SPRING Working Group
Internet-Draft
Intended status: Standards Track
Expires: 20 April 2026

K. Raza
J. Rajamanickam
Cisco Systems

S. Matsushima
SoftBank

P. Yu
Huawei Technologies

X. Liu
Individual

17 October 2025

YANG Data Model for SRv6 Static
draft-ietf-spring-srv6-yang-static-00

Abstract

This document describes a YANG data model for Segment Routing IPv6 (SRv6) Static for provisioning static SIDs. The SRv6 Static model augments the SRv6 base YANG model with specific data to configure and manage SRv6 Static SID(s).

The YANG module in this document conform to the Network Management Datastore Architecture (NMDA).

Status of This Memo

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

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

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

This Internet-Draft will expire on 20 April 2026.

Copyright Notice

Table of Contents

1. Introduction

The Network Configuration Protocol (NETCONF) [RFC6241] is one of the network management protocols that defines mechanisms to manage network devices. YANG [RFC6020] is a modular language that represents data structures in an XML tree format, and is used as a data modeling language for the NETCONF.

Segment Routing (SR), as defined in [RFC8402], leverages the source routing paradigm where a node steers a packet through an ordered list of instructions, called segments. SR, thus, allows enforcing a flow through any topological path and/or service chain while maintaining per-flow state only at the ingress nodes to the SR domain. When applied to ipv6 data-plane (i.e. SRv6), SR requires a type of routing header (SRH) in an IPv6 packet that is used to encode an ordered list of IPv6 addresses (SIDs). The active segment is indicated by the Destination Address of the packet, and the next segment is indicated

by a pointer in the SRH [RFC8754]. The various functions and behaviors corresponding to network programming using SRv6 are specified in [RFC8986].

This document introduces a YANG data model for Static SRv6 that allows static (aka explicit) provisioning of SRv6 SIDs. This model uses the building blocks and constructs that are defined under SRv6 base model [I-D.ietf-spring-srv6-yang-base]. More specifically, this model represents the "static" application (as defined under "srv6-sid-owner-type" identity) allocating a local SID with an explicit value (as defined by the "Explicit" type defined under "sid-alloc-type" enum).

The static YANG module augments the ietf-srv6-base YANG module defined in the SRv6 base document.

The Static model only defines the configuration constructs that are used for managing SRv6. The Static SID’s operational state and notification are covered under SRv6 base module’s SID state and notification [I-D.ietf-spring-srv6-yang-base].

2. Specification of Requirements

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

3. YANG Model

3.1. Overview

This document defines following new YANG module:

* ietf-srv6-static: specifies management model for SRv6-static application

The modeling in this document complies with the Network Management Datastore Architecture (NMDA) defined in [RFC8342]. The operational state data is combined with the associated configuration data in the same hierarchy [RFC8407]. This module covers the operational state that is retrieved from the NMDA operational state datastore for nodes defined with "config true" (rw) in the schema.

In this document, when a simplified graphical representation of YANG model is presented in a tree diagram, the meaning of the symbols in these tree diagrams is defined in [RFC8340].

3.2. SRv6 Static

SRv6-Static application allows a user to specify SRv6 local SIDs and program them in the forwarding plane. The SRv6-Static model is captured in the ietf-srv6-static module.

The associated YANG specification for this module is captured in Section 5.1.

3.2.1. Configuration

The SRv6-Static configuration augments the SRv6-base locator tree "/rt:routing/sr:segment-routing/srv6:srv6/srv6:locators/srv6:locator"

Following are salient features of the SRv6-Static config model:

* Allows static (explicit) configuration for local-SIDs under a given locator.

* Given that entry is scoped under a locator, the key for each entry is "function" value. The function value SHOULD come from explicit block and ranges as defined in SRv6 base model [I-D.ietf-spring-srv6-yang-base]

* A user must also specify end-behavior type (End*) associated with the entry.

* A user must also specify behavior-specific data with each entry. For example, for any end behavior requiring a table lookup, a lookup-table need be provided. Similarly, for any end behavior with forwarding next-hops need to specify next-hop information. The example of former include End, End.T, End.DT4, End.DT6, and End.DT46, whereas example of later include End.X, End.DX4, End.DX6, End.B6, End.BM etc.

* Each local-SID entry has zero or more forwarding paths specified.

* A forwarding path has next-hop type that depends on the end behavior, and could be either ipv6, or ipv4, or mpls, or l2 type. For example, End.X, End.DX4, End.DX6, End.B6, End.BM, and End.DX2 will have ipv6, ipv4, ipv6, ipv6, mpls, and l2 next-hop types respectively

* For each forwarding next-hop type, the appropriate path attributes are to be specified as well. For L2 type, the only other information required is the L2 interface name. Whereas for L3 (ipv6, ipv4, mpls) types, the information includes L3 interface name, next-hop IP address, weight, and protection information.

* Depending on the end behavior type, a forwarding path may have either MPLS or SRv6 encapsulation -- i.e., Stack of out-labels or Stack of SRv6 out-SIDs. The example of former is End.BM and example of later include the rest (End.X, End.DX4, End.DX6, End.B6 etc.).

