

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 28, 2018

Z. Hu
Z. Li
Huawei
S. Matsushima
K. Horiba
Softbank
October 25, 2017

YANG Data Model for SRv6
draft-hu-spring-srv6-yang-00

Abstract

This document defines a YANG data model that can be used to configure and manage SRv6

Requirements Language

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

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

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

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

This Internet-Draft will expire on April 28, 2018.

Copyright Notice

Copyright (c) 2017 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

Internet-Draft

SRv6 YANG Data Model

October 2017

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

Table of Contents

1.	Introduction	2
2.	Terminology	2
3.	Design of Data Model	3
3.1.	Overview	3
3.2.	SRv6 Configuration	4
3.3.	SRv6 Explicit-path Configuration	6
4.	SRv6 VPN YANG	7
5.	Yang Module	7
6.	IANA Considerations	13
7.	Security Considerations	13
8.	Acknowledgements	13
9.	References	13
9.1.	Normative References	13
9.2.	Informative References	15
	Authors' Addresses	15

[1.](#) Introduction

YANG[RFC6020] is a data definition language that was introduced to define the contents of a conceptual data store that allows networked devices to be managed using NETCONF[RFC6241]. YANG is proving relevant beyond its initial confines, as bindings to other interfaces(e.g. REST) and encoding other than XML (e.g. JSON) are being defined. Furthermore, YANG data models can be used as the basis of implementation for other interface, such as CLI and Programmatic APIs.

[2.](#) Terminology

- o SR: Segment Routing
- o SID: Segment ID
- o SRv6: Segment Routing IPv6 data plane

- o SRH:Segment Routing Header

[3.](#) Design of Data Model

[3.1.](#) Overview

This module augments the `"/rt:routing:"` with a `segment-routing-ipv6` container. This container defines the configuration and operation state parameters related to `segment-routing-ipv6`.

The YANG module includes :

- o `segment-routing-ipv6-global-enable`:Enable global SRv6 capability.
- o `segment-routing-ipv6-traffic-engineer-enable`:Enable SRv6 traffic engineering capability.
- o `srv6-node-capability`:the ability to encapsulate or handle SRv6 headers.
- o `my-local-sid-table`:this table contains all the local SRv6 segments explicitly instantiated at the node.

Internet-Draft

SRv6 YANG Data Model

October 2017

```
module: ietf-segment-routing-ipv6
augment /rt:routing:
  +--rw segment-routing-ipv6
    +--rw srv6-enable?          boolean
    +--rw traffic-engineering-enable?  boolean
    +--rw srv6-node-capabilities
      | +--rw encap-capability?      boolean
      | +--rw maximum-sl?            uint8
      | +--rw maximum-end-pop-srh?   uint8
      | +--rw maximum-insert-srh?   uint8
      | +--rw maximum-encap-srh?     uint8
      | +--rw maximum-end-d-srh?     uint8
    +--rw myLocalSidTable
      | +--ro srhEndSidForwTables
      | | +--ro srhEndSidForwTable* [endSidValue]
      | | | +--ro endSidValue      inet:ipv6-address-no-zone
      | | | +--ro endSidFlavor?    string
      | +--ro srhEndXSidForwTables
      | | +--ro srhEndXSidForwTable* [endXSidValue]
      | | | +--ro endXSidValue      inet:ipv6-address-no-zone
      | | | +--ro endXSidFlavor?    string
      | | | +--ro endXSidNhpInfos
      | | | | +--ro endXSidNhpInfo* [interface nextHop]
      | | | | +--ro interface      string
      | | | | +--ro nextHop        inet:ipv6-address-no-zone
    +--rw srv6-sid-list
      +--rw explicitPath* [explicitPathName]
        +--rw explicitPathName  string
        +--rw explicitPathHops
```

```

+--rw explicitPathHop* [tunnelHopIndex]
  +--rw tunnelHopIndex                uint32
  +--rw tunnelHopMode?                 erHopMode
  +--rw tunnelHopSidIPv6               inet:ipv6-address-no-zone
  +--rw tunnelHopSidIPv6Type?          erSidIPv6Type

```

[3.2.](#) SRv6 Configuration

The global SRv6 configuration includes the following :

1.The underlay transport-type is defined SRH. The SPRING architecture leverages the existing MPLS data plane without any modification, and it also leverages the IPv6 data plane with a new IPv6 Routing Header Type with a proposal for a new type of routing header is made by [SRH].

2.Enable the capability of SRv6 traffic engineering.All the SPRING use cases [[RFC7855](#)] are also applicable to the SRv6 data plane including traffic engineering.

