Internet Engineering Task Force (IETF)
Request for Comments: 9020
Category: Standards Track
ISSN: 2070-1721

S. Litkowski
Cisco Systems
Y. Qu
Futurewei
A. Lindem
Cisco Systems
P. Sarkar
VMware, Inc
J. Tantsura
Juniper Networks

May 2021

YANG Data Model for Segment Routing

Abstract

This document defines three YANG data models. The first is for Segment Routing (SR) configuration and operation, which is to be augmented by different Segment Routing data planes. The next is a YANG data model that defines a collection of generic types and groupings for SR. The third module defines the configuration and operational states for the Segment Routing MPLS data plane.

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/rfc9020.

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

This document defines three YANG data models [RFC7950]. The first one is for Segment Routing (SR) [RFC8402] configuration and operation. This document does not define the IGP extensions to support SR, but the second module defines generic groupings to be reused by IGP extension modules. The reason for this design choice is to not require implementations to support all IGP extensions. For example, an implementation may support the IS-IS extension but not the OSPF extension. The third YANG data model defines a module that is intended to be used on network elements to configure or operate the SR MPLS data plane [RFC8660].

The YANG modules in this document conform to the Network Management Datastore Architecture (NMDA) [RFC8342].

2. Terminology and Notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "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.

2.1. Tree Diagram

Tree diagrams used in this document follow the notation defined in [RFC8340].

2.2. Prefixes in Data Node Names

In this document, names of data nodes, actions, and other data model objects are often used without a prefix, as long as it is clear from the context in which YANG module each name is defined. Otherwise, names are prefixed using the standard prefix associated with the corresponding YANG module, as shown in Table 1.

Prefix	YANG module	Reference
if	ietf-interfaces	[RFC8343]
rt	ietf-routing	[RFC8349]
rt-types	ietf-routing-types	[RFC8294]

yang	ietf-yang-types	[RFC6991]
inet	ietf-inet-types	[RFC6991]

Table 1: Prefixes and Corresponding YANG Modules

3. Design of the Data Model

The ietf-segment-routing YANG module augments the routing container in the ietf-routing model [RFC8349] and defines generic SR configuration and operational state. This module is augmented by modules supporting different data planes.

Module ietf-segment-routing-mpls augments ietf-segment-routing and supports SR-MPLS data plane configuration and operational state.

Module ietf-segment-routing-common defines generic types and groupings that SHOULD be reused by IGP extension modules.

```
module: ietf-segment-routing
  augment /rt:routing:
    +--rw segment-routing

module: ietf-segment-routing-mpls
  augment /rt:routing/sr:segment-routing:
    +--rw sr-mpls
       +--rw bindings
       |  +--rw mapping-server {mapping-server}?
       |  |  +--rw policy* [name]
       |  |     +--rw name       string
       |  |     +--rw entries
       |  |        +--rw mapping-entry* [prefix algorithm]
       |  |           +--rw prefix        inet:ip-prefix
       |  |           +--rw value-type?   enumeration
       |  |           +--rw start-sid     uint32
       |  |           +--rw range?        uint32
       |  |           +--rw algorithm     identityref
       |  +--rw connected-prefix-sid-map
       |  |  +--rw connected-prefix-sid* [prefix algorithm]
       |  |     +--rw prefix               inet:ip-prefix
       |  |     +--rw value-type?          enumeration
       |  |     +--rw start-sid            uint32
       |  |     +--rw range?               uint32
       |  |     +--rw algorithm            identityref
       |  |     +--rw last-hop-behavior?   enumeration
       |  +--rw local-prefix-sid
       |     +--rw local-prefix-sid* [prefix algorithm]
       |        +--rw prefix        inet:ip-prefix
       |        +--rw value-type?   enumeration
       |        +--rw start-sid     uint32
       |        +--rw range?        uint32
       |        +--rw algorithm     identityref
       +--rw srgb
       |  +--rw srgb* [lower-bound upper-bound]
       |     +--rw lower-bound    uint32
       |     +--rw upper-bound    uint32
       +--rw srlb
       |  +--rw srlb* [lower-bound upper-bound]
       |     +--rw lower-bound    uint32
       |     +--rw upper-bound    uint32
       +--ro label-blocks* []
       |  +--ro lower-bound?   uint32
       |  +--ro upper-bound?   uint32
```

```
        |  +--ro size?          uint32
        |  +--ro free?          uint32
        |  +--ro used?          uint32
        |  +--ro scope?         enumeration
        +--ro sid-db
           +--ro sid* [target sid source source-protocol binding-type]
              +--ro target             string
              +--ro sid                uint32
              +--ro algorithm?         uint8
              +--ro source             inet:ip-address
              +--ro used?              boolean
              +--ro source-protocol    -> /rt:routing
                                          /control-plane-protocols
                                          /control-plane-protocol/name
              +--ro binding-type       enumeration
              +--ro scope?             enumeration

  notifications:
    +---n segment-routing-srgb-collision
    |  +--ro srgb-collisions* []
    |     +--ro lower-bound?          uint32
    |     +--ro upper-bound?          uint32
    |     +--ro routing-protocol?     -> /rt:routing
    |                                    /control-plane-protocols
    |                                    /control-plane-protocol/name
    |     +--ro originating-rtr-id?   router-or-system-id
    +---n segment-routing-global-sid-collision
    |  +--ro received-target?       string
    |  +--ro new-sid-rtr-id?        router-or-system-id
    |  +--ro original-target?       string
    |  +--ro original-sid-rtr-id?   router-or-system-id
    |  +--ro index?                 uint32
    |  +--ro routing-protocol?      -> /rt:routing
    |                                  /control-plane-protocols
    |                                  /control-plane-protocol/name
    +---n segment-routing-index-out-of-range
       +--ro received-target?    string
       +--ro received-index?     uint32
       +--ro routing-protocol?   -> /rt:routing
                                    /control-plane-protocols
                                    /control-plane-protocol/name
```

4. Configuration

The module ietf-segment-routing-mpls augments the "/rt:routing/ sr:segment-routing:" with an sr-mpls container. This container defines all the configuration parameters related to the SR MPLS data plane.