Following is a simplified graphical tree representation of the data model for SRv6 Static configuration

```
module: ietf-srv6-static
  augment /rt:routing/sr:segment-routing/srv6:srv6/srv6:locators/srv6:locator:
    +--rw static
       +--rw sids
          +--rw sid* [function]
             +--rw function             srv6-types:srv6-sid-func-value
             +--rw end-behavior-type    identityref
             +--rw end
             +--rw end_psp
             +--rw end_usp
             +--rw end_psp_usp
             +--rw end_usd
             +--rw end_psp_usd
             +--rw end_usp_usd
             +--rw end_psp_usp_usd
             +--rw end-t
             |  +--rw lookup-table-ipv6    srv6-types:table-id
             +--rw end-t_psp
             |  +--rw lookup-table-ipv6    srv6-types:table-id
             +--rw end-t_usp
             |  +--rw lookup-table-ipv6    srv6-types:table-id
             +--rw end-t_psp_usp
             |  +--rw lookup-table-ipv6    srv6-types:table-id
             +--rw end-t_usd
             |  +--rw lookup-table-ipv6    srv6-types:table-id
             +--rw end-t_psp_usd
             |  +--rw lookup-table-ipv6    srv6-types:table-id
             +--rw end-t_usp_usd
             |  +--rw lookup-table-ipv6    srv6-types:table-id
             +--rw end-t_psp_usp_usd
             |  +--rw lookup-table-ipv6    srv6-types:table-id
             +--rw end-x
             |  +--rw protected?   boolean
             |  +--rw paths
             |     +--rw path* [path-index]
             |        +--rw path-index           uint8
             |        +--rw interface?           if:interface-ref
             |        +--rw next-hop?            inet:ipv6-address
             |        +--rw table?               srv6-types:table-id
```

```
|        +--rw weight?              uint32
|        +--rw role?                enumeration
|        +--rw backup-path-index?   uint8
|        +--rw sid-list
|           +--rw out-sid* [sid]
|              +--rw sid    srv6-types:srv6-sid
+--rw end-x_psp
|  +--rw protected?   boolean
|  +--rw paths
|     +--rw path* [path-index]
|        +--rw path-index           uint8
|        +--rw interface?           if:interface-ref
|        +--rw next-hop?            inet:ipv6-address
|        +--rw table?               srv6-types:table-id
|        +--rw weight?              uint32
|        +--rw role?                enumeration
|        +--rw backup-path-index?   uint8
|        +--rw sid-list
|           +--rw out-sid* [sid]
|              +--rw sid    srv6-types:srv6-sid
+--rw end-x_usp
|  +--rw protected?   boolean
|  +--rw paths
|     +--rw path* [path-index]
|        +--rw path-index           uint8
|        +--rw interface?           if:interface-ref
|        +--rw next-hop?            inet:ipv6-address
|        +--rw table?               srv6-types:table-id
|        +--rw weight?              uint32
|        +--rw role?                enumeration
|        +--rw backup-path-index?   uint8
|        +--rw sid-list
|           +--rw out-sid* [sid]
|              +--rw sid    srv6-types:srv6-sid
+--rw end-x_psp_usp
|  +--rw protected?   boolean
|  +--rw paths
|     +--rw path* [path-index]
|        +--rw path-index           uint8
|        +--rw interface?           if:interface-ref
|        +--rw next-hop?            inet:ipv6-address
|        +--rw table?               srv6-types:table-id
|        +--rw weight?              uint32
|        +--rw role?                enumeration
|        +--rw backup-path-index?   uint8
|        +--rw sid-list
|           +--rw out-sid* [sid]
|              +--rw sid    srv6-types:srv6-sid
```

```
            +--rw end-x_usd
            |  +--rw protected?   boolean
            |  +--rw paths
            |     +--rw path* [path-index]
            |        +--rw path-index           uint8
            |        +--rw interface?           if:interface-ref
            |        +--rw next-hop?            inet:ipv6-address
            |        +--rw table?               srv6-types:table-id
            |        +--rw weight?              uint32
            |        +--rw role?                enumeration
            |        +--rw backup-path-index?   uint8
            |        +--rw sid-list
            |           +--rw out-sid* [sid]
            |              +--rw sid    srv6-types:srv6-sid
            +--rw end-x_psp_usd
            |  +--rw protected?   boolean
            |  +--rw paths
            |     +--rw path* [path-index]
            |        +--rw path-index           uint8
            |        +--rw interface?           if:interface-ref
            |        +--rw next-hop?            inet:ipv6-address
            |        +--rw table?               srv6-types:table-id
            |        +--rw weight?              uint32
            |        +--rw role?                enumeration
            |        +--rw backup-path-index?   uint8
            |        +--rw sid-list
            |           +--rw out-sid* [sid]
            |              +--rw sid    srv6-types:srv6-sid
            +--rw end-x_usp_usd
            |  +--rw protected?   boolean
            |  +--rw paths
            |     +--rw path* [path-index]
            |        +--rw path-index           uint8
            |        +--rw interface?           if:interface-ref
            |        +--rw next-hop?            inet:ipv6-address
            |        +--rw table?               srv6-types:table-id
            |        +--rw weight?              uint32
            |        +--rw role?                enumeration
            |        +--rw backup-path-index?   uint8
            |        +--rw sid-list
            |           +--rw out-sid* [sid]
            |              +--rw sid    srv6-types:srv6-sid
            +--rw end-x_psp_usp_usd
            |  +--rw protected?   boolean
            |  +--rw paths
            |     +--rw path* [path-index]
            |        +--rw path-index           uint8
            |        +--rw interface?           if:interface-ref
```

```
|        +--rw next-hop?            inet:ipv6-address
|        +--rw table?               srv6-types:table-id
|        +--rw weight?              uint32
|        +--rw role?                enumeration
|        +--rw backup-path-index?   uint8
|        +--rw sid-list
|           +--rw out-sid* [sid]
|              +--rw sid    srv6-types:srv6-sid
+--rw end-b6-encaps
|  +--rw policy-name       string
|  +--rw source-address    inet:ipv6-address
|  +--rw paths
|     +--rw path* [path-index]
|        +--rw path-index           uint8
|        +--rw interface?           if:interface-ref
|        +--rw next-hop?            inet:ipv6-address
|        +--rw table?               srv6-types:table-id
|        +--rw weight?              uint32
|        +--rw role?                enumeration
|        +--rw backup-path-index?   uint8
|        +--rw sid-list
|           +--rw out-sid* [sid]
|              +--rw sid    srv6-types:srv6-sid
+--rw end-bm
|  +--rw policy-name    string
|  +--rw paths
|     +--rw path* [path-index]
|        +--rw path-index           uint8
|        +--rw interface?           if:interface-ref
|        +--rw next-hop?            inet:ip-address
|        +--rw weight?              uint32
|        +--rw role?                enumeration
|        +--rw backup-path-index?   uint8
|        +--rw sid-list
|           +--rw out-sid* [sid]
|              +--rw sid    srv6-types:srv6-sid
+--rw end-dx6
|  +--rw paths
|     +--rw path* [path-index]
|        +--rw path-index           uint8
|        +--rw interface?           if:interface-ref
|        +--rw next-hop?            inet:ipv6-address
|        +--rw table?               srv6-types:table-id
|        +--rw weight?              uint32
|        +--rw role?                enumeration
|        +--rw backup-path-index?   uint8
|        +--rw sid-list
|           +--rw out-sid* [sid]
```

```
|              +--rw sid    srv6-types:srv6-sid
+--rw end-dx4
|  +--rw paths
|     +--rw path* [path-index]
|        +--rw path-index           uint8
|        +--rw interface?           if:interface-ref
|        +--rw next-hop?            inet:ipv4-address
|        +--rw table?               srv6-types:table-id
|        +--rw weight?              uint32
|        +--rw role?                enumeration
|        +--rw backup-path-index?   uint8
|        +--rw sid-list
|           +--rw out-sid* [sid]
|              +--rw sid    srv6-types:srv6-sid
+--rw end-dt6
|  +--rw lookup-table-ipv6    srv6-types:table-id
+--rw end-dt4
|  +--rw lookup-table-ipv4    srv6-types:table-id
+--rw end-dt46
|  +--rw lookup-table-ipv4    srv6-types:table-id
|  +--rw lookup-table-ipv6    srv6-types:table-id
+--rw end-dx2
|  +--rw path
|     +--rw l2-interface    if:interface-ref
+--rw end-dx2v
|  +--rw lookup-table-vlan    srv6-types:table-id
+--rw end-dt2u
|  +--rw lookup-table-mac    srv6-types:table-id
+--rw end-dt2m
   +--rw flooding-table    srv6-types:table-id
   +--rw paths
      +--rw path* [path-index]
         +--rw path-index      uint8
         +--rw l2-interface?   if:interface-ref
```