3.SRV6-node-capability:the ability to encapsulate or handle SRv6 headers.

encap-capability:the SRv6 encap capability of the node.

maximum-sl:the maximum value of the "SL" field in the SRH supported by the node.

maximum-end-pop-srh:the maximum number of SIDs in the top SRH in an SRH stack to which the node can apply "PSP" or USP" flavors.

maximum-insert-srh:the maximum number of SIDs that can be inserted as part of the "T.insert" behavior supported by the node.

maximum-encap-srh:the maximum number of SIDs that can be included as part of the "T.Encap" behavior supported by the node.

maximum-end-d-srh:the maximum number of SIDs in an SRH when applying

"End.DX6" and "End.DT6" functions supported by the node.

4:my-local-sid-table:this table contains all the local SRv6 segments explicitly instantiated at the node.

SRv6EndSidTables:Segment Routing IPv6 End Local-SID table.

SRV6EndXSidTables:Segment Routing IPv6 End.X Local-SID table.

the other SID will be added in a later revision.

```
module: ietf-segment-routing-ipv6
  augment /rt:routing:
    +--rw segment-routing-ipv6
      +--rw srv6-enable?                boolean
      +--rw traffic-engineer-enable?    boolean
      +--rw srv6-node-capabilities
        | +--rw encap-capability?       boolean
        | +--rw maximum-sl?             uint8
        | +--rw maximum-end-pop-srh?    uint8
        | +--rw maximum-insert-srh?    uint8
        | +--rw maximum-encap-srh?     uint8
        | +--rw maximum-end-d-srh?     uint8
      +--rw myLocalSidTable
        | +--ro srhEndSidForwTables
        | | +--ro srhEndSidForwTable* [endSidValue]
```

```

| |      +---ro endSidValue      inet:ipv6-address-no-zone
| |      +---ro endSidFlavor?    string
| +---ro srhEndXSidForwTables
| |      +---ro srhEndXSidForwTable* [endXSidValue]
| |      +---ro endXSidValue      inet:ipv6-address-no-zone
| |      +---ro endXSidFlavor?    string
| |      +---ro endXSidNhpInfos
| |      +---ro endXSidNhpInfo* [interface nextHop]
| |      +---ro interface        string
| |      +---ro nextHop          inet:ipv6-address-no-zone

```

3.3. SRv6 Explicit-path Configuration

The SRv6 explicit path configuration includes the following:

SRv6 tunnel explicit path, the SRv6 explicit path can be expressed by a set of IPv6 address or SRv6 sid or Mixed with both. It contains the following parameters:

- o explicitPathName: The path name of explicit path.
- o tunnelHopIndex: The path index of explicit path.
- o HopMode: Route hop mode, specifies the IPv6 address or the SRv6 SID.
- o tunnelHopSidIPv6 : IPv6 address or SRv6 SID.
- o tunnelHopSidIpv6Type: SRv6 SID type of the explicit route hop.

```

+---rw srv6-sid-list
|   +---rw explicitPath* [explicitPathName]
|   |   +---rw explicitPathName    string
|   |   +---rw explicitPathHops
|   |   |   +---rw explicitPathHop* [tunnelHopIndex]
|   |   |   |   +---rw tunnelHopIndex        uint32
|   |   |   |   +---rw tunnelHopMode?        erHopMode
|   |   |   |   +---rw tunnelHopSidIPv6      inet:ipv6-address-no-zone

```

4. SRv6 VPN YANG

The SRv6 VPN YANG module will be added in a later revision.

5. Yang Module

```
<CODE BEGINS> file "ietf-segment-routing-ipv6@2017-10-23.yang"
module ietf-segment-routing-ipv6 {
  namespace "urn:ietf:params:xml:ns:yang:ietf-segment-routing-ipv6";
  prefix sr;

  import ietf-inet-types {
    prefix inet;
  }
  import ietf-routing {
    prefix rt;
  }

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

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

    Author:   Zhenbin li
             <lizhenbin@huawei.com>

    Author:   S. Matsushima
             <satoru.matsushima@g.softbank.co.jp>

  ";
  description
    "The YANG module defines a generic configuration model for
    Segment routing IPv6 data plane."
```


