that 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:

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 0.

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 0.

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 [BGP-SR-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 provisioning is via configuration, this is a unique identifier for a candidate path; it is specific to the implementation's configuration model. The default value is 0.
- When signaling is via PCEP, the method to uniquely signal an individual candidate path along with its Discriminator is described in [PCEP-SR-POLICY-CP]. The default value is 0.
- When signaling is via BGP SR Policy, the BGP process receiving the route provides the distinguisher (refer to [BGP-SR-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 [BGP-SR-POLICY] and [PCEP-SR-POLICY-CP], 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 [BGP-SR-POLICY] and [PCEP-SR-POLICY-CP], 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. The higher value of Protocol-Origin is selected.
2. If specified by configuration, prefer the existing installed path.
3. The lower value of the Originator is selected.

4. Finally, the higher value of the 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:

- 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 as its Preference value is the highest, and if it becomes invalid, then CP2 with the next highest Preference value is selected, 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.
- The Protocol-Origin allows an operator to set up a default selection mechanism across protocol sources, e.g., to prefer configured paths over paths signaled via BGP SR Policy or PCEP.
- 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.
- The Discriminator performs the final tie-breaking step to ensure a deterministic outcome of selection regardless of the order in which candidate paths are signaled across multiple transport channels or sessions.

[SR-POLICY-CONSID] 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/Sw, where w is the weight of the segment list and Sw 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 re-computed 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 hashed on a per-flow basis 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:

- IGP information (topology, IGP metrics based on IS-IS [RFC1195] and OSPF [RFC2328] [RFC5340])
- Segment Routing information (such as Segment Routing Global Block, Segment Routing Local Block, Prefix-SIDs, Adj-SIDs, BGP Peering SID, SRv6 SIDs) [RFC8402] [RFC8986]
- TE Link Attributes (such as TE metric, Shared Risk Link Groups, attribute-flag, extended admin group) [RFC5305] [RFC3630] [RFC5329]
- Extended TE Link attributes (such as latency, loss) [RFC8570] [RFC7471]
- Inter-AS Topology information [RFC9086]

The attached domain topology may be learned via protocol/mechanisms such as IGP, Border Gateway Protocol - Link State (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 [BGP-LS-TE-POLICY] or PCEP [RFC8231] (along with [PCEP-SR-POLICY-CP] and [PCEP-BSID-LABEL]). 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 [SR-POLICY-CONSID].

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 data plane, 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. For example, 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 for identification of an 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 for identification of an 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 for identification of an 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 for identification of an 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 that 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 for 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 for 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 SIDs specific to the IGP Flex Algorithm [IGP-FLEX-ALGO] 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 SRv6 SID list from the segment list, the node instantiating the policy **MUST** interpret the set of Segments as follows:

- The first Segment represents the topmost MPLS label or the first SRv6 SID. It identifies the active segment the traffic will be directed toward along the explicit SR path.
- The last segment represents the bottommost MPLS label or the last SRv6 SID 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 [BGP-SR-POLICY]. 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:

- It is empty.
- Its weight is 0.
- It comprises a mix of SR-MPLS and SRv6 segment types.
- The headend is unable to perform path resolution for the first SID into one or more outgoing interface(s) and next-hop(s).
- 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.
- 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:

- The headend is unable to find the SID in its SR-DB

- 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.
- 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 :

- 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.
- 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 a set of 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.

[SR-POLICY-CONSID] 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. [SR-POLICY-CONSID] 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. A BSID is available when its value is not associated with any other usage, e.g., a label used by some other MPLS forwarding entry or an SRv6 SID used in some other context (such as to another segment, to another SR Policy, or that it is 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.Encaps and End.B6.Encaps.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, the SR Policy **MAY** dynamically bind a BSID to itself. Dynamically bound BSIDs **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 **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 the 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 [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 [SR-POLICY-YANG].

The SR Policy state can be reported by the headend node via BGP-LS [BGP-LS-TE-POLICY] or PCEP [RFC8231] [PCEP-BSID-LABEL].

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 [SR-TRAFFIC-COUNTERS] and [SR-TRAFFIC-ACCOUNTING], respectively.

Implementations **MAY** support an administrative state to control locally provisioned policies via mechanisms like command-line interface (CLI) or NETCONF.

8. Steering into an SR Policy

A headend can steer a packet flow into a valid SR Policy in various ways:

- Incoming packets have an active SID matching a local BSID at the headend.
- Per-Destination Steering: incoming packets match a BGP/Service route, which recurses on an SR Policy.
- Per-Flow Steering: incoming packets match or recurse on a forwarding array of which some of the entries are SR Policies.
- 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 Sections 2.10 and 5.

By default, upon transitioning to the invalid state,

- an SR Policy and its BSID are removed from the forwarding plane.
- 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. This would entail the following:

- an invalid SR Policy and its BSID is kept in the forwarding plane with an action to drop.
- 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.