Figure 1: SRv6 Static - Config Tree

3.2.2. State

As per NMDA model, the state related to configuration items specified in above section Section 3.2.1 can be retrieved from the same tree. The state regarding the local-SIDs created by SRv6-static model can be obtained using the state model of SRv6-base. Hence, there is no additional state identified at this time for SRv6-static.

3.2.3. Notification

None.

4. Pending Items

Following are the items that will be addressed in next revisions:

* Align model to support SRv6 compressed SIDs

5. YANG Specification

Following are actual YANG definition for SRv6 module defined earlier in the document.

5.1. SRv6 Static

This YANG module imports types defined in [RFC6991], [RFC8343], and [RFC8349].

```
<CODE BEGINS> file "ietf-srv6-static@2025-07-07.yang"
// RFC Editor: replace the above date with the date of
// publication and remove this note.

module ietf-srv6-static {
  yang-version 1.1;

  namespace "urn:ietf:params:xml:ns:yang:ietf-srv6-static";
  prefix srv6-static;

  import ietf-interfaces {
    prefix "if";
    reference "RFC 8343: A YANG Data Model for Interface Management";
  }

  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-segment-routing {
```

```
    prefix sr;
    reference "draft-ietf-spring-sr-yang";
  }

  import ietf-srv6-types {
    prefix srv6-types;
    reference "RFC XXXX: YANG Data Model for SRv6";
    // RFC Editor: replace XXXX with actual RFC number and remove
    // this note
  }

  import ietf-srv6-base {
    prefix srv6;
    reference "RFC XXXX: YANG Data Model for SRv6";
    // RFC Editor: replace XXXX with actual RFC number and remove
    // this note
  }

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

     Editor:   Kamran Raza
               <mailto:skraza@cisco.com>

     Editor:   Jaganbabu Rajamanickam
               <maito:jrajaman@cisco.com>

     Editor:   Xufeng Liu
               <mailto:xufeng.liu.ietf@gmail.com>

     Editor:   Zhibo Hu
               <mailto:huzhibo@huawei.com>

     Editor:   Iftekhar Hussain
               <mailto:iftekhar_hussain@yahoo.com>

     Editor:   Himanshu Shah
               <mailto:hshah@ciena.com>

     Editor:   Daniel Voyer
               <mailto:daniel.voyer@bell.ca>

     Editor:   Hani Elmalky
               <mailto:helmalky@google.com>
```

```
        Editor:   Satoru Matsushima
                  <mailto:satoru.matsushima@gmail.com>

        Editor:   Katsuhiro Horiba
                  <mailto:katsuhiro.horiba@gmail.com>

        Editor:   Ahmed AbdelSalam
                  <mailto:ahabdels@cisco.com>

        Editor:   Pingping Yu
                  <mailto:susana.yu@huawei.com>

        ";

      description
        "This YANG module defines the essential elements for the
         management of Static application for Segment-Routing with
         IPv6 dataplane (SRv6).

         Copyright (c) 2018 IETF Trust and the persons identified as
         authors of the code.  All rights reserved.

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

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

         // RFC Editor: replace XXXX with actual RFC number and remove
         // this note

      revision 2025-07-07 {
        description
          "Renamed local-sid to sid";
        reference
          "RFC XXXX: YANG Data Model for SRv6 Static";
        // RFC Editor: replace XXXX with actual RFC number and remove
        // this note
      }

      revision 2024-03-04 {
        description
          "Fixed static yang warnings";
        reference
```

```
        "RFC XXXX: YANG Data Model for SRv6";
      // RFC Editor: replace XXXX with actual RFC number and remove
      // this note
    }

    revision 2022-01-14 {
      description
        "Alignment with SRv6 network programming rev16";
      reference
        "RFC XXXX: YANG Data Model for SRv6";
      // RFC Editor: replace XXXX with actual RFC number and remove
      // this note
    }

    revision 2019-10-30 {
      description
        "Extended model for EVPN behaviors";
      reference
        "RFC XXXX: YANG Data Model for SRv6";
      // RFC Editor: replace XXXX with actual RFC number and remove
      // this note
    }

    revision 2019-07-08 {
      description
        "Alignment with SRv6 network programming";
      reference
        "RFC XXXX: YANG Data Model for SRv6";
      // RFC Editor: replace XXXX with actual RFC number and remove
      // this note
    }

    revision 2018-10-22 {
      description
        "Initial revision.";
      reference
        "RFC XXXX: YANG Data Model for SRv6";
      // RFC Editor: replace XXXX with actual RFC number and remove
      // this note
    }

   /*
    * Config and State
    */


    grouping path-attrs-v6 {
       description
```

```
          "IPv6 Path properties";

        leaf interface {
           type if:interface-ref;
           description "The outgoing interface";
        }

        leaf next-hop {
          type inet:ipv6-address;
          description "The IP address of the next-hop";
        }

        leaf table {
          type srv6-types:table-id;
          description "The routing table associated with the next-hop";
        }

      uses srv6:path-attrs-cmn;
    }

    grouping path-attrs-v4 {
       description
         "IPv4 Path properties";

       leaf interface {
          type if:interface-ref;
          description "The outgoing interface";
       }

       leaf next-hop {
         type inet:ipv4-address;
         description "The IP address of the next-hop";
       }

       leaf table {
         type srv6-types:table-id;
         description "The routing table associated with the next-hop";
       }

      uses srv6:path-attrs-cmn;
    }

    grouping path-attrs-mpls {
       description
         "MPLS Path properties";

       leaf interface {
          type if:interface-ref;
```

```
            description "The outgoing interface";
          }

          leaf next-hop {
            type inet:ip-address;
            description "The IP address of the next-hop";
          }

        uses srv6:path-attrs-cmn;
      }

      grouping multi-paths-v6 {
        description "Multipath grouping";

        container paths {
          description "List of outgoing paths";
          list path {
            key path-index;
            description "The list of paths associated with the SID";

            leaf path-index {
              type uint8;
              description "Index of the path";
            }

            uses path-attrs-v6;
            container sid-list {
              description "SID-list associated with the path";
              uses srv6:path-out-sids;
            }
            }
          }
       }

      grouping multi-paths-v4 {
        description "Multipath grouping";

        container paths {
          description "List of outgoing paths";
          list path {
            key path-index;
            description "The list of paths associated with the SID";

            leaf path-index {
              type uint8;
              description "Index of the path";
            }
```

```
            uses path-attrs-v4;
            container sid-list {
              description "SID-list associated with the path";
              uses srv6:path-out-sids;
            }
