Networking Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 18, 2019

P. Psenak, Ed.
C. Filsfils
Cisco Systems
A. Bashandy
Individual
B. Decraene
Orange
Z. Hu
Huawei Technologies
October 15, 2018

IS-IS Extensions to Support Routing over IPv6 Dataplane
draft-bashandy-isis-srv6-extensions-04.txt

Abstract

Segment Routing (SR) allows for a flexible definition of end-to-end paths by encoding paths as sequences of topological sub-paths, called "segments". Segment routing architecture can be implemented over an MPLS data plane as well as an IPv6 data plane. This draft describes the IS-IS extensions required to support Segment Routing over an IPv6 data plane.

Requirements Language

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

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 18, 2019.

Table of Contents

1. Introduction 3
2. SRv6 Capabilities sub-TLV 3
3. Advertising Supported Algorithms 4
4. Advertising Maximum SRv6 SID Depths 4
4.1. Maximum Segments Left MSD Type 5
4.2. Maximum End Pop MSD Type 5
4.3. Maximum T.Insert MSD Type 5
4.4. Maximum T.Encaps MSD Type 5
4.5. Maximum End D MSD Type 6
5. SRv6 SIDs and Reachability 6
6. Advertising Locators and End SIDs 7
6.1. SRv6 Locator TLV Format 8
6.2. SRv6 End SID sub-TLV 9
7. Advertising SRv6 End.X SIDs 11
7.1. SRv6 End.X SID sub-TLV 11
7.2. SRv6 LAN End.X SID sub-TLV 13
8. Advertising Endpoint Function Types 14
9. IANA Considerations 15
9.1. SRv6 Locator TLV 15
9.1.1. SRv6 End SID sub-TLV 16
9.1.2. Revised sub-TLV table 16
9.2. SRv6 Capabilities sub-TLV 16
9.3. SRv6 End.X SID and SRv6 LAN End.X SID sub-TLVs 17
9.4. MSD Types 17
10. Security Considerations 17
11. Contributors 18
12. References 19
12.1. Normative References 19
12.2. Informative References 20
Authors' Addresses 21

1. Introduction

With Segment Routing (SR) [I-D.ietf-spring-segment-routing], a node steers a packet through an ordered list of instructions, called segments.

Segments are identified through Segment Identifiers (SIDs).

Segment Routing can be directly instantiated on the IPv6 data plane through the use of the Segment Routing Header defined in [I-D.ietf-6man-segment-routing-header]. SRv6 refers to this SR instantiation on the IPv6 dataplane.

The network programming paradigm [I-D.filsfils-spring-srv6-network-programming] is central to SRv6. It describes how any function can be bound to a SID and how any network program can be expressed as a combination of SID’s.

This document specifies IS-IS extensions that allow the IS-IS protocol to encode some of these functions.

Familiarity with the network programming paradigm [I-D.filsfils-spring-srv6-network-programming] is necessary to understand the extensions specified in this document.

This document defines one new top level IS-IS TLV and several new IS-IS sub-TLVs.

The SRv6 Capabilities sub-TLV announces the ability to support SRv6 and some Endpoint functions listed in Section 7 as well as advertising limitations when applying such Endpoint functions.

The SRv6 Locator top level TLV announces SRv6 locators - a form of summary address for the set of topology/algorithm specific SIDs associated with a node.

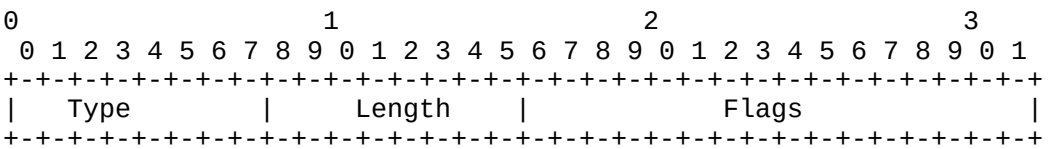
The SRv6 End SID sub-TLV, the SRv6 End.X SID sub-TLV, and the SRv6 LAN End.X SID sub-TLV are used to advertise which SIDs are instantiated at a node and what Endpoint function is bound to each instantiated SID.

2. SRv6 Capabilities sub-TLV

A node indicates that it has support for SRv6 by advertising a new SRv6- capabilities sub-TLV of the router capabilities TLV [RFC7981].

The SRv6 Capabilities sub-TLV may contain optional sub-sub-TLVs. No sub-sub-TLVs are currently defined.

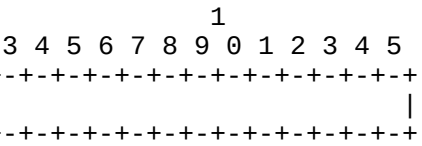
The SRv6 Capabilities sub-TLV has the following format:



Type: Suggested value 25, to be assigned by IANA

Length: 2 + length of sub-sub-TLVs

Flags: 2 octets The following flags are defined:



where:

O-flag: If set, the router supports use of the O-bit in the Segment Routing Header(SRH) as defined in [I-D.ali-spring-srv6-oam].

3. Advertising Supported Algorithms

SRv6 capable router indicates supported algorithm(s) by advertising the SR Algorithm TLV as defined in [I-D.ietf-isis-segment-routing-extensions].

4. Advertising Maximum SRv6 SID Depths

[I-D.ietf-isis-segment-routing-msd] defines the means to advertise node/link specific values for Maximum SID Depths (MSD) of various types. Node MSDs are advertised in a sub-TLV of the Router Capabilities TLV [RFC7981]. Link MSDs are advertised in a sub-TLV of TLVs 22, 23, 141, 222, and 223.

This document defines the relevant SRv6 MSDs and requests MSD type assignments in the MSD Types registry created by [I-D.ietf-isis-segment-routing-msd].

4.1. Maximum Segments Left MSD Type

The Maximum Segments Left MSD Type specifies the maximum value of the "SL" field [I-D.ietf-6man-segment-routing-header] in the SRH of a received packet before applying the Endpoint function associated with a SID.

SRH Max SL Type: 41 (Suggested value - to be assigned by IANA)

If no value is advertised the supported value is assumed to be 0.

4.2. Maximum End Pop MSD Type

The Maximum End Pop MSD Type specifies the maximum number of SIDs in the top SRH in an SRH stack to which the router can apply "PSP" or USP" as defined in [I-D.filsfils-spring-srv6-network-programming] flavors.

SRH Max End Pop Type: 42 (Suggested value - to be assigned by IANA)

If the advertised value is zero or no value is advertised then it is assumed that the router cannot apply PSP or USP flavors.

4.3. Maximum T.Insert MSD Type

The Maximum T.Insert MSD Type specifies the maximum number of SIDs that can be inserted as part of the "T.insert" behavior as defined in [I-D.filsfils-spring-srv6-network-programming].

SRH Max T.insert Type: 43 (Suggested value - to be assigned by IANA)

If the advertised value is zero or no value is advertised then the router is assumed not to support any variation of the "T.insert" behavior.

4.4. Maximum T.Encaps MSD Type

The Maximum T.Encaps MSD Type specifies the maximum number of SIDs that can be included as part of the "T.Encaps" behavior as defined in [I-D.filsfils-spring-srv6-network-programming] .

SRH Max T.encaps Type: 44 (Suggested value - to be assigned by IANA)

If the advertised value is zero then the router can apply T.Encaps only by encapsulating the incoming packet in another IPv6 header without SRH the same way IPinIP encapsulation is performed.

If the advertised value is non-zero then the router supports both IPinIP and SRH encapsulation subject to the SID limitation specified by the advertised value.

4.5. Maximum End D MSD Type

The Maximum End D MSD Type specifies the maximum number of SIDs in an SRH when performing decapsulation associated with "End.Dx" functions (e.g., "End.DX6" and "End.DT6") as defined in [I-D.filsfils-spring-srv6-network-programming].

SRH Max End D Type: 45 (Suggested value - to be assigned by IANA)

If the advertised value is zero or no value is advertised then it is assumed that the router cannot apply "End.DX6" or "End.DT6" functions if the extension header right underneath the outer IPv6 header is an SRH.

5. SRv6 SIDs and Reachability

As discussed in [I-D.filsfils-spring-srv6-network-programming], an SRv6 Segment Identifier (SID) is 128 bits and represented as

LOC:FUNCT

where LOC (the locator portion) is the L most significant bits and FUNCT is the 128-L least significant bits. L is called the locator length and is flexible. Each operator is free to use the locator length it chooses.

A node is provisioned with topology/algorithm specific locators for each of the topology/algorithm pairs supported by that node. Each locator is a covering prefix for all SIDs provisioned on that node which have the matching topology/algorithm.

Locators MUST be advertised in the SRv6 Locator TLV (see Section 6.1). Forwarding entries for the locators advertised in the SRv6 Locator TLV MUST be installed in the forwarding plane of receiving SRv6 capable routers when the associated topology/algorithm is supported by the receiving node.

Locators are routable and MAY also be advertised in Prefix Reachability TLVs (236 or 237).

Locators associated with algorithm 0 (for all supported topologies) SHOULD be advertised in a Prefix Reachability TLV (236 or 237) so that legacy routers (i.e., routers which do NOT support SRv6) will install a forwarding entry for algorithm 0 SRv6 traffic.

In cases where a locator advertisement is received in both in a Prefix Reachability TLV and an SRv6 Locator TLV, the Prefix Reachability advertisement MUST be preferred when installing entries in the forwarding plane. This is to prevent inconsistent forwarding entries on SRv6 capable/SRv6 incapable routers.

SRv6 SIDs are advertised as sub-TLVs in the SRv6 Locator TLV except for SRv6 End.X SIDs/LAN End.X SIDs which are associated with a specific Neighbor/Link and are therefore advertised as sub-TLVs in TLVs 22, 23, 222, 223, and 141.

SRv6 SIDs are not directly routable and MUST NOT be installed in the forwarding plane. Reachability to SRv6 SIDs depends upon the existence of a covering locator.

Adherence to the rules defined in this section will assure that SRv6 SIDs associated with a supported topology/algorithm pair will be forwarded correctly, while SRv6 SIDs associated with an unsupported topology/algorithm pair will be dropped. NOTE: The drop behavior depends on the absence of a default/summary route covering a given locator.

In order for forwarding to work correctly, the locator associated with SRv6 SID advertisements MUST be the longest match prefix installed in the forwarding plane for those SIDs. There are a number of ways in which this requirement could be compromised

- Another locator associated with a different topology/algorithm is the longest match
- A prefix advertisement (i.e., from TLV 236 or 237) is the longest match

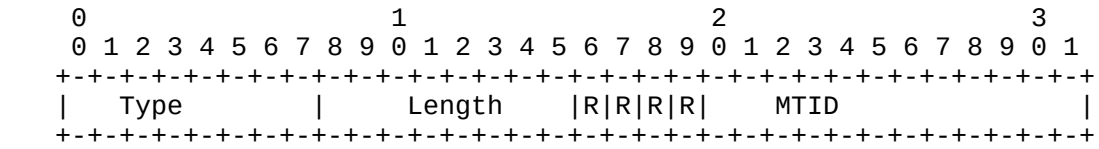
6. Advertising Locators and End SIDs

The SRv6 Locator TLV is introduced to advertise SRv6 Locators and End SIDs associated with each locator.

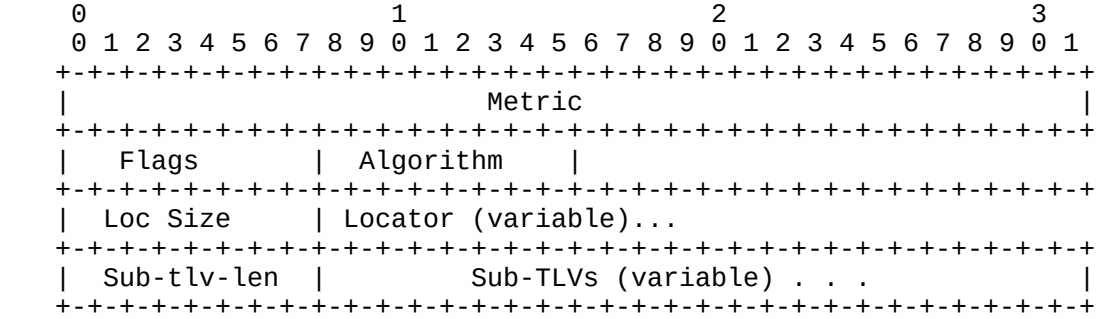
This new TLV shares the sub-TLV space defined for TLVs 135, 235, 236 and 237.

6.1. SRv6 Locator TLV Format

The SRv6 Locator TLV has the following format:



Followed by one or more locator entries of the form:



Type: 27 (Suggested value to be assigned by IANA)

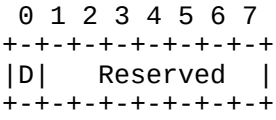
Length: variable.

MTID: Multitopology Identifier as defined in [RFC5120].
Note that the value 0 is legal.

Locator entry:

Metric: 4 octets. As described in [RFC5305].

Flags: 1 octet. The following flags are defined



where:
D bit: When the Locator is leaked from level-2 to level-1, the D bit MUST be set. Otherwise, this bit MUST be clear. Locators with the D bit set MUST NOT be leaked from level-1 to level-2.

This is to prevent looping.

The remaining bits are reserved for future use. They SHOULD be set to zero on transmission and MUST be ignored on receipt.

Algorithm: 1 octet. Associated algorithm. Algorithm values are defined in the IGP Algorithm Type registry.

Loc-Size: 1 octet. Number of bits in the Locator field. (1 - 128)

Locator: 1-16 octets. This field encodes the advertised SRv6 Locator. The Locator is encoded in the minimal number of octets for the given number of bits.

Sub-TLV-length: 1 octet. Number of octets used by sub-TLVs

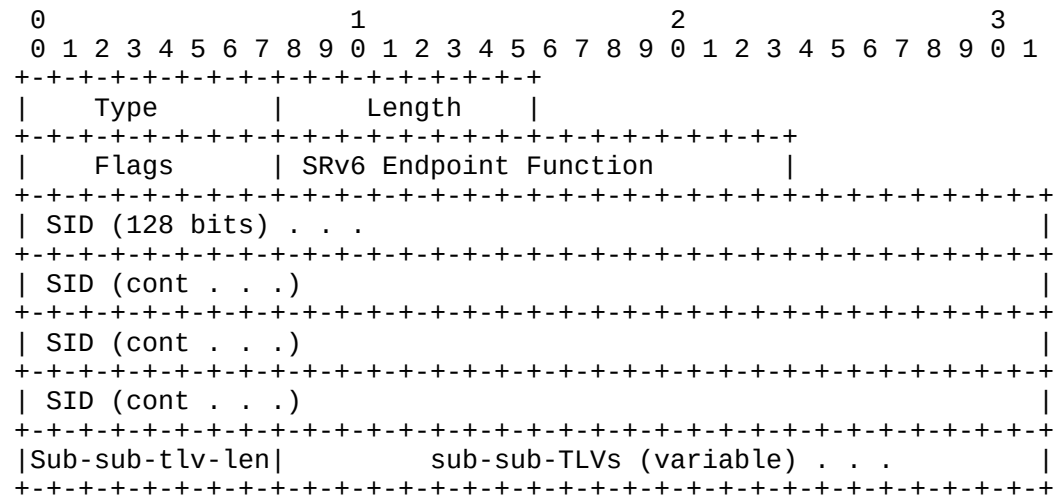
Optional sub-TLVs.

6.2. SRv6 End SID sub-TLV

The SRv6 End SID sub-TLV is introduced to advertise SRv6 Segment Identifiers (SID) with Endpoint functions which do not require a particular neighbor in order to be correctly applied [I-D.filsfils-spring-srv6-network-programming]. SRv6 SIDs associated with a neighbor are advertised using the sub-TLVs defined in Section 6.

This new sub-TLV is advertised in the SRv6 Locator TLV defined in the previous section. SRv6 End SIDs inherit the topology/algorithm from the parent locator.

The SRv6 End SID sub-TLV has the following format:



Type: 5 (Suggested value to be assigned by IANA)

Length: variable.

Flags: 1 octet. No flags are currently defined.

SRv6 Endpoint Function: 2 octets. As defined in [I-D.filsfils-spring-srv6-network-programming] Legal function values for this sub-TLV are defined in Section 7.

SID: 16 octets. This field encodes the advertised SRv6 SID.

Sub-sub-TLV-length: 1 octet. Number of octets used by sub-sub-TLVs

Optional sub-sub-TLVs

The SRv6 End SID MUST be a subnet of the associated Locator. SRv6 End SIDs which are NOT a subnet of the associated locator MUST be ignored.

Multiple SRv6 End SIDs MAY be associated with the same locator. In cases where the number of SRv6 End SID sub-TLVs exceeds the capacity of a single TLV, multiple Locator TLVs for the same locator MAY be advertised. For a given MTID/Locator the algorithm MUST be the same in all TLVs. If this restriction is not met all TLVs for that MTID/ Locator MUST be ignored.

7. Advertising SRv6 End.X SIDs

Certain SRv6 Endpoint functions [I-D.filsfils-spring-srv6-network-programming] must be associated with a particular neighbor, and in case of multiple layer 3 links to the same neighbor, with a particular link in order to be correctly applied.

This document defines two new sub-TLVs of TLV 22, 23, 222, 223, and 141 - namely "SRv6 End.X SID" and "SRv6 LAN End.X SID".

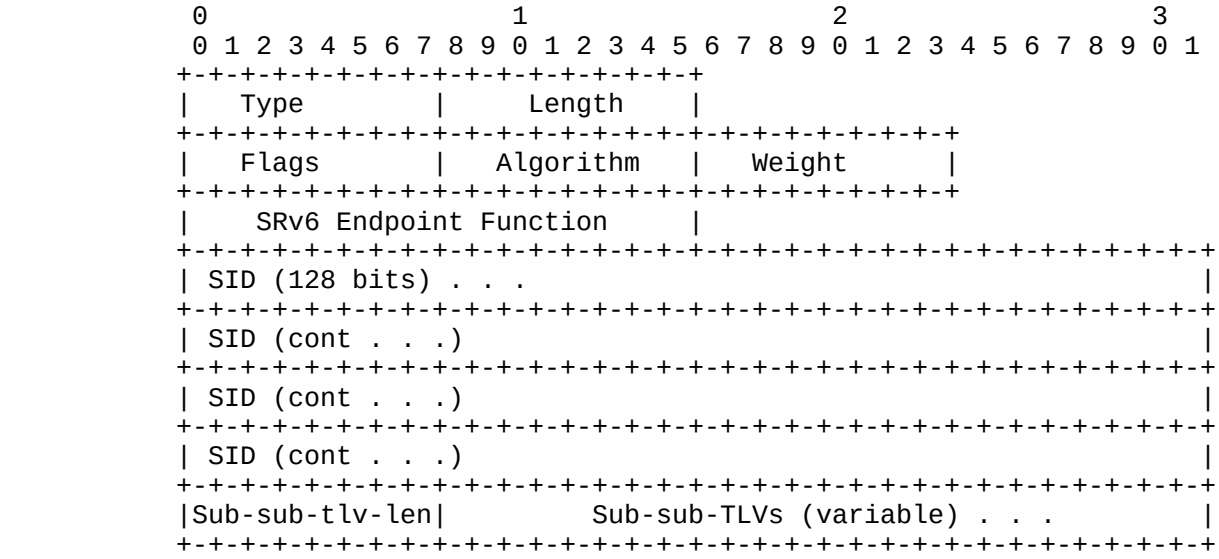
IS-IS Neighbor advertisements are topology specific - but not algorithm specific. End.X SIDs therefore inherit the topology from the associated neighbor advertisement, but the algorithm is specified in the individual SID.

All End.X SIDs MUST be a subnet of a Locator with matching topology and algorithm which is advertised by the same node in an SRv6 Locator TLV. End.X SIDs which do not meet this requirement MUST be ignored.

7.1. SRv6 End.X SID sub-TLV

This sub-TLV is used to advertise an SRv6 SID associated with a point to point adjacency. Multiple SRv6 End.X SID sub-TLVs MAY be associated with the same adjacency.

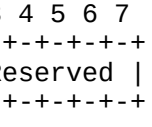
The SRv6 End.X SID sub-TLV has the following format:



Type: 43 (Suggested value to be assigned by IANA)

Length: variable.

Flags: 1 octet.



where:

B-Flag: Backup flag. If set, the End.X SID is eligible for protection (e.g., using IPFRR) as described in [RFC8355].

S-Flag. Set flag. When set, the S-Flag indicates that the End.X SID refers to a set of adjacencies (and therefore MAY be assigned to other adjacencies as well).

P-Flag. Persistent flag. When set, the P-Flag indicates that the End.X SID is persistently allocated, i.e., the End.X SID value remains consistent across router restart and/or interface flap.

Other bits: MUST be zero when originated and ignored when received.

Algorithm: 1 octet. Associated algorithm. Algorithm values are defined in the IGP Algorithm Type registry.

Weight: 1 octet. The value represents the weight of the End.X SID for the purpose of load balancing. The use of the weight is defined in [I-D.ietf-spring-segment-routing].

SRv6 Endpoint Function: 2 octets. As defined in [I-D.filsfils-spring-srv6-network-programming]
Legal function values for this sub-TLV are defined in Section 7.

SID: 16 octets. This field encodes the advertised SRv6 SID.

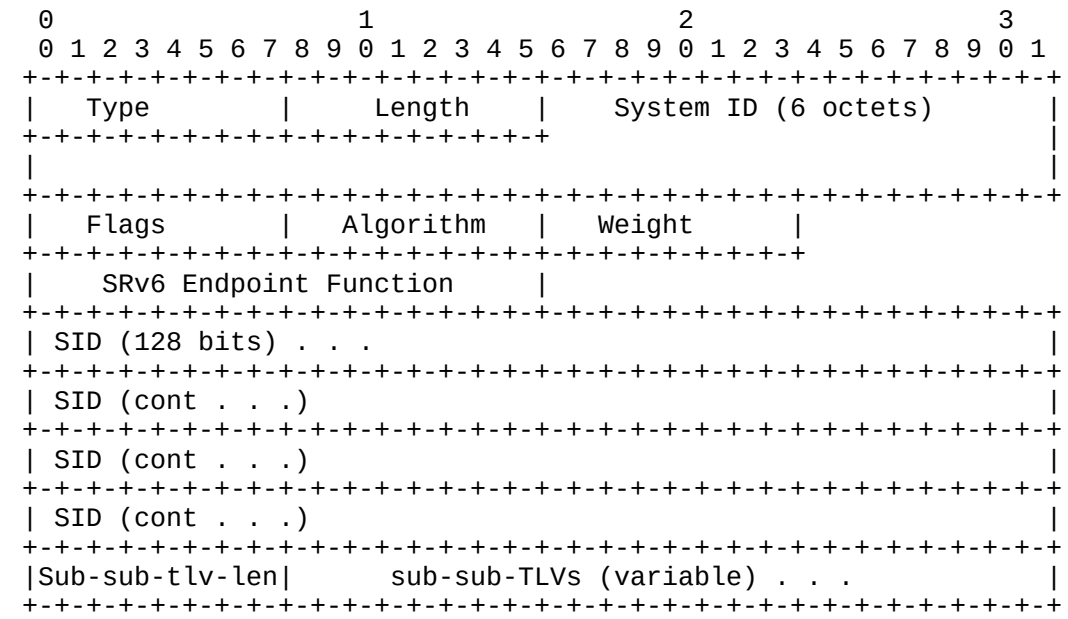
Sub-sub-TLV-length: 1 octet. Number of octets used by sub-sub-TLVs

Note that multiple TLVs for the same neighbor may be required in order to advertise all of the SRv6 End.X SIDs associated with that neighbor.

7.2. SRv6 LAN End.X SID sub-TLV

This sub-TLV is used to advertise an SRv6 SID associated with a LAN adjacency. Since the parent TLV is advertising an adjacency to the Designated Intermediate System(DIS) for the LAN, it is necessary to include the System ID of the physical neighbor on the LAN with which the SRv6 SID is associated. Given that a large number of neighbors may exist on a given LAN a large number of SRv6 LAN END.X SID sub-TLVs may be associated with the same LAN. Note that multiple TLVs for the same DIS neighbor may be required in order to advertise all of the SRv6 End.X SIDs associated with that neighbor.

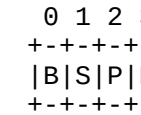
The SRv6 LAN End.X SID sub-TLV has the following format:



Type: 44 (Suggested value to be assigned by IANA)
Length: variable.

System-ID: 6 octets of IS-IS System-ID of length "ID Length" as defined in [ISO10589].

Flags: 1 octet.



where B,S, and P flags are as described in Section 6.1.
Other bits: MUST be zero when originated and ignored when received.

Algorithm: 1 octet. Associated algorithm. Algorithm values are defined in the IGP Algorithm Type registry.

Weight: 1 octet. The value represents the weight of the End.X SID for the purpose of load balancing. The use of the weight is defined in [I-D.ietf-spring-segment-routing].

SRv6 Endpoint Function: 2 octets. As defined in [I-D.filsfils-spring-srv6-network-programming]
Legal function values for this sub-TLV are defined in Section 7.

SID: 16 octets. This field encodes the advertised SRv6 SID.

Sub-sub-TLV-length: 1 octet. Number of octets used by sub-sub-TLVs.

8. Advertising Endpoint Function Types

Endpoint function types are defined in [I-D.filsfils-spring-srv6-network-programming]. The numerical values are defined in the "SRv6 Endpoint Types" registry defined in [I-D.filsfils-spring-srv6-network-programming]. This section lists the Endpoint function types which MAY be advertised by IS-IS and the SID sub-TLVs in which each type MAY appear.

Endpoint Function Type	End SID	End.X SID	Lan End.X SID
End(no PSP, no USP)	Y	N	N
End(with PSP)	Y	N	N
End(with USP)	Y	N	N
End(with PSP & USP)	Y	N	N
End.T(no PSP, no USP)	Y	N	N
End.T(with PSP)	Y	N	N
End.T(with USP)	Y	N	N
End.T(with PSP & USP)	Y	N	N
End.X(no PSP, no USP)	N	Y	Y
End.X(with PSP)	N	Y	Y
End.X(with USP)	N	Y	Y
End.X(with PSP & USP)	N	Y	Y
End.OTP	Y	N	N

9. IANA Considerations

This document requests allocation for the following TLVs, sub- TLVs, and sub-sub-TLVs as well updating the ISIS TLV registry and defining a new registry.

9.1. SRv6 Locator TLV

This document adds one new TLV to the IS-IS TLV Codepoints registry.

Value: 27 (suggested - to be assigned by IANA)

Name: SRv6 Locator

This TLV shares sub-TLV space with existing "Sub-TLVs for TLVs 135, 235, 236 and 237 registry". The name of this registry needs to be changed to "Sub-TLVs for TLVs 27, 135, 235, 236 and 237 registry".

9.1.1. SRv6 End SID sub-TLV

This document adds the following new sub-TLV to the (renamed) "Sub-TLVs for TLVs 27, 135, 235, 236 and 237 registry".

Value: 5 (suggested - to be assigned by IANA)

Name: SRv6 End SID

This document requests the creation of a new IANA managed registry for sub-sub-TLVs of the SRv6 End SID sub-TLV. The registration procedure is "Expert Review" as defined in [RFC7370]. Suggested registry name is "sub-sub-TLVs for SRv6 End SID sub-TLV". No sub-sub-TLVs are defined by this document except for the reserved value.

0: Reserved

1-255: Unassigned

9.1.2. Revised sub-TLV table

The revised table of sub-TLVs for the (renamed) "Sub-TLVs for TLVs 27, 135, 235, 236 and 237 registry" is shown below:

Type	27	135	235	236	237
1	n	y	y	y	y
2	n	y	y	y	y
3	n	y	y	y	y
4	y	y	y	y	y
5	y	n	n	n	n
11	y	y	y	y	y
12	y	y	y	y	y

9.2. SRv6 Capabilities sub-TLV

This document adds the definition of a new sub-TLV in the "Sub- TLVs for TLV 242 registry".

Type: 25 (Suggested - to be assigned by IANA)

Description: SRv6 Capabilities

This document requests the creation of a new IANA managed registry for sub-sub-TLVs of the SRv6 Capability sub-TLV. The registration procedure is "Expert Review" as defined in [RFC7370]. Suggested registry name is "sub-sub-TLVs for SRv6 Capability sub-TLV". No sub-sub-TLVs are defined by this document except for the reserved value.

0: Reserved

1-255: Unassigned

9.3. SRv6 End.X SID and SRv6 LAN End.X SID sub-TLVs

This document adds the definition of two new sub-TLVs in the "sub-TLVs for TLV 22, 23, 25, 141, 222 and 223 registry".

Type: 43 (suggested - to be assigned by IANA)

Description: SRv6 End.X SID

Type: 44 (suggested - to be assigned by IANA)

Description: SRv6 LAN End.X SID

Type	22	23	25	141	222	223
43	y	y	y	y	y	y
44	y	y	y	y	y	y

9.4. MSD Types

This document defines the following new MSD types. These types are to be defined in the IGP MSD Types registry defined in [I-D.ietf-isis-segment-routing-msd] .

All values are suggested values to be assigned by IANA.

Type	Description
41	SRH Max SL
42	SRH Max End Pop
43	SRH Max T.insert
44	SRH Max T.encaps
45	SRH Max End D

10. Security Considerations

Security concerns for IS-IS are addressed in [ISO10589], [RFC5304], and [RFC5310].

11. Contributors

The following people gave a substantial contribution to the content of this document and should be considered as co-authors:

Stefano Previdi
Huawei Technologies
Email: stefano@previdi.net

Paul Wells
Cisco Systems
Saint Paul,
Minnesota
United States
Email: pauwells@cisco.com

Daniel Voyer
Email: daniel.voyer@bell.ca

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

Bart Peirens
Email: bart.peirens@proximus.com

Hani Elmalky
Email: hani.elmalky@ericsson.com

Prem Jonnalagadda
Email: prem@barefootnetworks.com

Milad Sharif
Email: msharif@barefootnetworks.com>

Robert Hanzl
Cisco Systems
Millenium Plaza Building, V Celnici 10, Prague 1,
Prague, Czech Republic
Email rhanzl@cisco.com

Ketan Talaulikar
Cisco Systems, Inc.
Email: ketant@cisco.com

12. References

12.1. Normative References

[I-D.ali-spring-srv6-oam]
Ali, Z., Filsfils, C., Kumar, N., Pignataro, C., faiqbal@cisco.com, f., Gandhi, R., Leddy, J., Matsushima, S., Raszuk, R., daniel.voyer@bell.ca, d., Dawra, G., Peirens, B., Chen, M., and G. Naik, "Operations, Administration, and Maintenance (OAM) in Segment Routing Networks with IPv6 Data plane (SRv6)", draft-ali-spring-srv6-oam-01 (work in progress), July 2018.

[I-D.filsfils-spring-srv6-network-programming]
Filsfils, C., Camarillo, P., Leddy, J., daniel.voyer@bell.ca, d., Matsushima, S., and Z. Li, "SRv6 Network Programming", draft-filsfils-spring-srv6-network-programming-05 (work in progress), July 2018.

[I-D.ietf-6man-segment-routing-header]
Filsfils, C., Previdi, S., Leddy, J., Matsushima, S., and d. daniel.voyer@bell.ca, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-14 (work in progress), June 2018.

[I-D.ietf-isis-segment-routing-extensions]
Previdi, S., Ginsberg, L., Filsfils, C., Bashandy, A., Gredler, H., Litkowski, S., Decraene, B., and J. Tantsura, "IS-IS Extensions for Segment Routing", draft-ietf-isis-segment-routing-extensions-19 (work in progress), July 2018.

[I-D.ietf-isis-segment-routing-msd]
Tantsura, J., Chunduri, U., Aldrin, S., and L. Ginsberg, "Signaling MSD (Maximum SID Depth) using IS-IS", draft-ietf-isis-segment-routing-msd-19 (work in progress), October 2018.

[ISO10589]
"Intermediate system to Intermediate system intra-domain routeing information exchange protocol for use in conjunction with the protocol for providing the connectionless-mode Network Service (ISO 8473), ISO/IEC 10589:2002, Second Edition.", Nov 2002.

[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>.

[RFC5120] Przygienda, T., Shen, N., and N. Sheth, "M-ISIS: Multi Topology (MT) Routing in Intermediate System to Intermediate Systems (IS-ISs)", RFC 5120, DOI 10.17487/RFC5120, February 2008, <https://www.rfc-editor.org/info/rfc5120>.

[RFC5304] Li, T. and R. Atkinson, "IS-IS Cryptographic Authentication", RFC 5304, DOI 10.17487/RFC5304, October 2008, <https://www.rfc-editor.org/info/rfc5304>.

[RFC5305] Li, T. and H. Smit, "IS-IS Extensions for Traffic Engineering", RFC 5305, DOI 10.17487/RFC5305, October 2008, <https://www.rfc-editor.org/info/rfc5305>.

[RFC5310] Bhatia, M., Manral, V., Li, T., Atkinson, R., White, R., and M. Fanto, "IS-IS Generic Cryptographic Authentication", RFC 5310, DOI 10.17487/RFC5310, February 2009, <https://www.rfc-editor.org/info/rfc5310>.

[RFC7370] Ginsberg, L., "Updates to the IS-IS TLV Codepoints Registry", RFC 7370, DOI 10.17487/RFC7370, September 2014, <https://www.rfc-editor.org/info/rfc7370>.