The sr-mpls configuration is split into global configuration and interface configuration.

The global configuration includes:

Bindings: Defines Prefix to Segment Identifier (Prefix-SID) mappings. The operator can control advertisement of Prefix-SIDs independently for IPv4 and IPv6. Two types of mappings are available:

Mapping-server: Maps prefixes that are not local to a SID. Configuration of bindings does not automatically allow advertisement of those bindings. Advertisement must be controlled by each routing-protocol instance (see Section 5). Multiple mapping policies may be defined.

Connected prefixes: Maps connected prefixes to a SID.

Advertisement of the mapping will be done by IGP when enabled for SR (see Section 5). The SID value can be expressed as an index (default) or an absolute value. The "last-hop-behavior" configuration dictates the MPLS Penultimate Hop Popping (PHP) behavior: "explicit-null", "php", or "non-php".

Segment Routing Global Block (SRGB): Defines a list of label blocks represented by a pair of lower-bound/upper-bound labels. The SRGB is also agnostic to the control plane used. So, all local routing-protocol instances will have to advertise the same SRGB.

Segment Routing Local Block (SRLB): Defines a list of label blocks represented by a pair of lower-bound/upper-bound labels reserved for local SIDs.

5. IGP Control-Plane Configuration

Support of SR extensions for a particular IGP control plane is achieved by augmenting routing-protocol configuration with SR extensions. This augmentation SHOULD be part of the routing-protocol YANG modules as not to create any dependency for implementations to support SR extensions for all routing protocols.

This module defines groupings that SHOULD be used by IGP SR modules.

The "sr-control-plane" grouping defines the generic global configuration for the IGP.

The "enabled" leaf enables SR extensions for the routing-protocol instance.

The "bindings" container controls the routing-protocol instance’s advertisement of local bindings and the processing of received bindings.

5.1. IGP Interface Configuration

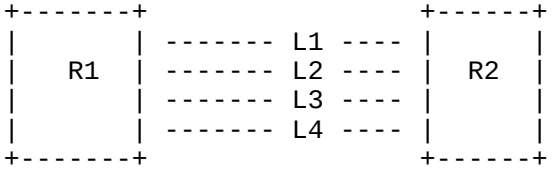
The interface configuration is part of the "igp-interface" grouping and includes Adjacency SID (Adj-SID) properties.

5.1.1. Adjacency SID (Adj-SID) Properties

5.1.1.1. Bundling

In case of parallel IP links between routers, an additional Adj-SID [RFC8402] may be advertised representing more than one adjacency (i.e., a bundle of adjacencies). The "advertise-adj-group-sid" configuration controls for which group(s) an additional Adj-SID is advertised.

The "advertise-adj-group-sid" is a list of group IDs. Each group ID will identify interfaces that are bundled together.



In the figure above, R1 and R2 are interconnected by four links. A routing protocol adjacency is established on each link. The operator would like to create Adj-SIDs that represent bundles of links. We can imagine two different bundles: L1/L2 and L3/L4. To achieve this behavior, the operator will configure a "group-id" X for interfaces L1 and L2 and a "group-id" Y for interfaces L3 and L4. This will

result in R1 advertising an additional Adj-SID for each adjacency. For example, an Adj-SID with a value of 400 will be added to L1 and L2, and an Adj-SID with a value of 500 will be added to L3 and L4. As L1/L2 and L3/L4 do not share the same "group-id", a different SID value will be allocated.

5.1.1.2. Protection

The "advertise-protection" defines how protection for an interface is advertised. It does not control the activation or deactivation of protection. If the "single" option is used, a single Adj-SID will be advertised for the interface. If the interface is protected, the B-Flag for the Adj-SID advertisement will be set. If the "dual" option is used and if the interface is protected, two Adj-SIDs will be advertised for the interface adjacencies. One Adj-SID will always have the B-Flag set, and the other will have the B-Flag clear. This option is intended to be used in the case of traffic engineering where a path must use either protected segments or unprotected segments.

6. State Data

The operational state contains information reflecting the usage of allocated SRGB labels.

It also includes a list of all global SIDs, their associated bindings, and other information, such as the associated source protocol and algorithm.

7. Notifications

The model defines the following notifications for SR.

segment-routing-srgb-collision: Raised when control-plane-advertised SRGB blocks have conflicts

segment-routing-global-sid-collision: Raised when a control-plane-advertised index is already associated with another target (in this version, the only defined targets are IPv4 and IPv6 prefixes)

segment-routing-index-out-of-range: Raised when a control-plane-advertised index falls outside the range of SRGBs configured for the network device

8. YANG Modules

There are three YANG modules included in this document.

The following RFCs are not referenced in the document text but are referenced in the ietf-segment-routing.yang, ietf-segment-routing-common.yang, and/or ietf-segment-routing-mpls.yang modules: [RFC6991], [RFC8294], [RFC8661], [RFC8665], [RFC8667], [RFC8669], and [RFC8814].