           }
         }
       }

      grouping multi-paths-mpls {
        description "Multipath grouping";

        container paths {
          description "List of outgoing paths";
          list path {
            key path-index;
            description "The list of paths associated with the SID";

            leaf path-index {
              type uint8;
              description "Index of the path";
            }

            uses path-attrs-mpls;
            container sid-list {
              description "SID-list associated with the path";
              uses srv6:path-out-sids;
            }
           }
          }
        }

        grouping multi-paths-v6-BUM {
         description
           "Multipath grouping for EVPN bridging BUM use case";

         container paths {
           description
             "List of outgoing paths for flooding";
           list path {
             key path-index;
             description "The list of paths associated with the SID";

             leaf path-index {
               type uint8;
               description "Index of the path";
             }
```

```
            leaf l2-interface {
              type if:interface-ref;
              description "The outgoing L2 interface for flooding";
            }
          }
        }
      }

      grouping srv6-sid-config {
        description
          "Configuration parameters relating to SRv6 sid.";

        leaf function {
          type srv6-types:srv6-sid-func-value;
          description
            "SRv6 function value.";
        }
        leaf end-behavior-type {
          type identityref {
            base srv6-types:srv6-endpoint-type;
          }
          mandatory true;
          description
            "Type of SRv6 end behavior.";
        }

        container end {
          when "../end-behavior-type = 'srv6-types:End'" {
            description
              "This container is valid only when the user chooses End
               behavior (variant: no PSP, no USP).";
          }
          description
            "The Endpoint function is the most basic function.
             FIB lookup on updated DA and forward accordingly
             to the matched entry.
             This is the SRv6 instantiation of a Prefix SID
             (variant: no PSP, no USP)";

        }

        container end_psp {
          when "../end-behavior-type = 'srv6-types:End_PSP'" {
            description
              "This container is valid only when the user chooses End
               behavior (variant: PSP only).";
          }
          description
```

```
          "The Endpoint function is the most basic function.
           FIB lookup on updated DA and forward accordingly
           to the matched entry.
           This is the SRv6 instantiation of a Prefix SID
           (variant: PSP only)";

      }

      container end_usp {
        when "../end-behavior-type = 'srv6-types:End_USP'" {
          description
            "This container is valid only when the user chooses End
             behavior (variant: USP only).";
        }
        description
          "The Endpoint function is the most basic function.
           FIB lookup on updated DA and forward accordingly
           to the matched entry.
           This is the SRv6 instantiation of a Prefix SID
           (variant: USP only)";

      }

      container end_psp_usp {
        when "../end-behavior-type = 'srv6-types:End_PSP_USP'" {
          description
            "This container is valid only when the user chooses End
             behavior (variant: PSP/USP).";
        }
        description
          "The Endpoint function is the most basic function.
           FIB lookup on updated DA and forward accordingly
           to the matched entry.
           This is the SRv6 instantiation of a Prefix SID
           (variant: PSP/USP)";

      }

      container end_usd {
        when "../end-behavior-type = 'srv6-types:End_USD'" {
          description
            "This container is valid only when the user chooses End
             behavior (variant: USD only).";
        }
        description
          "The Endpoint function is the most basic function.
           FIB lookup on updated DA and forward accordingly
           to the matched entry.
```

```
            This is the SRv6 instantiation of a Prefix SID
            (variant: USD)";
        }

        container end_psp_usd {
          when "../end-behavior-type = 'srv6-types:End_PSP_USD'" {
            description
              "This container is valid only when the user chooses End
               behavior (variant: PSP/USD).";
          }
          description
            "The Endpoint function is the most basic function.
             FIB lookup on updated DA and forward accordingly
             to the matched entry.
             This is the SRv6 instantiation of a Prefix SID
             (variant: PSP/USD)";

        }

        container end_usp_usd {
          when "../end-behavior-type = 'srv6-types:End_USP_USD'" {
            description
              "This container is valid only when the user chooses End
               behavior (variant: USP/USD).";
          }
          description
            "The Endpoint function is the most basic function.
             FIB lookup on updated DA and forward accordingly
             to the matched entry.
             This is the SRv6 instantiation of a Prefix SID
             (variant: USP/USD)";

        }

        container end_psp_usp_usd {
          when "../end-behavior-type = 'srv6-types:End_PSP_USP_USD'" {
            description
              "This container is valid only when the user chooses End
               behavior (variant: PSP/USP/USD).";
          }
          description
            "The Endpoint function is the most basic function.
             FIB lookup on updated DA and forward accordingly
             to the matched entry.
             This is the SRv6 instantiation of a Prefix SID
             (variant: PSP/USP/USD)";

        }
```

```
        container end-t {
          when "../end-behavior-type = 'srv6-types:End.T'" {
            description
              "This container is valid only when the user chooses
               End.T behavior (variant: no PSP, no USP).";
          }
          description
            "Endpoint with specific IPv6 table lookup (variant: no PSP,
            no USP).
             Lookup the next segment in IPv6 table T
             associated with the SID and forward via
             the matched table entry.
             The End.T is used for multi-table operation
             in the core.";

            // TODO presence "Mandatory child only if container is present";
            leaf lookup-table-ipv6 {
              type srv6-types:table-id;
              mandatory true;
              description
                "Table Id for lookup on updated DA (next segment)";
            }
        }

        container end-t_psp {
          when "../end-behavior-type = 'srv6-types:End.T_PSP'" {
            description
              "This container is valid only when the user chooses
               End.T behavior (variant: PSP only).";
          }
          description
            "Endpoint with specific IPv6 table lookup (variant: PSP only).
             Lookup the next segment in IPv6 table T