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."

```
reference "RFC XXXX";

revision 2017-10-23 {
  description
    "
      * Implement NMDA model
      *Conform to RFC6087BIS Appendix C
    ";
  reference "RFC XXXX: YANG Data Model for Segment Routing IPv6.";
}

typedef erHopMode {
  type enumeration {
    enum "IPv6_ADDRESS" {
      description
        "IPv6 address.";
    }
    enum "SRv6 Sid" {
      description
        "SRv6 SID.";
    }
  }
  description
    "TE Hop Address Type";
}

typedef erSidIpv6Type {
  type enumeration {
    enum "none" {
      description
        "None.";
    }
    enum "adjacency" {
      description
        "Adjacency SID.";
    }
  }
}
```

```
        enum "prefix" {
            description
                "Prefix SID.";
        }
    }
    description
        "TE Hop Address Type";
}

augment "/rt:routing" {
    description
        "This augments routing-instance
        configuration with segment-routing-ipv6.";
    container segment-routing-ipv6 {
        description
            "segment routing IPV6 global config.";
        leaf srv6-enable {
            type boolean;
            default "false";
            description
                "Enables segment-routing
                IPV6 dataplane.";
        }
        leaf traffic-engineering-enable {
            type boolean;
            default "false";
            description
                "Enables segment-routing
                IPV6 traffic-engineering.";
        }
        container SRV6-node-capabilities {
            description
                "Containing SRv6 node capabilities.";
            leaf encap-capability {
                type boolean;
                default true;
                description
                    "The SRv6 T.Encap capability of Node.";
            }
            leaf maximum-sl {
                type uint8;
                description
                    "Node maximum-sl is the maximum value of the 'SL' field
                    in the SRH supported by the node.";
            }
            leaf maximum-end-pop-srh {
```

```
type uint8;
description
```

```
        "Node maximum-end-pop-srh is the maximum number of SIDs
        in the top SRH in an SRH stack to which the node can
        apply 'PSP' or 'USP' flavors.";
    }
    leaf maximum-insert-srh {
        type uint8;
        description
            "Node maximum-insert-srh is the maximum number of SIDs
            that can be inserted as part of the 'T.insert'
            behavior supported by the node.";
    }
    leaf maximum-encap-srh {
        type uint8;
        description
            "Node maximum-encap-srh is the maximum number of SIDs
            that can be included as part of the 'T.Encap'
            behavior supported by the node.";
    }
    leaf maximum-end-d-srh {
        type uint8;
        description
            "Node maximum-end-d-srh is the maximum number of SIDs
            in an SRH when applying 'End.DX6' and 'End.DT6'
            functions supported by the node.";
    }
}

container myLocalSidTable {
    container srv6EndSidTables {
        config false;
        description
            "Segment Routing IPv6 End Local-Sid table.";
        list srhEndSidForwTable {
            key "endSidValue";
            config false;
            description
                "Segment Routing IPv6 End Local-Sid table.";
            leaf endSidValue {
                type inet:ipv6-address-no-zone;
```

```

        config false;
        description
            "End SID value.";
    }
    leaf endSidFlavor {
        type string {
            length "0..64";
        }
        config false;
    }

```

```

        description
            "End SID flavor value.";
    }
}
}
container srv6EndXSidTables {
    config false;
    description
        "Segment Routing IPv6 End.X Local-Sid table.";
    list srhEndXSidForwTable {
        key "endXSidValue";
        config false;
        description
            "Segment Routing IPv6 End.X Local-Sid table.";
        leaf endXSidValue {
            type inet:ipv6-address-no-zone;
            config false;
            description
                "End.X SID value.";
        }
        leaf endXSidFlavor {
            type string {
                length "0..64";
            }
            config false;
            description
                "End.X SID flavor value.";
        }
    }
    container endXSidNhpInfos {
        config false;
        description
            "Next hop information.";
    }
}

```

```

list endXSidNhpInfo {
  key "interface nextHop";
  config false;
  description
    "Next hop information.";
  leaf interface {
    type string {
      length "0..64";
    }
    config false;
    description
      "Exit interface.";
  }
  leaf nextHop {
    type inet:ipv6-address-no-zone;
    config false;
  }
}

```

```

    description
      "Next hop IPV6 address.";
  }
}
}
}
}
description
  "My local SID table contains all the local SRv6 segments
  explicitly instantiated at the node.";
}
container explicitPaths {
  description
    "Explicit path list.";
  list explicitPath {
    key "explicitPathName";
    description
      "Explicit path. When a TE LSP is to be established,
      specify route hop constraints for the LSP.";
    leaf explicitPathName {
      type string {
        length "1..63";
        pattern '^[^ \?]*';
      }
      description

```

```

        "Explicit path name.";
    }
    container explicitPathHops {
        description
            "Route hop list.";
        list explicitPathHop {
            key "tunnelHopIndex";
            max-elements "96";
            description
                "Route hop.";
            leaf tunnelHopIndex {
                type uint32 {
                    range "1..65535";
                }
                description
                    "Route hop index.";
            }
            leaf tunnelHopMode {
                type erHopMode;
                default "IPv6_ADDRESS";
                description
                    "Route hop mode, specifies the
                     IPv6 address or the SRv6 SID.";
            }
        }
    }
}