8.1. YANG Module for Segment Routing

ietf-segment-routing.yang: This module defines a generic framework for Segment Routing (SR), and it is to be augmented by models for different SR data planes.

```
<CODE BEGINS> file "ietf-segment-routing@2021-05-26.yang"
module ietf-segment-routing {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-segment-routing";
  prefix sr;
```

```
import ietf-routing {
  prefix rt;
  reference "RFC 8349: A YANG Data Model for Routing
                       Management (NMDA Version)";
}

organization
  "IETF SPRING - SPRING Working Group";
contact
  "WG Web:   <https://datatracker.ietf.org/wg/spring/>
   WG List:  <mailto:spring@ietf.org>

   Author:    Stephane Litkowski
              <mailto:slitkows.ietf@gmail.com>
   Author:    Yingzhen Qu
              <mailto:yingzhen.qu@futurewei.com>
   Author:    Acee Lindem
              <mailto:acee@cisco.com>
   Author:    Pushpasis Sarkar
              <mailto:pushpasis.ietf@gmail.com>
   Author:    Jeff Tantsura
              <jefftant.ietf@gmail.com>

  ";
description
  "This YANG module defines a generic framework for Segment
   Routing (SR).  It is to be augmented by models for different
   SR data planes.

   This YANG module conforms to the Network Management
   Datastore Architecture (NMDA), as described in RFC 8242.

   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 (RFC 2119) (RFC 8174) when, and only when,
   they appear in all capitals, as shown here.

   Copyright (c) 2021 IETF Trust and the persons identified as
   authors of the code.  All rights reserved.

   Redistribution and use in source and binary forms, with or
   without modification, is permitted pursuant to, and subject
   to the license terms contained in, the Simplified BSD License
   set forth in Section 4.c of the IETF Trust's Legal Provisions
   Relating to IETF Documents
   (https://trustee.ietf.org/license-info).

   This version of this YANG module is part of RFC 9020;
   see the RFC itself for full legal notices.";

reference
  "RFC 9020: YANG Data Model for Segment Routing.";

revision 2021-05-26 {
  description
    "Initial version";
  reference
    "RFC 9020: YANG Data Model for Segment Routing.";
}

augment "/rt:routing" {
  description
    "This module augments the routing data model (RFC 8349)
     with Segment Routing (SR).";
  container segment-routing {
```

```
        description
          "Segment Routing configuration.  This container
           is to be augmented by models for different SR
           data planes.";
        reference
          "RFC 8402: Segment Routing Architecture.";
      }
    }
  }
  <CODE ENDS>
```

8.2. YANG Module for Segment Routing Common Types

ietf-segment-routing-common.yang: This module defines a collection of generic types and groupings for SR, as defined in [RFC8402].

```
  <CODE BEGINS> file "ietf-segment-routing-common@2021-05-26.yang"
  module ietf-segment-routing-common {
    yang-version 1.1;
    namespace
     "urn:ietf:params:xml:ns:yang:ietf-segment-routing-common";
    prefix sr-cmn;

    import ietf-inet-types {
      prefix inet;
      reference
        "RFC 6991: Common YANG Data Types";
    }

    organization
      "IETF SPRING - SPRING Working Group";
    contact
      "WG Web:   <https://datatracker.ietf.org/wg/spring/>
       WG List:  <mailto:spring@ietf.org>

       Author:    Stephane Litkowski
                  <mailto:slitkows.ietf@gmail.com>
       Author:    Yingzhen Qu
                  <mailto:yingzhen.qu@futurewei.com>
       Author:    Acee Lindem
                  <mailto:acee@cisco.com>
       Author:    Pushpasis Sarkar
                  <mailto:pushpasis.ietf@gmail.com>
       Author:    Jeff Tantsura
                  <jefftant.ietf@gmail.com>

      ";
    description
      "This YANG module defines a collection of generic types and
       groupings for Segment Routing (SR), as described in RFC 8402.

       This YANG module conforms to the Network Management
       Datastore Architecture (NMDA), as described in RFC 8242.

       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 (RFC 2119) (RFC 8174) when, and only when,
       they appear in all capitals, as shown here.

       Copyright (c) 2021 IETF Trust and the persons identified as
       authors of the code.  All rights reserved.

       Redistribution and use in source and binary forms, with or
       without modification, is permitted pursuant to, and subject
       to the license terms contained in, the Simplified BSD License
```

```