[RFC7981] Ginsberg, L., Previdi, S., and M. Chen, "IS-IS Extensions for Advertising Router Information", RFC 7981, DOI 10.17487/RFC7981, October 2016, <https://www.rfc-editor.org/info/rfc7981>.

[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>.

12.2. Informative References

[I-D.ietf-spring-segment-routing] Filsfils, C., Previdi, S., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", draft-ietf-spring-segment-routing-15 (work in progress), January 2018.

[RFC8355] Filsfils, C., Ed., Previdi, S., Ed., Decraene, B., and R. Shakir, "Resiliency Use Cases in Source Packet Routing in Networking (SPRING) Networks", RFC 8355, DOI 10.17487/RFC8355, March 2018, <https://www.rfc-editor.org/info/rfc8355>.

Authors' Addresses

Peter Psenak (editor)
Cisco Systems
Pribinova Street 10
Bratislava 81109
Slovakia

Email: ppsenak@cisco.com

Clarence Filsfils
Cisco Systems
Brussels
Belgium

Email: cfilsfil@cisco.com

Ahmed Bashandy
Individual

Email: abashandy.ietf@gmail.com

Bruno Decraene
Orange
Issy-les-Moulineaux
France

Email: bruno.decraene@orange.com

Zhibo Hu
Huawei Technologies

Email: huzhibo@huawei.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: March 9, 2019

Z. Chen
Huawei
X. Xu
Alibaba
D. Cheng
Huawei

September 5, 2018

Avoiding Traffic Black-Holes for Route Aggregation in IS-IS
draft-chen-isis-black-hole-avoid-03

Abstract

When the Intermediate System to Intermediate System (IS-IS) routing protocol is adopted by a highly symmetric network such as the Leaf-Spine or Fat-Tree network, the Leaf nodes (e.g., Top of Rack switches in datacenters) are recommended to be prevented from receiving other nodes' explicit routes in order to achieve scalability. However, such a setup would cause traffic black-holes or suboptimal routing if link failure happens in the network. This document introduces INFINITE cost to IS-IS LSPs to solve this problem.

Requirements Language

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

Status of This Memo

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

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

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

This Internet-Draft will expire on March 9, 2019.

Table of Contents

1. Introduction 2
2. Problem Description 3
3. Solution 4
4. IANA Considerations 5
5. Security Considerations 5
6. Acknowledgements 5
7. References 5
Authors' Addresses 6

1. Introduction

When running the Intermediate System to Intermediate System (IS-IS) routing protocol in a highly symmetric network such as the Leaf-Spine or Fat-Tree network, the Leaf nodes (e.g., Top of Rack switches in datacenters) are recommended to be prevented from receiving other nodes' explicit routes in order to achieve scalability, as proposed in [IS-IS-SL-Extension], [IS-IS-Overhead-Reduction], [RIFT], and [OpenFabric]. In particular, each Leaf node SHOULD simply maintain a default (or aggregated) route (e.g., 0.0.0.0/0) in its routing table, of which the next hop SHOULD be an Equal Cost Multi Path (ECMP) group including all Spines nodes that the Leaf node connects to. However, such a setup would cause traffic black-holes or suboptimal routing if link failure happens in the network, since the Leaf nodes are not aware of any topology information.

To solve this problem, this document introduces INFINITE cost to IS-IS LSPs. When link failure happens between a Spine node and a Leaf node, the Spine node SHOULD advertise all prefixes attached to the Leaf node, whose costs SHOULD be set to be INFINITE, to every other Leaf node it connects to. On receiving the prefixes (with INFINITE

cost), each Leaf node SHOULD add the prefixes to its routing table, thus avoiding traffic black-holes and suboptimal routing.

2. Problem Description

This section illustrates why link failure would cause traffic black-hole or suboptimal routing when Leaf nodes only maintain default (or aggregated) routes.

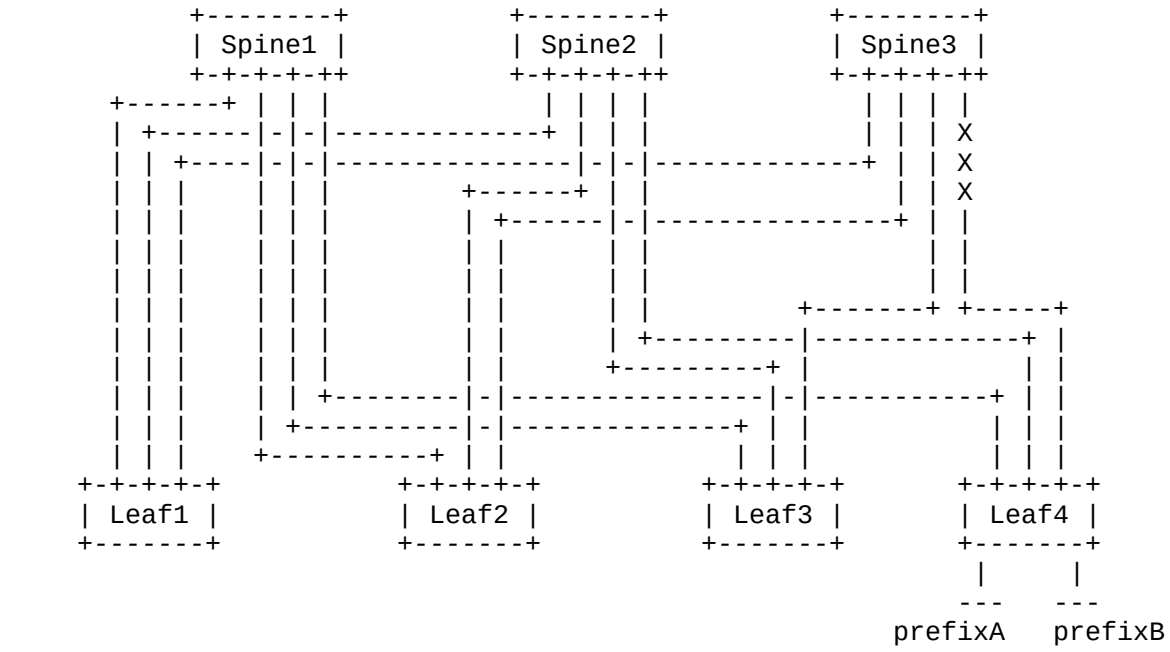


Figure 1: Topology Example

Figure 1 shows a Spine-Leaf topology example where Leaf1 to Leaf4 are connected to Spine1 to Spine3, and prefixA and prefixB are attached to Leaf4. To achieve scalability, as proposed in [IS-IS-SL-Extension], [IS-IS-Overhead-Reduction], [RIFT], and [OpenFabric], Leaf1 to Leaf4 SHOULD NOT receive explict routes from each other nor the Spine nodes. Instead, each of them maintains a default (or aggregated) route (e.g., 0.0.0.0/0) in the routing table, of which the next hop is an ECMP group including Spine1, Spine2, and Spine3. Flows from one Leaf node to another are shared among Spine1, Spine2, and Spine3 based on the well known 5-tuple hashing.

However, such a setup would cause traffic black-hole or suboptimal routing when link failure happens in the network. For example, if

the link between Spine3 and Leaf4 is broken, Leaf1, Leaf2, and Leaf3 could not get aware of the failure. As a result, these Leaf nodes will still send a portion of traffic destined for prefixA or prefixB toward Spine3, which makes the traffic be discarded at Spine3, causing traffic black-hole. On the other hand, if there are a set of links or a higher tier of switches interconnecting Spine1, Spine2, and Spine3, the traffic will be steered to other spine nodes or the higher-tier switches by Spine3, causing suboptimal routing.

Therefore, this document introduces INFINITE cost to IS-IS LSPs to solve this problem.

3. Solution

This document introduces the INFINITE cost to IS-IS LSPs, whose value is to be determined. When link failure happens between a Spine node and a Leaf node, the Spine node SHOULD 1) encode all prefixes attached to the Leaf node into the IP Reachability TLV, 2) set the cost of the prefixes to be INFINITE, 3) append the IP Reachability TLV to the IS-IS LSP, and 4) send the LSP to every other Leaf node it connects to.

When a Leaf node receives the prefixes (with INFINITE cost) advertised by a Spine node, it SHOULD install each of the prefixes into its routing table, of which the next hop SHOULD be set an ECMP group including all Spine nodes it connects to except the one who advertises the prefix.

For example, if the link between Spine3 and Leaf4 in Figure 1 is broken, Spine3 SHOULD advertise prefixA and prefixB to Leaf1, Leaf2, and Leaf3, by sending them an IS-IS LSP containing the IP Reachability TLV. The cost of prefixA and prefixB SHOULD be set INFINITE. On receiving the LSP, Leaf1, Leaf2, and Leaf3 SHOULD install prefixA and prefixB into their routing tables, and the next hop of prefixA or prefixB SHOULD be set an ECMP group including Spine1 and Spine2. For instance, the routing table of Leaf1 before and after the link failure is shown in Figure 2 and Figure 3, respectively.

Note that the mechanism described above could achieve minimal signaling latency, which helps to avoid black-hole or suboptimal routing rapidly when link failure happens.

Destination	Proto	Pre	Cost	Flags	NextHop	Interface
0.0.0.0/0	ISIS	15	20	D	Spine1	Ethernet0/0/0
	ISIS	15	20	D	Spine2	Ethernet0/0/1
	ISIS	15	20	D	Spine3	Ethernet0/0/2

Figure 2: Routing Table of Leaf1 before link failure

Destination	Proto	Pre	Cost	Flags	NextHop	Interface
0.0.0.0/0	ISIS	15	20	D	Spine1	Ethernet0/0/0
	ISIS	15	20	D	Spine2	Ethernet0/0/1
	ISIS	15	20	D	Spine3	Ethernet0/0/2
prefixA	ISIS	15	20	D	Spine1	Ethernet0/0/0
	ISIS	15	20	D	Spine2	Ethernet0/0/1
prefixB	ISIS	15	20	D	Spine1	Ethernet0/0/0
	ISIS	15	20	D	Spine2	Ethernet0/0/1

Figure 3: Routing Table of Leaf1 after link failure

4. IANA Considerations

TBD.

5. Security Considerations

TBD.

6. Acknowledgements

TBD.

7. References

[IS-IS-Overhead-Reduction]
Chen, Z., Xu, X., and D. Cheng, "Overheads Reduction for IS-IS Enabled Spine-Leaf Networks", draft-chen-isis-sl-overheads-reduction-03 (work in progress) , March 2018.

[IS-IS-SL-Extension]
Shen, N., Ginsberg, L., and S. Thyamagundalu, "IS-IS Routing for Spine-Leaf Topology", draft-shen-isis-spine-leaf-ext-06 (work in progress) , June 2018.

[OpenFabric]
White, R. and S. Zandi, "IS-IS Support for Openfabric", draft-white-openfabric-06 (work in progress) , June 2018.

[RFC1195] Callon, R., "Use of OSI IS-IS for Routing in TCP/IP and Dual Environments", RFC 1195 , December 1990.

[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>.

[RFC5305] Li, T. and H. Smit, "IS-IS Extensions for Traffic Engineering", RFC 5305 , October 2008.

[RIFT] Przygienda, T., Sharma, A., Drake, J., and A. Atlas, "RIFT: Routing in Fat Trees", draft-ietf-rift-rift-02 (work in progress) , June 2018.

Authors' Addresses

Zhe Chen
Huawei
No. 156 Beiqing Rd
Beijing 100095
China

Email: chenzhe17@huawei.com

Xiaohu Xu
Alibaba

Email: xiaohu.xxh@alibaba-inc.com

Dean Cheng
Huawei

Email: dean.cheng@huawei.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 6, 2018

Z. Chen
Huawei
X. Xu
Alibaba
D. Cheng
Huawei

March 5, 2018

Overheads Reduction for IS-IS Enabled Spine-Leaf Networks
draft-chen-isis-sl-overheads-reduction-03

Abstract

When a Spine-Leaf topology adopts the Intermediate System to Intermediate System (IS-IS) routing protocol, the Leaf node receives Link State Packets (LSPs) from all the other nodes thus having the entire routing information of the topology. This is usually considered unnecessary and costly. This document describes a solution to this problem by utilizing IS-IS's inherent multi-level and area partition features, which requires that an IS-IS router SHOULD check a level-1 LSP's area addresses before advertising it to a neighbor.

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 September 6, 2018.

Table of Contents

1. Introduction 2
2. Solution Description 3
2.1. Area Address Assignment 3
2.2. Area Address Checking 5
2.3. Default Route Advertising 7
3. Compatibility 7
3.1. Overlapping Areas Use Case 7
3.2. Maximum Area Addresses 7
4. IANA Considerations 8
5. Security Considerations 8
6. Acknowledgements 8
7. Normative References 8
Authors' Addresses 8

1. Introduction

Spine-Leaf topology (a.k.a., CLOS topology) is widely used in today's datacenter and campus networks. When the Spine-Leaf topology runs the Intermediate System to Intermediate System (IS-IS) routing protocol, each Leaf node receives Link State Packets (LSPs) from all the other nodes thus having the entire routing information of the topology. This is usually considered unnecessary and costly because the Leaf node only needs to know its default gateways (i.e., the Spine nodes it connects to) and the LSPs generated by the other Leaf nodes bring little benefit for it to forward traffic.

To avoid Leaf nodes from learning the unnecessary LSPs from one another, [IS-IS-SL-Extension] proposes a new TLV attached to the IS-IS Hello (IIH) PDU to carry an router's role (i.e., Spine or Leaf) in the topology. The Spine nodes then prevent all LSPs from being sent

to the Leaf nodes, and each Leaf node sets the Spine nodes it connects to as its default gateways.

This document proposes another solution to this problem, which utilizes IS-IS's inherent multi-level and area partition features. In particular, it requires that each Leaf node (configured as L1 router) SHOULD be assigned with a unique area address and each Spine node (configured as L1/L2 router) MUST NOT advertise level-1 LSPs of a given area to neighbors within another area. This prevents Leaf nodes from receiving routing information from one another, without introducing new message formats.

2. Solution Description

2.1. Area Address Assignment

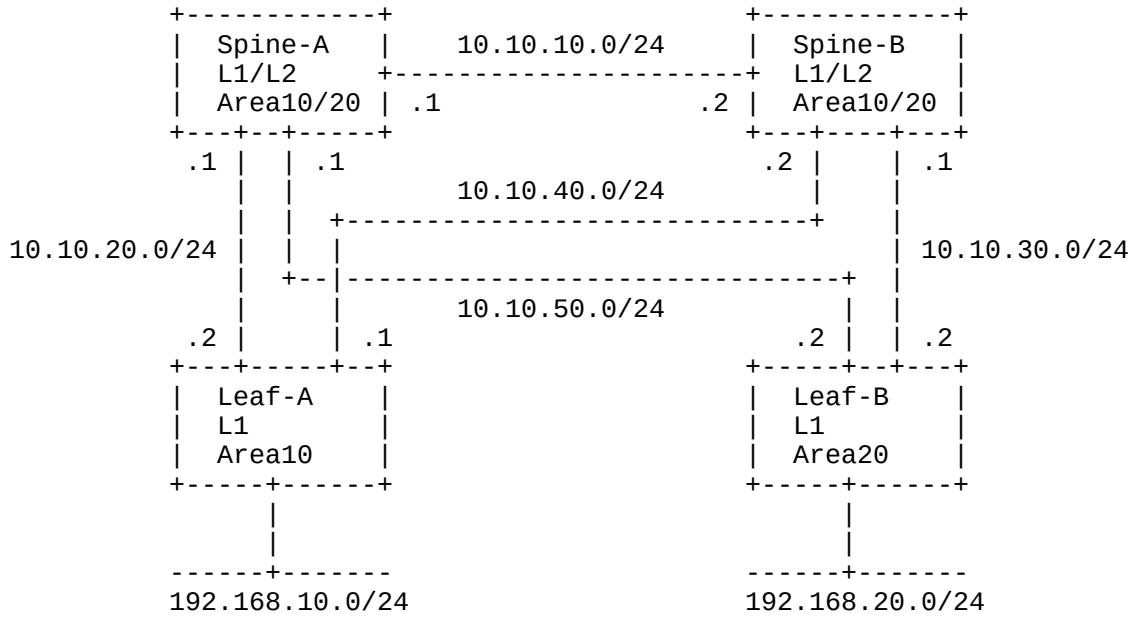


Figure 1: Topology Example

This section describes how to assign area addresses in the Spine-Leaf topology, and illustrates why IS-IS routers SHOULD check the area addresses before advertising level-1 LSPs. As shown in Figure 1, there are two Spine nodes (i.e., Spine-A and Spine-B) and two Leaf nodes (i.e., Leaf-A and Leaf-B). The System IDs of Spine-A, Spine-B, Leaf-A, and Leaf-B are 1111.1111.1111.00 to 4444.4444.4444.00, respectively.

To prevent a Leaf node from learning the routing information of the other ones, the following configurations are REQUIRED:

a. Leaf nodes SHOULD be configured as L1 routers and each of them SHOULD be assigned a unique area address.

b. Spine nodes SHOULD be configured as L1/L2 routers and SHOULD be assigned multiple area addresses with each being that of a given Leaf node connected to it.

As a result, Leaf-A and Leaf-B in Figure 1 are configured as L1 routers and are assigned 10 and 20 as their area addresses, respectively. Spine-A and Spine-B are configured as L1/L2 routers and are assigned both 10 and 20 as their area addresses.

Level-1 Link State Database (Spine-A):

LSPID	Seq Num	Checksum	Holdtime	Length	ATT/P/OL	Area
1111.1111.1111.00-00	0x0000006c	0x540b	743	124	0/0/0	10/20
2222.2222.2222.00-00	0x0000006d	0x933b	1068	124	0/0/0	10/20
3333.3333.3333.00-00	0x0000006b	0x1815	402	122	0/0/0	10
4444.4444.4444.00-00	0x0000006a	0xf543	431	122	0/0/0	20

Level-2 Link State Database (Spine-A):

LSPID	Seq Num	Checksum	Holdtime	Length	ATT/P/OL	Area
1111.1111.1111.00-00	0x0000006f	0x682f	743	150	0/0/0	10/20
2222.2222.2222.00-00	0x00000063	0x30eb	1068	150	0/0/0	10/20

Figure 2: Link State Database of Spine-A

Under such configurations, however, Leaf-A still receives Leaf-B’s LSPs (and vice versa) even though they are in different areas. This is because of the IS-IS definition that all routers in a specific area SHOULD share the same level-1 Link State Database (LSDB). In other words, IS-IS routers check area addresses during neighbor establishment, but are regardless of area addresses when advertising LSPs to a neighbor.

The example in Figure 1 and the LSDB of Spine-A (in Figure 2) further illustrate this. Since Spine-A and Leaf-B are both in area 20, Spine-A will receive LSP 4444.4444.4444.00-00 from Leaf-B and store the LSP into its level-1 LSDB. On the other hand, since Spine-A and Leaf-A are both in area 10, Spine-A will advertise LSP 4444.4444.4444.00-00 to Leaf-A although Leaf-A and Leaf-B (generator of the LSP) are in different areas. As a result, Leaf-A installs the route 192.168.20.0/24 into its routing table (Figure 3), even though it is an external area route.

Leaf-A Routing Table:

Destination	Proto	Pre	Cost	Flags	NextHop	Interface
10.10.10.0/24	ISIS-L1	15	20	D	10.10.20.1	Ethernet0/0/0
	ISIS-L1	15	20	D	10.10.40.2	Ethernet0/0/1
10.10.20.0/24	Direct	0	0	D	127.0.0.1	Ethernet0/0/0
10.10.30.0/24	ISIS-L1	15	20	D	10.10.40.2	Ethernet0/0/1
10.10.40.0/24	Direct	0	0	D	127.0.0.1	Ethernet0/0/1
10.10.50.0/24	ISIS-L1	15	20	D	10.10.20.1	Ethernet0/0/0
192.168.10.0/24	Direct	0	0	D	127.0.0.1	GEthernet0/0/0
192.168.20.0/24	ISIS-L1	15	30	D	10.10.20.1	Ethernet0/0/0
	ISIS-L1	15	30	D	10.10.40.2	Ethernet0/0/1
127.0.0.0/8	Direct	0	0	D	127.0.0.1	InLoopBack0
0.0.0.0/0	ISIS-L1	15	10	D	10.10.20.1	Ethernet0/0/0
	ISIS-L1	15	10	D	10.10.40.2	Ethernet0/0/1

Figure 3: Routing Table of Leaf-A

Therefore, the solution proposed in this document requires that an IS-IS router SHOULD check a level-1 LSP’s area addresses before advertising it to a neighbor (see Section 2.2).

2.2. Area Address Checking

Before advertising a level-1 LSP to a neighbor, an IS-IS router SHOULD compare the area addresses associated with the LSP and the ones associated with the neighbor. If they have at least one area

address in common, the router SHOULD advertise the LSP to the neighbor. Otherwise, the router MUST NOT advertise the LSP to the neighbor.

In the former case, the router SHOULD remove every area addresse in the LSP except the ones associated with the neighbor before the advertisement. This makes the solution more compatible since the Leaf nodes can be unaltered (see Section 3.2).

For instance, before Spine-A advertises LSP 1111.1111.1111.00-00 to Leaf-A, it compares the LSP's area addresses (i.e., 10 and 20) with Leaf-A's area address (i.e., 10). Since they have a common area address 10, Spine-A SHOULD remove area address 20 from the LSP and advertise the LSP to Leaf-A. On the other hand, before Spine-A advertises LSP 4444.4444.4444.00-00 to Leaf-A, it checks their area addresses and finds that they have no area address in common. So Spine-A MUST NOT advertise the LSP to Leaf-A. As a result, Leaf-A would not learn any routing information of Leaf-B, as shown in Figure 4.

Leaf-A Routing Table:

Destination	Proto	Pre	Cost	Flags	NextHop	Interface
10.10.10.0/24	ISIS-L1	15	20	D	10.10.20.1	Ethernet0/0/0
	ISIS-L1	15	20	D	10.10.40.2	Ethernet0/0/1
10.10.20.0/24	Direct	0	0	D	127.0.0.1	Ethernet0/0/0
10.10.30.0/24	ISIS-L1	15	20	D	10.10.40.2	Ethernet0/0/1
10.10.40.0/24	Direct	0	0	D	127.0.0.1	Ethernet0/0/1
10.10.50.0/24	ISIS-L1	15	20	D	10.10.20.1	Ethernet0/0/0
192.168.10.0/24	Direct	0	0	D	127.0.0.1	GEthernet0/0/0
127.0.0.0/8	Direct	0	0	D	127.0.0.1	InLoopBack0
0.0.0.0/0	ISIS-L1	15	10	D	10.10.20.1	Ethernet0/0/0
	ISIS-L1	15	10	D	10.10.40.2	Ethernet0/0/1

Figure 4: Routing Table of Leaf-A

2.3. Default Route Advertising

As defined in [RFC 1195], a L1/L2 router will indicate in its LSPs that it is "attached" by setting the ATT bits. Therefore, each Leaf node would set the Spine nodes as its default gateways and install a default route in its routing table, as shown in Figure 4.

However, a specific IS-IS implementation in this case may not let the L1/L2 router set the ATT bits, because it may speculate that the L1/L2 router has lost connectivity to the level-2 backbone. To solve this problem, operators can manually configure the L1/L2 router to advertise a default route.

3. Compatibility

3.1. Overlapping Areas Use Case

In most deployments, an IS-IS router is assigned only one area address, which will not be influenced by the area checking mechanism proposed in this document. However, an IS-IS router might be assigned more than one area addresses in some practical deployments for the following reasons: 1) it is desirable to change the area address of an area, 2) to merge two areas into one area, or 3) to partition an area into two areas.

For instance, to change an area’s address from X to Y, one can simply add area address Y to all routers in the area, and then remove X from them. Note that such operations would not disrupt live traffic in the network.

Although the solution in this document requires IS-IS router to check LSP’s area addresses before advertising it, the above use cases are still applicable and no compatible issue rises.

3.2. Maximum Area Addresses

The maximumAreaAddresses parameter in today’s IS-IS implementation is set to be 3 (or 0 which indicates 3) on consensus. Therefore, the solution in this document also requires that Spine node SHOULD be modified for supporting more area addresses. However, as LSPs sent to a given neighbor only carry the area address(es) of the neighbor (see Section 2.2), the solution does not require to modify Leaf nodes.

4. IANA Considerations

TBD.

5. Security Considerations

TBD.

6. Acknowledgements

TBD.

7. Normative References

[IS-IS-SL-Extension]
Shen, N., Ginsberg, L., and S. Thyamagundalu, "IS-IS Routing for Spine-Leaf Topology", draft-shen-isis-spine-leaf-ext-05 (work in progress) , January 2018.

[RFC1195] Callon, R., "Use of OSI IS-IS for Routing in TCP/IP and Dual Environments", RFC 1195 , December 1990.

[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

Zhe Chen
Huawei
No. 156 Beiqing Rd
Beijing 100095
China

Email: chenzhe17@huawei.com

Xiaohu Xu
Alibaba

Email: xiaohu.xxh@alibaba-inc.com

Dean Cheng
Huawei

Email: dean.cheng@huawei.com

Networking Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 22, 2017

L. Ginsberg
P. Psenak
Cisco Systems
S. Previdi
Individual
W. Henderickx
Nokia
June 20, 2017

IS-IS TE Attributes per application
draft-ginsberg-isis-te-app-03.txt

Abstract

Existing traffic engineering related link attribute advertisements have been defined and are used in RSVP-TE deployments. In cases where multiple applications wish to make use of these link attributes the current advertisements do not support application specific values for a given attribute nor do they support indication of which applications are using the advertised value for a given link.

This draft introduces new link attribute advertisements which address both of these shortcomings. It also discusses backwards compatibility issues and how to minimize duplicate advertisements in the presence of routers which do not support the extensions defined in this document.

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 http://datatracker.ietf.org/drafts/current/.

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

This Internet-Draft will expire on December 22, 2017.

Table of Contents

1. Introduction 3
2. Requirements Discussion 3
3. Legacy Advertisements 4
3.1. Legacy sub-TLVs 4
3.2. Legacy SRLG Advertisements 5
4. Advertising Application Specific Link Attributes 5
4.1. Application Identifier Bit Mask 5
4.2. Application Specific Link Attributes sub-TLV 7
4.3. Application Specific SRLG TLV 8
5. Deployment Considerations 9
6. Attribute Advertisements and Enablement 10
7. Interoperability, Backwards Compatibility and Migration Concerns 10
7.1. RSVP-TE only deployments 11
7.2. Multiple Applications: Common Attributes with RSVP-TE . 11
7.3. Multiple Applications: All Attributes Not Shared w RSVP-TE 11
7.4. Deprecating legacy advertisements 11
8. IANA Considerations 12
9. Security Considerations 13
10. Acknowledgements 13
11. References 13
11.1. Normative References 13
11.2. Informative References 14
Authors' Addresses 14

1. Introduction

Advertisement of link attributes by the Intermediate-System-to-Intermediate-System (IS-IS) protocol in support of traffic engineering (TE) was introduced by [RFC5305] and extended by [RFC5307], [RFC6119], and [RFC7810]. Use of these extensions has been associated with deployments supporting Traffic Engineering over Multiprotocol Label Switching (MPLS) in the presence of Resource Reservation Protocol (RSVP) - more succinctly referred to as RSVP-TE.

In recent years new applications have been introduced which have use cases for many of the link attributes historically used by RSVP-TE. Such applications include Segment Routing Traffic Engineering (SRTE) and Loop Free Alternates (LFA). This has introduced ambiguity in that if a deployment includes a mix of RSVP-TE support and SRTE support (for example) it is not possible to unambiguously indicate which advertisements are to be used by RSVP-TE and which advertisements are to be used by SRTE. If the topologies are fully congruent this may not be an issue, but any incongruence leads to ambiguity.

An additional issue arises in cases where both applications are supported on a link but the link attribute values associated with each application differ. Current advertisements do not support advertising application specific values for the same attribute on a specific link.

This document defines extensions which address these issues. Also, as evolution of use cases for link attributes can be expected to continue in the years to come, this document defines a solution which is easily extensible to the introduction of new applications and new use cases.

2. Requirements Discussion

As stated previously, evolution of use cases for link attributes can be expected to continue - so any discussion of existing use cases is limited to requirements which are known at the time of this writing. However, in order to determine the functionality required beyond what already exists in IS-IS, it is only necessary to discuss use cases which justify the key points identified in the introduction - which are:

1. Support for indicating which applications are using the link attribute advertisements on a link

2. Support for advertising application specific values for the same attribute on a link

[RFC7855] discusses use cases/requirements for SR. Included among these use cases is SRTE. If both RSVP-TE and SRTE are deployed in a network, link attribute advertisements can be used by one or both of these applications. As there is no requirement for the link attributes advertised on a given link used by SRTE to be identical to the link attributes advertised on that same link used by RSVP-TE, there is a clear requirement to indicate independently which link attribute advertisements are to be used by each application.

As the number of applications which may wish to utilize link attributes may grow in the future, an additional requirement is that the extensions defined allow the association of additional applications to link attributes without altering the format of the advertisements or introducing new backwards compatibility issues.

Finally, there may still be many cases where a single attribute value can be shared among multiple applications, so the solution must minimize advertising duplicate link/attribute pairs whenever possible.

3. Legacy Advertisements

There are existing advertisements used in support of RSVP-TE. These advertisements include sub-TLVs for TLVs 22, 23, 141, 222, and 223 and TLVs for SRLG advertisement.

3.1. Legacy sub-TLVs

Sub-TLVs for TLVs 22, 23, 141, 222, and 223

Code Point	Attribute Name
3	Administrative group (color)
9	Maximum link bandwidth
10	Maximum reservable link bandwidth
11	Unreserved bandwidth
14	Extended Administrative Group
33	Unidirectional Link Delay
34	Min/Max Unidirectional Link Delay
35	Unidirectional Delay Variation
36	Unidirectional Link Loss
37	Unidirectional Residual Bandwidth
38	Unidirectional Available Bandwidth
39	Unidirectional Utilized Bandwidth

3.2. Legacy SRLG Advertisements

TLV 138 GMPLS-SRLG
Supports links identified by IPv4 addresses and unnumbered links

TLV 139 IPv6 SRLG
Supports links identified by IPv6 addresses

Note that [RFC6119] prohibits the use of TLV 139 when it is possible to use TLV 138.

4. Advertising Application Specific Link Attributes

Two new code points are defined in support of Application Specific Link Attribute Advertisements:

1) Application Specific Link Attributes sub-TLV for TLVs 22, 23, 141, 222, and 223

2)Application Specific Shared Risk Link Group (SRLG) TLV

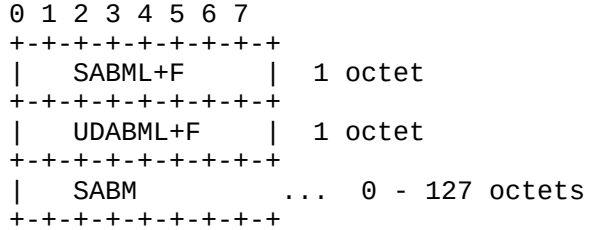
In support of these new advertisements, an application bit mask is defined which identifies the application(s) associated with a given advertisement.

The following sections define the format of these new advertisements.

4.1. Application Identifier Bit Mask

Identification of the set of applications associated with link attribute advertisements utilizes two bit masks. One bit mask is for standard applications where the definition of each bit is defined in a new IANA controlled registry. A second bit mask is for non-standard User Defined Applications(UDAs).

The encoding defined below is used by both the Application Specific Link Attributes sub-TLV and the Application Specific SRLG TLV.



```
|   UDABM        ...  0 - 127 octets
+-+-+-+-+-+-+-+-+
```

SABML+F (1 octet)
Standard Application Bit Mask Length/Flags

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|L|  SA-Length  |
+-+-+-+-+-+-+-+-+
```

L-flag: Applications listed (both Standard and User Defined) MUST use the legacy advertisements for the corresponding link found in TLVs 22, 23, 141, 222, and 223 or TLV 138 or TLV 139 as appropriate.

SA-Length: Indicates the length in octets (0-127) of the Bit Mask for Standard Applications.

UDABML+F (1 octet)
User Defined Application Bit Mask Length/Flags

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|R| UDA-Length  |
+-+-+-+-+-+-+-+-+
```

R: Reserved. Transmitted as 0 and ignored on receipt

UDA-Length: Indicates the length in octets (0-127) of the Bit Mask for User Defined Applications.

SABM (variable length)
Standard Application Bit Mask

(SA-Length * 8) bits

This is omitted if SA-Length is 0.

```
 0 1 2 3 4 5 6 7 ...
+-+-+-+-+-+-+-+-+...
|R|S|F|          ...
+-+-+-+-+-+-+-+-+...
```

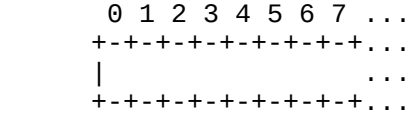
R-bit: RSVP-TE

S-bit: Segment Routing Traffic Engineering

F-bit: Loop Free Alternate

UDABM (variable length)
User Defined Application Bit Mask

(UDA Length * 8) bits



This is omitted if UDA-Length is 0.

NOTE: If both SA-length and UDA-Length are zero, then the attributes associated with this attribute identifier bit mask MAY be used by any Standard Application and any User Defined Application.