```

```

    }
    leaf tunnelHopSidIPv6 {
        when "../tunnelHopMode='SID_IPV6'";
        type inet:ipv6-address-no-zone;
        mandatory true;
        description
            "SRv6 SID of the explicit route hop.";
    }
    leaf tunnelHopSidIpv6Type {
        when "../tunnelHopMode='SID_IPV6'";
        type erSidIpv6Type;
        default "none";
        description
            "SRv6 SID type of the explicit route hop.";
    }
}
}
}
}

```

```
    }  
  }  
}  
<CODE ENDS>
```

[6.](#) IANA Considerations

This draft has no request to IANA.

[7.](#) Security Considerations

The data model defined in this document does not create any security implications. This draft does not change any underlying security issues inherent in [[I-D.ietf-netmod-routing-cfg](#)].

[8.](#) Acknowledgements

TBD

[9.](#) References

[9.1.](#) Normative References

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

Filsfils, C., Leddy, J., daniel.voyer@bell.ca, d., daniel.bernier@bell.ca, d., Steinberg, D., Raszuk, R., Matsushima, S., Lebrun, D., Decraene, B., Peirens, B., Salsano, S., Naik, G., Elmalky, H., Jonnalagadda, P., Sharif, M., Ayyangar, A., Mynam, S., Henderickx, W., Bashandy, A., Raza, K., Dukes, D., Clad, F., and P. Camarillo, "SRv6 Network Programming", [draft-filsfils-spring-srv6-network-programming-01](#) (work in progress), June 2017.

- [I-D.ietf-netmod-routing-cfg]
Lhotka, L. and A. Lindem, "A YANG Data Model for Routing Management", [draft-ietf-netmod-routing-cfg-25](#) (work in progress), November 2016.
- [RFC3973] Adams, A., Nicholas, J., and W. Siadak, "Protocol Independent Multicast – Dense Mode (PIM-DM): Protocol Specification (Revised)", [RFC 3973](#), DOI 10.17487/RFC3973, January 2005, <<https://www.rfc-editor.org/info/rfc3973>>.
- [RFC4601] Fenner, B., Handley, M., Holbrook, H., and I. Kouvelas, "Protocol Independent Multicast – Sparse Mode (PIM-SM): Protocol Specification (Revised)", [RFC 4601](#), DOI 10.17487/RFC4601, August 2006, <<https://www.rfc-editor.org/info/rfc4601>>.
- [RFC4607] Holbrook, H. and B. Cain, "Source-Specific Multicast for IP", [RFC 4607](#), DOI 10.17487/RFC4607, August 2006, <<https://www.rfc-editor.org/info/rfc4607>>.
- [RFC4608] Meyer, D., Rockell, R., and G. Shepherd, "Source-Specific Protocol Independent Multicast in 232/8", [BCP 120](#), [RFC 4608](#), DOI 10.17487/RFC4608, August 2006, <<https://www.rfc-editor.org/info/rfc4608>>.
- [RFC4610] Farinacci, D. and Y. Cai, "Anycast-RP Using Protocol Independent Multicast (PIM)", [RFC 4610](#), DOI 10.17487/RFC4610, August 2006, <<https://www.rfc-editor.org/info/rfc4610>>.
- [RFC5015] Handley, M., Kouvelas, I., Speakman, T., and L. Vicisano, "Bidirectional Protocol Independent Multicast (BIDIR-PIM)", [RFC 5015](#), DOI 10.17487/RFC5015, October 2007, <<https://www.rfc-editor.org/info/rfc5015>>.

- [RFC5059] Bhaskar, N., Gall, A., Lingard, J., and S. Venaas, "Bootstrap Router (BSR) Mechanism for Protocol Independent Multicast (PIM)", [RFC 5059](#), DOI 10.17487/RFC5059, January 2008, <<https://www.rfc-editor.org/info/rfc5059>>.

- [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>>.

9.2. Informative References

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

Authors' Addresses

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

Email: huzhibo@huawei.com

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

Email: lizhenbin@huawei.com

Satoru Matsushima
Softbank
1-9-1,Higashi-Shimbashi,Minato-Ku .
Tokyo 105-7322
Japan

Email: satoru.matsushima@g.softbank.co.jp

Katsuhiro Horiba
Softbank
1-9-1,Higashi-Shimbashi,Minato-Ku .
Tokyo 105-7322
Japan

Email: katsuhiro.horiba@g.softbank.co.jp