             associated with the SID and forward via
             the matched table entry.

             The End.T is used for multi-table operation
             in the core.";

            // TODO presence "Mandatory child only if container is present";

            leaf lookup-table-ipv6 {
              type srv6-types:table-id;
              mandatory true;
              description
                "Table Id for lookup on updated DA (next segment)";
            }
        }
```

```
      container end-t_usp {
        when "../end-behavior-type = 'srv6-types:End.T_USP'" {
          description
            "This container is valid only when the user chooses
             End.T behavior (variant: USP only).";
        }
        description
          "Endpoint with specific IPv6 table lookup (variant: USP only).
           Lookup the next segment in IPv6 table T
           associated with the SID and forward via
           the matched table entry.
           The End.T is used for multi-table operation
           in the core.";

          // TODO presence "Mandatory child only if container is present";

          leaf lookup-table-ipv6 {
            type srv6-types:table-id;
            mandatory true;
            description
              "Table Id for lookup on updated DA (next segment)";
          }
      }

      container end-t_psp_usp {
        when "../end-behavior-type = 'srv6-types:End.T_PSP_USP'" {
          description
            "This container is valid only when the user chooses
             End.T behavior (variant: USP/PSP).";
        }
        description
          "Endpoint with specific IPv6 table lookup (variant: USP/PSP).
           Lookup the next segment in IPv6 table T
           associated with the SID and forward via
           the matched table entry.
           The End.T is used for multi-table operation
           in the core.";

          // TODO presence "Mandatory child only if container is present";

          leaf lookup-table-ipv6 {
            type srv6-types:table-id;
            mandatory true;
            description
              "Table Id for lookup on updated DA (next segment)";
          }
      }
```

```
      container end-t_usd {
        when "../end-behavior-type = 'srv6-types:End.T_USD'" {
          description
            "This container is valid only when the user chooses
             End.T behavior (variant: USD only).";
        }
        description
          "Endpoint with specific IPv6 table lookup (variant: USD only).
           Lookup the next segment in IPv6 table T
           associated with the SID and forward via
           the matched table entry.
           The End.T is used for multi-table operation
           in the core.";

          // TODO presence "Mandatory child only if container is present";

          leaf lookup-table-ipv6 {
            type srv6-types:table-id;
            mandatory true;
            description
              "Table Id for lookup on updated DA (next segment)";
          }
      }

      container end-t_psp_usd {
        when "../end-behavior-type = 'srv6-types:End.T_PSP_USD'" {
          description
            "This container is valid only when the user chooses
             End.T behavior (variant: PSP/USD only).";
        }
        description
          "Endpoint with specific IPv6 table lookup (variant: PSP/USD
          only).
           Lookup the next segment in IPv6 table T
           associated with the SID and forward via
           the matched table entry.
           The End.T is used for multi-table operation
           in the core.";

          // TODO presence "Mandatory child only if container is present";

          leaf lookup-table-ipv6 {
            type srv6-types:table-id;
            mandatory true;
            description
              "Table Id for lookup on updated DA (next segment)";
          }
      }
```

```
      container end-t_usp_usd {
        when "../end-behavior-type = 'srv6-types:End.T_USP_USD'" {
          description
            "This container is valid only when the user chooses
             End.T behavior (variant: USP/USD only).";
        }
        description
          "Endpoint with specific IPv6 table lookup (variant:
          USP/USD only).
           Lookup the next segment in IPv6 table T
           associated with the SID and forward via
           the matched table entry.
           The End.T is used for multi-table operation
           in the core.";

          // TODO presence "Mandatory child only if container is present";

          leaf lookup-table-ipv6 {
            type srv6-types:table-id;
            mandatory true;
            description
              "Table Id for lookup on updated DA (next segment)";
          }
      }

      container end-t_psp_usp_usd {
        when "../end-behavior-type = 'srv6-types:End.T_PSP_USP_USD'" {
          description
            "This container is valid only when the user chooses
             End.T behavior (variant: USP only).";
        }
        description
          "Endpoint with specific IPv6 table lookup (variant:
          PSP/USP/USD only).
           Lookup the next segment in IPv6 table T
           associated with the SID and forward via
           the matched table entry.
           The End.T is used for multi-table operation
           in the core.";

          // TODO presence "Mandatory child only if container is present";

          leaf lookup-table-ipv6 {
            type srv6-types:table-id;
            mandatory true;
            description
              "Table Id for lookup on updated DA (next segment)";
          }
```

```
       }

      container end-x {
         when "../end-behavior-type = 'srv6-types:End.X'" {
           description
             "This container is valid only when the user chooses
              End.X behavior (variant: no USP/PSP)";
         }
         description
           "Endpoint with cross-connect to an array of
            layer-3 adjacencies (variant: no USP/PSP).
            Forward to layer-3 adjacency bound to the SID S.
            The End.X function is required to express any
            traffic-engineering policy.";

         leaf protected {
           type boolean;
           default false;
           description "Is Adj-SID protected?";
         }

         uses multi-paths-v6;
       }

      container end-x_psp {
         when "../end-behavior-type = 'srv6-types:End.X_PSP'" {
           description
             "This container is valid only when the user chooses
              End.X behavior (variant: PSP only)";
         }
         description
           "Endpoint with cross-connect to an array of
            layer-3 adjacencies (variant: PSP only).
            Forward to layer-3 adjacency bound to the SID S.
            The End.X function is required to express any
            traffic-engineering policy.";

         leaf protected {
           type boolean;
           default false;
           description "Is Adj-SID protected?";
         }

         uses multi-paths-v6;
       }

      container end-x_usp {
         when "../end-behavior-type = 'srv6-types:End.X_USP'" {
```

```
            description
              "This container is valid only when the user chooses
               End.X behavior (variant: USP only)";