   set forth in Section 4.c of the IETF Trust's Legal Provisions
   Relating to IETF Documents
   (https://trustee.ietf.org/license-info).

   This version of this YANG module is part of RFC 9020;
   see the RFC itself for full legal notices.";
reference
  "RFC 9020: YANG Data Model for Segment Routing";

revision 2021-05-26 {
  description
    "Initial version";
  reference
    "RFC 9020: YANG Data Model for Segment Routing";
}

feature sid-last-hop-behavior {
  description
    "Configurable last-hop behavior.";
  reference
    "RFC 8660: Segment Routing with the MPLS Data Plane";
}

identity prefix-sid-algorithm {
  description
    "Base identity for prefix-sid algorithm.";
  reference
    "RFC 8402: Segment Routing Architecture";
}

identity prefix-sid-algorithm-shortest-path {
  base prefix-sid-algorithm;
  description
    "Shortest Path First (SPF) Prefix-SID algorithm.  This
     is the default algorithm.";
}

identity prefix-sid-algorithm-strict-spf {
  base prefix-sid-algorithm;
  description
    "This algorithm mandates that the packet is forwarded
     according to the ECMP-aware SPF algorithm.";
}

grouping srlr {
  description
    "Grouping for SR Label Range configuration.";
  leaf lower-bound {
    type uint32;
    description
      "Lower value in the label range.";
  }
  leaf upper-bound {
    type uint32;
    must '../lower-bound < ../upper-bound' {
      error-message
        "The upper-bound must be greater than the lower-bound.";
      description
        "The value must be greater than lower-bound.";
    }
    description
      "Upper value in the label range.";
  }
}

grouping srgb {
```

```
  description
    "Grouping for SR Global Label Range.";
  list srgb {
    key "lower-bound upper-bound";
    ordered-by user;
    description
      "List of global blocks to be advertised.";
    uses srlr;
  }
}

grouping srlb {
  description
    "Grouping for SR Local Block Range.";
  list srlb {
    key "lower-bound upper-bound";
    ordered-by user;
    description
      "List of SRLBs.";
    uses srlr;
  }
}

grouping sid-value-type {
  description
    "Defines how the SID value is expressed.";
  leaf value-type {
    type enumeration {
      enum index {
        description
          "The value will be interpreted as an index.";
      }
      enum absolute {
        description
          "The value will become interpreted as an absolute
           value.";
      }
    }
    default "index";
    description
      "This leaf defines how the value must be interpreted.";
  }
}

grouping prefix-sid {
  description
    "This grouping defines configuration of a Prefix-SID.";
  leaf prefix {
    type inet:ip-prefix;
    description
      "Connected Prefix-SID.";
  }
  uses prefix-sid-attributes;
}

grouping ipv4-sid {
  description
    "Grouping for an IPv4 Prefix-SID.";
  leaf prefix {
    type inet:ipv4-prefix;
    description
      "Connected IPv4 Prefix-SID.";
  }
  uses prefix-sid-attributes;
}
```

```
    grouping ipv6-sid {
      description
        "Grouping for an IPv6 Prefix-SID.";
      leaf prefix {
        type inet:ipv6-prefix;
        description
          "Connected IPv6 Prefix-SID.";
      }
      uses prefix-sid-attributes;
    }

    grouping last-hop-behavior {
      description
        "Defines last-hop behavior.";
      leaf last-hop-behavior {
        if-feature "sid-last-hop-behavior";
        type enumeration {
          enum explicit-null {
            description
              "Use explicit-null for the SID.";
          }
          enum no-php {
            description
              "Do not use MPLS Penultimate Hop Popping (PHP)
               for the SID.";
          }
          enum php {
            description
              "Use MPLS PHP for the SID.";
          }
        }
        description
          "Configure last-hop behavior.";
      }
    }

    grouping prefix-sid-attributes {
      description
        "Grouping for Segment Routing (SR) prefix attributes.";
      uses sid-value-type;
      leaf start-sid {
        type uint32;
        mandatory true;
        description
          "Value associated with prefix.  The value must be
           interpreted in the context of sid-value-type.";
      }
      leaf range {
        type uint32;
        description
          "Indicates how many SIDs can be allocated.";
      }
      leaf algorithm {
        type identityref {
          base prefix-sid-algorithm;
        }
        description
          "Prefix-SID algorithm.";
      }
    }
  }
  <CODE ENDS>
```