Standard Application Bits are defined/sent starting with Bit 0. Additional bit definitions that may be defined in the future SHOULD be assigned in ascending bit order so as to minimize the number of octets that will need to be transmitted. Undefined bits MUST be transmitted as 0 and MUST be ignored on receipt. Bits that are NOT transmitted MUST be treated as if they are set to 0 on receipt.

User Defined Application bits have no relationship to Standard Application bits and are NOT managed by IANA or any other standards body. It is recommended that bits are used starting with Bit 0 so as to minimize the number of octets required to advertise all UDAs.

4.2. Application Specific Link Attributes sub-TLV

A new sub-TLV for TLVs 22, 23, 141, 222, and 223 is defined which supports specification of the applications and application specific attribute values.

Type: 15 (suggested value - to be assigned by IANA)
Length: Variable (1 octet)
Value:

Application Bit Mask (as defined in Section 3.1)

Link Attribute sub-sub-TLVs - format matches the existing formats defined in [RFC5305] and [RFC7810]

When the L-flag is set in the Application Identifiers, all of the applications specified in the bit mask MUST use the link attribute sub-TLV advertisements listed in Section 3.1 for the corresponding link. Application specific link attribute sub-sub-TLVs for the corresponding link attributes MUST NOT be advertised for the set of applications specified in the Standard/User Application Bit Masks and all such advertisements MUST be ignored on receipt.

Multiple sub-TLVs for the same link MAY be advertised. When multiple sub-TLVs for the same link are advertised, they SHOULD advertise non-conflicting application/attribute pairs. A conflict exists when the same application is associated with two different values of the same link attribute for a given link. In cases where conflicting values for the same application/attribute/link are advertised all the conflicting values MUST be ignored.

For a given application, the setting of the L-flag MUST be the same in all sub-TLVs for a given link. In cases where this constraint is violated, the L-flag MUST be considered set for this application.

A new registry of sub-sub-TLVs is to be created by IANA which defines the link attribute sub-sub-TLV code points. A sub-sub-TLV is defined for each of the existing sub-TLVs listed in Section 3.1. Format of the sub-sub-TLVs matches the format of the corresponding legacy sub-TLV and IANA is requested to assign the legacy sub-TLV identifer to the corresponding sub-sub-TLV.

4.3. Application Specific SRLG TLV

A new TLV is defined to advertise application specific SRLGs for a given link. Although similar in functionality to TLV 138 (defined by [RFC5307]) and TLV 139 (defined by [RFC6119], a single TLV provides support for IPv4, IPv6, and unnumbered identifiers for a link. Unlike TLVs 138/139, it utilizes sub-TLVs to encode the link identifiers in order to provide the flexible formatting required to support multiple link identifier types.

Type: 238 (Suggested value - to be assigned by IANA)
Length: Number of octets in the value field (1 octet)
Value:
  Neighbor System-ID + pseudo-node ID (7 octets)
  Application Bit Mask (as defined in Section 3.1)
  Length of sub-TLVs (1 octet)
  Link Identifier sub-TLVs (variable)
  0 or more SRLG Values (Each value is 4 octets)

The following Link Identifier sub-TLVs are defined. The type values are suggested and will be assigned by IANA - but as the formats are identical to existing sub-TLVs defined for TLVs 22, 23, 141, 222, and 223 the use of the suggested sub-TLV types is strongly encouraged.

Type	Description
4	Link Local/Remote Identifiers (see [RFC5307])
6	IPv4 interface address (see [RFC5305])
8	IPv4 neighbor address (see [RFC5305])
12	IPv6 Interface Address (see [RFC6119])
13	IPv6 Neighbor Address (see [RFC6119])

At least one set of link identifiers (IPv4, IPv6, or unnumbered) MUST be present. TLVs which do not meet this requirement MUST be ignored.

Multiple TLVs for the same link MAY be advertised.

When the L-flag is set in the Application Identifiers, SRLG values MUST NOT be included in the TLV. Any SRLG values which are advertised MUST be ignored. Based on the link identifiers advertised the corresponding legacy TLV (see Section 3.2) can be identified and the SRLG values advertised in the legacy TLV MUST be used by the set of applications specified in the Application Bit Mask.

For a given application, the setting of the L-flag MUST be the same in all TLVs for a given link. In cases where this constraint is violated, the L-flag MUST be considered set for this application.

5. Deployment Considerations

If link attributes are advertised associated with zero length application bit masks for both standard applications and user defined applications, then that set of link attributes MAY be used by any application. If support for a new application is introduced on any node in a network in the presence of such advertisements, these advertisements MAY be used by the new application. If this is not what is intended, then existing advertisements MUST be readvertised

with an explicit set of applications specified before a new application is introduced.

6. Attribute Advertisements and Enablement

This document defines extensions to support the advertisement of application specific link attributes. The presence or absence of link attribute advertisements for a given application on a link does NOT indicate the state of enablement of that application on that link. Enablement of an application on a link is controlled by other means.

For some applications, the concept of enablement is implicit. For example, SRTE implicitly is enabled on all links which are part of the Segment Routing enabled topology. Advertisement of link attributes supports constraints which may be applied when specifying an explicit path through that topology.

For other applications enablement is controlled by local configuration. For example, use of a link as an LFA can be controlled by local enablement/disablement and/or the use of administrative tags.

It is an application specific policy as to whether a given link can be used by that application even in the abscence of any application specififc link attributes.

7. Interoperability, Backwards Compatibility and Migration Concerns

Existing deployments of RSVP-TE utilize the legacy advertisements listed in Section 3. Routers which do not support the extensions defined in this document will only process legacy advertisements and are likely to infer that RSVP-TE is enabled on the links for which legacy advertisements exist. It is expected that deployments using the legacy advertisements will persist for a significant period of time - therefore deployments using the extensions defined in this document must be able to co-exist with use of the legacy advertisements by routers which do not support the extensions defined in this document. The following sub-sections discuss interoperability and backwards compatibility concerns for a number of deployment scenarios.

Note that in all cases the defined strategy can be employed on a per link basis.

7.1. RSVP-TE only deployments

In deployments where RSVP-TE is the only application utilizing link attribute advertisements, use of the the legacy advertisements can continue without change.

7.2. Multiple Applications: Common Attributes with RSVP-TE

In cases where multiple applications are utilizing a given link, one of the applications is RSVP-TE, and all link attributes for a given link are common to the set of applications utilizing that link, interoperability is achieved by using legacy advertisements and sending application specific advertisements with L-bit set and no link attribute values. This avoids duplication of link attribute advertisements.

7.3. Multiple Applications: All Attributes Not Shared w RSVP-TE

In cases where one or more applications other than RSVP-TE are utilizing a given link and one or more link attribute values are NOT shared with RSVP-TE, it is necessary to use application specific advertisements as defined in this document. Attributes for applications other than RSVP-TE MUST be advertised using application specific advertisements which have the L-bit clear. In cases where some link attributes are shared with RSVP-TE, this requires duplicate advertisements for those attributes.

The discussion in this section applies to cases where RSVP-TE is NOT using any advertised attributes on a link and to cases where RSVP-TE is using some link attribute advertisements on the link but some link attributes cannot be shared with RSVP-TE.

7.4. Deprecating legacy advertisements

The extensions defined in this document support RSVP-TE as one of the supported applications - so a long term goal for deployments would be to deprecate use of the legacy advertisements in support of RSVP-TE. This can be done in the following step-wise manner:

1)Upgrade all routers to support extensions in this document

2)Readvertise all legacy link attributes using application specific advertisements with L-bit clear and R-bit set.

3)Remove legacy advertisements

8. IANA Considerations

This document defines a new sub-TLV for TLVs 22, 23, 141, 222, and 223.

Type	Description	22	23	141	222	223
15	Application Specific Link Attributes	y	y	y	y	y

This document defines one new TLV:

Type	Description	IIH	SNP	LSP	Purge
238	Application Specific SRLG	n	n	y	n

This document requests a new IANA registry be created to control the assignment of sub-sub-TLV codepoints for the Application Specific Link Attributes sub-TLV. The suggested name of the new registry is "sub-sub-TLV code points for application link attributes". The registration procedure is "Expert Review" as defined in [RFC5226]. The following assignments are made by this document:

Type	Description
3	Administrative group (color)
9	Maximum link bandwidth
10	Maximum reservable link bandwidth
11	Unreserved bandwidth
14	Extended Administrative Group
33	Unidirectional Link Delay
34	Min/Max Unidirectional Link Delay
35	Unidirectional Delay Variation
36	Unidirectional Link Loss
37	Unidirectional Residual Bandwidth
38	Unidirectional Available Bandwidth
39	Unidirectional Utilized Bandwidth

This document requests a new IANA registry be created to control the assignment of application bit identifiers. The suggested name of the new registry is "Link Attribute Applications". The registration procedure is "Expert Review" as defined in [RFC5226]. The following assignments are made by this document:

Bit #	Name
0	RSVP-TE (R-bit)
1	Segment Routing Traffic Engineering (S-bit)
2	Loop Free Alternate (F-bit)

This document requests a new IANA registry be created to control the assignment of sub-TLV types for the application specific SRLG TLV. The suggested name of the new registry is "Sub-TLVs for TLV 238". The registration procedure is "Expert Review" as defined in [RFC5226]. The following assignments are made by this document:

Value	Description
4	Link Local/Remote Identifiers (see [RFC5307])
6	IPv4 interface address (see [RFC5305])
8	IPv4 neighbor address (see [RFC5305])
12	IPv6 Interface Address (see [RFC6119])
13	IPv6 Neighbor Address (see [RFC6119])

9. Security Considerations

Security concerns for IS-IS are addressed in [ISO10589, [RFC5304], and [RFC5310].

10. Acknowledgements

The authors would like to thank John Drake and Acee Lindem for their careful review and content suggestions.

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, <http://www.rfc-editor.org/info/rfc2119>.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, DOI 10.17487/RFC5226, May 2008, <http://www.rfc-editor.org/info/rfc5226>.

[RFC5304] Li, T. and R. Atkinson, "IS-IS Cryptographic Authentication", RFC 5304, DOI 10.17487/RFC5304, October 2008, <http://www.rfc-editor.org/info/rfc5304>.

[RFC5305] Li, T. and H. Smit, "IS-IS Extensions for Traffic Engineering", RFC 5305, DOI 10.17487/RFC5305, October 2008, <http://www.rfc-editor.org/info/rfc5305>.

[RFC5307] Kompella, K., Ed. and Y. Rekhter, Ed., "IS-IS Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)", RFC 5307, DOI 10.17487/RFC5307, October 2008, <http://www.rfc-editor.org/info/rfc5307>.

[RFC5310] Bhatia, M., Manral, V., Li, T., Atkinson, R., White, R., and M. Fanto, "IS-IS Generic Cryptographic Authentication", RFC 5310, DOI 10.17487/RFC5310, February 2009, <http://www.rfc-editor.org/info/rfc5310>.

[RFC6119] Harrison, J., Berger, J., and M. Bartlett, "IPv6 Traffic Engineering in IS-IS", RFC 6119, DOI 10.17487/RFC6119, February 2011, <http://www.rfc-editor.org/info/rfc6119>.

[RFC7810] Previdi, S., Ed., Giacalone, S., Ward, D., Drake, J., and Q. Wu, "IS-IS Traffic Engineering (TE) Metric Extensions", RFC 7810, DOI 10.17487/RFC7810, May 2016, <http://www.rfc-editor.org/info/rfc7810>.

11.2. Informative References

[RFC7855] Previdi, S., Ed., Filsfils, C., Ed., Decraene, B., Litkowski, S., Horneffer, M., and R. Shakir, "Source Packet Routing in Networking (SPRING) Problem Statement and Requirements", RFC 7855, DOI 10.17487/RFC7855, May 2016, <http://www.rfc-editor.org/info/rfc7855>.

Authors' Addresses

Les Ginsberg
Cisco Systems
821 Alder Drive
Milpitas, CA 95035
USA

Email: ginsberg@cisco.com

Peter Psenak
Cisco Systems
Apollo Business Center Mlynske nivy 43
Bratislava 821 09
Slovakia

Email: ppsenak@cisco.com

Stefano Previdi
Individual

Email: stefano@previdi.net

Wim Henderickx
Nokia
Copernicuslaan 50
Antwerp 2018 94089
Belgium

Email: wim.henderickx@nokia.com

IS-IS Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 31, 2018

S. Litkowski
Orange
Y. Qu
Huawei
P. Sarkar
Individual
I. Chen
Jabil
J. Tantsura
Individual
June 29, 2018

YANG Data Model for IS-IS Segment Routing
draft-ietf-isis-sr-yang-04

Abstract

This document defines a YANG data model that can be used to configure and manage IS-IS Segment Routing ([I-D.ietf-isis-segment-routing-extensions].

Requirements Language

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

Status of This Memo

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

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

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

This Internet-Draft will expire on December 31, 2018.

Table of Contents

1. Overview 2
2. IS-IS Segment Routing 3
3. IS-IS Segment Routing configuration 6
3.1. Segment Routing activation 6
3.2. Advertising mapping server policy 6
3.3. IP Fast reroute 6
4. IS-IS Segment Routing YANG Module 6
5. Security Considerations 20
6. Contributors 20
7. Acknowledgements 21
8. IANA Considerations 21
9. Change log for ietf-isis-sr YANG module 21
9.1. From version -03 to version -04 21
9.2. From version -02 to version -03 21
9.3. From isis-sr document version -01 to version -02 21
9.4. From isis-sr document version -00 to version -01 22
9.5. From isis document version -12 to isis-sr document version -00 22
9.6. From isis document version -12 to version -13 22
9.7. From isis document version -09 to version -11 22
9.8. From isis document version -08 to version -09 22
9.9. From isis document version -07 to version -08 22
10. Normative References 22
Authors' Addresses 23

1. Overview

YANG [RFC6020] [RFC7950] is a data definition language used 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 encodings other than XML (e.g., JSON) are being defined. Furthermore, YANG data models can be used as the basis for implementation of other interfaces, such as CLI and programmatic APIs.

This document defines a YANG data model that can be used to configure and manage IS-IS Segment Routing and it is an augmentation to the IS-IS YANG data model.

2. IS-IS Segment Routing

This document defines a model for IS-IS Segment Routing feature. It is an augmentation of the IS-IS base model.

The IS-IS SR YANG module requires support for the base segment routing module [I-D.ietf-spring-sr-yang], which defines the global segment routing configuration independent of any specific routing protocol configuration, and support of IS-IS base model [I-D.ietf-isis-yang-isis-cfg] which defines basic IS-IS configuration and state.

The figure below describes the overall structure of the isis-sr YANG module:

```
 module: ietf-isis-sr
   augment /rt:routing/rt:control-plane-protocols
           /rt:control-plane-protocol/isis:isis:
     +--rw segment-routing
     |  +--rw enabled?    boolean
     |  +--rw bindings
     |     +--rw advertise
     |     |  +--rw policies*   string
     |     +--rw receive?     boolean
     +--rw protocol-srgb {sr:protocol-srgb}?
        +--rw srgb* [lower-bound upper-bound]
           +--rw lower-bound    uint32
           +--rw upper-bound    uint32
   augment /rt:routing/rt:control-plane-protocols
           /rt:control-plane-protocol/isis:isis/isis:interfaces
           /isis:interface:
     +--rw segment-routing
        +--rw adjacency-sid
           +--rw advertise-adj-group-sid* [group-id]
           |  +--rw group-id    uint32
           +--rw advertise-protection?      enumeration
   augment /rt:routing/rt:control-plane-protocols
           /rt:control-plane-protocol/isis:isis/isis:interfaces
           /isis:interface/isis:fast-reroute:
```

```
     +--rw ti-lfa {ti-lfa}?
        +--rw enable?   boolean
   augment /rt:routing/rt:control-plane-protocols
           /rt:control-plane-protocol/isis:isis/isis:interfaces
           /isis:interface/isis:fast-reroute/isis:lfa/isis:remote-lfa:
     +--rw use-segment-routing-path?   boolean {remote-lfa-sr}?
   augment /rt:routing/rt:control-plane-protocols
           /rt:control-plane-protocol/isis:isis/isis:interfaces
           /isis:interface/isis:adjacencies/isis:adjacency:
     +--ro adjacency-sid* [value]
        +--ro af?                     iana-rt-types:address-family
        +--ro value                   uint32
        +--ro weight?                 uint8
        +--ro protection-requested?   boolean
   augment /rt:routing/rt:control-plane-protocols
           /rt:control-plane-protocol/isis:isis/isis:database
           /isis:level-db/isis:lsp/isis:router-capabilities:
     +--ro sr-capability
     |  +--ro flags?           bits
     |  +--ro global-blocks
     |     +--ro global-block*
     |        +--ro range-size?    uint32
     |        +--ro sid-sub-tlv
     |           +--ro sid?   uint32
     +--ro sr-algorithms
     |  +--ro sr-algorithm*   uint8
     +--ro local-blocks
     |  +--ro local-block*
     |     +--ro range-size?    uint32
     |     +--ro sid-sub-tlv
     |        +--ro sid?   uint32
     +--ro srms-preference
        +--ro preference?   uint8
   augment /rt:routing/rt:control-plane-protocols
           /rt:control-plane-protocol/isis:isis/isis:database
           /isis:level-db/isis:lsp/isis:extended-is-neighbor
           /isis:neighbor:
     +--ro sid-list* [value]
        +--ro flags?         bits
        +--ro weight?        uint8
        +--ro neighbor-id?   isis:system-id
        +--ro value          uint32
   augment /rt:routing/rt:control-plane-protocols
           /rt:control-plane-protocol/isis:isis/isis:database
           /isis:level-db/isis:lsp/isis:mt-is-neighbor/isis:neighbor:
     +--ro sid-list* [value]
        +--ro flags?         bits
        +--ro weight?        uint8
```

```
        +--ro neighbor-id?   isis:system-id
        +--ro value          uint32
   augment /rt:routing/rt:control-plane-protocols
           /rt:control-plane-protocol/isis:isis/isis:database
           /isis:level-db/isis:lsp/isis:extended-ipv4-reachability
           /isis:prefixes:
     +--ro sid-list* [value]
        +--ro flags?       bits
        +--ro algorithm?   uint8
        +--ro value        uint32
   augment /rt:routing/rt:control-plane-protocols
           /rt:control-plane-protocol/isis:isis/isis:database
           /isis:level-db/isis:lsp/isis:mt-extended-ipv4-reachability
           /isis:prefixes:
     +--ro sid-list* [value]
        +--ro flags?       bits
        +--ro algorithm?   uint8
        +--ro value        uint32
   augment /rt:routing/rt:control-plane-protocols
           /rt:control-plane-protocol/isis:isis/isis:database
           /isis:level-db/isis:lsp/isis:ipv6-reachability/isis:prefixes:
     +--ro sid-list* [value]
        +--ro flags?       bits
        +--ro algorithm?   uint8
        +--ro value        uint32
   augment /rt:routing/rt:control-plane-protocols
           /rt:control-plane-protocol/isis:isis/isis:database
           /isis:level-db/isis:lsp/isis:mt-ipv6-reachability
           /isis:prefixes:
     +--ro sid-list* [value]
        +--ro flags?       bits
        +--ro algorithm?   uint8
        +--ro value        uint32
   augment /rt:routing/rt:control-plane-protocols
           /rt:control-plane-protocol/isis:isis/isis:database
           /isis:level-db/isis:lsp:
     +--ro segment-routing-bindings* [fec range]
        +--ro fec        string
        +--ro range      uint16
        +--ro flags?     bits
        +--ro binding
           +--ro prefix-sid
              +--ro sid-list* [value]
                 +--ro flags?       bits
                 +--ro algorithm?   uint8
                 +--ro value        uint32
```

3. IS-IS Segment Routing configuration

3.1. Segment Routing activation

Activation of segment-routing IS-IS is done by setting the "enable" leaf to true. This triggers advertisement of segment-routing extensions based on the configuration parameters that have been setup using the base segment routing module.

3.2. Advertising mapping server policy

The base segment routing module defines mapping server policies. By default, IS-IS will not advertise nor receive any mapping server entry. The IS-IS segment-routing module allows to advertise one or multiple mapping server policies through the "bindings/advertise/ policies" leaf-list. The "bindings/receive" leaf allows to enable the reception of mapping server entries.

3.3. IP Fast reroute

IS-IS SR model augments the fast-reroute container under interface. It brings the ability to activate TI-LFA (topology independent LFA) and also enhances remote LFA to use segment-routing tunneling instead of LDP.

4. IS-IS Segment Routing YANG Module

```
<CODE BEGINS> file "ietf-isis-sr@2018-06-25.yang"
module ietf-isis-sr {
  namespace "urn:ietf:params:xml:ns:"
          + "yang:ietf-isis-sr";
  prefix isis-sr;


  import ietf-routing {
    prefix "rt";
  }

  import ietf-segment-routing-common {
  prefix "sr-cmn";
  }


  import ietf-segment-routing {
  prefix "sr";
  }

  import ietf-isis {
```

```
    prefix "isis";
    }

    import iana-routing-types {
      prefix "iana-rt-types";
    }

    organization
     "IETF LSR - LSR Working Group";

    contact
      "WG List:  <mailto:lsr@ietf.org>

      Editor:    Stephane Litkowski
                 <mailto:stephane.litkowski@orange.com>

      Author:    Acee Lindem
                 <mailto:acee@cisco.com>
      Author:    Yingzhen Qu
                 <mailto:yiqu@cisco.com>
      Author:    Pushpasis Sarkar
                 <mailto:pushpasis.ietf@gmail.com>
      Author:    Ing-Wher Chen
                 <mailto:ichen@kuatrotech.com>
      Author:    Jeff Tantsura
                 <mailto:jefftant.ietf@gmail.com>
      ";

    description
      "The YANG module defines a generic configuration model for
      Segment routing ISIS extensions common across all of the vendor
      implementations.";

    revision 2018-06-25 {
    description
      "Initial revision.";
    reference "RFC XXXX";
    }


    /* Identities */


    /* Features */

    feature remote-lfa-sr {
      description
       "Enhance rLFA to use SR path.";
```

```
    }

    feature ti-lfa {
      description
       "Enhance IPFRR with ti-lfa
       support";
    }

    /* Groupings */


    grouping sid-sub-tlv {
      description "SID/Label sub-TLV grouping.";
      container sid-sub-tlv {
        description
          "Used to advertise the SID/Label associated with a
        prefix or adjacency.";
        leaf sid {
          type uint32;
        description
          "Segment Identifier (SID) - A 20 bit label or
           32 bit SID.";
        }
      }
    }

    grouping sr-capability {
      description
        "SR capability grouping.";
      container sr-capability {
        description
          "Segment Routing capability.";
        leaf flags {
          type bits {
            bit mpls-ipv4 {
              position 0;
              description
                "If set, then the router is capable of
                 processing SR MPLS encapsulated IPv4 packets
                 on all interfaces.";
            }
            bit mpls-ipv6 {
              position 1;
              description
                "If set, then the router is capable of
                 processing SR MPLS encapsulated IPv6 packets
                 on all interfaces.";
            }
```

```
          }
          description
            "Flags.";
        }
        container global-blocks {
          description
            "Segment Routing Global Blocks.";
          list global-block {
            description "Segment Routing Global Block.";
            leaf range-size {
              type uint32;
              description "The SID range.";
            }
            uses sid-sub-tlv;
          }
        }
      }
    }

    grouping sr-algorithm {
      description
        "SR algorithm grouping.";
      container sr-algorithms {
        description "All SR algorithms.";
        leaf-list sr-algorithm {
          type uint8;
          description
            "The Segment Routing (SR) algorithms that the router is
             currently using.";
        }
      }
    }

    grouping srlb {
      description
        "SR Local Block grouping.";
      container local-blocks {
        description "List of SRLBs.";
        list local-block {
          description "Segment Routing Local Block.";
          leaf range-size {
            type uint32;
            description "The SID range.";
          }
          uses sid-sub-tlv;
        }
      }
    }
```

```
    grouping srms-preference {
      description "The SRMS preference TLV is used to advertise
                   a preference associated with the node that acts
                   as an SR Mapping Server.";
      container srms-preference {
        description "SRMS Preference TLV.";
        leaf preference {
          type uint8 {
            range "0 .. 255";
          }
          description "SRMS preference TLV, vlaue from 0 to 255.";
        }
      }
    }

    grouping adjacency-state {
      description
        "This group will extend adjacency state.";
      list adjacency-sid {
        key value;
        config false;
        leaf af {
          type iana-rt-types:address-family;
          description
            "Address-family associated with the
             segment ID";
        }
        leaf value {
          type uint32;
          description
            "Value of the Adj-SID.";
        }
        leaf weight {
          type uint8;
          description
            "Weight associated with
            the adjacency SID.";
        }
        leaf protection-requested {
          type boolean;
          description
            "Describe if the adjacency SID
             must be protected.";
        }
        description
          "List of adjacency Segment IDs.";
      }
    }
```

```
    grouping prefix-segment-id {
      description
        "This group defines segment routing extensions
         for prefixes.";

      list sid-list {
        key value;

        leaf flags {
          type bits {
            bit readvertisment {
              position 7;
              description
                "If set, then the prefix to
                 which this Prefix-SID is attached,
                 has been propagated by the
                 router either from another level
                 or from redistribution.";
            }
            bit php {
              position 5;
              description
                "If set, then the penultimate hop MUST NOT
                 pop the Prefix-SID before delivering the packet
                 to the node that advertised the Prefix-SID.";
            }
            bit explicit-null {
              position 4;
              description
                "If set, any upstream neighbor of
                 the Prefix-SID originator MUST replace
                 the Prefix-SID with a
                 Prefix-SID having an
                 Explicit-NULL value (0 for IPv4 and 2 for
                 IPv6) before forwarding the packet.";
            }
            bit value {
              position 3;
              description
                "If set, then the Prefix-SID carries a
                 value (instead of an index).
                 By default the flag is UNSET.";
            }
            bit local {
              position 2;
              description
                "If set, then the value/index carried by
                 the Prefix-SID has local significance.
```

```
                By default the flag is UNSET.";
            }
          }
          description
            "Describes flags associated with the
             segment ID.";
        }

        leaf algorithm {
          type uint8;
          description
            "Algorithm to be used for path computation.";
        }
        leaf value {
          type uint32;
          description
           "Value of the prefix-SID.";
        }
        description
          "List of segments.";
      }
    }

    grouping adjacency-segment-id {
      description
        "This group defines segment routing extensions
         for adjacencies.";

      list sid-list {
        key value;

        leaf flags {
          type bits {
            bit address-family {
              position 7;
              description
                "If unset, then the Adj-SID refers
                 to an adjacency with outgoing IPv4 encapsulation.
                 If set then the Adj-SID refers to an adjacency
                 with outgoing IPv6 encapsulation.";
            }
            bit backup {
              position 6;
              description
                "If set, the Adj-SID refers to an
                 adjacency being protected
                 (e.g.: using IPFRR or MPLS-FRR)";
            }
```

```
            bit value {
              position 5;
              description
                "If set, then the SID carries a
                 value (instead of an index).
                 By default the flag is SET.";
            }
            bit local {
              position 4;
              description
                "If set, then the value/index carried by
                 the SID has local significance.
                 By default the flag is SET.";
            }
            bit set {
              position 3;
              description
                "When set, the S-Flag indicates that the
                 Adj-SID refers to a set of adjacencies.";
            }
            bit persistent {
              position 2;
              description
                "When set, the P-Flag indicates that the
                 Adj-SID is persistently allocated.";
            }
          }

          description
            "Describes flags associated with the
             segment ID.";
        }
        leaf weight {
          type uint8;
          description
            "The value represents the weight of the Adj-SID
             for the purpose of load balancing.";
        }
        leaf neighbor-id {
          type isis:system-id;
          description
            "Describes the system ID of the neighbor
             associated with the SID value. This is only
             used on LAN adjacencies.";
        }
        leaf value {
          type uint32;
          description
```

```
            "Value of the Adj-SID.";
        }
        description
          "List of segments.";
      }
    }

    grouping segment-routing-binding-tlv {
      list segment-routing-bindings {
        key "fec range";

        leaf fec {
          type string;
          description
          "IP (v4 or v6) range to be bound to SIDs.";
        }

        leaf range {
          type uint16;
          description
            "Describes number of elements to assign
             a binding to.";
        }

        leaf flags {
          type bits {
            bit address-family {
              position 7;
              description
                "If unset, then the Prefix FEC
                 carries an IPv4 Prefix.
                 If set then the Prefix FEC carries an
                 IPv6 Prefix.";
            }
            bit mirror {
              position 6;
              description
                "Set if the advertised SID/path
                 corresponds to a mirrored context.";
            }
            bit flooding {
              position 5;
              description
                "If the S bit is set(1),
                 the IS-IS Router CAPABILITY TLV
                 MUST be flooded across the entire routing domain.
                 If the S bit is
                 not set(0), the TLV MUST NOT be leaked between levels.
```

```
                This bit MUST NOT be altered during the TLV leaking.";
            }
            bit down {
              position 4;
              description
                "When the IS-IS Router CAPABILITY TLV is
                 leaked from level-2 to level-1, the D bit
                 MUST be set.  Otherwise, this bit MUST
                 be clear.  IS-IS Router capability TLVs
                 with the D bit set MUST NOT
                 be leaked from level-1 to level-2.
                 This is to prevent TLV looping.";
            }
            bit attached {
              position 3;
              description
                "The originator of the SID/Label Binding TLV MAY set the
                 A bit in order to signal that the prefixes and SIDs
                 advertised in the SID/Label Binding TLV are directly
                 connected to their originators.";
            }

          }
          description
            "Flags of the binding.";
        }

        container binding {
          container prefix-sid {
            uses prefix-segment-id;
            description
              "Binding prefix SID to the range.";
          }
          description
            "Bindings associated with the range.";
        }

        description
          "This container describes list of SID/Label bindings.
           ISIS reference is TLV 149.";
      }
      description
        "Defines binding TLV for database.";
    }

    /* Cfg */

    augment "/rt:routing/" +
```

```
           "rt:control-plane-protocols/rt:control-plane-protocol"+
           "/isis:isis" {
      when "/rt:routing/rt:control-plane-protocols/"+
           "rt:control-plane-protocol/rt:type = 'isis:isis'" {
        description
          "This augment ISIS routing protocol when used";
      }
      description
        "This augments ISIS protocol configuration
         with segment routing.";

      uses sr:controlplane-cfg;
      container protocol-srgb {
        if-feature sr:protocol-srgb;
        uses sr-cmn:srgb-cfg;
        description
          "Per-protocol SRGB.";
      }
    }

    augment "/rt:routing/" +
           "rt:control-plane-protocols/rt:control-plane-protocol"+
           "/isis:isis/isis:interfaces/isis:interface" {
      when "/rt:routing/rt:control-plane-protocols/"+
           "rt:control-plane-protocol/rt:type = 'isis:isis'" {
        description
          "This augment ISIS routing protocol when used";
      }
      description
        "This augments ISIS protocol configuration
         with segment routing.";

      uses sr:igp-interface-cfg;
    }

    augment "/rt:routing/" +
           "rt:control-plane-protocols/rt:control-plane-protocol"+
           "/isis:isis/isis:interfaces/isis:interface"+
           "/isis:fast-reroute" {
      when "/rt:routing/rt:control-plane-protocols/"+
           "rt:control-plane-protocol/rt:type = 'isis:isis'" {
        description
          "This augment ISIS routing protocol when used";
      }
      description
        "This augments ISIS IP FRR with TILFA.";

      container ti-lfa {
```

```
        if-feature ti-lfa;
        leaf enable {
          type boolean;
          description
            "Enables TI-LFA computation.";
        }
        description
          "TILFA configuration.";
      }
    }

    augment "/rt:routing/" +
            "rt:control-plane-protocols/rt:control-plane-protocol"+
            "/isis:isis/isis:interfaces/isis:interface"+
            "/isis:fast-reroute/isis:lfa/isis:remote-lfa" {
      when "/rt:routing/rt:control-plane-protocols/"+
           "rt:control-plane-protocol/rt:type = 'isis:isis'" {
        description
          "This augment ISIS routing protocol when used";
      }
      description
        "This augments ISIS remoteLFA config with
         use of segment-routing path.";

      leaf use-segment-routing-path {
        if-feature remote-lfa-sr;
        type boolean;
        description
          "force remote LFA to use segment routing
           path instead of LDP path.";
      }
    }

    /* Operational states */

    augment "/rt:routing/" +
            "rt:control-plane-protocols/rt:control-plane-protocol"+
            "/isis:isis/isis:interfaces/isis:interface" +
            "/isis:adjacencies/isis:adjacency" {
      when "/rt:routing/rt:control-plane-protocols/"+
           "rt:control-plane-protocol/rt:type = 'isis:isis'" {
        description
          "This augment ISIS routing protocol when used";
      }
      description
        "This augments ISIS protocol configuration
         with segment routing.";
```

```
      uses adjacency-state;
    }

    augment "/rt:routing/" +
            "rt:control-plane-protocols/rt:control-plane-protocol"+
            "/isis:isis/isis:database/isis:level-db/isis:lsp"+
            "/isis:router-capabilities" {
      when "/rt:routing/rt:control-plane-protocols/"+
           "rt:control-plane-protocol/rt:type = 'isis:isis'" {
        description
          "This augment ISIS routing protocol when used";
      }
      description
        "This augments ISIS protocol LSDB router capability.";

      uses sr-capability;
      uses sr-algorithm;
      uses srlb;
      uses srms-preference;
    }

    augment "/rt:routing/" +
            "rt:control-plane-protocols/rt:control-plane-protocol"+
            "/isis:isis/isis:database/isis:level-db/isis:lsp"+
            "/isis:extended-is-neighbor/isis:neighbor" {
      when "/rt:routing/rt:control-plane-protocols/"+
           "rt:control-plane-protocol/rt:type = 'isis:isis'" {
        description
          "This augment ISIS routing protocol when used";
      }
      description
        "This augments ISIS protocol LSDB neighbor.";
         uses adjacency-segment-id;
    }

    augment "/rt:routing/" +
            "rt:control-plane-protocols/rt:control-plane-protocol"+
            "/isis:isis/isis:database/isis:level-db/isis:lsp"+
            "/isis:mt-is-neighbor/isis:neighbor" {
      when "/rt:routing/rt:control-plane-protocols/"+
           "rt:control-plane-protocol/rt:type = 'isis:isis'" {
        description
          "This augment ISIS routing protocol when used";
      }
      description
        "This augments ISIS protocol LSDB neighbor.";
         uses adjacency-segment-id;
    }
```

```
    augment "/rt:routing/" +
            "rt:control-plane-protocols/rt:control-plane-protocol"+
            "/isis:isis/isis:database/isis:level-db/isis:lsp"+
            "/isis:extended-ipv4-reachability/isis:prefixes" {
      when "/rt:routing/rt:control-plane-protocols/"+
           "rt:control-plane-protocol/rt:type = 'isis:isis'" {
        description
          "This augment ISIS routing protocol when used";
      }
      description
        "This augments ISIS protocol LSDB prefix.";
         uses prefix-segment-id;
    }

    augment "/rt:routing/" +
            "rt:control-plane-protocols/rt:control-plane-protocol"+
            "/isis:isis/isis:database/isis:level-db/isis:lsp"+
            "/isis:mt-extended-ipv4-reachability/isis:prefixes" {
      when "/rt:routing/rt:control-plane-protocols/"+
           "rt:control-plane-protocol/rt:type = 'isis:isis'" {
        description
          "This augment ISIS routing protocol when used";
      }
      description
        "This augments ISIS protocol LSDB prefix.";
      uses prefix-segment-id;
    }

    augment "/rt:routing/" +
            "rt:control-plane-protocols/rt:control-plane-protocol"+
            "/isis:isis/isis:database/isis:level-db/isis:lsp"+
            "/isis:ipv6-reachability/isis:prefixes" {
      when "/rt:routing/rt:control-plane-protocols/"+
           "rt:control-plane-protocol/rt:type = 'isis:isis'" {
        description
          "This augment ISIS routing protocol when used";
      }
      description
        "This augments ISIS protocol LSDB prefix.";
      uses prefix-segment-id;
    }

    augment "/rt:routing/" +
            "rt:control-plane-protocols/rt:control-plane-protocol"+
            "/isis:isis/isis:database/isis:level-db/isis:lsp"+
            "/isis:mt-ipv6-reachability/isis:prefixes" {
      when "/rt:routing/rt:control-plane-protocols/"+
           "rt:control-plane-protocol/rt:type = 'isis:isis'" {
```

```
        description
          "This augment ISIS routing protocol when used";
      }
      description
        "This augments ISIS protocol LSDB prefix.";
      uses prefix-segment-id;
    }

    augment "/rt:routing/" +
            "rt:control-plane-protocols/rt:control-plane-protocol"+
            "/isis:isis/isis:database/isis:level-db/isis:lsp" {
      when "/rt:routing/rt:control-plane-protocols/"+
           "rt:control-plane-protocol/rt:type = 'isis:isis'" {
        description
          "This augment ISIS routing protocol when used";
      }
      description
        "This augments ISIS protocol LSDB.";
      uses segment-routing-binding-tlv;
    }


    /* Notifications */

  }
  <CODE ENDS>
```

5. Security Considerations

Configuration and state data defined in this document are designed to be accessed via the NETCONF protocol [RFC6241].