          }
          description
            "Endpoint with cross-connect to an array of
             layer-3 adjacencies (variant: USP only).
             Forward to layer-3 adjacency bound to the SID S.
             The End.X function is required to express any
             traffic-engineering policy.";

          leaf protected {
            type boolean;
            default false;
            description "Is Adj-SID protected?";
          }

          uses multi-paths-v6;
        }

      container end-x_psp_usp {
         when "../end-behavior-type = 'srv6-types:End.X_PSP_USP'" {
           description
             "This container is valid only when the user chooses
              End.X behavior (variant: PSP/USP)";
         }
         description
           "Endpoint with cross-connect to an array of
            layer-3 adjacencies (variant: PSP/USP).
            Forward to layer-3 adjacency bound to the SID S.
            The End.X function is required to express any
            traffic-engineering policy.";

         leaf protected {
           type boolean;
           default false;
           description "Is Adj-SID protected?";
         }

         uses multi-paths-v6;
        }


      container end-x_usd {
         when "../end-behavior-type = 'srv6-types:End.X_USD'" {
           description
             "This container is valid only when the user chooses
              End.X behavior (variant: USD only)";
```

```
          }
          description
            "Endpoint with cross-connect to an array of
             layer-3 adjacencies (variant: PSP/USP).
             Forward to layer-3 adjacency bound to the SID S.
             The End.X function is required to express any
             traffic-engineering policy.";

          leaf protected {
            type boolean;
            default false;
            description "Is Adj-SID protected?";
          }

          uses multi-paths-v6;
        }

      container end-x_psp_usd {
         when "../end-behavior-type = 'srv6-types:End.X_PSP_USD'" {
           description
             "This container is valid only when the user chooses
              End.X behavior (variant: PSP/USD only)";
         }
         description
           "Endpoint with cross-connect to an array of
            layer-3 adjacencies (variant: PSP/USP).
            Forward to layer-3 adjacency bound to the SID S.
            The End.X function is required to express any
            traffic-engineering policy.";

         leaf protected {
           type boolean;
           default false;
           description "Is Adj-SID protected?";
         }

         uses multi-paths-v6;
        }

      container end-x_usp_usd {
         when "../end-behavior-type = 'srv6-types:End.X_USP_USD'" {
           description
             "This container is valid only when the user chooses
              End.X behavior (variant: USP/USD only)";
         }
         description
           "Endpoint with cross-connect to an array of
            layer-3 adjacencies (variant: PSP/USP).
```

```
            Forward to layer-3 adjacency bound to the SID S.
            The End.X function is required to express any
            traffic-engineering policy.";

          leaf protected {
            type boolean;
            default false;
            description "Is Adj-SID protected?";
          }

          uses multi-paths-v6;
        }

       container end-x_psp_usp_usd {
          when "../end-behavior-type = 'srv6-types:End.X_PSP_USP_USD'" {
            description
              "This container is valid only when the user chooses
               End.X behavior (variant: PSP/USP/USD only)";
          }
          description
            "Endpoint with cross-connect to an array of
             layer-3 adjacencies (variant: PSP/USP).
             Forward to layer-3 adjacency bound to the SID S.
             The End.X function is required to express any
             traffic-engineering policy.";

          leaf protected {
            type boolean;
            default false;
            description "Is Adj-SID protected?";
          }

          uses multi-paths-v6;
        }

        container end-b6-encaps {
          when "../end-behavior-type = 'srv6-types:End.B6.Encaps' or
                ../end-behavior-type = 'srv6-types:End.B6.Encaps.Red'" {
            description
              "This container is valid only when the user chooses
               End.B6.Encaps or End.B6.Encaps.Red behavior.";
          }
          description
             "Endpoint bound to an SRv6 Policy.
             Insert SRH based on the policy and forward the
             packet toward the first hop configured in the policy.
             This is the SRv6 instantiation of a Binding SID.
             This behavior also adds an outer IPv6 header";
```

```
          // TODO presence  "Mandatory child only if container is present";

          leaf policy-name {
            type string;
            mandatory true;
            description "SRv6 policy name.";
          }
          leaf source-address {
            type inet:ipv6-address;
            mandatory true;
            description
              "IPv6 source address for Encap.";
          }

          uses multi-paths-v6;
        }

        container end-bm {
          when "../end-behavior-type = 'srv6-types:End.BM'" {
            description
              "This container is valid only when the user chooses
               End.BM behavior.";
          }

          description
            "Endpoint bound to an SR-MPLS Policy.
             push an MPLS label stack <L1, L2, L3> on the
             received packet and forward the according to
             Lable L1.
             This is an SRv6 instantiation of an SR-MPLS Binding SID.";

          // TODO presence  "Mandatory child only if container is present";

          leaf policy-name {
            type string;
            mandatory true;
            description "SRv6 policy name";
          }
          uses multi-paths-mpls;
        }

        container end-dx6 {
          when "../end-behavior-type = 'srv6-types:End.DX6'" {
            description
              "This container is valid only when the user chooses
               End.DX6 behavior.";
          }
          description
```

```
            "Endpoint with decapsulation and cross-connect to
             an array of IPv6 adjacencies. Pop the (outer)
             IPv6 header and its extension headers and forward
             to layer-3 adjacency bound to the SID S.
             The End.DX6 used in the L3VPN use-case.";

          uses multi-paths-v6;
          // TODO: Backup path of type "Lookup in table"
        }

        container end-dx4 {
          when "../end-behavior-type = 'srv6-types:End.DX4'" {
            description
              "This container is valid only when the user chooses
               End.DX4 behavior.";
          }
          description
            "Endpoint with decapsulation and cross-connect to
             an array of IPv4 adjacencies.
             Pop the (outer) IPv6 header and its extension
             header and forward to layer-3 adjacency bound
             to the SID S.
             This would be equivalent to the per-CE VPN
             label in MPLS.";

          uses multi-paths-v4;