8.3. YANG Module for Segment Routing MPLS

ietf-segment-routing-mpls.yang: This module defines the

```
   configuration and operational states for the Segment Routing MPLS
   data plane.

<CODE BEGINS> file "ietf-segment-routing-mpls@2021-05-26.yang"
module ietf-segment-routing-mpls {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-segment-routing-mpls";
  prefix sr-mpls;

  import ietf-inet-types {
    prefix inet;
    reference
      "RFC 6991: Common YANG Data Types";
  }
  import ietf-routing {
    prefix rt;
    reference
      "RFC 8349: A YANG Data Model for Routing
                 Management (NMDA Version)";
  }
  import ietf-routing-types {
    prefix rt-types;
    reference
      "RFC 8294: Common YANG Data Types for the
                 Routing Area";
  }
  import ietf-segment-routing {
    prefix sr;
    reference
      "RFC 9020: YANG Data Model for Segment Routing";
  }
  import ietf-segment-routing-common {
    prefix sr-cmn;
    reference
      "RFC 9020: YANG Data Model for Segment Routing";
  }

  organization
    "IETF SPRING - SPRING Working Group";
  contact
    "WG Web:   <https://datatracker.ietf.org/wg/spring/>
     WG List:  <mailto:spring@ietf.org>

     Author:    Stephane Litkowski
                <mailto:slitkows.ietf@gmail.com>
     Author:    Yingzhen Qu
                <mailto:yingzhen.qu@futurewei.com>
     Author:    Acee Lindem
                <mailto:acee@cisco.com>
     Author:    Pushpasis Sarkar
                <mailto:pushpasis.ietf@gmail.com>
     Author:    Jeff Tantsura
                <jefftant.ietf@gmail.com>

    ";
  description
    "This YANG module defines a generic configuration model for
     the Segment Routing MPLS data plane.

     This YANG module conforms to the Network Management
     Datastore Architecture (NMDA), as described in RFC 8242.

     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 (RFC 2119) (RFC 8174) when, and only when,
```

```
   they appear in all capitals, as shown here.

   Copyright (c) 2021 IETF Trust and the persons identified as
   authors of the code.  All rights reserved.

   Redistribution and use in source and binary forms, with or
   without modification, is permitted pursuant to, and subject
   to the license terms contained in, the Simplified BSD License
   set forth in Section 4.c of the IETF Trust's Legal Provisions
   Relating to IETF Documents
   (https://trustee.ietf.org/license-info).

   This version of this YANG module is part of RFC 9020;
   see the RFC itself for full legal notices.";
reference
  "RFC 9020: YANG Data Model for Segment Routing";

revision 2021-05-26 {
  description
    "Initial version";
  reference
    "RFC 9020: YANG Data Model for Segment Routing";
}

feature mapping-server {
  description
    "Support for Segment Routing Mapping Server (SRMS).";
  reference
    "RFC 8661: Segment Routing MPLS Interworking
               with LDP";
}

feature protocol-srgb {
  description
    "Support for per-protocol Segment Routing Global Block
     (SRGB) configuration.";
  reference
    "RFC 8660: Segment Routing with the MPLS
               Data Plane";
}

typedef system-id {
  type string {
    pattern '[0-9A-Fa-f]{4}\.[0-9A-Fa-f]{4}\.[0-9A-Fa-f]{4}';
  }
  description
    "This type defines an IS-IS system-id using a pattern.
     An example system-id is 0143.0438.AEF0.";
}

typedef router-or-system-id {
  type union {
    type rt-types:router-id;
    type system-id;
  }
  description
    "OSPF/BGP router-id or IS-IS system ID.";
}

grouping sr-control-plane {
  description
    "Defines protocol configuration.";
  container segment-routing {
    description
      "Segment Routing global configuration.";
    leaf enabled {
```

```
      type boolean;
      default "false";
      description
        "Enables Segment Routing control-plane protocol
         extensions.";
    }
    container bindings {
      if-feature "mapping-server";
      description
        "Control of binding advertisement and reception.";
      container advertise {
        description
          "Control advertisement of local mappings
           in binding TLVs.";
        leaf-list policies {
          type leafref {
            path "/rt:routing/sr:segment-routing/sr-mpls:sr-mpls"
               + "/sr-mpls:bindings/sr-mpls:mapping-server"
               + "/sr-mpls:policy/sr-mpls:name";
          }
          description
            "List of binding advertisement policies.";
        }
      }
      leaf receive {
        type boolean;
        default "true";
        description
          "Allow the reception and usage of binding TLVs.";
      }
    }
  }
}

grouping igp-interface {
  description
    "Grouping for IGP interface configuration.";
  container segment-routing {
    description
      "Container for SR interface configuration.";
    container adjacency-sid {
      description
        "Adjacency SID (Adj-SID) configuration.";
      reference
        "RFC 8660: Segment Routing with the MPLS
                   Data Plane";
      list adj-sids {
        key "value";
        uses sr-cmn:sid-value-type;
        leaf value {
          type uint32;
          description
            "Value of the Adj-SID.";
        }
        leaf protected {
          type boolean;
          default "false";
          description
            "It is used to protect the Adj-SID, e.g., using
             IP Fast Reroute (IPFRR) or MPLS-FRR.";
        }
        leaf weight {
          type uint8;
          description
            "The load-balancing factor over parallel adjacencies.";
          reference
```

```
            "RFC 8402: Segment Routing Architecture
             RFC 8665: OSPF Extensions for Segment Routing
             RFC 8667: IS-IS Extensions for Segment
                       Routing";
        }
        description
          "List of Adj-SIDs and their configuration.";
      }
      list advertise-adj-group-sid {
        key "group-id";
        description
          "Control advertisement of S-flag or G-flag.  Enable
           advertisement of a common Adj-SID for parallel
           links.";
        reference
          "RFC 8665: OSPF Extensions for Segment Routing,
                     Section 6.1
           RFC 8667: IS-IS Extensions for Segment
                     Routing, Section 2.2.1";
        leaf group-id {
          type uint32;
          description
            "The value is an internal value to identify a
             group-ID. Interfaces with the same group-ID
             will be bundled together.";
        }
      }
      leaf advertise-protection {
        type enumeration {
          enum single {
            description
              "A single Adj-SID is associated with the
               adjacency and reflects the protection
               configuration.";
          }
          enum dual {
            description
              "Two Adj-SIDs will be associated with the adjacency
               if the interface is protected.  In this case, one
               Adj-SID will be advertised with the backup-flag
               set and the other with the backup-flag clear.  In
               the case where protection is not configured, a
               single Adj-SID will be advertised with the
               backup-flag clear.";
          }
        }
        description
          "If set, the Adj-SID refers to a protected adjacency.";
        reference
          "RFC 8665: OSPF Extensions for Segment Routing,
                     Section 6.1
           RFC 8667: IS-IS Extensions for Segment
                     Routing, Section 2.2.1";
      }
    }
  }
}

augment "/rt:routing/sr:segment-routing" {
  description
    "This augments the routing data model (RFC 8349)
     with Segment Routing (SR) using the MPLS data plane.";