As IS-IS is an IGP protocol (critical piece of the network), ensuring stability and security of the protocol is mandatory for the network service.

Authors recommends to implement NETCONF access control model ([RFC6536]) to restrict access to all or part of the configuration to specific users.

6. Contributors

Authors would like to thank Derek Yeung, Acee Lindem, Yi Yang for their major contributions to the draft.

7. Acknowledgements

TBD.

8. IANA Considerations

The IANA is requested to assign two new URIs from the IETF XML registry ([RFC3688]). Authors are suggesting the following URI:

```
URI: urn:ietf:params:xml:ns:yang:ietf-isis-sr
Registrant Contact: IS-IS WG
XML: N/A, the requested URI is an XML namespace
```

This document also requests one new YANG module name in the YANG Module Names registry ([RFC6020]) with the following suggestion :

```
name: ietf-isis-sr
namespace: urn:ietf:params:xml:ns:yang:ietf-isis-sr
prefix: isis-sr
reference: RFC XXXX
```

9. Change log for ietf-isis-sr YANG module

9.1. From version -03 to version -04

- Fixed yang module indentations.

9.2. From version -02 to version -03

- Change address-family type according to routing types.

9.3. From isis-sr document version -01 to version -02

- NMDA compliancy.
- Added SRLB in configuration and LSDB.
- Added SR capability in LSDB.
- Added SR algorithms in LSDB.
- Added SRMS preference in LSDB.
- Alignment with iana-rt-types module.
- Align binding SID with draft-ietf-isis-segment-routing-extensions-13.

9.4. From isis-sr document version -00 to version -01

- Added P-Flag in Adj-SID.

9.5. From isis document version -12 to isis-sr document version -00

- Separate document for IS-IS SR extensions.

9.6. From isis document version -12 to version -13

- Align with new segment routing common module.

9.7. From isis document version -09 to version -11

- Fixed XPATH in 'when' expressions.

9.8. From isis document version -08 to version -09

- Align to draft-ietf-netmod-routing-cfg-23.

9.9. From isis document version -07 to version -08

- Align to draft-ietf-netmod-routing-cfg-21.

10. Normative References

[I-D.ietf-isis-segment-routing-extensions]
Previdi, S., Ginsberg, L., Filsfils, C., Bashandy, A., Gredler, H., Litkowski, S., Decraene, B., and J. Tantsura, "IS-IS Extensions for Segment Routing", draft-ietf-isis-segment-routing-extensions-15 (work in progress), December 2017.

[I-D.ietf-isis-yang-isis-cfg]
Litkowski, S., Yeung, D., Lindem, A., Zhang, Z., and L. Lhotka, "YANG Data Model for IS-IS protocol", draft-ietf-isis-yang-isis-cfg-19 (work in progress), November 2017.

[I-D.ietf-spring-sr-yang]
Litkowski, S., Qu, Y., Sarkar, P., and J. Tantsura, "YANG Data Model for Segment Routing", draft-ietf-spring-sr-yang-08 (work in progress), December 2017.

[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>.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, DOI 10.17487/RFC6536, March 2012, <https://www.rfc-editor.org/info/rfc6536>.

[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>.

Authors' Addresses

Stephane Litkowski
Orange

Email: stephane.litkowski@orange.com

Yinghzen Qu
Huawei

Email: yingzhen.qu@huawei.com

Pushpasis Sarkar
Individual

Email: pushpasis.ietf@gmail.com

Ing-Wher Chen
Jabil

Email: ing-wher_chen@jabil.com

Jeff Tantsura
Individual

Email: jefftant.ietf@gmail.com

IS-IS Working Group
Internet-Draft
Intended status: Standards Track
Expires: July 25, 2019

S. Litkowski
Orange
D. Yeung
Arrcus, Inc
A. Lindem
Cisco Systems
J. Zhang
Juniper Networks
L. Lhotka
CZ.NIC
January 21, 2019

YANG Data Model for IS-IS Protocol
draft-ietf-isis-yang-isis-cfg-34

Abstract

This document defines a YANG data model that can be used to configure and manage IS-IS protocol on network elements.

Requirements Language

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

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 http://datatracker.ietf.org/drafts/current/.

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

This Internet-Draft will expire on July 25, 2019.

Table of Contents

1. Introduction 3
2. Design of the Data Model 3
2.1. IS-IS Configuration 9
2.2. Multi-topology Parameters 10
2.3. Per-Level Parameters 10
2.4. Per-Interface Parameters 12
2.5. Authentication Parameters 19
2.6. IGP/LDP synchronization 20
2.7. ISO parameters 20
2.8. IP FRR 20
2.9. Operational States 21
3. RPC Operations 21
4. Notifications 21
5. Interaction with Other YANG Modules 23
6. IS-IS YANG Module 23
7. Security Considerations 104
8. Contributors 105
9. IANA Considerations 105
10. Change log for ietf-isis YANG module 105
10.1. From version -29 to version -30 105
10.2. From version -28 to version -29 106
10.3. From version -27 to version -28 106
10.4. From version -26 to version -27 106
10.5. From version -25 to version -26 106
10.6. From version -24 to version -25 106
10.7. From version -22 to version -24 107
10.8. From version -21 to version -22 107
10.9. From version -20 to version -21 108
10.10. From version -19 to version -20 108
10.11. From version -18 to version -19 108
10.12. From version -17 to version -18 108

10.13. From version -16 to version -17 108
10.14. From version -15 to version -16 108
10.15. From version -14 to version -15 109
10.16. From version -13 to version -14 109
10.17. From version -12 to version -13 109
10.18. From version -09 to version -12 109
10.19. From version -08 to version -09 110
10.20. From version -07 to version -08 110
10.21. From version -05 to version -07 110
10.22. From version -03 to version -05 110
10.23. From version -02 to version -03 111
10.24. From version -01 to version -02 111
10.25. From version -00 to version -01 111
11. References 112
11.1. Normative References 112
11.2. Informative References 116
Appendix A. Example of IS-IS configuration in XML 116
Authors' Addresses 118

1. Introduction

This document defines a YANG ([RFC7950]) data model for IS-IS routing protocol.

The data model covers configuration of an IS-IS routing protocol instance as well as operational states.

A simplified tree representation of the data model is presented in Section 2. Tree diagrams used in this document follow the notation defined in [RFC8340].

The module is designed as per NMDA (Network Management Datastore Architecture) [RFC8342].

2. Design of the Data Model

The IS-IS YANG module augments the "control-plane-protocol" list in ietf-routing module (defined in [RFC8349]) with specific IS-IS parameters.

The figure below describes the overall structure of the isis YANG module:

```
module: ietf-isis
  augment /rt:routing/rt:ribs/rt:rib/rt:routes/rt:route:
    +--ro metric?       uint32
    +--ro tag*          uint64
```

```
       +--ro route-type?   enumeration
     augment /if:interfaces/if:interface:
       +--rw clns-mtu?   uint16
     augment
   /rt:routing/rt:control-plane-protocols/rt:control-plane-protocol:
       +--rw isis
          +--rw enable?                    boolean {admin-control}?
          +--rw level-type?                level
          +--rw system-id?                 system-id
          +--rw maximum-area-addresses?    uint8
          |       {maximum-area-addresses}?
          +--rw area-address*              area-address
          +--rw lsp-mtu?                   uint16
          +--rw lsp-lifetime?              uint16
          +--rw lsp-refresh?               rt-types:timer-value-seconds16
          |       {lsp-refresh}?
          +--rw poi-tlv?                   boolean {poi-tlv}?
          +--rw graceful-restart {graceful-restart}?
          |  +--rw enable?              boolean
          |  +--rw restart-interval?    rt-types:timer-value-seconds16
          |  +--rw helper-enable?       boolean
          +--rw nsr {nsr}?
          |  +--rw enable?   boolean
          +--rw node-tags {node-tag}?
          |  +--rw node-tag* [tag]
          |        ...
          +--rw metric-type
          |  +--rw value?     enumeration
          |  +--rw level-1
          |  |     ...
          |  +--rw level-2
          |        ...
          +--rw default-metric
          |  +--rw value?     wide-metric
          |  +--rw level-1
          |  |     ...
          |  +--rw level-2
          |        ...
          +--rw auto-cost {auto-cost}?
          |  +--rw enable?                 boolean
          |  +--rw reference-bandwidth?    uint32
          +--rw authentication
          |  +--rw (authentication-type)?
          |  |     ...
          |  +--rw level-1
          |  |     ...
          |  +--rw level-2
          |        ...
```

```
          +--rw address-families {nlpid-control}?
          |  +--rw address-family-list* [address-family]
          |        ...
          +--rw mpls
          |  +--rw te-rid {te-rid}?
          |  |     ...
          |  +--rw ldp
          |        ...
          +--rw spf-control
          |  +--rw paths?            uint16 {max-ecmp}?
          |  +--rw ietf-spf-delay {ietf-spf-delay}?
          |        ...
          +--rw fast-reroute {fast-reroute}?
          |  +--rw lfa {lfa}?
          +--rw preference
          |  +--rw (granularity)?
          |        ...
          +--rw overload
          |  +--rw status?   boolean
          +--rw overload-max-metric {overload-max-metric}?
          |  +--rw timeout?   rt-types:timer-value-seconds16
          +--ro spf-log
          |  +--ro event* [id]
          |        ...
          +--ro lsp-log
          |  +--ro event* [id]
          |        ...
          +--ro hostnames
          |  +--ro hostname* [system-id]
          |        ...
          +--ro database
          |  +--ro levels* [level]
          |        ...
          +--ro local-rib
          |  +--ro route* [prefix]
          |        ...
          +--ro system-counters
          |  +--ro level* [level]
          |        ...
          +--ro protected-routes
          |  +--ro address-family-stats*
          |        [address-family prefix alternate]
          |        ...
          +--ro unprotected-routes
          |  +--ro address-family-stats* [address-family prefix]
          |        ...
          +--ro protection-statistics* [frr-protection-method]
          |  +--ro frr-protection-method string
```

```
          |  +--ro address-family-stats* [address-family]
          |        ...
          +--rw topologies {multi-topology}?
          |  +--rw topology* [name]
          |        ...
          +--rw interfaces
             +--rw interface* [name]
                   ...

     rpcs:
       +---x clear-adjacency
       |  +---w input
       |     +---w routing-protocol-instance-name    ->
   /rt:routing/control-plane-protocols/control-plane-protocol/name
       |     +---w level?                            level
       |     +---w interface?                        if:interface-ref
       +---x clear-database
          +---w input
             +---w routing-protocol-instance-name    ->
   /rt:routing/control-plane-protocols/control-plane-protocol/name
             +---w level?                             level

     notifications:
       +---n database-overload
       |  +--ro routing-protocol-name?   ->
   /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro overload?                enumeration
       +---n lsp-too-large
       |  +--ro routing-protocol-name?   ->
   /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro pdu-size?                uint32
       |  +--ro lsp-id?                  lsp-id
       +---n if-state-change
       |  +--ro routing-protocol-name?   ->
   /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro state?                   if-state-type
       +---n corrupted-lsp-detected
       |  +--ro routing-protocol-name?   ->
   /rt:routing/control-plane-protocols/control-plane-protocol/name
```

```
       |  +--ro isis-level?              level
       |  +--ro lsp-id?                  lsp-id
       +---n attempt-to-exceed-max-sequence
       |  +--ro routing-protocol-name?   ->
  /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro lsp-id?                  lsp-id
       +---n id-len-mismatch
       |  +--ro routing-protocol-name?   ->
  /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro pdu-field-len?           uint8
       |  +--ro raw-pdu?                 binary
       +---n max-area-addresses-mismatch
       |  +--ro routing-protocol-name?   ->
  /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro max-area-addresses?      uint8
       |  +--ro raw-pdu?                 binary
       +---n own-lsp-purge
       |  +--ro routing-protocol-name?   ->
  /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro lsp-id?                  lsp-id
       +---n sequence-number-skipped
       |  +--ro routing-protocol-name?   ->
  /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro lsp-id?                  lsp-id
       +---n authentication-type-failure
       |  +--ro routing-protocol-name?   ->
  /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
```

```
       |  +--ro raw-pdu?                 binary
       +---n authentication-failure
       |  +--ro routing-protocol-name?   ->
     /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro raw-pdu?                 binary
       +---n version-skew
       |  +--ro routing-protocol-name?   ->
     /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro protocol-version?        uint8
       |  +--ro raw-pdu?                 binary
       +---n area-mismatch
       |  +--ro routing-protocol-name?   ->
     /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro raw-pdu?                 binary
       +---n rejected-adjacency
       |  +--ro routing-protocol-name?   ->
     /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro raw-pdu?                 binary
       |  +--ro reason?                  string
       +---n protocols-supported-mismatch
       |  +--ro routing-protocol-name?   ->
     /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro raw-pdu?                 binary
       |  +--ro protocols*               uint8
       +---n lsp-error-detected
       |  +--ro routing-protocol-name?   ->
     /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
```

```
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro lsp-id?                  lsp-id
       |  +--ro raw-pdu?                 binary
       |  +--ro error-offset?            uint32
       |  +--ro tlv-type?                uint8
       +---n adjacency-state-change
       |  +--ro routing-protocol-name?   ->
  /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro neighbor?                string
       |  +--ro neighbor-system-id?      system-id
       |  +--ro state?                   adj-state-type
       |  +--ro reason?                  string
       +---n lsp-received
       |  +--ro routing-protocol-name?   ->
  /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro lsp-id?                  lsp-id
       |  +--ro sequence?                uint32
       |  +--ro received-timestamp?      yang:timestamp
       |  +--ro neighbor-system-id?      system-id
       +---n lsp-generation
          +--ro routing-protocol-name?   ->
  /rt:routing/control-plane-protocols/control-plane-protocol/name
          +--ro isis-level?               level
          +--ro lsp-id?                   lsp-id
          +--ro sequence?                 uint32
          +--ro send-timestamp?           yang:timestamp
```

2.1. IS-IS Configuration

The IS-IS configuration is divided in:

- Global parameters.
- Per interface configuration (see Section 2.4).

Additional modules may be created to support any additional parameters. These additional modules MUST augment the ietf-isis module.

The model implements features, thus some of the configuration statement becomes optional. As an example, the ability to control the administrative state of a particular IS-IS instance is optional. By advertising the feature "admin-control", a device communicates to the client that it supports the ability to shut down a particular IS-IS instance.

The global configuration contains usual IS-IS parameters such as lsp-mtu, lsp-lifetime, lsp-refresh, default-metric...

2.2. Multi-topology Parameters

The model supports multi-topology (MT) IS-IS as defined in [RFC5120].

The "topologies" container is used to enable support of MT extensions.

The "name" used in the topology list should refer to an existing RIB of the device.

Some specific parameters could be defined on a per topology basis both at global level and at interface level: for example, an interface metric can be defined per topology.

Multiple address families (like IPv4 or IPv6) can also be activated within the default topology. This can be achieved using the address-families container (requiring "nlpid-control" feature to be advertised).

2.3. Per-Level Parameters

Some parameters allow a per level configuration. In this case, the parameter is modeled as a container with three configuration locations:

- a top-level container: corresponds to level-1-2, so the configuration applies to both levels.
- a level-1 container: corresponds to level-1 specific parameters.
- a level-2 container: corresponds to level-2 specific parameters.

```
          +--rw priority
          |  +--rw value?     uint8
          |  +--rw level-1
          |  |  +--rw value?   uint8
          |  +--rw level-2
          |     +--rw value?   uint8
```

Example:

```
        <priority>
            <value>250</value>
            <level-1>
                <value>100</value>
            </level-1>
            <level-2>
                <value>200</value>
            </level-2>
        </priority>
```

An implementation SHOULD prefer a level specific parameter over a level-all parameter. As example, if the priority is 100 for the level-1, 200 for the level-2 and 250 for the top-level configuration, the implementation should use 100 for the level-1 and 200 for the level-2.

Some parameters like "overload bit" and "route preference" are not modeled to support a per level configuration. If an implementation supports per level configuration for such parameter, this implementation SHOULD augment the current model by adding both level-1 and level-2 containers and SHOULD reuse existing configuration groupings.

Example of augmentation:

```
augment "/rt:routing/" +
        "rt:control-plane-protocols/rt:control-plane-protocol"+
        "/isis:isis/isis:overload" {
        when "rt:type = 'isis:isis'" {
          description
           "This augment IS-IS routing protocol when used";
        }
        description
          "This augments IS-IS overload configuration
           with per level configuration.";

        container level-1 {
          uses isis:overload-global-cfg;
          description
            "Level 1 configuration.";
        }
        container level-2 {
          uses isis:overload-global-cfg;
          description
            "Level 2 configuration.";
        }
}
```

If an implementation does not support per level configuration for a parameter modeled with per level configuration, the implementation SHOULD advertise a deviation to announce the non-support of the level-1 and level-2 containers.

Finally, if an implementation supports per level configuration but does not support the level-1-2 configuration, it SHOULD also advertise a deviation.

2.4. Per-Interface Parameters

The per-interface section of the IS-IS instance describes the interface specific parameters.

The interface is modeled as a reference to an existing interface defined in the "ietf-interfaces" YANG model ([RFC8343].

Each interface has some interface-specific parameters that may have a different per level value as described in previous section. An interface-specific parameter always overrides an IS-IS global parameter.

Some parameters like hello-padding are defined as containers to allow easy extension by vendor specific modules.

```
          +--rw interfaces
             +--rw interface* [name]
                +--rw name                          if:interface-ref
                +--rw level-type?                   level
                +--rw lsp-pacing-interval?
                |       rt-types:timer-value-milliseconds
                +--rw lsp-retransmit-interval?
                |       rt-types:timer-value-seconds16
                +--rw passive?                      boolean
                +--rw csnp-interval?
                |       rt-types:timer-value-seconds16
                +--rw hello-padding
                |  +--rw enable?   boolean
                +--rw mesh-group-enable?       mesh-group-state
                +--rw mesh-group?              uint8
                +--rw interface-type?          interface-type
                +--rw enable?                  boolean {admin-control}?
                +--rw tag*                     uint32 {prefix-tag}?
                +--rw tag64*                   uint64 {prefix-tag64}?
                +--rw node-flag?               boolean {node-flag}?
                +--rw hello-authentication
                |  +--rw (authentication-type)?
                |  |  +--:(key-chain) {key-chain}?
                |  |  |  +--rw key-chain?
                |  |  |           key-chain:key-chain-ref
                |  |  +--:(password)
                |  |     +--rw key?                string
                |  |     +--rw crypto-algorithm?   identityref
                |  +--rw level-1
                |  |  +--rw (authentication-type)?
                |  |     +--:(key-chain) {key-chain}?
                |  |     |  +--rw key-chain?
                |  |     |         key-chain:key-chain-ref
                |  |     +--:(password)
                |  |        +--rw key?                string
                |  |        +--rw crypto-algorithm?   identityref
                |  +--rw level-2
                |     +--rw (authentication-type)?
                |        +--:(key-chain) {key-chain}?
                |        |  +--rw key-chain?
                |        |           key-chain:key-chain-ref
                |        +--:(password)
                |           +--rw key?                string
                |           +--rw crypto-algorithm?   identityref
                +--rw hello-interval
                |  +--rw value?     rt-types:timer-value-seconds16
                |  +--rw level-1
                |  |  +--rw value?   rt-types:timer-value-seconds16
```

```
|  +--rw level-2
|     +--rw value?   rt-types:timer-value-seconds16
+--rw hello-multiplier
|  +--rw value?     uint16
|  +--rw level-1
|  |  +--rw value?   uint16
|  +--rw level-2
|     +--rw value?   uint16
+--rw priority
|  +--rw value?     uint8
|  +--rw level-1
|  |  +--rw value?   uint8
|  +--rw level-2
|     +--rw value?   uint8
+--rw metric
|  +--rw value?     wide-metric
|  +--rw level-1
|  |  +--rw value?   wide-metric
|  +--rw level-2
|     +--rw value?   wide-metric
+--rw bfd {bfd}?
|  +--rw enable?                              boolean
|  +--rw local-multiplier?                    multiplier
|  +--rw (interval-config-type)?
|     +--:(tx-rx-intervals)
|     |  +--rw desired-min-tx-interval?    uint32
|     |  +--rw required-min-rx-interval?   uint32
|     +--:(single-interval) {single-minimum-interval}?
|        +--rw min-interval?                uint32
+--rw address-families {nlpid-control}?
|  +--rw address-family-list* [address-family]
|     +--rw address-family iana-rt-types:address-family
+--rw mpls
|  +--rw ldp
|     +--rw igp-sync?   boolean {ldp-igp-sync}?
+--rw fast-reroute {fast-reroute}?
|  +--rw lfa {lfa}?
|     +--rw candidate-enable?   boolean
|     +--rw enable?             boolean
|     +--rw remote-lfa {remote-lfa}?
|     |  +--rw enable?   boolean
|     +--rw level-1
|     |  +--rw candidate-enable?   boolean
|     |  +--rw enable?             boolean
|     |  +--rw remote-lfa {remote-lfa}?
|     |     +--rw enable?   boolean
|     +--rw level-2
|        +--rw candidate-enable?   boolean
|
```

```
|        +--rw enable?             boolean
|        +--rw remote-lfa {remote-lfa}?
|           +--rw enable?   boolean
+--ro adjacencies
|  +--ro adjacency* []
|     +--ro neighbor-sys-type?              level
|     +--ro neighbor-sysid?                 system-id
|     +--ro neighbor-extended-circuit-id?
|     |       extended-circuit-id
|     +--ro neighbor-snpa?                  snpa
|     +--ro usage?                          level
|     +--ro hold-timer?
|     |       rt-types:timer-value-seconds16
|     +--ro neighbor-priority?              uint8
|     +--ro lastuptime?                   yang:timestamp
|     +--ro state?                        adj-state-type
+--ro event-counters
|  +--ro adjacency-changes?             uint32
|  +--ro adjacency-number?              uint32
|  +--ro init-fails?                    uint32
|  +--ro adjacency-rejects?             uint32
|  +--ro id-len-mismatch?               uint32
|  +--ro max-area-addresses-mismatch?   uint32
|  +--ro authentication-type-fails?     uint32
|  +--ro authentication-fails?          uint32
|  +--ro lan-dis-changes?               uint32
+--ro packet-counters
|  +--ro level* [level]
|     +--ro level      level-number
|     +--ro iih
|     |  +--ro in?    uint32
|     |  +--ro out?   uint32
|     +--ro ish
|     |  +--ro in?    uint32
|     |  +--ro out?   uint32
|     +--ro esh
|     |  +--ro in?    uint32
|     |  +--ro out?   uint32
|     +--ro lsp
|     |  +--ro in?    uint32
|     |  +--ro out?   uint32
|     +--ro psnp
|     |  +--ro in?    uint32
|     |  +--ro out?   uint32
|     +--ro csnp
|     |  +--ro in?    uint32
|     |  +--ro out?   uint32
|     +--ro unknown
```

```
                |        +--ro in?    uint32
                |        +--ro out?   uint32
                +--rw topologies {multi-topology}?
                   +--rw topology* [name]
                      +--rw name      ->
   ../../../../../../../../rt:ribs/rib/name
                      +--rw metric
                         +--rw value?     wide-metric
                         +--rw level-1
                         |  +--rw value?   wide-metric
                         +--rw level-2
                            +--rw value?   wide-metric

     rpcs:
       +---x clear-adjacency
       |  +---w input
       |     +---w routing-protocol-instance-name    ->
   /rt:routing/control-plane-protocols/control-plane-protocol/name
       |     +---w level?                            level
       |     +---w interface?                        if:interface-ref
       +---x clear-database
          +---w input
             +---w routing-protocol-instance-name    ->
   /rt:routing/control-plane-protocols/control-plane-protocol/name
             +---w level?                            level

     notifications:
       +---n database-overload
       |  +--ro routing-protocol-name?   ->
   /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro overload?                enumeration
       +---n lsp-too-large
       |  +--ro routing-protocol-name?   ->
   /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro pdu-size?                uint32
       |  +--ro lsp-id?                  lsp-id
       +---n if-state-change
       |  +--ro routing-protocol-name?   ->
   /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
```

```
       |  +--ro state?                   if-state-type
       +---n corrupted-lsp-detected
       |  +--ro routing-protocol-name?   ->
     /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro lsp-id?                  lsp-id
       +---n attempt-to-exceed-max-sequence
       |  +--ro routing-protocol-name?   ->
     /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro lsp-id?                  lsp-id
       +---n id-len-mismatch
       |  +--ro routing-protocol-name?   ->
     /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro pdu-field-len?           uint8
       |  +--ro raw-pdu?                 binary
       +---n max-area-addresses-mismatch
       |  +--ro routing-protocol-name?   ->
     /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro max-area-addresses?      uint8
       |  +--ro raw-pdu?                 binary
       +---n own-lsp-purge
       |  +--ro routing-protocol-name?   ->
     /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro lsp-id?                  lsp-id
       +---n sequence-number-skipped
       |  +--ro routing-protocol-name?   ->
     /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro lsp-id?                  lsp-id
       +---n authentication-type-failure
       |  +--ro routing-protocol-name?   ->
     /rt:routing/control-plane-protocols/control-plane-protocol/name
```

```
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro raw-pdu?                 binary
       +---n authentication-failure
       |  +--ro routing-protocol-name?   ->
  /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro raw-pdu?                 binary
       +---n version-skew
       |  +--ro routing-protocol-name?   ->
  /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro protocol-version?        uint8
       |  +--ro raw-pdu?                 binary
       +---n area-mismatch
       |  +--ro routing-protocol-name?   ->
  /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro raw-pdu?                 binary
       +---n rejected-adjacency
       |  +--ro routing-protocol-name?   ->
  /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro raw-pdu?                 binary
       |  +--ro reason?                  string
       +---n protocols-supported-mismatch
       |  +--ro routing-protocol-name?   ->
  /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro raw-pdu?                 binary
       |  +--ro protocols*               uint8
```

```
       +---n lsp-error-detected
       |  +--ro routing-protocol-name?   ->
   /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro lsp-id?                  lsp-id
       |  +--ro raw-pdu?                 binary
       |  +--ro error-offset?            uint32
       |  +--ro tlv-type?                uint8
       +---n adjacency-state-change
       |  +--ro routing-protocol-name?   ->
   /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro neighbor?                string
       |  +--ro neighbor-system-id?      system-id
       |  +--ro state?                   adj-state-type
       |  +--ro reason?                  string
       +---n lsp-received
       |  +--ro routing-protocol-name?   ->
   /rt:routing/control-plane-protocols/control-plane-protocol/name
       |  +--ro isis-level?              level
       |  +--ro interface-name?          if:interface-ref
       |  +--ro interface-level?         level
       |  +--ro extended-circuit-id?     extended-circuit-id
       |  +--ro lsp-id?                  lsp-id
       |  +--ro sequence?                uint32
       |  +--ro received-timestamp?      yang:timestamp
       |  +--ro neighbor-system-id?      system-id
       +---n lsp-generation
          +--ro routing-protocol-name?   ->
   /rt:routing/control-plane-protocols/control-plane-protocol/name
          +--ro isis-level?               level
          +--ro lsp-id?                   lsp-id
          +--ro sequence?                 uint32
          +--ro send-timestamp?           yang:timestamp
```

2.5. Authentication Parameters

The module enables authentication configuration through the IETF key-chain module ([RFC8177]). The IS-IS module imports the "ietf-key-chain" module and reuses some groupings to allow global and per interface configuration of authentication. If a global authentication is configured, an implementation SHOULD authenticate

PSNPs (Partial Sequence Number Packets), CSNPs (Complete Sequence Number Packets) and LSPs (Link State Packets) with the authentication parameters supplied. The authentication of HELLO PDUs (Protocol Data Units) can be activated on a per interface basis.

2.6. IGP/LDP synchronization

[RFC5443] defines a mechanism where IGP (Interior Gateway Protocol) needs to be synchronized with LDP (Label Distribution Protocol). An "ldp-igp-sync" feature has been defined in the model to support this mechanism. The "mpls/ldp/igp-sync" leaf under "interface" allows activation of the mechanism on a per interface basis. The "mpls/ldp/igp-sync" container in the global configuration is empty on purpose and is not required for the activation. The goal of this empty container is to allow easy augmentation with additional parameters like timers for example.

2.7. ISO parameters

As IS-IS protocol is based on ISO protocol suite, some ISO parameters may be required.

This module augments interface configuration model to support ISO configuration parameters.

The clns-mtu can be defined under the interface.

2.8. IP FRR

This YANG module supports LFA (Loop Free Alternates) ([RFC5286]) and remote LFA ([RFC7490]) as IP FRR techniques. The "fast-reroute" container may be augmented by other models to support other IPFRR flavors (MRT, TILFA ...).

The current version of the model supports activation of LFA and remote LFA at interface only. The global "lfa" container is present but kept empty to allow augmentation with vendor specific properties like policies.

Remote LFA is considered as a child of LFA. Remote LFA cannot be enabled if LFA is not enabled.

The "candidate-disabled" allows to mark an interface to not be used as a backup.

2.9. Operational States

Operational states are provided in the module in various places:

- system-counters: provides statistical informations about the global system.
- interface : provides configuration state informations for each interface.
- adjacencies: provides state informations about current IS-IS adjacencies.
- spf-log: provides informations about SPF events on the node. This SHOULD be implemented as a wrapping buffer.
- lsp-log: provides informations about LSP events on the node (reception of an LSP or modification of local LSP). This SHOULD be implemented as a wrapping buffer and an implementation MAY decide to log refresh LSPs or not.
- local-rib: provides the IS-IS internal routing table view.
- database: provides details on the current LSDB.
- hostnames: provides informations about system-id to hostname mappings (as defined in [RFC5301]).
- fast-reroute: provides informations about IP FRR.