          // TODO: Backup path of type "Lookup in table"
        }
        container end-dt6 {
          when "../end-behavior-type = 'srv6-types:End.DT6'" {
            description
              "This container is valid only when the user chooses
               End.DT6 behavior.";
          }
          description
            "Endpoint with decapsulation and specific IPv6 table
             lookup.
             Pop the (outer) IPv6 header and its extension
             headers.
             Lookup the exposed inner IPv6 DA in IPv6
             table T and forward via the matched table entry.
             End.DT6 function is used in L3VPN use-case.";

          // TODO presence  "Mandatory child only if container is present";

          leaf lookup-table-ipv6 {
            type srv6-types:table-id;
            mandatory true;
```

```
          description "IPv6 table";
        }
      }
      container end-dt4 {
        when "../end-behavior-type = 'srv6-types:End.DT4'" {
          description
            "This container is valid only when the user chooses
             End.DT4 behavior.";
        }
        description
          "Endpoint with decapsulation and specific
           IPv4 table lookup.
           Pop the (outer) IPv6 header and its extension
           headers.
           Lookup the exposed inner IPv4 DA in IPv4
           table T and forward via the matched table entry.
           This would be equivalent to the per-VRF VPN label
           in MPLS.";

        // TODO presence "Mandatory child only if container is present";

        leaf lookup-table-ipv4 {
          type srv6-types:table-id;
          mandatory true;
          description "IPv4 table";
        }
      }
      container end-dt46 {
        when "../end-behavior-type = 'srv6-types:End.DT46'" {
          description
            "This container is valid only when the user chooses
             End.DT46 behavior.";
        }
        description
          "Endpoint with decapsulation and specific
           IP table lookup.
           Depending on the protocol type (IPv4 or IPv6)
           of the inner ip packet and the specific VRF name
           forward the packet.
           This would be equivalent to the per-VRF VPN
           label in MPLS.";

        // TODO presence  "Mandatory child only if container is present";

        leaf lookup-table-ipv4 {
          type srv6-types:table-id;
          mandatory true;
          description "IPv4 table";
```

```
          }
          leaf lookup-table-ipv6 {
            type srv6-types:table-id;
            mandatory true;
            description "IPv6 table";
          }
        }

        /* EVPN END behavior types */
        container end-dx2 {
          when "../end-behavior-type = 'srv6-types:End.DX2'" {
            description
              "This container is valid only when the user chooses
               End.DX2 behavior.";
          }
          description
            "This is an Endpoint with decapsulation and Layer-2
             cross-connect to OIF.
             Pop the (outer) IPv6 header and its extension headers.
             Forward the resulting frame via OIF associated to the SID.
             The End.DX2 function is the L2VPN/EVPN VPWS use-case.";

            container path {
              description "Outgoing path";
              leaf l2-interface {
                type if:interface-ref;
                mandatory true;
                description "Outgoing L2 interface";
              }
            }
        }

        container end-dx2v {
          when "../end-behavior-type = 'srv6-types:End.DX2V'" {
            description
              "This container is valid only when the user chooses
               End.DX2V behavior.";
          }
          description
            "Endpoint with decapsulation and specific VLAN
             L2 table lookup.
             Pop the (outer) IPv6 header and its extension headers.
             Lookup the exposed inner VLANs in L2 table T.
             Forward via the matched table entry.
             The End.DX2V is used for EVPN Flexible cross-connect
             use-cases";

          leaf lookup-table-vlan {
```

```
            type srv6-types:table-id;
            mandatory true;
            description
              "VLAN lookup table. There could be multiple
               vlan demux tables on the node, where a DX2V SID
               points to one vlan table";
          }
        }

        container end-dt2u {
          when "../end-behavior-type = 'srv6-types:End.DT2U'" {
            description
              "This container is valid only when the user chooses
               End.DT2U behavior.";
          }
          description
            "Endpoint with decapsulation and specific
             unicast L2 MAC table lookup.
             Pop the (outer) IPv6 header and its extension headers.
             Learn the exposed inner MAC SA in L2 MAC table T.
             Lookup the exposed inner MAC DA in L2 MAC table T.
             Forward via the matched T entry else to all L2OIF in T.
             The End.DT2U is used for EVPN Bridging unicast use cases";

          leaf lookup-table-mac {
            type srv6-types:table-id;
            mandatory true;
            description "MAC L2 lookup table";
          }
        }

        container end-dt2m {
          when "../end-behavior-type = 'srv6-types:End.DT2M'" {
            description
              "This container is valid only when the user chooses
               End.DT2M behavior.";
          }
          description
            "Endpoint with decapsulation and specific flooding table.
             Pop the (outer) IPv6 header and its extension headers.
             Learn the exposed inner MAC SA in L2 MAC table T.
             Forward on all L2OIF (in the flooding table) excluding the one
             identfied by Arg.FE2.
             The End.DT2M is used for EVPN Bridging BUM use case with
             ESI (Split Horizon) filtering capability.";

          leaf flooding-table {
            type srv6-types:table-id;
```

```
            mandatory true;
            description "L2 Flooding table (list of OIFs)";
          }

          uses multi-paths-v6-BUM;

          /* TODO - Support for argument Arg.FE2. It is an argument specific
             to EVPN ESI filtering and EVPN-ETREE used to exclude specific
             OIF (or set of OIFs) from flooding table. */
        }

        /* End of EVPN END behavior types */
      }

      grouping srv6-static-sid-cfg {
        description
          "Grouping configuration and operation for SRv6 sid.";

          list sid {
            key "function";
            description "List of locally instantiated SIDs";

            uses srv6-sid-config;
          }
      }

     augment "/rt:routing/sr:segment-routing/srv6:srv6/srv6:locators/srv6:locator" {
         description
          "This augments locator leaf within SRv6.";

       container static {
        description "Static SRv6";

        /* Local/My SIDs */
        container sids {
          description
               "Locally instantiated explicit SRv6 SIDs";

          uses srv6-static-sid-cfg;
          /* no state for now; SID state accessible through base model */
       }

      }
     }
    } // module
    <CODE ENDS>
```