  container sr-mpls {
    description
      "Segment Routing global configuration and
       operational state.";
```

```
container bindings {
  description
    "List of bindings.";
  container mapping-server {
    if-feature "mapping-server";
    description
      "Configuration of mapping-server local entries.";
    list policy {
      key "name";
      description
        "List mapping-server policies.";
      leaf name {
        type string;
        description
          "Name of the mapping policy.";
      }
      container entries {
        description
          "IPv4/IPv6 mapping entries.";
        list mapping-entry {
          key "prefix algorithm";
          description
            "Mapping entries.";
          uses sr-cmn:prefix-sid;
        }
      }
    }
  }
  container connected-prefix-sid-map {
    description
      "Prefix-SID configuration.";
    list connected-prefix-sid {
      key "prefix algorithm";
      description
        "List of mappings of Prefix-SIDs to IPv4/IPv6
         local prefixes.";
      uses sr-cmn:prefix-sid;
      uses sr-cmn:last-hop-behavior;
    }
  }
  container local-prefix-sid {
    description
      "Local SID configuration.";
    list local-prefix-sid {
      key "prefix algorithm";
      description
        "List of local IPv4/IPv6 Prefix-SIDs.";
      uses sr-cmn:prefix-sid;
    }
  }
}
container srgb {
  description
    "Global SRGB configuration.";
  uses sr-cmn:srgb;
}
container srlb {
  description
    "Segment Routing Local Block (SRLB) configuration.";
  uses sr-cmn:srlb;
}
list label-blocks {
  config false;
  description
    "List of label blocks currently in use.";
  leaf lower-bound {
```

```
      type uint32;
      description
        "Lower bound of the label block.";
    }
    leaf upper-bound {
      type uint32;
      description
        "Upper bound of the label block.";
    }
    leaf size {
      type uint32;
      description
        "Number of indexes in the block.";
    }
    leaf free {
      type uint32;
      description
        "Number of free indexes in the block.";
    }
    leaf used {
      type uint32;
      description
        "Number of indexes in use in the block.";
    }
    leaf scope {
      type enumeration {
        enum global {
          description
            "Global SID.";
        }
        enum local {
          description
            "Local SID.";
        }
      }
      description
        "Scope of this label block.";
    }
  }
  container sid-db {
    config false;
    description
      "List of prefix and SID associations.";
    list sid {
      key "target sid source source-protocol binding-type";
      ordered-by system;
      description
        "SID binding.";
      leaf target {
        type string;
        description
          "Defines the target of the binding.  It can be a
           prefix or something else.";
      }
      leaf sid {
        type uint32;
        description
          "Index associated with the prefix.";
      }
      leaf algorithm {
        type uint8;
        description
          "Algorithm to be used for the Prefix-SID.";
        reference
          "RFC 8665: OSPF Extensions for Segment Routing
           RFC 8667: IS-IS Extensions for Segment
```

```
                    Routing
             RFC 8669: Segment Routing Prefix Segment
                    Identifier Extensions to BGP";
        }
        leaf source {
          type inet:ip-address;
          description
            "IP address of the router that owns the binding.";
        }
        leaf used {
          type boolean;
          description
            "Indicates if the binding is installed in the
             forwarding plane.";
        }
        leaf source-protocol {
          type leafref {
            path "/rt:routing/rt:control-plane-protocols/"
               + "rt:control-plane-protocol/rt:name";
          }
          description
            "Routing protocol that owns the binding.";
        }
        leaf binding-type {
          type enumeration {
            enum prefix-sid {
              description
                "Binding is learned from a Prefix-SID.";
            }
            enum binding-tlv {
              description
                "Binding is learned from a binding TLV.";
            }
          }
          description
            "Type of binding.";
        }
        leaf scope {
          type enumeration {
            enum global {
              description
                "Global SID.";
            }
            enum local {
              description
                "Local SID.";
            }
          }
          description
            "SID scoping.";
        }
      }
    }
  }
}

notification segment-routing-srgb-collision {
  description
    "This notification is sent when SRGB blocks received from
     different routers collide.";
  list srgb-collisions {
    description
      "List of SRGB blocks that collide.";
    leaf lower-bound {
      type uint32;
      description
```

```
        "Lower value in the block.";
    }
    leaf upper-bound {
      type uint32;
      description
        "Upper value in the block.";
    }
    leaf routing-protocol {
      type leafref {
        path "/rt:routing/rt:control-plane-protocols/"
           + "rt:control-plane-protocol/rt:name";
      }
      description
        "Routing protocol reference for SRGB collision.";
    }
    leaf originating-rtr-id {
      type router-or-system-id;
      description
        "Originating router ID of this SRGB block.";
    }
  }
}

notification segment-routing-global-sid-collision {
  description
    "This notification is sent when a new mapping is learned
     containing a mapping where the SID is already used.
     The notification generation must be throttled with at least
     a 5-second gap between notifications.";
  leaf received-target {
    type string;
    description
      "Target received in the router advertisement that caused
       the SID collision.";
  }
  leaf new-sid-rtr-id {
    type router-or-system-id;
    description
      "Router ID that advertised the colliding SID.";
  }
  leaf original-target {
    type string;
    description
      "Target already available in the database with the same SID
       as the received target.";
  }
  leaf original-sid-rtr-id {
    type router-or-system-id;
    description
      "Router ID for the router that originally advertised the
       colliding SID, i.e., the instance in the database.";
  }
  leaf index {
    type uint32;
    description
      "Value of the index used by two different prefixes.";
  }
  leaf routing-protocol {
    type leafref {
      path "/rt:routing/rt:control-plane-protocols/"
         + "rt:control-plane-protocol/rt:name";
    }
    description
      "Routing protocol reference for colliding SID.";
  }
}
```

```
  notification segment-routing-index-out-of-range {
    description
      "This notification is sent when a binding is received
       containing a segment index that is out of the local
       configured ranges.  The notification generation must be
       throttled with at least a 5-second gap between
       notifications.";
    leaf received-target {
      type string;
      description
        "A human-readable string representing the target
         received in the protocol-specific advertisement
         corresponding to the out-of-range index.";
    }
    leaf received-index {
      type uint32;
      description
        "Value of the index received.";
    }
    leaf routing-protocol {
      type leafref {
        path "/rt:routing/rt:control-plane-protocols/"
           + "rt:control-plane-protocol/rt:name";
      }
      description
        "Routing protocol reference for out-of-range indexed.";
    }
  }
}
<CODE ENDS>
```