3. RPC Operations

The "ietf-isis" module defines two RPC operations:

- clear-database: reset the content of a particular IS-IS database and restart database synchronization with the neighbors.
- clear-adjacency: restart a particular set of IS-IS adjacencies.

4. Notifications

The "ietf-isis" module introduces some notifications :

database-overload: raised when overload condition is changed.

lsp-too-large: raised when the system tries to propagate a too large PDU.

if-state-change: raised when the state of an interface changes.

corrupted-lsp-detected: raised when the system finds that an LSP that was stored in memory has become corrupted.

attempt-to-exceed-max-sequence: This notification is sent when the system wraps the 32-bit sequence counter of an LSP.

id-len-mismatch: This notification is sent when we receive a PDU with a different value for the System ID length.

max-area-addresses-mismatch: This notification is sent when we receive a PDU with a different value for the Maximum Area Addresses.

own-lsp-purge: This notification is sent when the system receives a PDU with its own system ID and zero age.

sequence-number-skipped: This notification is sent when the system receives a PDU with its own system ID and different contents. The system has to reissue the LSP with a higher sequence number.

authentication-type-failure: This notification is sent when the system receives a PDU with the wrong authentication type field.

authentication-failure: This notification is sent when the system receives a PDU with the wrong authentication information.

version-skew: This notification is sent when the system receives a PDU with a different protocol version number.

area-mismatch: This notification is sent when the system receives a Hello PDU from an IS that does not share any area address.

rejected-adjacency: This notification is sent when the system receives a Hello PDU from an IS but does not establish an adjacency for some reason.

protocols-supported-mismatch: This notification is sent when the system receives a non-pseudonode LSP that has no matching protocol supported.

lsp-error-detected: This notification is sent when the system receives an LSP with a parse error.

adjacency-state-change: This notification is sent when an IS-IS adjacency moves to Up state or to Down state.

lsp-received: This notification is sent when an LSP is received.

lsp-generation: This notification is sent when an LSP is regenerated.

5. Interaction with Other YANG Modules

The "isis" container augments the "/rt:routing/rt:control-plane-protocols/control-plane-protocol" container of the ietf-routing [RFC8349] module by defining IS-IS specific parameters.

The "isis" module augments "/if:interfaces/if:interface" defined by [RFC8343] with ISO specific parameters.

The "isis" operational state container augments the "/rt:routing-state/rt:control-plane-protocols/control-plane-protocol" container of the ietf-routing module by defining IS-IS specific operational states.

Some IS-IS specific routes attributes are added to route objects of the ietf-routing module by augmenting "/rt:routing-state/rt:ribs/rt:rib/rt:routes/rt:route".

The modules defined in this document use some groupings from ietf-keychain [RFC8177].

The module reuses types from [RFC6991] and [RFC8294].

To support BFD for fast detection, the module relies on [I-D.ietf-bfd-yang].

6. IS-IS YANG Module

The following RFCs, drafts and external standards are not referenced in the document text but are referenced in the ietf-isis.yang module: [ISO-10589], [RFC1195], [RFC5029], [RFC5130], [RFC5305], [RFC5306], [RFC5308], [RFC5880], [RFC5881], [RFC6119], [RFC6232], [RFC7794], [RFC7810], [RFC7917], [RFC8405].

```
<CODE BEGINS> file "ietf-ospf@2019-01-24.yang"
module ietf-isis {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-isis";

  prefix isis;

  import ietf-routing {
```

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

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

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

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

    import ietf-key-chain {
      prefix "key-chain";
      reference "RFC 8177 - YANG Data Model for Key Chains";
    }

    import ietf-routing-types {
      prefix "rt-types";
      reference "RFC 8294 - Common YANG Data Types for the
                 Routing Area";
    }

    import iana-routing-types {
      prefix "iana-rt-types";
      reference "RFC 8294 - Common YANG Data Types for the
                 Routing Area";
    }

    import ietf-bfd-types {
      prefix "bfd-types";
      reference "RFC YYYY - YANG Data Model for Bidirectional
                 Forwarding Detection (BFD).

   -- Note to RFC Editor Please replace YYYY with published RFC
     number for draft-ietf-bfd-yang.";

    }
```

```
     organization
       "IETF LSR Working Group";

     contact
       "WG List:  <mailto:lsr@ietf.org>

       Editor:    Stephane Litkowski
             <mailto:stephane.litkowski@orange.com>

           Derek Yeung
             <mailto:derek@arrcus.com>
           Acee Lindem
             <mailto:acee@cisco.com>
           Jeffrey Zhang
             <mailto:zzhang@juniper.net>
           Ladislav Lhotka
             <mailto:llhotka@nic.cz>
           Yi Yang
             <mailto:yiya@cisco.com>
           Dean Bogdanovic
             <mailto:deanb@juniper.net>
           Kiran Agrahara Sreenivasa
             <mailto:kkoushik@brocade.com>
           Yingzhen Qu
             <mailto:yiqu@cisco.com>
           Jeff Tantsura
             <mailto:jefftant.ietf@gmail.com>

       ";

     description
       "This YANG module defines the generic configuration and
         operational state for the IS-IS protocol.
         It is intended that the module
         will be extended by vendors to define vendor-specific
         IS-IS configuration parameters and policies,
         for example, route maps or route policies.

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

         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.

          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.
          ";

      revision 2019-01-24 {
        description
          "Initial revision.";
        reference "RFC XXXX";
      }

      /* Identities */

      identity isis {
        base rt:routing-protocol;
        description "Identity for the IS-IS routing protocol.";
      }

      identity lsp-log-reason {
        description "Base identity for an LSP change log reason.";
      }

      identity refresh {
        base lsp-log-reason;
        description
          "Identity used when the LSP log reason is
           a refresh LSP received.";
      }

      identity content-change {
        base lsp-log-reason;
        description
          "Identity used when the LSP log reason is
           a change in the content of the LSP.";
      }


      /* Feature definitions */
```

```
    feature poi-tlv {
      description "Support of Purge Originator Identification.";
      reference "RFC 6232 - Purge Originator Identification TLV
                 for IS-IS";
    }
    feature ietf-spf-delay {
      description
        "Support for IETF SPF delay algorithm.";
      reference "RFC 8405 - SPF Back-off algorithm for link
                 state IGPs";
    }
    feature bfd {
      description
        "Support for BFD detection of IS-IS neighbor reachability.";
      reference "RFC 5880 - Bidirectional Forwarding Detection (BFD)
                 RFC 5881 - Bidirectional Forwarding Detection
                 (BFD) for IPv4 and IPv6 (Single Hop)";
    }
    feature key-chain {
      description
        "Support of keychain for authentication.";
      reference "RFC8177 - YANG Data Model for Key Chains";
    }
    feature node-flag {
      description
        "Support for node-flag for IS-IS prefixes.";
      reference "RFC7794 - IS-IS Prefix Attributes for
                 Extended IP and IPv6 Reachability";
    }
    feature node-tag {
      description
        "Support for node admin tag for IS-IS routing instances.";
      reference "RFC7917 - Advertising Node Administrative Tags
                 in IS-IS";
    }
    feature ldp-igp-sync {
      description
        "LDP IGP synchronization.";
      reference "RFC5443 - LDP IGP Synchronization.";
    }
    feature fast-reroute {
      description
        "Support for IP Fast Reroute (IP-FRR).";
    }
    feature nsr {
      description
        "Non-Stop-Routing (NSR) support.";
    }
```

```
    feature lfa {
      description
        "Support for Loop-Free Alternates (LFAs).";
      reference "RFC5286 - Basic Specification of IP Fast-Reroute:
                 Loop-free Alternates";
    }
    feature remote-lfa {
      description
        "Support for Remote Loop-Free Alternates (R-LFAs).";
      reference "RFC7490 - Remote Loop-Free Alternate Fast Reroute";
    }

    feature overload-max-metric {
      description
        "Support of overload by setting
         all links to max metric.";
    }
    feature prefix-tag {
      description
        "Support for 32-bit prefix tags";
      reference "RFC5130 - A Policy Control Mechanism in
                 IS-IS Using Administrative Tags";
    }
    feature prefix-tag64 {
      description
        "Support for 64-bit prefix tags";
      reference "RFC5130 - A Policy Control Mechanism in
                 IS-IS Using Administrative Tags";
    }
    feature auto-cost {
      description
        "Calculate IS-IS interface metric according to
         reference bandwidth.";
    }

    feature te-rid {
      description
        "Traffic-Engineering Router-ID.";
      reference "RFC5305 - IS-IS Extensions for Traffic Engineering
                 RFC6119 - IPv6 Traffic Engineering in IS-IS";
    }
    feature max-ecmp {
      description
        "Setting maximum number of ECMP paths.";
    }
    feature multi-topology {
      description
        "Support for Multiple-Topology Routing (MTR).";
```

```
      reference "RFC5120 - M-IS-IS: Multi Topology Routing in IS-IS";
    }
    feature nlpid-control {
      description
        "This feature controls the advertisement
         of support NLPID within IS-IS configuration.";
    }
    feature graceful-restart {
      description
        "IS-IS Graceful restart support.";
      reference "RFC5306 - Restart Signaling in IS-IS";
    }

    feature lsp-refresh {
      description
        "Configuration of LSP refresh interval.";
    }

    feature maximum-area-addresses {
      description
        "Support of maximum-area-addresses config.";
    }

    feature admin-control {
      description
        "Administrative control of the protocol state.";
    }

    /* Type definitions */

    typedef circuit-id {
      type uint8;
      description
        "This type defines the circuit ID
         associated with an interface.";
    }

    typedef extended-circuit-id {
      type uint32;
      description
        "This type defines the extended circuit ID
         associated with an interface.";
    }

    typedef interface-type {
      type enumeration {
        enum broadcast {
          description
```

```
            "Broadcast interface type.";
        }
        enum point-to-point {
          description
            "Point-to-point interface type.";
        }
      }
      description
        "This type defines the type of adjacency
         to be established on the interface.
         The interface-type determines the type
         of hello message that is used.";

    }

    typedef level {
      type enumeration {
        enum "level-1" {
          description
            "This enum indicates L1-only capability.";
        }
        enum "level-2" {
          description
            "This enum indicates L2-only capability.";
        }
        enum "level-all" {
          description
            "This enum indicates capability for both levels.";
        }
      }
      default "level-all";
      description
        "This type defines IS-IS level of an object.";

    }

    typedef adj-state-type {
      type enumeration {
        enum "up" {
          description
            "State indicates the adjacency is established.";
        }
        enum "down" {
          description
            "State indicates the adjacency is NOT established.";
        }
        enum "init" {
          description
```

```
            "State indicates the adjacency is establishing.";
        }
        enum "failed" {
          description
            "State indicates the adjacency is failed.";
        }
      }
      description
        "This type defines states of an adjacency";
    }

    typedef if-state-type {
      type enumeration {
        enum "up" {
          description "Up state.";

        }
        enum "down" {
          description "Down state";
        }
      }
      description
        "This type defines the state of an interface";
    }

    typedef level-number {
      type uint8 {
        range "1 .. 2";
      }
      description
        "This type defines the current IS-IS level.";
    }

    typedef lsp-id {
      type string {
        pattern
          '[0-9A-Fa-f]{4}\.[0-9A-Fa-f]{4}\.[0-9A-Fa-f]'
          +'{4}\.[0-9][0-9]-[0-9][0-9]';
      }
      description
        "This type defines the IS-IS LSP ID format using a
         pattern. An example LSP ID is 0143.0438.AEF0.02-01";
    }

    typedef area-address {
      type string {
        pattern '[0-9A-Fa-f]{2}(\.[0-9A-Fa-f]{4}){0,6}';
      }
```

```
      description
        "This type defines the area address format.";
    }

    typedef snpa {
      type string {
        length "0 .. 20";
      }
      description
        "This type defines the Subnetwork Point
         of Attachment (SNPA) format.
         The SNPA should be encoded according to the rules
         specified for the particular type of subnetwork
         being used. As an example, for an ethernet subnetwork,
         the SNPA is encoded as a MAC address like
         '00aa.bbcc.ddee'.";
    }

    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 IS-IS system-id using pattern,
         An example system-id is 0143.0438.AEF0";
    }
    typedef extended-system-id {
      type string {
        pattern
          '[0-9A-Fa-f]{4}\.[0-9A-Fa-f]{4}\.[0-9A-Fa-f]{4}\.'
          +'[0-9][0-9]';
      }
      description
        "This type defines IS-IS system-id using pattern. The extended
         system-id contains the pseudonode number in addition to the
         system-id.
         An example system-id is 0143.0438.AEF0.00";
    }

    typedef wide-metric {
      type uint32 {
        range "0 .. 16777215";
      }
      description
        "This type defines wide style format of IS-IS metric.";
    }
```

```
    typedef std-metric {
      type uint8 {
        range "0 .. 63";
      }
      description
        "This type defines old style format of IS-IS metric.";
    }

    typedef mesh-group-state {
      type enumeration {
        enum "mesh-inactive" {
          description
            "Interface is not part of a mesh group.";
        }
        enum "mesh-set" {
          description
            "Interface is part of a mesh group.";
        }
        enum "mesh-blocked" {
          description
            "LSPs must not be flooded over this interface.";
        }
      }
      description
        "This type describes mesh group state of an interface";
    }

    /* Grouping for notifications */

    grouping notification-instance-hdr {
      description
        "Instance specific IS-IS notification data grouping";
      leaf routing-protocol-name {
        type leafref {
          path "/rt:routing/rt:control-plane-protocols/"
             + "rt:control-plane-protocol/rt:name";
        }
        description "Name of the IS-IS instance.";
      }
      leaf isis-level {
        type level;
        description "IS-IS level of the instance.";
      }
    }

    grouping notification-interface-hdr {
      description
        "Interface specific IS-IS notification data grouping";
```

```
      leaf interface-name {
        type if:interface-ref;
        description "IS-IS interface name";
      }
      leaf interface-level {
        type level;
        description "IS-IS level of the interface.";
      }
      leaf extended-circuit-id {
        type extended-circuit-id;
        description "Extended circuit-id of the interface.";
      }
    }


  /* Groupings for IP Fast Reroute */

  grouping instance-fast-reroute-config {
     description
       "This group defines global configuration of IP
        Fast ReRoute (FRR).";
     container fast-reroute {
       if-feature fast-reroute;
       description
         "This container may be augmented with global
          parameters for IP-FRR.";
       container lfa {
         if-feature lfa;
         description
           "This container may be augmented with
            global parameters for Loop-Free Alternatives (LFA).
            Container creation has no effect on LFA activation.";
       }
     }
   }

  grouping interface-lfa-config {
     leaf candidate-enable {
       type boolean;
       default true;
       description
         "Enable the interface to be used as backup.";
     }
     leaf enable {
       type boolean;
       default false;
       description
         "Activates LFA - Per-prefix LFA computation
```

```
           is assumed.";
      }
      container remote-lfa {
        if-feature remote-lfa;
        leaf enable {
          type boolean;
          default false;
          description
            "Activates Remote LFA (R-LFA).";
        }
        description
          "Remote LFA configuration.";
      }
      description "Grouping for LFA interface configuration";
    }
    grouping interface-fast-reroute-config {
      description
        "This group defines interface configuration of IP-FRR.";
      container fast-reroute {
        if-feature fast-reroute;
        container lfa {
          if-feature lfa;
          uses interface-lfa-config;
          container level-1 {
            uses interface-lfa-config;
            description
              "LFA level 1 config";
          }
          container level-2 {
          uses interface-lfa-config;
           description
            "LFA level 2 config";
          }
          description
            "LFA configuration.";
        }
        description
          "Interface IP Fast-reroute configuration.";
      }
    }
    grouping instance-fast-reroute-state {
      description "IPFRR state data grouping";
      container protected-routes {
        config false;
        list address-family-stats {
          key "address-family prefix alternate";

          leaf address-family {
```

```
            type iana-rt-types:address-family;
            description
              "Address-family";
          }
          leaf prefix {
            type inet:ip-prefix;
            description
              "Protected prefix.";
          }
          leaf alternate {
            type inet:ip-address;
            description
              "Alternate next hop for the prefix.";
          }
          leaf alternate-type {
            type enumeration {
              enum equal-cost {
                description
                  "ECMP alternate.";
              }
              enum lfa {
                description
                  "LFA alternate.";
              }
              enum remote-lfa {
                description
                  "Remote LFA alternate.";
              }
              enum tunnel {
                description
                  "Tunnel based alternate
                   (like RSVP-TE or GRE).";
              }
              enum ti-lfa {
                description
                  "TI-LFA alternate.";
              }
              enum mrt {
                description
                  "MRT alternate.";
              }
              enum other {
                description
                  "Unknown alternate type.";
              }
            }
            description
              "Type of alternate.";
```

```
          }
          leaf best {
            type boolean;
            description
              "Is set when the alternate is the preferred one,
               is unset otherwise.";
          }
          leaf non-best-reason {
            type string {
              length "1..255";
            }
            description
              "Information field to describe why the alternate
               is not best. The length should be limited to 255
               unicode characters. The expected format is a single
               line text.";
          }
          leaf protection-available {
            type bits {
              bit node-protect {
                position 0;
                description
                  "Node protection available.";
              }
              bit link-protect {
                position 1;
                description
                  "Link protection available.";
              }
              bit srlg-protect {
                position 2;
                description
                  "SRLG protection available.";
              }
              bit downstream-protect {
                position 3;
                description
                  "Downstream protection available.";
              }
              bit other {
                position 4;
                description
                  "Other protection available.";
              }
            }
            description "Protection provided by the alternate.";
          }
          leaf alternate-metric1 {
```

```
            type uint32;
            description
              "Metric from Point of Local Repair (PLR) to
               destination through the alternate path.";
          }
          leaf alternate-metric2 {
            type uint32;
            description
              "Metric from PLR to the alternate node";
          }
          leaf alternate-metric3 {
            type uint32;
            description
              "Metric from alternate node to the destination";
          }
          description
            "Per-AF protected prefix statistics.";
        }
        description
          "List of prefixes that are protected.";
      }

      container unprotected-routes {
        config false;
        list address-family-stats {
          key "address-family prefix";

          leaf address-family {
            type iana-rt-types:address-family;

            description "Address-family";
          }
          leaf prefix {
            type inet:ip-prefix;
            description "Unprotected prefix.";
          }
          description
            "Per AF unprotected prefix statistics.";
        }
        description
          "List of prefixes that are not protected.";
      }

      list protection-statistics {
        key frr-protection-method;
        config false;
        leaf frr-protection-method {
          type string;
```

```
          description "Protection method used.
                       The expected format is a single word.
                       As example: LFA,rLFA, MRT, RSVP-TE...";
        }
        list address-family-stats {
          key address-family;

          leaf address-family {
            type iana-rt-types:address-family;

            description "Address-family";
          }
          leaf total-routes {
            type uint32;
            description "Total prefixes.";
          }
          leaf unprotected-routes {
            type uint32;
            description
              "Total prefixes that are not protected.";
          }
          leaf protected-routes {
            type uint32;
            description
              "Total prefixes that are protected.";
          }
          leaf linkprotected-routes {
            type uint32;
            description
              "Total prefixes that are link protected.";
          }
          leaf nodeprotected-routes {
            type uint32;
            description
              "Total prefixes that are node protected.";
          }
          description
            "Per AF protected prefix statistics.";
        }

        description "Global protection statistics.";
      }
    }

      /* Route table and local RIB groupings */

    grouping local-rib {
      description "Local-rib - RIB for Routes computed by the local
```

```
                  IS-IS routing instance.";
      container local-rib {
        config false;
        description "Local-rib.";
        list route {
          key "prefix";
          description "Routes";
          leaf prefix {
            type inet:ip-prefix;
            description "Destination prefix.";
          }
          container next-hops {
            description "Next hops for the route.";
            list next-hop {
              key "next-hop";
              description "List of next hops for the route";
              leaf outgoing-interface {
                type if:interface-ref;
                description
                  "Name of the outgoing interface.";
              }
              leaf next-hop {
               type inet:ip-address;
               description "Next hop address.";
              }
            }
          }
          leaf metric {
            type uint32;
            description "Metric for this route.";
          }
          leaf level {
            type level-number;
            description "Level number for this route.";
          }
          leaf route-tag {
            type uint32;
            description "Route tag for this route.";
          }
        }
      }
    }

    grouping route-content {
      description
        "IS-IS protocol-specific route properties grouping.";
      leaf metric {
        type uint32;
```

```
        description "IS-IS metric of a route.";
      }
      leaf-list tag {
        type uint64;
        description
          "List of tags associated with the route. The leaf
           describes both 32-bit and 64-bit tags.";
      }
      leaf route-type {
        type enumeration {
          enum l2-up-internal {
            description "Level 2 internal route
                         and not leaked to a lower level";
          }
          enum l1-up-internal {
            description "Level 1 internal route
                         and not leaked to a lower level";
          }
          enum l2-up-external {
            description "Level 2 external route
                         and not leaked to a lower level";
          }
          enum l1-up-external {
            description "Level 1 external route
                         and not leaked to a lower level";
          }
          enum l2-down-internal {
            description "Level 2 internal route
                         and leaked to a lower level";
          }
          enum l1-down-internal {
            description "Level 1 internal route
                         and leaked to a lower level";
          }
          enum l2-down-external {
            description "Level 2 external route
                         and leaked to a lower level";
          }
          enum l1-down-external {
            description "Level 1 external route
                         and leaked to a lower level";
          }
        }
        description "IS-IS route type.";
      }
    }
```

```
          /* Grouping definitions for configuration and ops state */


     grouping adjacency-state {
       container adjacencies {
         config false;
         list adjacency {
           leaf neighbor-sys-type {
             type level;
             description
               "Level capability of neighboring system";
           }
           leaf neighbor-sysid {
             type system-id;
             description
               "The system-id of the neighbor";
           }
           leaf neighbor-extended-circuit-id {
             type extended-circuit-id;
             description
               "Circuit ID of the neighbor";
           }
           leaf neighbor-snpa {
             type snpa;
             description
               "SNPA of the neighbor";
           }
           leaf usage {
             type level;
             description
               "Define the level(s) activated on the adjacency.
                On a p2p link this might be level 1 and 2,
                but on a LAN, the usage will be level 1
                between peers at level 1 or level 2 between
                peers at level 2.";
           }
           leaf hold-timer {
             type rt-types:timer-value-seconds16;
             units seconds;
             description
               "The holding time in seconds for this
                adjacency. This value is based on
                received hello PDUs and the elapsed
                time since receipt.";
           }
           leaf neighbor-priority {
             type uint8 {
               range "0 .. 127";
```

```
            }
            description
              "Priority of the neighboring IS for becoming
               the DIS.";
          }
          leaf lastuptime {
            type yang:timestamp;
            description
              "When the adjacency most recently entered
               state 'up', measured in hundredths of a
               second since the last reinitialization of
               the network management subsystem.
               The value is 0 if the adjacency has never
               been in state 'up'.";
          }
          leaf state {
            type adj-state-type;
            description
              "This leaf describes the state of the interface.";
          }

          description
            "List of operational adjacencies.";
        }
        description
          "This container lists the adjacencies of
           the local node.";
      }
      description
        "Adjacency state";
    }

    grouping admin-control {
      leaf enable {
        if-feature admin-control;
        type boolean;
        default true;
        description
          "Enable/Disable the protocol.";
      }
      description
        "Grouping for admin control.";
    }

    grouping ietf-spf-delay {
      leaf initial-delay {
        type rt-types:timer-value-milliseconds;
        units msec;
```

```
        description
          "Delay used while in QUIET state (milliseconds).";
      }
      leaf short-delay {
        type rt-types:timer-value-milliseconds;
        units msec;
        description
          "Delay used while in SHORT_WAIT state (milliseconds).";
      }
      leaf long-delay {
        type rt-types:timer-value-milliseconds;
        units msec;
        description
          "Delay used while in LONG_WAIT state (milliseconds).";
      }

      leaf hold-down {
        type rt-types:timer-value-milliseconds;
        units msec;
        description
          "Timer used to consider an IGP stability period
                                (milliseconds).";
      }
      leaf time-to-learn {
        type rt-types:timer-value-milliseconds;
        units msec;
        description
          "Duration used to learn all the IGP events
           related to a single component failure (milliseconds).";
      }
      leaf current-state {
        type enumeration {
          enum "quiet" {
            description "QUIET state";
          }
          enum "short-wait" {
            description "SHORT_WAIT state";
          }
          enum "long-wait" {
            description "LONG_WAIT state";
          }
        }
        config false;
        description
          "Current SPF back-off algorithm state.";
      }
      leaf remaining-time-to-learn {
        type rt-types:timer-value-milliseconds;
```

```
        units "msec";
        config false;
        description
          "Remaining time until time-to-learn timer fires.";
      }
      leaf remaining-hold-down {
        type rt-types:timer-value-milliseconds;
        units "msec";
        config false;
        description
          "Remaining time until hold-down timer fires.";
      }
      leaf last-event-received {
        type yang:timestamp;
        config false;
        description
          "Time of last IGP event received";
      }
      leaf next-spf-time {
        type yang:timestamp;
        config false;
        description
          "Time when next SPF has been scheduled.";
      }
      leaf last-spf-time {
        type yang:timestamp;
        config false;
        description
          "Time of last SPF computation.";
      }
      description
        "Grouping for IETF SPF delay configuration and state.";
    }



    grouping node-tag-config {
      description
        "IS-IS node tag config state.";
      container node-tags {
        if-feature node-tag;
        list node-tag {
          key tag;
          leaf tag {
            type uint32;
              description
                "Node tag value.";
          }
```

```
          description
            "List of tags.";
        }
        description
          "Container for node admin tags.";
      }
    }

    grouping authentication-global-cfg {
      choice authentication-type {
        case key-chain {
          if-feature key-chain;
          leaf key-chain {
            type key-chain:key-chain-ref;
            description
              "Reference to a key-chain.";
          }
        }
        case password {
          leaf key {
            type string;
            description
              "This leaf specifies the authentication key. The
               length of the key may be dependent on the
               cryptographic algorithm. In cases where it is
               not, a key length of at least 32 octets should be
               supported to allow for interoperability with
               strong keys.";
          }
          leaf crypto-algorithm {
            type identityref {
              base key-chain:crypto-algorithm;
            }
            description
              "Cryptographic algorithm associated with key.";
          }
        }
        description "Choice of authentication.";
      }
      description "Grouping for global authentication config.";
    }

    grouping metric-type-global-cfg {
      leaf value {
        type enumeration {
          enum wide-only {
            description
              "Advertise new metric style only (RFC5305)";
```

```
          }
          enum old-only {
            description
              "Advertise old metric style only (RFC1195)";
          }
          enum both {
            description "Advertise both metric styles";
          }
        }
        default wide-only;
        description
          "Type of metric to be generated:
           - wide-only means only new metric style
              is generated,
           - old-only means that only old-style metric
             is generated,
           - both means that both are advertised.
           This leaf is only affecting IPv4 metrics.";
      }
      description
        "Grouping for global metric style config.";
    }

    grouping default-metric-global-cfg {
      leaf value {
        type wide-metric;
        default "10";
        description  "Value of the metric";
      }
      description
        "Global default metric config grouping.";
    }


    grouping overload-global-cfg {
      leaf status {
        type boolean;
        default false;
        description
          "This leaf specifies the overload status.";
      }
      description "Grouping for overload bit config.";
    }

    grouping overload-max-metric-global-cfg {
      leaf timeout {
        type rt-types:timer-value-seconds16;
        units "seconds";
```

```
        description
          "Timeout (in seconds) of the overload condition.";
      }
      description
        "Overload maximum metric configuration grouping";
    }

    grouping route-preference-global-cfg {
      choice granularity {
        case detail {
          leaf internal {
            type uint8;
            description
              "Protocol preference for internal routes.";
          }
          leaf external {
            type uint8;
            description
              "Protocol preference for external routes.";
          }
        }
        case coarse {
          leaf default {
            type uint8;
            description
              "Protocol preference for all IS-IS routes.";
          }
        }
        description
          "Choice for implementation of route preference.";
      }
      description
        "Global route preference grouping";
    }

    grouping hello-authentication-cfg {
      choice authentication-type {
        case key-chain {
          if-feature key-chain;
          leaf key-chain {
            type key-chain:key-chain-ref;
            description "Reference to a key-chain.";
          }
        }
        case password {
          leaf key {
            type string;
            description "Authentication key specification - The
```

```
                          length of the key may be dependent on the
                          cryptographic algorithm. In cases where
                          it is not, a key length of at least 32 octets
                          should be supported to allow for
                          interoperability with strong keys.";
            }
            leaf crypto-algorithm {
              type identityref {
                base key-chain:crypto-algorithm;
              }
              description
                "Cryptographic algorithm associated with key.";
            }
          }
          description "Choice of authentication.";
        }
        description "Grouping for hello authentication.";
      }

      grouping hello-interval-cfg {
        leaf value {
          type rt-types:timer-value-seconds16;
          units "seconds";
          default 10;
          description
            "Interval (in seconds) between successive hello
             messages.";
        }

        description "Interval between hello messages.";
      }

      grouping hello-multiplier-cfg {
        leaf value {
          type uint16;
          default 3;
          description
            "Number of missed hello messages prior to
             declaring the adjacency down.";
        }
        description
            "Number of missed hello messages prior to
             adjacency down grouping.";
      }

      grouping priority-cfg {
        leaf value {
          type uint8 {
```

```
          range "0 .. 127";
        }
        default 64;
        description
          "Priority of interface for DIS election.";
      }

      description "Interface DIS election priority grouping";
    }

    grouping metric-cfg {
      leaf value {
        type wide-metric;
        default "10";
        description "Metric value.";
      }
      description "Interface metric grouping";
    }


    grouping metric-parameters {
      container metric-type {
        uses metric-type-global-cfg;
        container level-1 {
          uses metric-type-global-cfg;
          description "level-1 specific configuration";
        }
        container level-2 {
          uses metric-type-global-cfg;
          description "level-2 specific configuration";
        }
        description "Metric style global configuration";
      }

      container default-metric {
        uses default-metric-global-cfg;
        container level-1 {
          uses default-metric-global-cfg;
          description "level-1 specific configuration";
        }
        container level-2 {
          uses default-metric-global-cfg;
          description "level-2 specific configuration";
        }
        description "Default metric global configuration";
      }
      container auto-cost {
        if-feature auto-cost;
```

```
        description
          "Interface Auto-cost configuration state.";
        leaf enable {
          type boolean;
          description
            "Enable/Disable interface auto-cost.";
        }
        leaf reference-bandwidth {
          when "../enable = 'true'" {
            description "Only when auto cost is enabled";
          }
          type uint32 {
            range "1..4294967";
          }
          units Mbits;
          description
            "Configure reference bandwidth used to automatically
             determine interface cost (Mbits). The cost is the
             reference bandwidth divided by the interface speed
             with 1 being the minimum cost.";
        }
      }

      description "Grouping for global metric parameters.";
    }

    grouping high-availability-parameters {
      container graceful-restart {
        if-feature graceful-restart;
        leaf enable {
          type boolean;
          default false;
          description "Enable graceful restart.";
        }
        leaf restart-interval {
          type rt-types:timer-value-seconds16;
          units "seconds";
          description
            "Interval (in seconds) to attempt graceful restart prior
             to failure.";
        }
        leaf helper-enable {
          type boolean;
          default true;
          description
            "Enable local IS-IS router as graceful restart helper.";
        }
        description "Graceful-Restart Configuration.";
```

```
      }
      container nsr {
        if-feature nsr;
        description "Non-Stop Routing (NSR) configuration.";
        leaf enable {
          type boolean;
          default false;
          description "Enable/Disable Non-Stop Routing (NSR).";
        }
      }
      description "Grouping for High Availability parameters.";
    }

    grouping authentication-parameters {
      container authentication {
        uses authentication-global-cfg;

        container level-1 {
          uses authentication-global-cfg;
          description "level-1 specific configuration";
        }
        container level-2 {
          uses authentication-global-cfg;
          description "level-2 specific configuration";
        }
        description "Authentication global configuration for
                     both LSPs and SNPs.";
      }
      description "Grouping for authentication parameters";
    }
    grouping address-family-parameters {
      container address-families {
        if-feature nlpid-control;
        list address-family-list {
          key address-family;
          leaf address-family {
            type iana-rt-types:address-family;
            description "Address-family";
          }
          leaf enable {
            type boolean;
            description "Activate the address family.";
          }
          description
            "List of address families and whether or not they
             are activated.";
        }
        description "Address Family configuration";
```

```
      }
      description "Grouping for address family parameters.";
    }

    grouping mpls-parameters {
      container mpls {
        container te-rid {
          if-feature te-rid;
          description
            "Stable ISIS Router IP Address used for Traffic
             Engineering";
          leaf ipv4-router-id {
            type inet:ipv4-address;
            description
              "Router ID value that would be used in TLV 134.";
          }
          leaf ipv6-router-id {
            type inet:ipv6-address;
            description
              "Router ID value that would be used in TLV 140.";
          }
        }
        container ldp {
          container igp-sync {
            if-feature ldp-igp-sync;
            description
              "This container may be augmented with global
               parameters for igp-ldp-sync.";
          }
          description "LDP configuration.";
        }
        description "MPLS configuration";
      }
      description "Grouping for MPLS global parameters.";
    }

    grouping lsp-parameters {
      leaf lsp-mtu {
        type uint16;
        units "bytes";
        default 1492;
        description
          "Maximum size of an LSP PDU in bytes.";
      }
      leaf lsp-lifetime {
        type uint16 {
          range "1..65535";
        }
```