Figure 2: ietf-srv6-static.yang

6. Security Considerations

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8446].

The Network Configuration Access Control Model (NACM) [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

The data nodes defined in this YANG module 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.

It goes without saying that this specification also inherits the security considerations captured in the SRv6 specification document [RFC8986].

7. IANA Considerations

This document requests the registration of the following URIs in the IETF "XML registry" [RFC3688]:

URI	Registrant	XML
urn:ietf:params:xml:ns:yang:ietf-srv6-static	The IESG	N/A

Table 1

This document requests the registration of the following YANG modules in the "YANG Module Names" registry [RFC6020]:

Name	Namespace	Prefix	Reference
ietf-srv6-static	urn:ietf:params:xml:ns:yang:ietf-srv6-static	srv6-static	This document

Table 2

-- RFC Editor: Replace "This document" with the document RFC number at time of publication, and remove this note.

8. References

8.1. Normative References

[I-D.ietf-spring-srv6-yang-base]
Raza, S. K., Rajamanickam, J., Matsushima, S., Yu, P., and X. Liu, "YANG Data Model for SRv6 Base", Work in Progress, Internet-Draft, draft-ietf-spring-srv6-yang-base-00, 17 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-spring-srv6-yang-base-00>.

[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>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

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

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

[RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <https://www.rfc-editor.org/info/rfc8341>.

[RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <https://www.rfc-editor.org/info/rfc8342>.

[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>.

[RFC8407] Bierman, A., "Guidelines for Authors and Reviewers of Documents Containing YANG Data Models", BCP 216, RFC 8407, DOI 10.17487/RFC8407, October 2018, <https://www.rfc-editor.org/info/rfc8407>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC8986] Filsfils, C., Ed., Camarillo, P., Ed., Leddy, J., Voyer, D., Matsushima, S., and Z. Li, "Segment Routing over IPv6 (SRv6) Network Programming", RFC 8986, DOI 10.17487/RFC8986, February 2021, <https://www.rfc-editor.org/info/rfc8986>.

8.2. Informative References

[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>.

Appendix A. Acknowledgments

The authors would like to acknowledge Darren Dukes, Les Ginsberg, Kris Michielson for their review of some of the contents in this draft.

Appendix B. Contributors

Zhibo Hu
Huawei Technologies
Email: huzhibo@huawei.com

Sonal Agarwal
Individual

Katsuhiro Horiba
Individual
Email: katsuhiro.horiba@gmail.com>

Himanshu Shah
Ciena Corporation
Email: hshah@ciena.com

Iftekhar Hussain
Individual
Email: iftekhar_hussain@yahoo.com

Ahmed AbdelSalam
Cisco Systems
Email: ahabdels@cisco.com

Daniel Voyer
Individual
Email: danvoyerwork@gmail.com

Hani Elmalky
Individual
Email: helmalky@google.com

Authors' Addresses

Kamran Raza
Cisco Systems
Email: skraza@cisco.com

Jaganbabu Rajamanickam
Cisco Systems
Email: jrajaman@cisco.com

Satoru Matsushima
SoftBank
Email: satoru.matsushima@g.softbank.co.jp

Pingping Yu
Huawei Technologies
Email: susana.yu@huawei.com

Xufeng Liu
Individual
Email: xufeng.liu.ietf@gmail.com