9. Security Considerations

The YANG modules specified in this document define 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 the modules 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. These are the subtrees and data nodes and their sensitivity/vulnerability:

* /segment-routing

* /segment-routing/mpls

* /segment-routing/mpls/bindings -- Modification to the local bindings could result in a Denial-of-Service (DoS) attack. An attacker may also try to create segment conflicts (using the same segment identifier for different purposes) to redirect traffic within the trusted domain. However, the traffic will remain within the trusted domain. Redirection could be used to route the traffic to compromised nodes within the trusted domain or to avoid

certain security functions (e.g., firewall). Refer to Section 8.1 of [RFC8402] for a discussion of the SR-MPLS trusted domain.

* /segment-routing/mpls/srgb -- Modification of the Segment Routing Global Block (SRGB) could be used to mount a DoS attack. For example, if the SRGB size is reduced to a very small value, a lot of existing segments could no longer be installed leading to a traffic disruption.

* /segment-routing/mpls/srlb -- Modification of the Segment Routing Local Block (SRLB) could be used to mount a DoS attack similar to those applicable to the SRGB.

Some of the readable data nodes in these YANG modules 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. These are the subtrees and data nodes and their sensitivity/vulnerability:

* /segment-routing/mpls/bindings -- Knowledge of these data nodes can be used to attack the local router with a Denial-of-Service (DoS) attack.

* /segment-routing/mpls/sid-db -- Knowledge of these data nodes can be used to attack the other routers in the SR domain with either a Denial-of-Service (DoS) attack or redirection traffic destined for those routers.

10. IANA Considerations

This document registers a URI in the "IETF XML Registry" [RFC3688]. Following the format in [RFC3688], the following registration is requested to be made:

ID: yang:ietf-segment-routing-common
URI: urn:ietf:params:xml:ns:yang:ietf-segment-routing-common
Registrant Contact: The IESG.
XML: N/A, the requested URI is an XML namespace.

ID: yang:ietf-segment-routing
URI: urn:ietf:params:xml:ns:yang:ietf-segment-routing
Registrant Contact: The IESG.
XML: N/A, the requested URI is an XML namespace.

ID: yang:ietf-segment-routing-mpls
URI: urn:ietf:params:xml:ns:yang:ietf-segment-routing-mpls
Registrant Contact: The IESG.
XML: N/A, the requested URI is an XML namespace.

This document registers YANG modules in the "YANG Module Names" registry [RFC6020].

Name: ietf-segment-routing-common
Maintained by IANA: N
Namespace: urn:ietf:params:xml:ns:yang:ietf-segment-routing-common
Prefix: sr-cmn
Reference: RFC 9020

Name: ietf-segment-routing
Maintained by IANA: N
Namespace: urn:ietf:params:xml:ns:yang:ietf-segment-routing
Prefix: sr
Reference: RFC 9020

Name: ietf-segment-routing-mpls
Maintained by IANA: N

Namespace: urn:ietf:params:xml:ns:yang:ietf-segment-routing-mpls
Prefix: sr-mpls
Reference: RFC 9020

11. References

11.1. 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>.

[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>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/info/rfc6991>.

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/info/rfc7950>.

[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>.

[RFC8294] Liu, X., Qu, Y., Lindem, A., Hopps, C., and L. Berger, "Common YANG Data Types for the Routing Area", RFC 8294, DOI 10.17487/RFC8294, December 2017, <https://www.rfc-editor.org/info/rfc8294>.

[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>.

[RFC8343] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 8343, DOI 10.17487/RFC8343, March 2018, <https://www.rfc-editor.org/info/rfc8343>.

[RFC8349] Lhotka, L., Lindem, A., and Y. Qu, "A YANG Data Model for Routing Management (NMDA Version)", RFC 8349, DOI 10.17487/RFC8349, March 2018, <https://www.rfc-editor.org/info/rfc8349>.

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

[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>.

[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>.

[RFC8661] Bashandy, A., Ed., Filsfils, C., Ed., Previdi, S., Decraene, B., and S. Litkowski, "Segment Routing MPLS Interworking with LDP", RFC 8661, DOI 10.17487/RFC8661, December 2019, <https://www.rfc-editor.org/info/rfc8661>.

[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>.

[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>.

[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>.

[W3C.REC-xml11-20060816] Bray, T., Paoli, J., Sperberg-McQueen, M., Maler, E., Yergeau, F., and J. Cowan, "Extensible Markup Language (XML) 1.1 (Second Edition)", World Wide Web Consortium Recommendation REC-xml11-20060816, 16 August 2006, <https://www.w3.org/TR/2006/REC-xml11-20060816>.

11.2. Informative References

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <https://www.rfc-editor.org/info/rfc8340>.

[RFC8792] Watsen, K., Auerswald, E., Farrel, A., and Q. Wu, "Handling Long Lines in Content of Internet-Drafts and RFCs", RFC 8792, DOI 10.17487/RFC8792, June 2020, <https://www.rfc-editor.org/info/rfc8792>.

Appendix A. Configuration Examples

Note: '\' line wrapping per [RFC8792].

A.1. SR-MPLS with IPv4

The following is an XML [W3C.REC-xml11-20060816] example using the SR-MPLS YANG modules with IPv4 addresses.

```
<routing xmlns="urn:ietf:params:xml:ns:yang:ietf-routing">
  <segment-routing
    xmlns="urn:ietf:params:xml:ns:yang:ietf-segment-routing">
    <sr-mpls
      xmlns="urn:ietf:params:xml:ns:yang:ietf-segment-routing-mpls">
      <bindings>
        <mapping-server>
          <policy>
            <name>mapping 1</name>
            <entries>
              <mapping-entry>
                <prefix>198.51.100.0/24</prefix>
                <algorithm xmlns:sr-cmn="urn:ietf:params:xml:ns:yang\
                  :ietf-segment-routing-common">\
                  sr-cmn:prefix-sid-algorithm-shortest-path\
                </algorithm>
                <start-sid>200</start-sid>
                <range>100</range>
              </mapping-entry>
            </entries>
          </policy>
        </mapping-server>
        <connected-prefix-sid-map>
          <connected-prefix-sid>
            <prefix>192.0.2.0/24</prefix>
            <algorithm xmlns:sr-cmn="urn:ietf:params:xml:ns:yang:\
              ietf-segment-routing-common">\
              sr-cmn:prefix-sid-algorithm-strict-spf</algorithm>
            <start-sid>100</start-sid>
            <range>1</range>
            <last-hop-behavior>php</last-hop-behavior>
          </connected-prefix-sid>
        </connected-prefix-sid-map>
      </bindings>
      <srgb>
        <srgb>
          <lower-bound>45000</lower-bound>
          <upper-bound>55000</upper-bound>
        </srgb>
      </srgb>
    </sr-mpls>
  </segment-routing>
</routing>
```