```
        units "seconds";
        description
          "Lifetime of the router's LSPs in seconds.";
      }
      leaf lsp-refresh {
        if-feature lsp-refresh;
        type rt-types:timer-value-seconds16;
        units "seconds";
        description
          "Refresh interval of the router's LSPs in seconds.";
      }
      leaf poi-tlv {
        if-feature poi-tlv;
        type boolean;
        default false;
        description
          "Enable advertisement of IS-IS purge TLV.";
      }
      description "Grouping for LSP global parameters.";
    }
    grouping spf-parameters {
      container spf-control {
          leaf paths {
            if-feature max-ecmp;
            type uint16 {
              range "1..32";
            }
            description
              "Maximum number of Equal-Cost Multi-Path (ECMP) paths.";
          }
          container ietf-spf-delay {
            if-feature ietf-spf-delay;
            uses ietf-spf-delay;
            description "IETF SPF delay algorithm configuration.";
          }
          description
            "SPF calculation control.";
      }
                  description "Grouping for SPF global parameters.";
    }
    grouping instance-config {
      description "IS-IS global configuration grouping";

      uses admin-control;

      leaf level-type {
        type level;
        default "level-all";
```

```
        description
          "Level of an IS-IS node - can be level-1,
           level-2 or level-all.";
      }

      leaf system-id {
        type system-id;
        description "system-id of the node.";
      }

      leaf maximum-area-addresses {
        if-feature maximum-area-addresses;
        type uint8;
        default 3;
        description "Maximum areas supported.";
      }

      leaf-list area-address {
        type area-address;
        description
          "List of areas supported by the protocol instance.";
      }

      uses lsp-parameters;
      uses high-availability-parameters;
      uses node-tag-config;
      uses metric-parameters;
      uses authentication-parameters;
      uses address-family-parameters;
      uses mpls-parameters;
      uses spf-parameters;
      uses instance-fast-reroute-config;

      container preference {
        uses route-preference-global-cfg;
        description "Router preference configuration for IS-IS
                     protocol instance route installation";
      }

      container overload {
        uses overload-global-cfg;
        description "Router protocol instance overload state
                     configuration";
      }

      container overload-max-metric {
        if-feature overload-max-metric;
        uses overload-max-metric-global-cfg;
```

```
        description
          "Router protocol instance overload maximum
           metric advertisement configuration.";
      }
    }

    grouping instance-state {
        description
         "IS-IS instance operational state.";
       uses spf-log;
        uses lsp-log;
        uses hostname-db;
        uses lsdb;
        uses local-rib;
        uses system-counters;
        uses instance-fast-reroute-state;
    }

    grouping multi-topology-config {
      description "Per-topology configuration";
      container default-metric {
        uses default-metric-global-cfg;
        container level-1 {
          uses default-metric-global-cfg;
          description "level-1 specific configuration";
        }
        container level-2 {
          uses default-metric-global-cfg;
          description "level-2 specific configuration";
        }
        description "Default metric per-topology configuration";
      }
      uses node-tag-config;
    }

    grouping interface-config {
      description "Interface configuration grouping";
      leaf level-type {
        type level;
        default "level-all";
        description "IS-IS level of the interface.";
      }
      leaf lsp-pacing-interval {
        type rt-types:timer-value-milliseconds;
        units "milliseconds";
        default 33;
        description
          "Interval (in milli-seconds) between LSP
```

```
          transmissions.";
      }
      leaf lsp-retransmit-interval {
        type rt-types:timer-value-seconds16;
        units "seconds";
        description
          "Interval (in seconds) between LSP
           retransmissions.";
      }
      leaf passive {
        type boolean;
        default "false";
        description
          "Indicates whether the interface is in passive mode (IS-IS
           not running but network is advertised).";
      }
      leaf csnp-interval {
        type rt-types:timer-value-seconds16;
        units "seconds";
        default 10;
        description
           "Interval (in seconds) between CSNP messages.";
      }
      container hello-padding {
        leaf enable {
          type boolean;
          default "true";
          description
            "IS-IS Hello-padding activation - enabled by default.";
        }
        description "IS-IS hello padding configuration.";
      }
      leaf mesh-group-enable {
        type mesh-group-state;
        description "IS-IS interface mesh-group state";
      }
      leaf mesh-group {
        when "../mesh-group-enable = 'mesh-set'" {
          description
            "Only valid when mesh-group-enable equals mesh-set";
        }
        type uint8;
        description "IS-IS interface mesh-group ID.";
      }
      leaf interface-type {
        type interface-type;
        default "broadcast";
        description
```

```
          "Type of adjacency to be established on the interface. This
           dictates the type of hello messages that are used.";
      }

      uses admin-control;

      leaf-list tag {
        if-feature prefix-tag;
        type uint32;
        description
          "List of tags associated with the interface.";
      }
      leaf-list tag64 {
        if-feature prefix-tag64;
        type uint64;
        description
          "List of 64-bit tags associated with the interface.";
      }
      leaf node-flag {
        if-feature node-flag;
        type boolean;
        default false;
        description
          "Set prefix as a node representative prefix.";
      }
      container hello-authentication {
        uses hello-authentication-cfg;
        container level-1 {
          uses hello-authentication-cfg;
          description "level-1 specific configuration";
        }
        container level-2 {
          uses hello-authentication-cfg;
          description "level-2 specific configuration";
        }
        description
          "Authentication type to be used in hello messages.";
      }
      container hello-interval {
        uses hello-interval-cfg;
        container level-1 {
          uses hello-interval-cfg;
          description "level-1 specific configuration";
        }
        container level-2 {
          uses hello-interval-cfg;
          description "level-2 specific configuration";
        }
```

```
        description "Interval between hello messages.";
      }
      container hello-multiplier {
        uses hello-multiplier-cfg;
        container level-1 {
          uses hello-multiplier-cfg;
          description "level-1 specific configuration";
        }
        container level-2 {
          uses hello-multiplier-cfg;
          description "level-2 specific configuration";
        }
        description "Hello multiplier configuration.";
      }
      container priority {
        must '../interface-type = "broadcast"' {
          error-message
            "Priority only applies to broadcast interfaces.";
          description "Check for broadcast interface.";
        }
        uses priority-cfg;
        container level-1 {
          uses priority-cfg;
          description "level-1 specific configuration";
        }
        container level-2 {
          uses priority-cfg;
          description "level-2 specific configuration";
        }
        description "Priority for DIS election.";
      }
      container metric {
        uses metric-cfg;
        container level-1 {
          uses metric-cfg;
          description "level-1 specific configuration";
        }
        container level-2 {
          uses metric-cfg;
          description "level-2 specific configuration";
        }
        description "Metric configuration.";
      }
      container bfd {
        if-feature bfd;
        description "BFD Client Configuration.";
        uses bfd-types:client-cfg-parms;
```

```
        reference "RFC YYYY - YANG Data Model for Bidirectional
                  Forwarding Detection (BFD).

   -- Note to RFC Editor Please replace YYYY with published FC
      number for draft-ietf-bfd-yang.";

       }
       container address-families {
         if-feature nlpid-control;
         list address-family-list {
           key address-family;
           leaf address-family {
             type iana-rt-types:address-family;
             description  "Address-family";
           }
           description "List of AFs.";
         }
         description "Interface address-families";
       }
       container mpls {
         container ldp {
           leaf igp-sync {
             if-feature ldp-igp-sync;
             type boolean;
             default false;
             description "Enables IGP/LDP synchronization";
           }
           description "LDP protocol related configuration.";
         }
         description "MPLS configuration for IS-IS interfaces";
       }
       uses interface-fast-reroute-config;
     }

     grouping multi-topology-interface-config {
       description "IS-IS interface topology configuration.";
       container metric {
         uses metric-cfg;
         container level-1 {
           uses metric-cfg;
           description "level-1 specific configuration";
         }
         container level-2 {
           uses metric-cfg;
           description "level-2 specific configuration";
         }
         description "Metric IS-IS interface configuration.";
       }
```

```
    }
    grouping interface-state {
      description
        "IS-IS interface operational state.";
      uses adjacency-state;
      uses event-counters;
      uses packet-counters;
    }

   /* Grouping for the hostname database */

    grouping hostname-db {
      container hostnames {
        config false;
        list hostname {
          key system-id;
          leaf system-id {
            type system-id;
            description
              "system-id associated with the hostname.";
          }
          leaf hostname {
            type string {
              length "1..255";
            }
            description
              "Hostname associated with the system-id
               as defined in RFC5301.";
          }
          description
            "List of system-id/hostname associations.";
        }
        description
          "Hostname to system-id mapping database.";
      }
      description
        "Grouping for hostname to system-id mapping database.";
    }

    /* Groupings for counters */

    grouping system-counters {
      container system-counters {
        config false;
        list level {
          key level;

          leaf level {
```

```
            type level-number;
            description "IS-IS level.";
          }
          leaf corrupted-lsps {
            type uint32;
            description
              "Number of corrupted in-memory LSPs detected.
               LSPs received from the wire with a bad
               checksum are silently dropped and not counted.
               LSPs received from the wire with parse errors
               are counted by lsp-errors.";
          }
          leaf authentication-type-fails {
            type uint32;
            description
               "Number of authentication type mismatches.";
          }
          leaf authentication-fails {
            type uint32;
            description
              "Number of authentication key failures.";
          }
          leaf database-overload {
            type uint32;
            description
              "Number of times the database has become
               overloaded.";
          }
          leaf own-lsp-purge {
            type uint32;
            description
              "Number of times a zero-aged copy of the system's
               own LSP is received from some other IS-IS node.";
          }
          leaf manual-address-drop-from-area {
            type uint32;
            description
              "Number of times a manual address
               has been dropped from the area.";
          }
          leaf max-sequence {
            type uint32;
            description
              "Number of times the system has attempted
               to exceed the maximum sequence number.";
          }
          leaf sequence-number-skipped {
            type uint32;
```

```
            description
              "Number of times a sequence number skip has
               occurred.";
          }
          leaf id-len-mismatch {
            type uint32;
            description
              "Number of times a PDU is received with a
               different value for the ID field length
               than that of the receiving system.";
          }
          leaf partition-changes {
            type uint32;
            description
              "Number of partition changes detected.";
          }
          leaf lsp-errors {
            type uint32;
            description
              "Number of LSPs with errors we have received.";
          }
          leaf spf-runs {
            type uint32;
            description
              "Number of times we ran SPF at this level.";
          }
          description
            "List of supported levels.";
        }
        description
          "List counters for the IS-IS protocol instance";
      }
      description "System counters grouping.";
    }

    grouping event-counters {
      container event-counters {
        config false;
        leaf adjacency-changes {
          type uint32;
          description
            "The number of times an adjacency state change has
             occurred on this interface.";
        }
        leaf adjacency-number {
          type uint32;
          description
            "The number of adjacencies on this interface.";
```

```
        }
        leaf init-fails {
          type uint32;
          description
            "The number of times initialization of this
             interface has failed. This counts events such
             as PPP NCP failures. Failures to form an
             adjacency are counted by adjacency-rejects.";
        }
        leaf adjacency-rejects {
          type uint32;
          description
            "The number of times an adjacency has been
             rejected on this interface.";
        }
        leaf id-len-mismatch {
          type uint32;
          description
            "The number of times an IS-IS PDU with an ID
             field length different from that for this
             system has been received on this interface.";
        }
        leaf max-area-addresses-mismatch {
          type uint32;
          description
            "The number of times an IS-IS PDU has been
             received on this interface with the
             max area address field differing from that of
             this system.";
        }
        leaf authentication-type-fails {
          type uint32;
          description
            "Number of authentication type mismatches.";
        }
        leaf authentication-fails {
          type uint32;
          description
            "Number of authentication key failures.";
        }
        leaf lan-dis-changes {
          type uint32;
          description
            "The number of times the DIS has changed on this
             interface at this level. If the interface type is
             point-to-point, the count is zero.";
        }
        description "IS-IS interface event counters.";
```

```
      }
      description
        "Grouping for IS-IS interface event counters";
    }

    grouping packet-counters {
      container packet-counters {
        config false;
        list level {
          key level;

          leaf level {
            type level-number;
            description "IS-IS level.";
          }
          container iih {
            leaf in {
              type uint32;
              description "Received IIH PDUs.";
            }
            leaf out {
              type uint32;
              description "Sent IIH PDUs.";
            }
            description "Number of IIH PDUs received/sent.";
          }
          container ish {
            leaf in {
              type uint32;
              description "Received ISH PDUs.";
            }
            leaf out {
              type uint32;
              description "Sent ISH PDUs.";
            }
            description
              "ISH PDUs received/sent.";
          }
          container esh {
            leaf in {
              type uint32;
              description "Received ESH PDUs.";
            }
            leaf out {
              type uint32;
              description "Sent ESH PDUs.";
            }
            description "Number of ESH PDUs received/sent.";
```

```
          }
          container lsp {
            leaf in {
              type uint32;
              description "Received LSP PDUs.";
            }
            leaf out {
              type uint32;
              description "Sent LSP PDUs.";
            }
            description "Number of LSP PDUs received/sent.";
          }
          container psnp {
            leaf in {
              type uint32;
              description "Received PSNP PDUs.";
            }
            leaf out {
              type uint32;
              description "Sent PSNP PDUs.";
            }
            description "Number of PSNP PDUs received/sent.";
          }
          container csnp {
            leaf in {
              type uint32;
              description "Received CSNP PDUs.";
            }
            leaf out {
              type uint32;
              description "Sent CSNP PDUs.";
            }
            description "Number of CSNP PDUs received/sent.";
          }
          container unknown {
            leaf in {
              type uint32;
              description "Received unknown PDUs.";
            }
            leaf out {
              type uint32;
              description "Sent unknown PDUs.";
            }
            description "Number of unknown PDUs received/sent.";
          }
          description
            "List of packet counter for supported levels.";
        }
```

```
        description "Packet counters per IS-IS level.";
      }
      description
        "Grouping for per IS-IS Level packet counters.";
    }

      /* Groupings for various log buffers */
    grouping spf-log {
      container spf-log {
        config false;
        list event {
          key id;

          leaf id {
            type uint32;
            description
              "Event identifier -  purely internal value.";
          }
          leaf spf-type {
            type enumeration {
              enum full {
                description "Full SPF computation.";
              }
              enum route-only {
                description
                  "Route reachability only SPF computation";
              }
            }
            description "Type of SPF computation performed.";
          }
          leaf level {
            type level-number;
            description
              "IS-IS level number for SPF computation";
          }
          leaf schedule-timestamp {
            type yang:timestamp;
            description
              "Timestamp of when the SPF computation was
               scheduled.";
          }
          leaf start-timestamp {
            type yang:timestamp;
            description
              "Timestamp of when the SPF computation started.";
          }
          leaf end-timestamp {
            type yang:timestamp;
```

```
            description
              "Timestamp of when the SPF computation ended.";
          }
          list trigger-lsp {
            key "lsp";
            leaf lsp {
              type lsp-id;
              description
                "LSP ID of the LSP triggering SPF computation.";
            }
            leaf sequence {
              type uint32;
              description
                "Sequence number of the LSP triggering SPF
                 computation";
            }
            description
              "This list includes the LSPs that triggered the
               SPF computation.";
          }
          description
            "List of computation events - implemented as a
             wrapping buffer.";
        }

        description
          "This container lists the SPF computation events.";
      }
      description "Grouping for spf-log events.";
    }

    grouping lsp-log {
      container lsp-log {
        config false;
        list event {
          key id;

          leaf id {
            type uint32;
            description
              "Event identifier -  purely internal value.";
          }
          leaf level {
            type level-number;
            description
              "IS-IS level number for LSP";
          }
          container lsp {
```

```
            leaf lsp {
              type lsp-id;
              description
                "LSP ID of the LSP.";
            }
            leaf sequence {
              type uint32;
              description
                "Sequence number of the LSP.";
            }
            description
              "LSP identification container - either the received
               LSP or the locally generated LSP.";
          }

          leaf received-timestamp {
            type yang:timestamp;
            description
              "This is the timestamp when the LSA was received.
               In case of local LSA update, the timestamp refers
               to the LSA origination time.";
          }

          leaf reason {
            type identityref {
              base lsp-log-reason;
            }
            description "Type of LSP change.";
          }

          description
            "List of LSP events - implemented as a
             wrapping buffer.";
        }

        description
          "This container lists the LSP log.
           Local LSP modifications are also included
           in the list.";

      } description "Grouping for LSP log.";
    }



    /* Groupings for the LSDB description */

    /* Unknown TLV and sub-TLV description */
```

```
    grouping tlv {
      description
        "Type-Length-Value (TLV)";
      leaf type {
        type uint16;
        description "TLV type.";
      }
      leaf length {
        type uint16;
        description "TLV length (octets).";
      }
      leaf value {
        type yang:hex-string;
        description "TLV value.";
      }
    }

    grouping unknown-tlvs {
      description
        "Unknown TLVs grouping - Used for unknown TLVs or
         unknown sub-TLVs.";
      container unknown-tlvs {
        description "All unknown TLVs.";
        list unknown-tlv {
          description "Unknown TLV.";
          uses tlv;
        }
      }
    }

    /* TLVs and sub-TLVs for prefixes */

    grouping prefix-reachability-attributes {
      description
        "Grouping for extended reachability attributes of an
         IPv4 or IPv6 prefix.";

      leaf external-prefix-flag {
        type boolean;
        description "External prefix flag.";
      }
      leaf readvertisement-flag {
        type boolean;
        description "Re-advertisement flag.";
      }
      leaf node-flag {
        type boolean;
        description "Node flag.";
```

```
      }
    }

    grouping prefix-ipv4-source-router-id {
      description
        "Grouping for the IPv4 source router ID of a prefix
         advertisement.";

      leaf ipv4-source-router-id {
        type inet:ipv4-address;
        description "IPv4 Source router ID address.";
      }
    }

    grouping prefix-ipv6-source-router-id {
      description
        "Grouping for the IPv6 source router ID of a prefix
         advertisement.";

      leaf ipv6-source-router-id {
        type inet:ipv6-address;
        description "IPv6 Source router ID address.";
      }
    }

    grouping prefix-attributes-extension {
      description "Prefix extended attributes
                   as defined in RFC7794.";

      uses prefix-reachability-attributes;
      uses prefix-ipv4-source-router-id;
      uses prefix-ipv6-source-router-id;
    }

    grouping prefix-ipv4-std {
      description
        "Grouping for attributes of an IPv4 standard prefix
         as defined in RFC1195.";
      leaf ip-prefix {
        type inet:ipv4-address;
        description "IPv4 prefix address";
      }
      leaf prefix-len {
        type uint8;
        description "IPv4 prefix length (in bits)";
      }
      leaf i-e {
        type boolean;
```

```
        description "Internal or External (I/E) Metric bit value.";
      }
      container default-metric {
        leaf metric {
          type std-metric;
          description "Default IS-IS metric for IPv4 prefix";
        }
        description "IS-IS default metric container.";
      }
      container delay-metric {
        leaf metric {
          type std-metric;
          description "IS-IS delay metric for IPv4 prefix";
        }
        leaf supported {
          type boolean;
          default "false";
          description
            "Indicates whether IS-IS delay metric is supported.";
        }
        description "IS-IS delay metric container.";
      }
      container expense-metric {
        leaf metric {
          type std-metric;
          description "IS-IS expense metric for IPv4 prefix";
        }
        leaf supported {
          type boolean;
          default "false";
          description
            "Indicates whether IS-IS delay metric is supported.";
        }
        description "IS-IS expense metric container.";
      }
      container error-metric {
        leaf metric {
          type std-metric;
          description
            "This leaf describes the IS-IS error metric value";
        }
        leaf supported {
          type boolean;
          default "false";
          description "IS-IS error metric for IPv4 prefix";
        }
        description "IS-IS error metric container.";
      }
```

```
    }

    grouping prefix-ipv4-extended {
      description
        "Grouping for attributes of an IPv4 extended prefix
         as defined in RFC5305.";
      leaf up-down {
        type boolean;
        description  "Value of up/down bit.";
      }
      leaf ip-prefix {
        type inet:ipv4-address;
        description "IPv4 prefix address";
      }
      leaf prefix-len {
        type uint8;
        description "IPv4 prefix length (in bits)";
      }
      leaf metric {
        type wide-metric;
        description "IS-IS wide metric value";
      }
      leaf-list tag {
        type uint32;
        description
          "List of 32-bit tags associated with the IPv4 prefix.";
      }
      leaf-list tag64 {
        type uint64;
        description
          "List of 32-bit tags associated with the IPv4 prefix.";
      }
      uses prefix-attributes-extension;
    }

    grouping prefix-ipv6-extended {
      description "Grouping for attributes of an IPv6 prefix
                   as defined in RFC5308.";
      leaf up-down {
        type boolean;
        description "Value of up/down bit.";
      }
      leaf ip-prefix {
        type inet:ipv6-address;
        description "IPv6 prefix address";
      }
      leaf prefix-len {
        type uint8;
```

```
        description  "IPv4 prefix length (in bits)";
      }
      leaf metric {
        type wide-metric;
        description  "IS-IS wide metric value";
      }
      leaf-list tag {
        type uint32;
        description
          "List of 32-bit tags associated with the IPv4 prefix.";
      }
      leaf-list tag64 {
        type uint64;
        description
          "List of 32-bit tags associated with the IPv4 prefix.";
      }
      uses prefix-attributes-extension;
    }

    /* TLVs and sub-TLVs for neighbors */

    grouping neighbor-link-attributes {
      description
        "Grouping for link attributes as defined
        in RFC5029";
      leaf link-attributes-flags {
        type uint16;
        description
          "Flags for the link attributes";
      }
    }
    grouping neighbor-gmpls-extensions {
      description
        "Grouping for GMPLS attributes of a neighbor as defined
        in RFC5307";
      leaf link-local-id {
        type uint32;
        description
          "Local identifier of the link.";
      }
      leaf remote-local-id {
        type uint32;
        description
          "Remote identifier of the link.";
      }
      leaf protection-capability {
        type uint8;
        description
```

```
          "Describes the protection capabilities
          of the link. This is the value of the
          first octet of the sub-TLV type 20 value.";
        }
        container interface-switching-capability {
          description
            "Interface switching capabilities of the link.";
          leaf switching-capability {
            type uint8;
            description
              "Switching capability of the link.";
          }
          leaf encoding {
            type uint8;
            description
              "Type of encoding of the LSP being used.";
          }
          container max-lsp-bandwidths {
            description "Per priority max LSP bandwidths.";
            list max-lsp-bandwidth {
              leaf priority {
                type uint8 {
                  range "0 .. 7";
                }
                description "Priority from 0 to 7.";
              }
              leaf bandwidth {
                type rt-types:bandwidth-ieee-float32;
                description "max LSP bandwidth.";
              }
              description
                "List of max LSP bandwidths for different
                 priorities.";
            }
          }
          container tdm-specific {
            when "../switching-capability = 100";
            description
              "Switching Capability-specific information applicable
              when switching type is TDM.";

            leaf minimum-lsp-bandwidth {
              type rt-types:bandwidth-ieee-float32;
              description "minimum LSP bandwidth.";
            }
            leaf indication {
              type uint8;
              description
```

```
              "The indication whether the interface supports Standard
              or Arbitrary SONET/SDH.";
          }
        }
        container psc-specific {
          when "../switching-capability >= 1 and
                ../switching-capability <= 4";
          description
            "Switching Capability-specific information applicable
            when switching type is PSC1,PSC2,PSC3 or PSC4.";

          leaf minimum-lsp-bandwidth {
            type rt-types:bandwidth-ieee-float32;
            description "minimum LSP bandwidth.";
          }
          leaf mtu {
            type uint16;
            units bytes;
            description
              "Interface MTU";
          }
        }
      }
    }

    grouping neighbor-extended-te-extensions {
      description
        "Grouping for TE attributes of a neighbor as defined
        in RFC7810";

      container unidirectional-link-delay {
        description
          "Container for the average delay
          from the local neighbor to the remote one.";
        leaf flags {
          type bits {
            bit A {
              position 7;
              description
                "The A bit represents the Anomalous (A) bit.
                The A bit is set when the measured value of
                this parameter exceeds its configured
                maximum threshold.
                The A bit is cleared when the measured value
                falls below its configured reuse threshold.
                If the A bit is clear,
                the value represents steady-state link performance.";
            }
```

```
          }
          description
            "Flags.";
        }
        leaf value {
          type uint32;
          units usec;
          description
            "Delay value expressed in microseconds.";
        }
      }
      container min-max-unidirectional-link-delay {
        description
          "Container for the min and max delay
          from the local neighbor to the remote one.";
        leaf flags {
          type bits {
            bit A {
              position 7;
              description
                "The A bit represents the Anomalous (A) bit.
                The A bit is set when the measured value of
                this parameter exceeds its configured
                maximum threshold.
                The A bit is cleared when the measured value
                falls below its configured reuse threshold.
                If the A bit is clear,
                the value represents steady-state link performance.";
            }
          }
          description
            "Flags.";
        }
        leaf min-value {
          type uint32;
          units usec;
          description
            "Minimum delay value expressed in microseconds.";
        }
        leaf max-value {
          type uint32;
          units usec;
          description
            "Maximum delay value expressed in microseconds.";
        }
      }
      container unidirectional-link-delay-variation {
        description
```

```
          "Container for the average delay variation
          from the local neighbor to the remote one.";
        leaf value {
          type uint32;
          units usec;
          description
            "Delay variation value expressed in microseconds.";
        }
      }
      container unidirectional-link-loss{
        description
          "Container for the packet loss
          from the local neighbor to the remote one.";
        leaf flags {
          type bits {
            bit A {
              position 7;
              description
                "The A bit represents the Anomalous (A) bit.
                The A bit is set when the measured value of
                this parameter exceeds its configured
                maximum threshold.
                The A bit is cleared when the measured value
                falls below its configured reuse threshold.
                If the A bit is clear,
                the value represents steady-state link performance.";
            }
          }
          description
            "Flags.";
        }
        leaf value {
          type uint32;
          units percent;
          description
            "Link packet loss expressed as a percentage
            of the total traffic sent over a configurable interval.";
        }
      }
      container unidirectional-link-residual-bandwidth {
        description
          "Container for the residual bandwidth
          from the local neighbor to the remote one.";
        leaf value {
          type rt-types:bandwidth-ieee-float32;
          units Bps;
          description
            "Residual bandwidth.";
```

```
        }
      }
      container unidirectional-link-available-bandwidth {
        description
          "Container for the available bandwidth
          from the local neighbor to the remote one.";
        leaf value {
          type rt-types:bandwidth-ieee-float32;
          units Bps;
          description
            "Available bandwidth.";
        }
      }
      container unidirectional-link-utilized-bandwidth {
        description
          "Container for the utilized bandwidth
          from the local neighbor to the remote one.";
        leaf value {
          type rt-types:bandwidth-ieee-float32;
          units Bps;
          description
            "Utilized bandwidth.";
        }
      }
    }

    grouping neighbor-te-extensions {
      description
        "Grouping for TE attributes of a neighbor as defined
        in RFC5305";
      leaf admin-group {
        type uint32;
        description
          "Administrative group/Resource Class/Color.";
      }
      container local-if-ipv4-addrs {
        description "All local interface IPv4 addresses.";
        leaf-list local-if-ipv4-addr {
          type inet:ipv4-address;
          description
            "List of local interface IPv4 addresses.";
        }
      }
      container remote-if-ipv4-addrs {
        description "All remote interface IPv4 addresses.";
        leaf-list remote-if-ipv4-addr {
          type inet:ipv4-address;
          description
```

```
            "List of remote interface IPv4 addresses.";
        }
      }
      leaf te-metric {
        type uint32;
        description "TE metric.";
      }
      leaf max-bandwidth {
        type rt-types:bandwidth-ieee-float32;
        description "Maximum bandwidth.";
      }
      leaf max-reservable-bandwidth {
        type rt-types:bandwidth-ieee-float32;
        description "Maximum reservable bandwidth.";
      }
      container unreserved-bandwidths {
        description "All unreserved bandwidths.";
        list unreserved-bandwidth {
          leaf priority {
            type uint8 {
              range "0 .. 7";
            }
            description "Priority from 0 to 7.";
          }
          leaf unreserved-bandwidth {
            type rt-types:bandwidth-ieee-float32;
            description "Unreserved bandwidth.";
          }
          description
            "List of unreserved bandwidths for different
             priorities.";
        }
      }
    }

    grouping neighbor-extended {
      description
       "Grouping for attributes of an IS-IS extended neighbor.";
      leaf neighbor-id {
        type extended-system-id;
        description "system-id of the extended neighbor.";
      }
     container instances {
        description "List of all adjacencies between the local
                     system and the neighbor system-id.";
        list instance {
          key id;
```

```
          leaf id {
            type uint32;
            description "Unique identifier of an instance of a
                         particular neighbor.";
          }
          leaf metric {
            type wide-metric;
            description "IS-IS wide metric for extended neighbor";
          }
          uses neighbor-gmpls-extensions;
          uses neighbor-te-extensions;
          uses neighbor-extended-te-extensions;
          uses neighbor-link-attributes;
          uses unknown-tlvs;
          description "Instance of a particular adjacency.";
        }
      }
    }

    grouping neighbor {
      description  "IS-IS standard neighbor grouping.";
      leaf neighbor-id {
        type extended-system-id;
        description "IS-IS neighbor system-id";
      }
     container instances {
        description "List of all adjacencies between the local
                     system and the neighbor system-id.";
        list instance {
          key id;

          leaf id {
            type uint32;
            description "Unique identifier of an instance of a
                         particular neighbor.";
          }
          leaf i-e {
            type boolean;
            description
              "Internal or External (I/E) Metric bit value";
          }
          container default-metric {
            leaf metric {
              type std-metric;
              description "IS-IS default metric value";
            }
           description "IS-IS default metric container";
          }
```

```
          container delay-metric {
            leaf metric {
              type std-metric;
              description "IS-IS delay metric value";
            }
            leaf supported {
              type boolean;
              default "false";
              description "IS-IS delay metric supported";
            }
            description "IS-IS delay metric container";
          }
          container expense-metric {
            leaf metric {
              type std-metric;
              description "IS-IS delay expense metric value";
            }
            leaf supported {
              type boolean;
              default "false";
              description "IS-IS delay expense metric supported";
            }
            description "IS-IS delay expense metric container";
          }
          container error-metric {
            leaf metric {
              type std-metric;
              description "IS-IS error metric value";
            }
            leaf supported {
              type boolean;
              default "false";
              description "IS-IS error metric supported";
            }
            description "IS-IS error metric container";
          }
          description "Instance of a particular adjacency
                       as defined in ISO10589.";
        }
      }
    }

    /* Top-level TLVs */

    grouping tlv132-ipv4-addresses {
      leaf-list ipv4-addresses {
        type inet:ipv4-address;
        description
```

```
          "List of IPv4 addresses of the IS-IS node - IS-IS
           reference is TLV 132.";
      }
      description "Grouping for TLV132.";
    }
    grouping tlv232-ipv6-addresses {
      leaf-list ipv6-addresses {
        type inet:ipv6-address;
        description
          "List of IPv6 addresses of the IS-IS node - IS-IS
           reference is TLV 232.";
      }
      description "Grouping for TLV232.";
    }
    grouping tlv134-ipv4-te-rid {
      leaf ipv4-te-routerid {
        type inet:ipv4-address;
        description
          "IPv4 Traffic Engineering router ID of the IS-IS node -
           IS-IS reference is TLV 134.";
      }
      description "Grouping for TLV134.";
    }
    grouping tlv140-ipv6-te-rid {
      leaf ipv6-te-routerid {
        type inet:ipv6-address;
        description
          "IPv6 Traffic Engineering router ID of the IS-IS node -
           IS-IS reference is TLV 140.";
      }
      description "Grouping for TLV140.";
    }
    grouping tlv129-protocols {
      leaf-list protocol-supported {
        type uint8;
        description
        "List of supported protocols of the IS-IS node -
         IS-IS reference is TLV 129.";
      }
      description "Grouping for TLV129.";
    }
    grouping tlv137-hostname {
      leaf dynamic-hostname {
        type string;
        description
          "Host Name of the IS-IS node - IS-IS reference
           is TLV 137.";
      }
```

```
      description "Grouping for TLV137.";
    }
    grouping tlv10-authentication {
      container authentication {
        leaf authentication-type {
          type identityref {
              base key-chain:crypto-algorithm;
          }
          description
            "Authentication type to be used with IS-IS node.";
        }
        leaf authentication-key {
          type string;
          description
            "Authentication key to be used. For security reasons,
             the authentication key MUST NOT be presented in
             a clear text format in response to any request
            (e.g., via get, get-config).";
        }
        description
          "IS-IS node authentication information container -
           IS-IS reference is TLV 10.";
      }
      description "Grouping for TLV10.";
    }
    grouping tlv229-mt {
      container mt-entries {
        list topology {
          description
            "List of topologies supported";

          leaf mt-id {
            type uint16 {
              range "0 .. 4095";
            }
            description
              "Multi-Topology identifier of topology.";
          }

          leaf attributes {
            type bits {
              bit overload {
                description
                  "If set, the originator is overloaded,
                   and must be avoided in path calculation.";
              }
              bit attached {
                description
```