> "Forwards the resulting packet according to SID 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 to 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:

- learns a BGP route R/r via next-hop N, Color Extended community C, and VPN label V.
- has a valid SR Policy P to (color = C, endpoint = N) of segment list <S1, S2, S3> and BSID B.
- 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 the Locator/ID Separation Protocol (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 next-hop 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:

- learns a BGP route R/r via next-hop N, Color Extended communities C1 and C2.
- has a valid SR Policy P1 to (color = C1, endpoint = N) of segment list <S1, S2, S3> and BSID B1.
- has a valid SR Policy P2 to (color = C2, endpoint = N) of segment list <S4, S5, S6> and BSID B2.
- 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:

- the BGP process learns of a route R/r via N and with Color Extended community C.
- 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 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

This section provides an example of how a headend might apply per-flow steering in practice.

Let us assume that headend H:

- has a valid SR Policy P1 to (color = C1, endpoint = N) of segment list <S1, S2, S3> and BSID B1.
- has a valid SR Policy P2 to (color = C2, endpoint = N) of segment list <S4, S5, S6> and BSID B2.
- 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].
- is configured to match flows in its ingress interfaces (upon any field such as Ethernet destination/source/VLAN/TOS or IP destination/source/Differentiated Services Code Point (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:

- N via recursion on an array A (instead of the immediate outgoing link associated with the IGP shortest path to N).
- Entry A(0) set to the immediate outgoing link of the IGP shortest path to N.
- Entry A(1) set to SR Policy P1 of BSID=B1.

- 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 that recurses on N. H colors these 3 packets respectively with forwarding-class 0, 1, and 2.

As a result, for SR-MPLS:

- H forwards K along the shortest path to N (i.e., pushes the Prefix-SID of N).
- H pushes <S1, S2, S3> on packet K1 and forwards the resulting frame along the shortest path to S1.
- 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 entry 0, 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 that 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 [BGP-SR-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:

- SR Policy (NH, C1).
- SR Policy (null, C1).
- SR Policy (NH, C2).
- SR Policy (null, C2).
- 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.

- H learns a BGP route R/r via next-hop N, Color Extended community C.
- H has a valid SR Policy P to (color = C, endpoint = N) of segment list <S1, S2, S3> and BSID B.
- 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) [SR-TI-LFA] provides a 50 msec 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

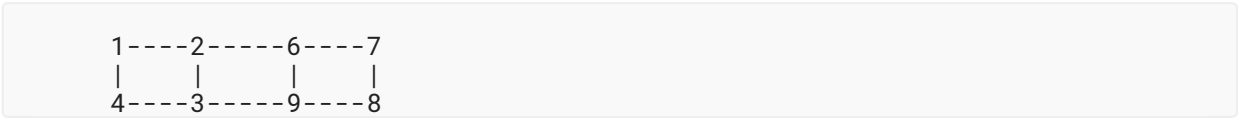

Figure 1: Local Protection Using SR Policy

An SR Policy can be instantiated at node 2 to protect link 2-to-6. 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 2-to-6 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 2-to-6.

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:

- A pair of disjoint candidate paths are provisioned with one of them as primary and the other identified as its backup.
- A specific candidate path is provisioned as the backup for any (active) candidate path.
- 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-reroute mechanism **MAY** then be used to trigger sub-50 msec 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 the 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.

Section 8 specifies mechanisms for the 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 the forwarding of packets for those routes along undesired paths.

In Sections 2.1 and 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 free-form 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 the 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 [SR-POLICY-YANG].

12. IANA Considerations

IANA has created a new subregistry called "Segment Types" under the "Segment Routing" registry that was created by [RFC8986]. This subregistry 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 ZZ. This subregistry follows the Specification Required allocation policy as specified in [RFC8126].

The initial registrations for this subregistry are as follows:

Value	Description	Reference
A	SR-MPLS Label	RFC 9256
B	SRv6 SID	RFC 9256
C	IPv4 Prefix with optional SR Algorithm	RFC 9256
D	IPv6 Global Prefix with optional SR Algorithm for SR-MPLS	RFC 9256
E	IPv4 Prefix with Local Interface ID	RFC 9256
F	IPv4 Addresses for link endpoints as Local, Remote pair	RFC 9256
G	IPv6 Prefix and Interface ID for link endpoints as Local, Remote pair for SR-MPLS	RFC 9256
H	IPv6 Addresses for link endpoints as Local, Remote pair for SR-MPLS	RFC 9256
I	IPv6 Global Prefix with optional SR Algorithm for SRv6	RFC 9256
J	IPv6 Prefix and Interface ID for link endpoints as Local, Remote pair for SRv6	RFC 9256
K	IPv6 Addresses for link endpoints as Local, Remote pair for SRv6	RFC 9256

Table 2: Segment Types

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. References

13.1. 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., Medved, J., Previdi, S., Farrel, A., and S. Ray, "North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP", DOI 10.17487/RFC7752, RFC 7752, 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] Filsfils, C., Previdi, S., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", DOI 10.17487/RFC8402, RFC 8402, 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>.

[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>.

[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>.