The following is the same example using JSON format.

```
{
  "ietf-routing:routing": {
    "ietf-segment-routing:segment-routing": {
      "ietf-segment-routing-mpls:sr-mpls": {
        "bindings": {
          "mapping-server": {
            "policy": [
              {
                "name": "mapping 1",
                "entries": {
```

```
                "mapping-entry": [
                  {
                    "prefix": "198.51.100.0/24",
                    "algorithm": "ietf-segment-routing-common:\
                    prefix-sid-algorithm-shortest-path",
                    "start-sid": 200,
                    "range": 100
                  }
                ]
              }
            }
          ]
        },
        "connected-prefix-sid-map": {
          "connected-prefix-sid": [
            {
              "prefix": "192.0.2.0/24",
              "algorithm": "ietf-segment-routing-common:\
              prefix-sid-algorithm-strict-spf",
              "start-sid": 100,
              "range": 1,
              "last-hop-behavior": "php"
            }
          ]
        }
      },
      "srgb": {
        "srgb": [
          {
            "lower-bound": 45000,
            "upper-bound": 55000
          }
        ]
      }
    }
  }
}
```

A.2. SR-MPLS with IPv6

The following is an XML [W3C.REC-xml11-20060816] example using the SR-MPLS YANG modules with IPv6 addresses.

```
<routing xmlns="urn:ietf:params:xml:ns:yang:ietf-routing">
  <segment-routing
    xmlns="urn:ietf:params:xml:ns:yang:ietf-segment-routing">
    <sr-mpls
      xmlns="urn:ietf:params:xml:ns:yang:ietf-segment-routing-mpls">
      <bindings>
        <mapping-server>
          <policy>
            <name>mapping 1</name>
            <entries>
              <mapping-entry>
                <prefix>2001:db8:aaaa:bbbb::/64</prefix>
                <algorithm xmlns:sr-cmn="urn:ietf:params:xml:ns:yang\
                  :ietf-segment-routing-common">\
                  sr-cmn:prefix-sid-algorithm-shortest-path\
                </algorithm>
                <start-sid>200</start-sid>
                <range>100</range>
              </mapping-entry>
            </entries>
          </policy>
        </mapping-server>
```

```
        <connected-prefix-sid-map>
          <connected-prefix-sid>
            <prefix>2001:db8:aaaa:cccc::/64</prefix>
            <algorithm xmlns:sr-cmn="urn:ietf:params:xml:ns:yang:\
              ietf-segment-routing-common">\
              sr-cmn:prefix-sid-algorithm-strict-spf</algorithm>
            <start-sid>100</start-sid>
            <range>1</range>
            <last-hop-behavior>php</last-hop-behavior>
          </connected-prefix-sid>
        </connected-prefix-sid-map>
      </bindings>
      <srgb>
        <srgb>
          <lower-bound>45000</lower-bound>
          <upper-bound>55000</upper-bound>
        </srgb>
      </srgb>
    </sr-mpls>
  </segment-routing>
</routing>
```

The following is the same example using JSON format.

```
{
  "ietf-routing:routing": {
    "ietf-segment-routing:segment-routing": {
      "ietf-segment-routing-mpls:sr-mpls": {
        "bindings": {
          "mapping-server": {
            "policy": [
              {
                "name": "mapping 1",
                "entries": {
                  "mapping-entry": [
                    {
                      "prefix": "2001:db8:aaaa:bbbb::/64",
                      "algorithm": "ietf-segment-routing-common:\
                      prefix-sid-algorithm-shortest-path",
                      "start-sid": 200,
                      "range": 100
                    }
                  ]
                }
              }
            ]
          },
          "connected-prefix-sid-map": {
            "connected-prefix-sid": [
              {
                "prefix": "2001:db8:aaaa:cccc::/64",
                "algorithm": "ietf-segment-routing-common:\
                prefix-sid-algorithm-strict-spf",
                "start-sid": 100,
                "range": 1,
                "last-hop-behavior": "php"
              }
            ]
          }
        },
        "srgb": {
          "srgb": [
            {
              "lower-bound": 45000,
              "upper-bound": 55000
            }
```

```
            ]
          }
        }
      }
    }
  }
```

Acknowledgements

The authors would like to thank Derek Yeung, Greg Hankins, Hannes Gredler, Uma Chunduri, Jeffrey Zhang, Shradda Hedge, and Les Ginsberg for their contributions.

Thanks to Ladislav Lhotka and Tom Petch for their thorough reviews and helpful comments.

The authors would like to thank Benjamin Kaduk, Alvaro Retana, and Roman Danyliw for IESG review and comments.

Authors' Addresses

Stephane Litkowski
Cisco Systems

Email: slitkows.ietf@gmail.com

Yingzhen Qu
Futurewei

Email: yingzhen.qu@futurewei.com

Acee Lindem
Cisco Systems
301 Mindenhall Way
Cary, NC 27513
United States of America

Email: acee@cisco.com

Pushpasis Sarkar
VMware, Inc

Email: pushpasis.ietf@gmail.com

Jeff Tantsura
Juniper Networks

Email: jefftant.ietf@gmail.com