```
                  "If set, the originator is attached to
                  another area using the referred metric.";
              }
            }
            description
              "Attributes of the LSP for the associated
               topology.";
          }
        }
        description
          "IS-IS node topology information container -
           IS-IS reference is TLV 229.";
      }
      description "Grouping for TLV229.";
    }

    grouping tlv242-router-capabilities {
      container router-capabilities {
        list router-capability {
            leaf flags {
              type bits {
                bit flooding {
                  position 0;
                  description
                    "If the S bit is set, the IS-IS Router CAPABILITY
                     TLV MUST be flooded across the entire routing
                     domain. If the S bit is clear, the TLV MUST NOT
                     be leaked between levels. This bit MUST NOT
                      be altered during the TLV leaking.";
                }
                bit down {
                  position 1;
                  description
                    "When the IS-IS Router CAPABILITY TLV is leaked
                     from level-2 to level-1, the D bit MUST be set.
                     Otherwise, this bit MUST be clear.  IS-IS Router
                     capability TLVs with the D bit set MUST NOT be
                     leaked from level-1 to level-2 in to prevent
                     TLV looping.";
                }
              }
              description "Router Capability Flags";
            }
            container node-tags {
              if-feature node-tag;
              list node-tag {
                leaf tag {
                  type uint32;
```

```
                  description "Node tag value.";
                }
                description "List of tags.";
              }
              description "Container for node admin tags";
            }

            uses unknown-tlvs;

            leaf binary {
              type binary;
              description
                "Binary encoding of the IS-IS node capabilities";
            }
            description
              "IS-IS node capabilities. This list element may
               be extended with detailed information -  IS-IS
               reference is TLV 242.";
          }
          description "List of router capability TLVs.";
        }
        description "Grouping for TLV242.";
      }

      grouping tlv138-srlg {
        description
          "Grouping for TLV138.";
        container links-srlgs {
          list links {
            leaf neighbor-id {
              type extended-system-id;
              description "system-id of the extended neighbor.";
            }
            leaf flags {
              type uint8;
              description
                "Flags associated with the link.";
            }
            leaf link-local-id {
              type union {
                type inet:ip-address;
                type uint32;
              }
              description
                "Local identifier of the link.
                It could be an IPv4 address or a local identifier.";
            }
            leaf link-remote-id {
```

```
            type union {
              type inet:ip-address;
              type uint32;
            }
            description
              "Remote identifier of the link.
              It could be an IPv4 address or a remotely learned
              identifier.";
          }
          container srlgs {
            description "List of SRLGs.";
            leaf-list srlg {
              type uint32;
              description
                "SRLG value of the link.";
            }
          }
          description
            "SRLG attribute of a link.";
        }
        description
          "List of links with SRLGs";
      }
    }

    /* Grouping for LSDB description */

    grouping lsp-entry {
      description "IS-IS LSP database entry grouping";

      leaf decoded-completed {
        type boolean;
        description "IS-IS LSP body fully decoded.";
      }
      leaf raw-data {
        type yang:hex-string;
        description
          "The hexadecimal representation of the complete LSP in
           network-byte order (NBO) as received or originated.";
      }
      leaf lsp-id {
        type lsp-id;
        description "LSP ID of the LSP";
      }
      leaf checksum {
        type uint16;
        description "LSP checksum";
      }
```

```
      leaf remaining-lifetime {
        type uint16;
        units "seconds";
        description
          "Remaining lifetime (in seconds) until LSP expiration.";
      }
      leaf sequence {
        type uint32;
        description
          "This leaf describes the sequence number of the LSP.";
      }
      leaf attributes {
        type bits {
          bit partitioned {
            description "Originator partition repair supported";
          }
          bit attached-error {
            description
              "If set, the originator is attached to
               another area using the referred metric.";
          }
          bit attached-expense {
            description
              "If set, the originator is attached to
               another area using the referred metric.";
          }
          bit attached-delay {
            description
              "If set, the originator is attached to
               another area using the referred metric.";
          }
          bit attached-default {
            description
              "If set, the originator is attached to
               another area using the referred metric.";
          }
          bit overload {
            description
              "If set, the originator is overloaded,
               and must be avoided in path calculation.";
          }
        }
        description "LSP attributes";
      }

      uses tlv132-ipv4-addresses;
      uses tlv232-ipv6-addresses;
      uses tlv134-ipv4-te-rid;
```

```
      uses tlv140-ipv6-te-rid;
      uses tlv129-protocols;
      uses tlv137-hostname;
      uses tlv10-authentication;
      uses tlv229-mt;
      uses tlv242-router-capabilities;
      uses tlv138-srlg;
      uses unknown-tlvs;

      container is-neighbor {
        list neighbor {
          key neighbor-id;

          uses neighbor;
          description "List of neighbors.";
        }
        description
          "Standard IS neighbors container - IS-IS reference is
           TLV 2.";
      }

      container extended-is-neighbor {
        list neighbor {
          key neighbor-id;

          uses neighbor-extended;
          description
            "List of extended IS neighbors";
        }
        description
          "Standard IS extended neighbors container - IS-IS
           reference is TLV 22";
      }

      container ipv4-internal-reachability {
        list prefixes {
          uses prefix-ipv4-std;
          description "List of prefixes.";
        }
        description
        "IPv4 internal reachability information container - IS-IS
         reference is TLV 128.";
      }

      container ipv4-external-reachability {
        list prefixes {
          uses prefix-ipv4-std;
          description "List of prefixes.";
```

```
        }
        description
          "IPv4 external reachability information container -
           IS-IS reference is TLV 130.";
      }

      container extended-ipv4-reachability {
        list prefixes {
          uses prefix-ipv4-extended;
          uses unknown-tlvs;
          description "List of prefixes.";
        }
        description
          "IPv4 extended reachability information container -
           IS-IS reference is TLV 135.";
      }

      container mt-is-neighbor {
        list neighbor {
          leaf mt-id {
            type uint16 {
              range "0 .. 4095";
            }
            description "Multi-topology (MT) identifier";
          }
          uses neighbor-extended;
          description "List of neighbors.";
        }
        description
          "IS-IS multi-topology neighbor container - IS-IS
           reference is TLV 223.";
      }

      container mt-extended-ipv4-reachability {
        list prefixes {
          leaf mt-id {
            type uint16 {
              range "0 .. 4095";
            }
            description  "Multi-topology (MT) identifier";
          }
          uses prefix-ipv4-extended;
          uses unknown-tlvs;
          description "List of extended prefixes.";
        }
        description
          "IPv4 multi-topology (MT) extended reachability
           information container - IS-IS reference is TLV 235.";
```

```
      }

      container mt-ipv6-reachability {
        list prefixes {
          leaf MT-ID {
            type uint16 {
              range "0 .. 4095";
            }
            description "Multi-topology (MT) identifier";
          }
          uses prefix-ipv6-extended;
          uses unknown-tlvs;
          description "List of IPv6 extended prefixes.";
        }
        description
          "IPv6 multi-topology (MT) extended reachability
           information container - IS-IS reference is TLV 237.";
      }

      container ipv6-reachability {
        list prefixes {
          uses prefix-ipv6-extended;
          uses unknown-tlvs;
          description "List of IPv6 prefixes.";
        }
        description
          "IPv6 reachability information container - IS-IS
           reference is TLV 236.";
      }
    }

    grouping lsdb {
      description "Link State Database (LSDB) grouping";
      container database {
        config false;
        list levels {
          key level;

          leaf level {
            type level-number;
            description "LSDB level number (1 or 2)";
          }
          list lsp {
            key lsp-id;
            uses lsp-entry;
            description "List of LSPs in LSDB";
          }
          description "List of LSPs for the LSDB level container";
```

```
        }
        description "IS-IS Link State database container";
      }
    }



    /* Augmentations */

    augment "/rt:routing/"
      +"rt:ribs/rt:rib/rt:routes/rt:route" {
      when "rt:source-protocol = 'isis:isis'" {
        description "IS-IS-specific route attributes.";
      }
      uses route-content;
      description
        "This augments route object in RIB with IS-IS-specific
         attributes.";
    }


    augment "/if:interfaces/if:interface" {
      leaf clns-mtu {
        type uint16;
        description "CLNS MTU of the interface";
      }
      description "ISO specific interface parameters.";
    }

    augment "/rt:routing/rt:control-plane-protocols/"
      +"rt:control-plane-protocol" {
      when "rt:type = 'isis:isis'" {
        description
          "This augment is only valid when routing protocol
           instance type is 'isis'";
      }
      description
        "This augments a routing protocol instance with IS-IS
         specific parameters.";
      container isis {
        must "count(area-address) > 0" {
          error-message
            "At least one area-address must be configured.";
          description
            "Enforce configuration of at least one area.";
        }
```

```
        uses instance-config;
        uses instance-state;

        container topologies {
          if-feature multi-topology;
          list topology {
            key "name";
            leaf enable {
              type boolean;
              description "Topology enable configuration";
            }
            leaf name {
              type leafref {
                path "../../../../../../rt:ribs/rt:rib/rt:name";
              }
              description
                "Routing Information Base (RIB) corresponding
                 to topology.";
            }

            uses multi-topology-config;

            description "List of topologies";
          }
          description "Multi-topology container";
        }
        container interfaces {
          list interface {
            key "name";
            leaf name {
              type if:interface-ref;

              description
                "Reference to the interface within
                 the routing-instance.";
            }
            uses interface-config;
            uses interface-state;
            container topologies {
              if-feature multi-topology;
              list topology {
                key name;

                leaf name {
                  type leafref {
                    path "../../../../../../../../"+
                      "rt:ribs/rt:rib/rt:name";
                  }
```

```
                  description
                    "Routing Information Base (RIB) corresponding
                     to topology.";
                }
                uses multi-topology-interface-config;
                description "List of interface topologies";
              }
              description "Multi-topology container";
            }
            description "List of IS-IS interfaces.";
          }
          description
            "IS-IS interface specific configuration container";
        }

        description
          "IS-IS configuration/state top-level container";
      }
    }


    /* RPC methods */

    rpc clear-adjacency {
      description
        "This RPC request clears a particular set of IS-IS
         adjacencies. If the operation fails due to an internal
         reason, then the error-tag and error-app-tag should be
         set indicating the reason for the failure.";
      input {

        leaf routing-protocol-instance-name {
          type leafref {
            path "/rt:routing/rt:control-plane-protocols/"
               + "rt:control-plane-protocol/rt:name";
          }
          mandatory "true";
          description
            "Name of the IS-IS protocol instance whose IS-IS
             adjacency is being cleared.

             If the corresponding IS-IS instance doesn't exist,
             then the operation will fail with an error-tag of
             'data-missing' and an error-app-tag of
             'routing-protocol-instance-not-found'.";
        }
        leaf level {
          type level;
```

```
          description
            "IS-IS level of the adjacency to be cleared. If the
             IS-IS level is level-1-2, both level 1 and level 2
             adjacencies would be cleared.

             If the value provided is different from the one
             authorized in the enum type, then the operation
             SHALL fail with an error-tag of 'data-missing' and
             an error-app-tag of 'bad-isis-level'.";
        }
        leaf interface {
          type if:interface-ref;
          description
            "IS-IS interface name.

             If the corresponding IS-IS interface doesn't exist,
             then the operation SHALL fail with an error-tag of
             'data-missing' and an error-app-tag of
             'isis-interface-not-found'.";
        }
      }
    }

    rpc clear-database {
      description
        "This RPC request clears a particular  IS-IS database. If
         the operation fails for an IS-IS internal reason, then
         the error-tag and error-app-tag should be set
         indicating the reason for the failure.";
      input {
        leaf routing-protocol-instance-name {
          type leafref {
            path "/rt:routing/rt:control-plane-protocols/"
               + "rt:control-plane-protocol/rt:name";
          }
          mandatory "true";
          description
            "Name of the IS-IS protocol instance whose IS-IS
             database(s) is/are being cleared.

             If the corresponding IS-IS instance doesn't exist,
             then the operation will fail with an error-tag of
             'data-missing' and an error-app-tag of
             'routing-protocol-instance-not-found'.";
        }
        leaf level {
          type level;
          description
```

```
            "IS-IS level of the adjacency to be cleared. If the
             IS-IS level is level-1-2, both level 1 and level 2
             databases would be cleared.

             If the value provided is different from the one
             authorized in the enum type, then the operation
             SHALL fail with an error-tag of 'data-missing' and
             an error-app-tag of 'bad-isis-level'.";
        }
      }
    }


    /* Notifications */

    notification database-overload {
      uses notification-instance-hdr;

      leaf overload {
        type enumeration {
          enum off {
            description
              "Indicates IS-IS instance has left overload state";
          }
          enum on {
            description
              "Indicates IS-IS instance has entered overload state";
          }

        }
        description "New overload state of the IS-IS instance";
      }
      description
        "This notification is sent when an IS-IS instance
         overload state changes.";
    }

    notification lsp-too-large {
      uses notification-instance-hdr;
      uses notification-interface-hdr;

      leaf pdu-size {
        type uint32;
        description "Size of the LSP PDU";
      }
      leaf lsp-id {
        type lsp-id;
        description "LSP ID";
```

```
      }
      description
        "This notification is sent when we attempt to propagate
         an LSP that is larger than the dataLinkBlockSize for the
         circuit.  The notification generation must be throttled
         with at least 5 seconds between successive
         notifications.";
    }

    notification if-state-change {
      uses notification-instance-hdr;
      uses notification-interface-hdr;

      leaf state {
        type if-state-type;
        description "Interface state.";
      }
      description
        "This notification is sent when an interface
         state change is detected.";
    }

    notification corrupted-lsp-detected {
      uses notification-instance-hdr;
      leaf lsp-id {
        type lsp-id;
        description "LSP ID";
      }
      description
        "This notification is sent when we find that
         an LSP that was stored in memory has become
         corrupted.";
    }

    notification attempt-to-exceed-max-sequence {
      uses notification-instance-hdr;
      leaf lsp-id {
        type lsp-id;
        description "LSP ID";
      }
      description
        "This notification is sent when the system
         wraps the 32-bit sequence counter of an LSP.";
    }

    notification id-len-mismatch {
      uses notification-instance-hdr;
      uses notification-interface-hdr;
```

```
      leaf pdu-field-len {
        type uint8;
        description "Size of the ID length in the received PDU";
      }
      leaf raw-pdu {
        type binary;
        description "Received raw PDU.";
      }
      description
        "This notification is sent when we receive a PDU
         with a different value for the system-id length.
         The notification generation must be throttled
         with at least 5 seconds between successive
         notifications.";
    }

    notification max-area-addresses-mismatch {
      uses notification-instance-hdr;
      uses notification-interface-hdr;

      leaf max-area-addresses {
        type uint8;
        description "Received number of supported areas";
      }
      leaf raw-pdu {
        type binary;
        description "Received raw PDU.";
      }
      description
        "This notification is sent when we receive a PDU
         with a different value for the Maximum Area Addresses.
         The notification generation must be throttled
         with at least 5 seconds between successive
         notifications.";
    }

    notification own-lsp-purge {
      uses notification-instance-hdr;
      uses notification-interface-hdr;
      leaf lsp-id {
        type lsp-id;
        description "LSP ID";
      }
      description
        "This notification is sent when the system receives
         a PDU with its own system-id and zero age.";
    }
```

```
    notification sequence-number-skipped {
      uses notification-instance-hdr;
      uses notification-interface-hdr;
      leaf lsp-id {
        type lsp-id;
        description "LSP ID";
      }
      description
        "This notification is sent when the system receives a
         PDU with its own system-id and different contents. The
         system has to originate the LSP with a higher sequence
         number.";
    }

    notification authentication-type-failure {
      uses notification-instance-hdr;
      uses notification-interface-hdr;
      leaf raw-pdu {
        type binary;
        description "Received raw PDU.";
      }
      description
        "This notification is sent when the system receives a
         PDU with the wrong authentication type field.
         The notification generation must be throttled
         with at least 5 seconds between successive
         notifications.";
    }

    notification authentication-failure {
      uses notification-instance-hdr;
      uses notification-interface-hdr;
      leaf raw-pdu {
        type binary;
        description "Received raw PDU.";
      }
      description
        "This notification is sent when the system receives
         a PDU with the wrong authentication information.
         The notification generation must be throttled
         with at least 5 seconds between successive
         notifications.";
    }

    notification version-skew {
      uses notification-instance-hdr;
      uses notification-interface-hdr;
      leaf protocol-version {
```

```
        type uint8;
        description "Protocol version received in the PDU.";
      }
      leaf raw-pdu {
        type binary;
        description "Received raw PDU.";
      }
      description
        "This notification is sent when the system receives a
         PDU with a different protocol version number.
         The notification generation must be throttled
         with at least 5 seconds between successive
         notifications.";
    }

    notification area-mismatch {
      uses notification-instance-hdr;
      uses notification-interface-hdr;
      leaf raw-pdu {
        type binary;
        description "Received raw PDU.";
      }
      description
        "This notification is sent when the system receives a
         Hello PDU from an IS that does not share any area
         address. The notification generation must be throttled
         with at least 5 seconds between successive
         notifications.";
    }

    notification rejected-adjacency {
      uses notification-instance-hdr;
      uses notification-interface-hdr;
      leaf raw-pdu {
        type binary;
        description
          "Received raw PDU.";
      }
      leaf reason {
        type string {
          length "1..255";
        }
        description
          "The system may provide a reason to reject the
           adjacency. If the reason is not available,
           an empty string will be returned.
           The expected format is a single line text.";
      }
```

```
      description
        "This notification is sent when the system receives a
         Hello PDU from an IS but does not establish an adjacency
         for some reason. The notification generation must be
         throttled with at least 5 seconds between successive
         notifications.";
    }


    notification protocols-supported-mismatch {
      uses notification-instance-hdr;
      uses notification-interface-hdr;
      leaf raw-pdu {
        type binary;
        description "Received raw PDU.";
      }
      leaf-list protocols {
        type uint8;
        description
          "List of protocols supported by the remote system.";
      }
      description
        "This notification is sent when the system receives a
         non-pseudonode LSP that has no matching protocols
         supported. The notification generation must be throttled
         with at least 5 seconds between successive
         notifications.";
    }

    notification lsp-error-detected {
      uses notification-instance-hdr;
      uses notification-interface-hdr;
      leaf lsp-id {
        type lsp-id;
        description "LSP ID.";
      }
      leaf raw-pdu {
        type binary;
        description "Received raw PDU.";
      }
      leaf error-offset {
        type uint32;
        description
          "If the problem is a malformed TLV, the error-offset
           points to the start of the TLV. If the problem is with
           the LSP header, the error-offset points to the errant
           byte";
      }
```

```
      leaf tlv-type {
        type uint8;
        description
          "If the problem is a malformed TLV, the tlv-type is set
           to the type value of the suspicious TLV. Otherwise,
           this leaf is not present.";
      }
      description
        "This notification is sent when the system receives an
         LSP with a parse error. The notification generation must
         be throttled with at least 5 seconds between successive
         notifications.";
    }

    notification adjacency-state-change {
      uses notification-instance-hdr;
      uses notification-interface-hdr;
      leaf neighbor {
        type string {
          length "1..255";
        }
        description
          "Name of the neighbor.
           It corresponds to the hostname associated
           with the system-id of the neighbor in the
           mapping database (RFC5301).
           If the name of the neighbor is
           not available, it is not returned.";
      }
      leaf neighbor-system-id {
        type system-id;
        description "Neighbor system-id";
      }
      leaf state {
        type adj-state-type;

        description "New state of the IS-IS adjacency.";
      }
      leaf reason {
        type string {
          length "1..255";
        }
        description
          "If the adjacency is going to DOWN,  this leaf provides
           a reason for the adjacency going down. The reason is
           provided as a text. If the adjacency is going to UP, no
           reason is provided. The expected format is a single line
           text.";
```

```
      }
      description
        "This notification is sent when an IS-IS adjacency
         moves to Up state or to Down state.";
    }

    notification lsp-received {
      uses notification-instance-hdr;
      uses notification-interface-hdr;

      leaf lsp-id {
        type lsp-id;
        description "LSP ID";
      }
      leaf sequence {
        type uint32;
        description "Sequence number of the received LSP.";
      }
      leaf received-timestamp {
        type yang:timestamp;

        description "Timestamp when the LSP was received.";
      }
      leaf neighbor-system-id {
        type system-id;
        description "Neighbor system-id of LSP sender";
      }
      description
        "This notification is sent when an LSP is received.
         The notification generation must be throttled with at
         least 5 seconds between successive notifications.";
     }

    notification lsp-generation {
      uses notification-instance-hdr;

      leaf lsp-id {
        type lsp-id;
        description "LSP ID";
      }
      leaf sequence {
        type uint32;
        description "Sequence number of the received LSP.";
      }
      leaf send-timestamp {
        type yang:timestamp;

        description "Timestamp when our LSP was regenerated.";
```

```
      }
      description
        "This notification is sent when an LSP is regenerated.
         The notification generation must be throttled with at
         least 5 seconds between successive notifications.";
    }
  }
  <CODE ENDS>
```

7. Security Considerations

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

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

There are a number of data nodes defined in this YANG module that are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) to these data nodes without proper protection can have a negative effect on network operations. For IS-IS, the ability to modify IS-IS configuration will allow the entire IS-IS domain to be compromised including creating adjacencies with unauthorized routers to misroute traffic, isolate routers, or mount a massive Denial-of-Service (DoS) attack. A user should consider all the configuration nodes are sensible.

Some of the readable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. The exposure of the Link State Database (LSDB) will expose the detailed topology of the network including authentication parameters. Implementations MUST NOT provide a configured authentication key in a clear text format in response to any request (e.g., via get, get-config).

For IS-IS authentication, configuration is supported via the specification of key-chain [RFC8177] or the direction specification of key and authentication algorithm. Hence, authentication

configuration using the "auth-table-trailer" case in the "authentication" container inherits the security considerations of [RFC8177]. This includes the considerations with respect to the local storage and handling of authentication keys.

Some of the RPC operations in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control access to these operations. The OSPF YANG module support the "clear-adjacency" and "clear-database" RPCs. If access to either of these is compromised, they can result in temporary network outages be employed to mount DoS attacks.

8. Contributors

Authors would like to thank Kiran Agrahara Sreenivasa, Dean Bogdanovic, Yingzhen Qu, Yi Yang, Jeff Tanstura for their major contributions to the draft.

9. IANA Considerations

The IANA is requested to assign two new URIs from the IETF XML registry ([RFC3688]). Authors are suggesting the following URI:

```
URI: urn:ietf:params:xml:ns:yang:ietf-isis
Registrant Contact: The IESG
XML: N/A, the requested URI is an XML namespace
```

This document also requests one new YANG module name in the YANG Module Names registry ([RFC6020]) with the following suggestion:

```
name: ietf-isis
namespace: urn:ietf:params:xml:ns:yang:ietf-isis
prefix: isis
reference: RFC XXXX
```

10. Change log for ietf-isis YANG module

10.1. From version -29 to version -30

- Set a maximum-length of 255 for the non-best-reason leaf.
- Updated some descriptions.
- Renamed "level-db" container to "levels" for consistency reasons.
- Added some RFC references in the YANG model.

- Leaf default-metric of container prefix-ipv4-std is now a container.
- Changed the description of the authentication and removed the reference to MD5 as a possible way to hide the password.
- Changed leaf type for authentication type within tlv10-authentication container. Moved from string to keychain:crypto-algorithm identity.
- Some admin strings are now limited to a range of 1..255.
- Hostname string in hostname DB is limited to 255 as per RFC5301.
- In FRR stats, "prefix" is now encoded as inet:ip-prefix rather than a string. "alternate" is now encoded as ip-address.

10.2. From version -28 to version -29

- Fixed area-address type pattern. The authorized number of bytes was wrong.

10.3. From version -27 to version -28

- Fixed inaccurate description of level-type leaf.

10.4. From version -26 to version -27

- Fixed XPATH to switching-capability leaf in "when" statement of psc-specific and tdm-specific containers.

10.5. From version -25 to version -26

- Modify BFD reference descriptions.
- Fix indentation.
- "uses interface-state" was missing.

10.6. From version -24 to version -25

- RPC clear-adjacency uses now an interface reference instead of a string.
- Use expanded "address-family" rather than "af" in the model.
- Fix pattern for area-address.

- Removed unnecessary identities.
- Added an instances container and list in the modeling of neighbors in the LSDB. This is to manage the case where there are parallel links between two neighbors.
- Notification instance header: removed routing-instance leaf (alignment with OSPF).
- Notification instance header: changed routing-protocol-name to a reference.
- Notification interface header: changed interface to a reference.
- RPCs: changed instance-state-ref typedef to a leafref. Removed associated typedef.
- Revised some groupings and their names (alignment with OSPF).
- Some description alignment with OSPF.
- auto-cost container alignment with OSPF.
- MT-ID leaf in database description renamed (lower-case) mt-id.
- In the lsp-log grouping, leaf change is renamed reason to align with OSPF.
- Added some link attributes (GMPLS, TE, extended TE, SRLGs) in the LSDB.
- Added extended-system-id typedef which contains the system-id plus the pseudonode number.
- Add support of POI Purge Originator Identification.

10.7. From version -22 to version -24

- Fix revision date of the module.

10.8. From version -21 to version -22

- TE router-id modeling alignment with OSPF.
- Add max-ecmp + feature in spf-control container (alignment with OSPF).

10.9. From version -20 to version -21

- Model revision date fix

10.10. From version -19 to version -20

- Moved to Yang 1.1
- Lower case enumerations
- Add RFC references to features
- Remove segment-routing feature
- Modified BFD activation modeling

10.11. From version -18 to version -19

- Align with draft-ietf-netmod-rfc8022bis.
- Modify address family types as per draft-ietf-rtgwg-routing-types-17.

10.12. From version -17 to version -18

- NMDA compliancy.
- Set some default values.
- Align with iana-rt-types module.

10.13. From version -16 to version -17

- Cosmetic fixes.
- Use of rt-types model.

10.14. From version -15 to version -16

- Alignment with last IETF key chain model.
- lsp-log "change" leaf moved as an identity.
- Incremental SPF removed from spf-log types.

10.15. From version -14 to version -15

- Alignment with OSPF model done:
  - Added spf-control container with IETF SPF delay algorithm as a feature.
  - Added graceful-restart options.
  - Added nsr as a feature.
  - Removed per topology FRR. Need to be augmented if necessary.
  - Created an ldp container within mpls.
  - Renamed igp-ldp-sync to igp-sync.
  - Added auto-cost container.
  - Moved reference-bandwidth under auto-cost container.
  - Added IS-IS local RIB as operational state.
  - Added decode-completed and raw-data leaves in the LSDB model.
  - Modified the notification header.

10.16. From version -13 to version -14

- Segment Routing extensions are now in a separate document.

10.17. From version -12 to version -13

- Move feature nlpid-control to container rather than list.
- Rename multi-topology to topologies to align with OSPF.
- Rename bfd/enabled to bfd/enable for consistency reason.
- Add support for NSR with a feature.

10.18. From version -09 to version -12

- Rename node-tag container to node-tags.

10.19. From version -08 to version -09

- Added container before af list.
- Added container before topology list.
- Aligned LFA if per level cfg.
- Align to draft-ietf-netmod-routing-cfg-23.

10.20. From version -07 to version -08

- Remove selector from system-id type.
- Add some default values.
- Moved lists to containers and groupings for per level configuration.
- Remove routing-instance as per core routing model v21.
- added BFD leaf (no more BFD protocol model).
- changed keychain module reference.

10.21. From version -05 to version -07

- Move Overload config from list to container.
- Move Overload-max-metric config from list to container.
- Move preference config from list to container.
- Add Node flag in config.
- Removed BFD config => moved to isis-bfd module.
- Remove call to routing policy model.

10.22. From version -03 to version -05

- Correct invalid references to previous versions of core routing model.
- Remove BFD config and replace by groupings from ietf-bfd.
- Adding routing-policy support through routing-policy model.

10.23. From version -02 to version -03

- Reviewed config and op state groupings.
- Add default value to lfa candidate-disabled.
- Add enable leaf to isis container to reflect admin state.
- Move to VRF centric only.
- Segment routing is part of a separate module.

10.24. From version -01 to version -02

- Adding IPFRR.
- Adding igp-ldp-sync.
- Adding segment-routing.
- Adding instance reference to operational states.
- Move AF type from string to identity.
- Updated router-capability in LSDB description.
- packet counters moved to interface-packet-counters.
- Added modification information in lsp-log.
- Removing igp-ldp-sync timer in IS-IS.
- Defining hierarchy for operational states.
- Adding clns-mtu.
- Adding key-chain.

10.25. From version -00 to version -01

- Interface metric move from af container to interface container.
- Hello-padding on interface moved to hello-padding-disable with empty type.
- three-way-handshake removed.
- route preference changed to a choice.

- csnp-authentication/psnp-authentication merged to authentication container.
- lsp-gen-interval-exp-delay removed.
- Added overload-max-metric feature.
- overload-max-metric is in a separate container.
- Change hello-padding to container.
- Change bfd to container.
- Make BFD a feature.
- Create mpls-te container and put router-id inside.
- Remove GR helper disable and timers.

11. References

11.1. Normative References

[I-D.ietf-bfd-yang]
Rahman, R., Zheng, L., Jethanandani, M., Networks, J., and G. Mirsky, "YANG Data Model for Bidirectional Forwarding Detection (BFD)", draft-ietf-bfd-yang-17 (work in progress), August 2018.

[ISO-10589]
ISO, , "Intermediate System to Intermediate System intra-domain routing information exchange protocol for use in conjunction with the protocol for providing the connectionless-mode network service (ISO 8473)", International Standard 10589: 2002, Second Edition, 2002.

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

[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>.

[RFC5029] Vasseur, JP. and S. Previdi, "Definition of an IS-IS Link Attribute Sub-TLV", RFC 5029, DOI 10.17487/RFC5029, September 2007, <https://www.rfc-editor.org/info/rfc5029>.

[RFC5120] Przygienda, T., Shen, N., and N. Sheth, "M-ISIS: Multi Topology (MT) Routing in Intermediate System to Intermediate Systems (IS-ISs)", RFC 5120, DOI 10.17487/RFC5120, February 2008, <https://www.rfc-editor.org/info/rfc5120>.

[RFC5130] Previdi, S., Shand, M., Ed., and C. Martin, "A Policy Control Mechanism in IS-IS Using Administrative Tags", RFC 5130, DOI 10.17487/RFC5130, February 2008, <https://www.rfc-editor.org/info/rfc5130>.

[RFC5286] Atlas, A., Ed. and A. Zinin, Ed., "Basic Specification for IP Fast Reroute: Loop-Free Alternates", RFC 5286, DOI 10.17487/RFC5286, September 2008, <https://www.rfc-editor.org/info/rfc5286>.

[RFC5301] McPherson, D. and N. Shen, "Dynamic Hostname Exchange Mechanism for IS-IS", RFC 5301, DOI 10.17487/RFC5301, October 2008, <https://www.rfc-editor.org/info/rfc5301>.

[RFC5305] Li, T. and H. Smit, "IS-IS Extensions for Traffic Engineering", RFC 5305, DOI 10.17487/RFC5305, October 2008, <https://www.rfc-editor.org/info/rfc5305>.

[RFC5306] Shand, M. and L. Ginsberg, "Restart Signaling for IS-IS", RFC 5306, DOI 10.17487/RFC5306, October 2008, <https://www.rfc-editor.org/info/rfc5306>.

[RFC5308] Hopps, C., "Routing IPv6 with IS-IS", RFC 5308, DOI 10.17487/RFC5308, October 2008, <https://www.rfc-editor.org/info/rfc5308>.

[RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <https://www.rfc-editor.org/info/rfc5880>.

[RFC5881] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD) for IPv4 and IPv6 (Single Hop)", RFC 5881, DOI 10.17487/RFC5881, June 2010, <https://www.rfc-editor.org/info/rfc5881>.

[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>.

[RFC6119] Harrison, J., Berger, J., and M. Bartlett, "IPv6 Traffic Engineering in IS-IS", RFC 6119, DOI 10.17487/RFC6119, February 2011, <https://www.rfc-editor.org/info/rfc6119>.

[RFC6232] Wei, F., Qin, Y., Li, Z., Li, T., and J. Dong, "Purge Originator Identification TLV for IS-IS", RFC 6232, DOI 10.17487/RFC6232, May 2011, <https://www.rfc-editor.org/info/rfc6232>.

[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>.

[RFC7490] Bryant, S., Filsfils, C., Previdi, S., Shand, M., and N. So, "Remote Loop-Free Alternate (LFA) Fast Reroute (FRR)", RFC 7490, DOI 10.17487/RFC7490, April 2015, <https://www.rfc-editor.org/info/rfc7490>.

[RFC7794] Ginsberg, L., Ed., Decraene, B., Previdi, S., Xu, X., and U. Chunduri, "IS-IS Prefix Attributes for Extended IPv4 and IPv6 Reachability", RFC 7794, DOI 10.17487/RFC7794, March 2016, <https://www.rfc-editor.org/info/rfc7794>.

[RFC7810] Previdi, S., Ed., Giacalone, S., Ward, D., Drake, J., and Q. Wu, "IS-IS Traffic Engineering (TE) Metric Extensions", RFC 7810, DOI 10.17487/RFC7810, May 2016, <https://www.rfc-editor.org/info/rfc7810>.

[RFC7917] Sarkar, P., Ed., Gredler, H., Hegde, S., Litkowski, S., and B. Decraene, "Advertising Node Administrative Tags in IS-IS", RFC 7917, DOI 10.17487/RFC7917, July 2016, <https://www.rfc-editor.org/info/rfc7917>.

[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>.

[RFC8177] Lindem, A., Ed., Qu, Y., Yeung, D., Chen, I., and J. Zhang, "YANG Data Model for Key Chains", RFC 8177, DOI 10.17487/RFC8177, June 2017, <https://www.rfc-editor.org/info/rfc8177>.

[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>.

[RFC8405] Decraene, B., Litkowski, S., Gredler, H., Lindem, A., Francois, P., and C. Bowers, "Shortest Path First (SPF) Back-Off Delay Algorithm for Link-State IGPs", RFC 8405, DOI 10.17487/RFC8405, June 2018, <https://www.rfc-editor.org/info/rfc8405>.

[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>.

11.2. Informative References

[RFC5443] Jork, M., Atlas, A., and L. Fang, "LDP IGP Synchronization", RFC 5443, DOI 10.17487/RFC5443, March 2009, <https://www.rfc-editor.org/info/rfc5443>.

[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>.

Appendix A. Example of IS-IS configuration in XML

This section gives an example of configuration of an IS-IS instance on a device. The example is written in XML.