13.2. Informative References

[BGP-LS-TE-POLICY] Previdi, S., Talaulikar, K., Ed., Dong, J., Ed., Chen, M., Gredler, H., and J. Tantsura, "Distribution of Traffic Engineering (TE) Policies and State using BGP-LS", Work in Progress, Internet-Draft, draft-ietf-idr-te-lsp-distribution-17, April 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-idr-te-lsp-distribution-17>.

[BGP-SR-POLICY] Previdi, S., Filsfils, C., Talaulikar, K., Ed., Mattes, P., Jain, D., and S. Lin, "Advertising Segment Routing Policies in BGP", Work in Progress, Internet-Draft, draft-ietf-idr-segment-routing-te-policy-18, June 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-idr-segment-routing-te-policy-18>.

[IGP-FLEX-ALGO] Psenak, P., Ed., Hegde, S., Filsfils, C., Talaulikar, K., and A. Gulko, "IGP Flexible Algorithm", Work in Progress, Internet-Draft, draft-ietf-lsr-flex-algo-20, May 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-lsr-flex-algo-20>.

[PCEP-BSID-LABEL] Sivabalan, S., Filsfils, C., Tantsura, J., Previdi, S., and C. Li, Ed., "Carrying Binding Label/Segment Identifier (SID) in PCE-based Networks.", Work in Progress, Internet-Draft, draft-ietf-pce-binding-label-sid-15, March 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-pce-binding-label-sid-15>.

[PCEP-SR-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-07, 21 April 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-pce-segment-routing-policy-cp-07>.

[POI-SR] Anand, M., Bardhan, S., Subrahmaniam, R., Tantsura, J., Mukhopadhyaya, U., and C. Filsfils, "Packet-Optical Integration in Segment Routing", Work in Progress, Internet-Draft, draft-anand-spring-poi-sr-08, 29 July 2019, <https://datatracker.ietf.org/doc/html/draft-anand-spring-poi-sr-08>.

[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 W., "Use of OSI IS-IS for routing in TCP/IP and dual environments", DOI 10.17487/RFC1195, RFC 1195, December 1990, <https://www.rfc-editor.org/info/rfc1195>.

[RFC2328] Moy, J., "OSPF Version 2", DOI 10.17487/RFC2328, STD 54, RFC 2328, 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", DOI 10.17487/RFC5340, RFC 5340, 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. Triantafillis, "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>.

[SR-POLICY-CONSID] Filsfils, C., Talaulikar, K., Ed., Krol, P., Horneffer, M., and P. Mattes, "SR Policy Implementation and Deployment Considerations", Work in Progress, Internet-Draft, draft-filsfils-spring-sr-policy-considerations-09, 24 April 2022, <https://datatracker.ietf.org/doc/html/draft-filsfils-spring-sr-policy-considerations-09>.

[SR-POLICY-YANG] Raza, K., Ed., Sawaya, S., Shunwan, Z., Voyer, D., Durrani, M., Matsushima, S., and V. Beeram, "YANG Data Model for Segment Routing Policy", Work in Progress, Internet-Draft, draft-ietf-spring-sr-policy-yang-01, April 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-spring-sr-policy-yang-01>.

[SR-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://datatracker.ietf.org/doc/html/draft-ietf-rtgwg-segment-routing-ti-lfa-08>.

[SR-TRAFFIC-ACCOUNTING] Ali, Z., Filsfils, C., Talaulikar, K., Sivabalan, S., Horneffer, M., Raszuk, R., Litkowski, S., Voyer, D., Morton, R., and G. Dawra, "Traffic Accounting in Segment Routing Networks", Work in Progress, Internet-Draft, draft-ali-spring-sr-traffic-accounting-07, May 2022, <https://datatracker.ietf.org/doc/html/draft-ali-spring-sr-traffic-accounting-07>.

[SR-TRAFFIC-COUNTERS] Filsfils, C., Ali, Z., Ed., Horneffer, M., Voyer, D., Durrani, M., and R. Raszuk, "Segment Routing Traffic Accounting Counters", Work in Progress, Internet-Draft, draft-filsfils-spring-sr-traffic-counters-02, October 2021, <https://datatracker.ietf.org/doc/html/draft-filsfils-spring-sr-traffic-counters-02>.

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 J. Bernardos for their review, comments, and suggestions.

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

Authors' Addresses

Clarence Filsfils
Cisco Systems, Inc.
Pegasus Parc
De kleetlaan 6a
1831 Diegem
Belgium
Email: cfilsfil@cisco.com

Ketan Talaulikar (EDITOR)
Cisco Systems, Inc.
India
Email: ketant.ietf@gmail.com

Daniel Voyer
Bell Canada
671 de la gauchetiere W
Montreal Quebec H3B 2M8
Canada
Email: daniel.voyer@bell.ca

Alex Bogdanov
British Telecom
Email: alex.bogdanov@bt.com

Paul Mattes
Microsoft
One Microsoft Way
Redmond, WA 98052-6399
United States of America
Email: pamattes@microsoft.com