```
<?xml version="1.0" encoding="utf-8"?>
<data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <routing xmlns="urn:ietf:params:xml:ns:yang:ietf-routing">
      <name>SLI</name>
      <router-id>1.1.1.1</router-id>
      <control-plane-protocols>
        <control-plane-protocol>
          <name>ISIS-example</name>
          <description/>
          <type>
            <type
xmlns:isis="urn:ietf:params:xml:ns:yang:ietf-isis">
             isis:isis
            </type>
          </type>
          <isis xmlns="urn:ietf:params:xml:ns:yang:ietf-isis">
              <enable>true</enable>
              <level-type>level-2</level-type>
              <system-id>87FC.FCDF.4432</system-id>
              <area-address>49.0001</area-address>
              <mpls>
                <te-rid>
                  <ipv4-router-id>192.0.2.1</ipv4-router-id>
                </te-rid>
              </mpls>
              <lsp-lifetime>65535</lsp-lifetime>
              <lsp-refresh>65000</lsp-refresh>
              <metric-type>
                <value>wide-only</value>
```

```
                </metric-type>
                <default-metric>
                  <value>111111</value>
                </default-metric>
                <address-families>
                  <address-family-list>
                    <address-family>ipv4</address-family>
                    <enable>true</enable>
                  </address-family-list>
                  <address-family-list>
                    <address-family>ipv6</address-family>
                    <enable>true</enable>
                  </address-family-list>
                </address-families>
                <interfaces>
                  <interface>
                    <name>Loopback0</name>
                    <tag>200</tag>
                    <metric>
                      <value>0</value>
                    </metric>
                    <passive>true</passive>
                  </interface>
                  <interface>
                    <name>Eth1</name>
                    <level-type>level-2</level-type>

   <interface-type>point-to-point</interface-type>
                    <metric>
                      <value>167890</value>
                    </metric>
                  </interface>
                </interfaces>
            </isis>
          </control-plane-protocol>
        </control-plane-protocols>
    </routing>
    <interfaces xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
      <interface>
        <name>Loopback0</name>
        <description/>
        <type
   xmlns:ianaift="urn:ietf:params:xml:ns:yang:iana-if-type">
        ianaift:softwareLoopback
        </type>

   <link-up-down-trap-enable>enabled</link-up-down-trap-enable>
        <ipv4 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
```

```
            <address>
              <ip>192.0.2.1</ip>
              <prefix-length>32</prefix-length>
            </address>
          </ipv4>
          <ipv6 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
            <address>
              <ip>2001:DB8::1</ip>
              <prefix-length>128</prefix-length>
            </address>
          </ipv6>
        </interface>
        <interface>
          <name>Eth1</name>
          <description/>
          <type
    xmlns:ianaift="urn:ietf:params:xml:ns:yang:iana-if-type">
          ianaift:ethernetCsmacd
          </type>

    <link-up-down-trap-enable>enabled</link-up-down-trap-enable>
          <ipv4 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
            <address>
              <ip>198.51.100.1</ip>
              <prefix-length>30</prefix-length>
            </address>
          </ipv4>
          <ipv6 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
            <address>
              <ip>2001:DB8:0:0:FF::1</ip>
              <prefix-length>64</prefix-length>
            </address>
          </ipv6>
        </interface>
      </interfaces>
    </data>
```

Authors' Addresses

Stephane Litkowski
Orange

Email: stephane.litkowski@orange.com

Derek Yeung
Arrcus, Inc

Email: derek@arrcus.com

Acee Lindem
Cisco Systems

Email: acee@cisco.com

Jeffrey Zhang
Juniper Networks

Email: zzhang@juniper.net

Ladislav Lhotka
CZ.NIC

Email: lhotka@nic.cz

Networking Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 19, 2019

N. Shen
L. Ginsberg
Cisco Systems
S. Thyamagundalu

October 16, 2018

IS-IS Routing for Spine-Leaf Topology
draft-shen-isis-spine-leaf-ext-07

Abstract

This document describes a mechanism for routers and switches in a Spine-Leaf type topology to have non-reciprocal Intermediate System to Intermediate System (IS-IS) routing relationships between the leafs and spines. The leaf nodes do not need to have the topology information of other nodes and exact prefixes in the network. This extension also has application in the Internet of Things (IoT).

Table of Contents

1. Introduction 3
1.1. Requirements Language 3
2. Motivations 3
3. Spine-Leaf (SL) Extension 4
3.1. Topology Examples 4
3.2. Applicability Statement 5
3.3. Spine-Leaf TLV 6
3.3.1. Spine-Leaf Sub-TLVs 7
3.3.1.1. Leaf-Set Sub-TLV 7
3.3.1.2. Info-Req Sub-TLV 8
3.3.2. Advertising IPv4/IPv6 Reachability 8
3.3.3. Advertising Connection to RF-Leaf Node 8
3.4. Mechanism 8
3.4.1. Pure CLOS Topology 10
3.5. Implementation and Operation 11
3.5.1. CSNP PDU 11
3.5.2. Overload Bit 11
3.5.3. Spine Node Hostname 11
3.5.4. IS-IS Reverse Metric 11
3.5.5. Spine-Leaf Traffic Engineering 12
3.5.6. Other End-to-End Services 12
3.5.7. Address Family and Topology 12
3.5.8. Migration 13
4. IANA Considerations 13
5. Security Considerations 14
6. Acknowledgments 14
7. Document Change Log 14
7.1. Changes to draft-shen-isis-spine-leaf-ext-05.txt 14
7.2. Changes to draft-shen-isis-spine-leaf-ext-04.txt 14
7.3. Changes to draft-shen-isis-spine-leaf-ext-03.txt 14
7.4. Changes to draft-shen-isis-spine-leaf-ext-02.txt 14
7.5. Changes to draft-shen-isis-spine-leaf-ext-01.txt 15
7.6. Changes to draft-shen-isis-spine-leaf-ext-00.txt 15
8. References 15
8.1. Normative References 15
8.2. Informative References 16
Authors' Addresses 17

1. Introduction

The IS-IS routing protocol defined by [ISO10589] has been widely deployed in provider networks, data centers and enterprise campus environments. In the data center and enterprise switching networks, a Spine-Leaf topology is commonly used. This document describes a mechanism where IS-IS routing can be optimized for a Spine-Leaf topology.

In a Spine-Leaf topology, normally a leaf node connects to a number of spine nodes. Data traffic going from one leaf node to another leaf node needs to pass through one of the spine nodes. Also, the decision to choose one of the spine nodes is usually part of equal cost multi-path (ECMP) load sharing. The spine nodes can be considered as gateway devices to reach destinations on other leaf nodes. In this type of topology, the spine nodes have to know the topology and routing information of the entire network, but the leaf nodes only need to know how to reach the gateway devices to which are the spine nodes they are uplinked.

This document describes the IS-IS Spine-Leaf extension that allows the spine nodes to have all the topology and routing information, while keeping the leaf nodes free of topology information other than the default gateway routing information. The leaf nodes do not even need to run a Shortest Path First (SPF) calculation since they have no topology information.

1.1. Requirements Language

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

2. Motivations

- The leaf nodes in a Spine-Leaf topology do not require complete topology and routing information of the entire domain since their forwarding decision is to use ECMP with spine nodes as default gateways

- The spine nodes in a Spine-Leaf topology are richly connected to leaf nodes, which introduces significant flooding duplication if they flood all Link State PDUs (LSPs) to all the leaf nodes. It saves both spine and leaf nodes' CPU and link bandwidth resources if flooding is blocked to leaf nodes. For small Top of the Rack (ToR) leaf switches in data centers, it is meaningful to prevent full topology routing information and massive database flooding through those devices.

- When a spine node advertises a topology change, every leaf node connected to it will flood the update to all the other spine nodes, and those spine nodes will further flood them to all the leaf nodes, causing a O(n^2) flooding storm which is largely redundant.

- Similar to some of the overlay technologies which are popular in data centers, the edge devices (leaf nodes) may not need to contain all the routing and forwarding information on the device's control and forwarding planes. "Conversational Learning" can be utilized to get the specific routing and forwarding information in the case of pure CLOS topology and in the events of link and node down.

- Small devices and appliances of Internet of Things (IoT) can be considered as leafs in the routing topology sense. They have CPU and memory constrains in design, and those IoT devices do not have to know the exact network topology and prefixes as long as there are ways to reach the cloud servers or other devices.

3. Spine-Leaf (SL) Extension

3.1. Topology Examples

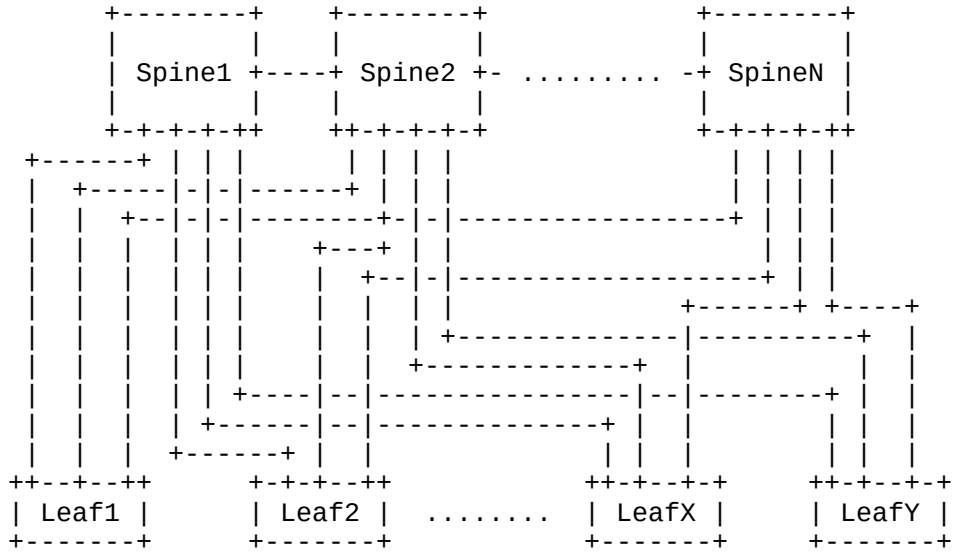


Figure 1: A Spine-Leaf Topology

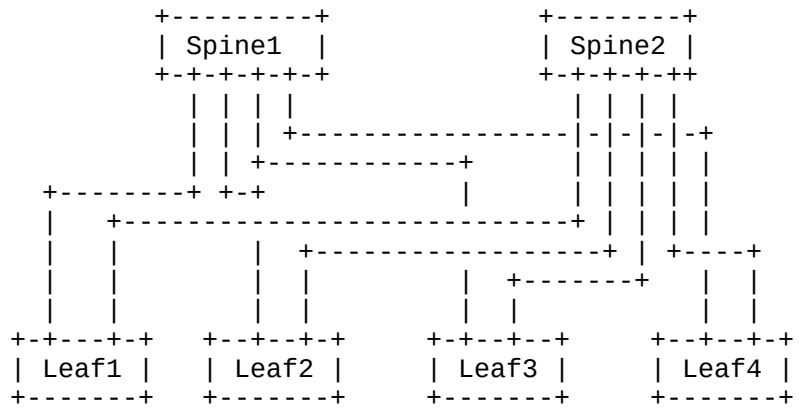


Figure 2: A CLOS Topology

3.2. Applicability Statement

This extension assumes the network is a Spine-Leaf topology, and it should not be applied in an arbitrary network setup. The spine nodes can be viewed as the aggregation layer of the network, and the leaf nodes as the access layer of the network. The leaf nodes use a load sharing algorithm with spine nodes as nexthops in routing and forwarding.

This extension works when the spine nodes are inter-connected, and it works with a pure CLOS or Fat Tree topology based network where the spines are NOT horizontally interconnected.

Although the example diagram in Figure 1 shows a fully meshed Spine-Leaf topology, this extension also works in the case where they are partially meshed. For instance, leaf1 through leaf10 may be fully meshed with spine1 through spine5 while leaf11 through leaf20 is fully meshed with spine4 through spine8, and all the spines are inter-connected in a redundant fashion.

This extension can also work in multi-level spine-leaf topology. The lower level spine node can be a 'leaf' node to the upper level spine node. A spine-leaf 'Tier' can be exchanged with IS-IS hello packets to allow tier X to be connected with tier X+1 using this extension. Normally tier-0 will be the TOR routers and switches if provisioned.

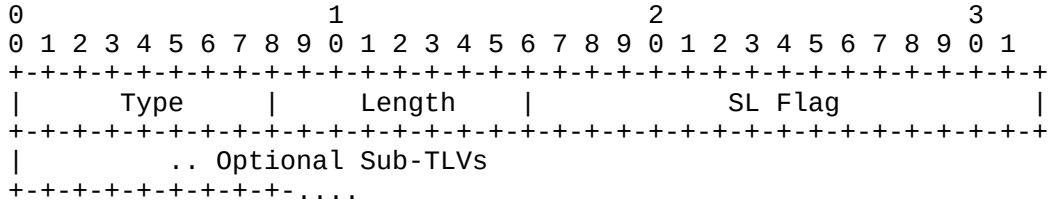
This extension also works with normal IS-IS routing in a topology with more than two layers of spine and leaf. For instance, in example diagrams Figure 1 and Figure 2, there can be another Core layer of routers/switches on top of the aggregation layer. From an IS-IS routing point of view, the Core nodes are not affected by this

extension and will have the complete topology and routing information just like the spine nodes. To make the network even more scalable, the Core layer can operate as a level-2 IS-IS sub-domain while the Spine and Leaf layers operate as stays at the level-1 IS-IS domain.

This extension assumes the link between the spine and leaf nodes are point-to-point, or point-to-point over LAN [RFC5309]. The links connecting among the spine nodes or the links between the leaf nodes can be any type.

3.3. Spine-Leaf TLV

This extension introduces a new TLV, the Spine-Leaf TLV, which may be advertised in IS-IS Hello (IIH) PDUs, LSPs, or in Circuit Scoped Link State PDUs (CS-LSP) [RFC7356]. It is used by both spine and leaf nodes in this Spine-Leaf mechanism.

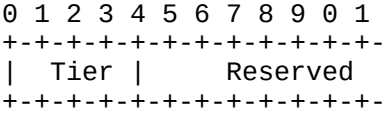


The fields of this TLV are defined as follows:

Type: 1 octet Suggested value 150 (to be assigned by IANA)

Length: 1 octet (2 + length of sub-TLVs).

SL Flags: 16 bits



Tier: A value from 0 to 15. It represents the spine-leaf tier level. The value 15 is reserved to indicate the tier level is unknown. This value is only valid when the 'T' bit (see below) is set. If the 'T' bit is clear, this value MUST be set to zero on transmission, and it MUST be ignored on receipt.

L bit (0x01): Only leaf node sets this bit. If the L bit is set in the SL flag, the node indicates it is in 'Leaf-Mode'.

R bit (0x02): Only Spine node sets this bit. If the R bit is set, the node indicates to the leaf neighbor that it can be used as the default route gateway.

T bit (0x04): If set, the value in the "Tier" field (see above) is valid.

Optional Sub-TLV: Not defined in this document, for future extension

sub-TLVs MAY be included when the TLV is in a CS-LSP. sub-TLVs MUST NOT be included when the TLV is in an IIH

3.3.1. Spine-Leaf Sub-TLVs

If the data center topology is a pure CLOS or Fat Tree, there are no link connections among the spine nodes. If we also assume there is not another Core layer on top of the aggregation layer, then the traffic from one leaf node to another may have a problem if there is a link outage between a spine node and a leaf node. For instance, in the diagram of Figure 2, if Leaf1 sends data traffic to Leaf3 through Spine1 node, and the Spine1-Leaf3 link is down, the data traffic will be dropped on the Spine1 node.

To address this issue spine and leaf nodes may send/request specific reachability information via the sub-TLVs defined below.

Two Spine-Leaf sub-TLVs are defined. The Leaf-Set sub-TLV and the Info-Req sub-TLV.

3.3.1.1. Leaf-Set Sub-TLV

This sub-TLV is used by spine nodes to optionally advertise Leaf neighbors to other Leaf nodes. The fields of this sub-TLV are defined as follows:

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

Length: 1 octet MUST be a multiple of 6 octets.

Leaf-Set: A list of IS-IS System-ID of the leaf node neighbors of this spine node.

3.3.1.2. Info-Req Sub-TLV

This sub-TLV is used by leaf nodes to request the advertisement of more specific prefix information from a selected spine node. The list of leaf nodes in this sub-TLV reflects the current set of leaf-nodes for which not all spine node neighbors have indicated the presence of connectivity in the Leaf-Set sub-TLV (See Section 3.3.1.1). The fields of this sub-TLV are defined as follows:

Type: 1 octet Suggested value 2 (to be assigned by IANA)

Length: 1 octet. It MUST be a multiple of 6 octets.

Info-Req: List of IS-IS System-IDs of leaf nodes for which connectivity information is being requested.

3.3.2. Advertising IPv4/IPv6 Reachability

In cases where connectivity between a leaf node and a spine node is down, the leaf node MAY request reachability information from a spine node as described in Section 3.3.1.2. The spine node utilizes TLVs 135 [RFC5305] and TLVs 236 [RFC5308] to advertise this information. These TLVs MAY be included either in IIHs or CS-LSPs [RFC7356] sent from the spine to the requesting leaf node. Sending such information in IIHs has limited scale - all reachability information MUST fit within a single IIH. It is therefore recommended that CS-LSPs be used.

3.3.3. Advertising Connection to RF-Leaf Node

For links between Spine and Leaf Nodes on which the Spine Node has set the R-bit and the Leaf node has set the L-bit in their respective Spine-Leaf TLVs, spine nodes may advertise the link with a bit in the "link-attribute" sub-TLV [RFC5029] to express this link is not used for LSP flooding. This information can be used by nodes computing a flooding topology e.g., [DYNAMIC-FLOODING], to exclude the RF-Leaf nodes from the computed flooding topology.

3.4. Mechanism

Leaf nodes in a spine-leaf application using this extension are provisioned with two attributes:

1)Tier level of 0. This indicates the node is a Leaf Node. The value 0 is advertised in the Tier field of Spine-Leaf TLV defined above.

2)Flooding reduction enabled/disabled. If flooding reduction is enabled the L-bit is set to one in the Spine-Leaf TLV defined above

A spine node does not need explicit configuration. Spine nodes can dynamically discover their tier level by computing the number of hops to a leaf node. Until a spine node determines its tier level it MUST advertise level 15 (unknown tier level) in the Spine-Leaf TLV defined above. Each tier level can also be statically provisioned on the node.

When a spine node receives an IIH which includes the Spine-Leaf TLV with Tier level 0 and 'L' bit set, it labels the point-to-point interface and adjacency to be a 'Reduced Flooding Leaf-Peer (RF-Leaf)'. IIHs sent by a spine node on a link to an RF-Leaf include the Spine-Leaf TLV with the 'R' bit set in the flags field. The 'R' bit indicates to the RF-Leaf neighbor that the spine node can be used as a default routing nexthop.

There is no change to the IS-IS adjacency bring-up mechanism for Spine-Leaf peers.

A spine node blocks LSP flooding to RF-Leaf adjacencies, except for the LSP PDUs in which the IS-IS System-ID matches the System-ID of the RF-Leaf neighbor. This exception is needed since when the leaf node reboots, the spine node needs to forward to the leaf node non-purged LSPs from the RF-Leaf's previous incarnation.

Leaf nodes will perform IS-IS LSP flooding as normal over all of its IS-IS adjacencies, but in the case of RF-Leafs only self-originated LSPs will exist in its LSP database.

Spine nodes will receive all the LSP PDUs in the network, including all the spine nodes and leaf nodes. It will perform Shortest Path First (SPF) as a normal IS-IS node does. There is no change to the route calculation and forwarding on the spine nodes.

The LSPs of a node only floods north bound towards the upper layer spine nodes. The default route is generated with loadsharing also towards the upper layer spine nodes.

RF-Leaf nodes do not have any LSP in the network except for its own. Therefore there is no need to perform SPF calculation on the RF-Leaf node. It only needs to download the default route with the nexthops of those Spine Neighbors which have the 'R' bit set in the Spine-Leaf TLV in IIH PDUs. IS-IS can perform equal cost or unequal cost load sharing while using the spine nodes as nexthops. The aggregated metric of the outbound interface and the 'Reverse Metric' [REVERSE-METRIC] can be used for this purpose.

3.4.1. Pure CLOS Topology

In a data center where the topology is pure CLOS or Fat Tree, there is no interconnection among the spine nodes, and there is not another Core layer above the aggregation layer with reachability to the leaf nodes. When flooding reduction to RF-Leafs is in use, if the link between a spine and a leaf goes down, there is then a possibility of black holing the data traffic in the network.

As in the diagram Figure 2, if the link Spine1-Leaf3 goes down, there needs to be a way for Leaf1, Leaf2 and Leaf4 to avoid the Spine1 if the destination of data traffic is to Leaf3 node.

In the above example, the Spine1 and Spine2 are provisioned to advertise the Leaf-Set sub-TLV of the Spine-Leaf TLV. Originally both Spines will advertise Leaf1 through Leaf4 as their Leaf-Set. When the Spine1-Leaf3 link is down, Spine1 will only have Leaf1, Leaf2 and Leaf4 in its Leaf-Set. This allows the other leaf nodes to know that Spine1 has lost connectivity to the leaf node of Leaf3.

Each RF-Leaf node can select another spine node to request for some prefix information associated with the lost leaf node. In this diagram of Figure 2, there are only two spine nodes (Spine-Leaf topology can have more than two spine nodes in general). Each RF-Leaf node can independently select a spine node for the leaf information. The RF-Leaf nodes will include the Info-Req sub-TLV in the Spine-Leaf TLV in hellos sent to the selected spine node, Spine2 in this case.

The spine node, upon receiving the request from one or more leaf nodes, will find the IPv6/IPv4 prefixes advertised by the leaf nodes listed in the Info-Req sub-TLV. The spine node will use the mechanism defined in Section 3.3.2 to advertise these prefixes to the RF-Leaf node. For instance, it will include the IPv4 loopback prefix of leaf3 based on the policy configured or administrative tag attached to the prefixes. When the leaf nodes receive the more specific prefixes, they will install the advertised prefixes towards the other spine nodes (Spine2 in this example).

For instance in the data center overlay scenario, when any IP destination or MAC destination uses the leaf3’s loopback as the tunnel nexthop, the overlay tunnel from leaf nodes will only select Spine2 as the gateway to reach leaf3 as long as the Spine1-Leaf3 link is still down.

In cases where multiple links or nodes fail at the same time, the RF-leaf node may need to send the Info-Req to multiple upper layer spine

nodes in order to obtain reachability information for all the partially connected nodes.

This negative routing is more useful between tier 0 and tier 1 spine-leaf levels in a multi-level spine-leaf topology when the reduced flooding extension is in use. Nodes in tiers 1 or greater may have much richer topology information and alternative paths.

3.5. Implementation and Operation

3.5.1. CSNP PDU

In Spine-Leaf extension, Complete Sequence Number PDU (CSNP) does not need to be transmitted over the Spine-Leaf link to an RF-Leaf. Some IS-IS implementations send periodic CSNPs after the initial adjacency bring-up over a point-to-point interface. There is no need for this optimization here since the RF-Leaf does not need to receive any other LSPs from the network, and the only LSPs transmitted across the Spine-Leaf link is the leaf node LSP.

Also in the graceful restart case[RFC5306], for the same reason, there is no need to send the CSNPs over the Spine-Leaf interface to an RF-Leaf. Spine nodes only need to set the SRMflag on the LSPs belonging to the RF-Leaf.

3.5.2. Overload Bit

The leaf node SHOULD set the ’overload’ bit on its LSP PDU, since if the spine nodes were to forward traffic not meant for the local node, the leaf node does not have the topology information to prevent a routing/forwarding loop.

3.5.3. Spine Node Hostname

This extension creates a non-reciprocal relationship between the spine node and leaf node. The spine node will receive leaf’s LSP and will know the leaf’s hostname, but the leaf does not have spine’s LSP. This extension allows the Dynamic Hostname TLV [RFC5301] to be optionally included in spine’s IIH PDU when sending to a ’Leaf-Peer’. This is useful in troubleshooting cases.

3.5.4. IS-IS Reverse Metric

This metric is part of the aggregated metric for leaf’s default route installation with load sharing among the spine nodes. When a spine node is in ’overload’ condition, it should use the IS-IS Reverse Metric TLV in IIH [REVERSE-METRIC] to set this metric to maximum to discourage the leaf using it as part of the loadsharing.

In some cases, certain spine nodes may have less bandwidth in link provisioning or in real-time condition, and it can use this metric to signal to the leaf nodes dynamically.

In other cases, such as when the spine node loses a link to a particular leaf node, although it can redirect the traffic to other spine nodes to reach that destination leaf node, but it MAY want to increase this metric value if the inter-spine connection becomes over utilized, or the latency becomes an issue.

In the leaf-leaf link as a backup gateway use case, the 'Reverse Metric' SHOULD always be set to very high value.

3.5.5. Spine-Leaf Traffic Engineering

Besides using the IS-IS Reverse Metric by the spine nodes to affect the traffic pattern for leaf default gateway towards multiple spine nodes, the IPv6/IPv4 Info-Advertise sub-TLVs can be selectively used by traffic engineering controllers to move data traffic around the data center fabric to alleviate congestion and to reduce the latency of a certain class of traffic pairs. By injecting more specific leaf node prefixes, it will allow the spine nodes to attract more traffic on some underutilized links.

3.5.6. Other End-to-End Services

Losing the topology information will have an impact on some of the end-to-end network services, for instance, MPLS TE or end-to-end segment routing. Some other mechanisms such as those described in PCE [RFC4655] based solution may be used. In this Spine-Leaf extension, the role of the leaf node is not too much different from the multi-level IS-IS routing while the level-1 IS-IS nodes only have the default route information towards the node which has the Attach Bit (ATT) set, and the level-2 backbone does not have any topology information of the level-1 areas. The exact mechanism to enable certain end-to-end network services in Spine-Leaf network is outside the scope of this document.

3.5.7. Address Family and Topology

IPv6 Address families[RFC5308], Multi-Topology (MT)[RFC5120] and Multi-Instance (MI)[RFC8202] information is carried over the IIH PDU. Since the goal is to simplify the operation of IS-IS network, for the simplicity of this extension, the Spine-Leaf mechanism is applied the same way to all the address families, MTs and MIs.

3.5.8. Migration

For this extension to be deployed in existing networks, a simple migration scheme is needed. To support any leaf node in the network, all the involved spine nodes have to be upgraded first. So the first step is to migrate all the involved spine nodes to support this extension, then the leaf nodes can be enabled with 'Leaf-Mode' one by one. No flag day is needed for the extension migration.

4. IANA Considerations

A new TLV codepoint is defined in this document and needs to be assigned by IANA from the "IS-IS TLV Codepoints" registry. It is referred to as the Spine-Leaf TLV and the suggested value is 150. This TLV is only to be optionally inserted either in the IIH PDU or in the Circuit Flooding Scoped LSP PDU. IANA is also requested to maintain the SL-flag bit values in this TLV, and 0x01, 0x02 and 0x04 bits are defined in this document.

Value	Name	IIH	LSP	SNP	Purge	CS-LSP
150	Spine-Leaf	y	y	n	n	y

This extension also proposes to have the Dynamic Hostname TLV, already assigned as code 137, to be allowed in IIH PDU.

Value	Name	IIH	LSP	SNP	Purge
137	Dynamic Name	y	y	n	y

Two new sub-TLVs are defined in this document and needs to be added assigned by IANA from the "IS-IS TLV Codepoints". They are referred to in this document as the Leaf-Set sub-TLV and the Info-Req sub-TLV. It is suggested to have the values 1 and 2 respectively.

This document also requests that IANA allocate from the registry of link-attribute bit values for sub-TLV 19 of TLV 22 (Extended IS reachability TLV). This new bit is referred to as the "Connect to RF-Leaf Node" bit.

Value	Name	Reference
0x3	Connect to RF-Leaf Node	This document

5. Security Considerations

Security concerns for IS-IS are addressed in [ISO10589], [RFC5304], [RFC5310], and [RFC7602]. This extension does not raise additional security issues.

6. Acknowledgments

The authors would like to thank Tony Przygienda for his discussion and contributions. The authors also would like to thank Acee Lindem, Russ White and Christian Hopps for their review and comments of this document.

7. Document Change Log

7.1. Changes to draft-shen-isis-spine-leaf-ext-05.txt

- Submitted January 2018.
- Just a refresh.

7.2. Changes to draft-shen-isis-spine-leaf-ext-04.txt

- Submitted June 2017.
- Added the Tier level information to handle the multi-level spine-leaf topology using this extension.

7.3. Changes to draft-shen-isis-spine-leaf-ext-03.txt

- Submitted March 2017.
- Added the Spine-Leaf sub-TLVs to handle the case of data center pure CLOS topology and mechanism.
- Added the Spine-Leaf TLV and sub-TLVs can be optionally inserted in either IIH PDU or CS-LSP PDU.
- Allow use of prefix Reachability TLVs 135 and 236 in IIHs/CS-LSPs sent from spine to leaf.

7.4. Changes to draft-shen-isis-spine-leaf-ext-02.txt

- Submitted October 2016.
- Removed the ’Default Route Metric’ field in the Spine-Leaf TLV and changed to using the IS-IS Reverse Metric in IIH.

7.5. Changes to draft-shen-isis-spine-leaf-ext-01.txt

- Submitted April 2016.
- No change. Refresh the draft version.

7.6. Changes to draft-shen-isis-spine-leaf-ext-00.txt

- Initial version of the draft is published in November 2015.

8. References

8.1. Normative References

[ISO10589] ISO "International Organization for Standardization", "Intermediate system to Intermediate system intra-domain routeing information exchange protocol for use in conjunction with the protocol for providing the connectionless-mode Network Service (ISO 8473), ISO/IEC 10589:2002, Second Edition.", Nov 2002.

[REVERSE-METRIC] Shen, N., Amante, S., and M. Abrahamsson, "IS-IS Routing with Reverse Metric", draft-ietf-isis-reverse-metric-07 (work in progress), 2017.

[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>.

[RFC5029] Vasseur, JP. and S. Previdi, "Definition of an IS-IS Link Attribute Sub-TLV", RFC 5029, DOI 10.17487/RFC5029, September 2007, <https://www.rfc-editor.org/info/rfc5029>.

[RFC5120] Przygienda, T., Shen, N., and N. Sheth, "M-ISIS: Multi Topology (MT) Routing in Intermediate System to Intermediate Systems (IS-ISs)", RFC 5120, DOI 10.17487/RFC5120, February 2008, <https://www.rfc-editor.org/info/rfc5120>.

[RFC5301] McPherson, D. and N. Shen, "Dynamic Hostname Exchange Mechanism for IS-IS", RFC 5301, DOI 10.17487/RFC5301, October 2008, <https://www.rfc-editor.org/info/rfc5301>.

[RFC5304] Li, T. and R. Atkinson, "IS-IS Cryptographic Authentication", RFC 5304, DOI 10.17487/RFC5304, October 2008, <https://www.rfc-editor.org/info/rfc5304>.

[RFC5305] Li, T. and H. Smit, "IS-IS Extensions for Traffic Engineering", RFC 5305, DOI 10.17487/RFC5305, October 2008, <https://www.rfc-editor.org/info/rfc5305>.

[RFC5306] Shand, M. and L. Ginsberg, "Restart Signaling for IS-IS", RFC 5306, DOI 10.17487/RFC5306, October 2008, <https://www.rfc-editor.org/info/rfc5306>.

[RFC5308] Hopps, C., "Routing IPv6 with IS-IS", RFC 5308, DOI 10.17487/RFC5308, October 2008, <https://www.rfc-editor.org/info/rfc5308>.

[RFC5310] Bhatia, M., Manral, V., Li, T., Atkinson, R., White, R., and M. Fanto, "IS-IS Generic Cryptographic Authentication", RFC 5310, DOI 10.17487/RFC5310, February 2009, <https://www.rfc-editor.org/info/rfc5310>.

[RFC7356] Ginsberg, L., Previdi, S., and Y. Yang, "IS-IS Flooding Scope Link State PDUs (LSPs)", RFC 7356, DOI 10.17487/RFC7356, September 2014, <https://www.rfc-editor.org/info/rfc7356>.

[RFC7602] Chunduri, U., Lu, W., Tian, A., and N. Shen, "IS-IS Extended Sequence Number TLV", RFC 7602, DOI 10.17487/RFC7602, July 2015, <https://www.rfc-editor.org/info/rfc7602>.

[RFC8202] Ginsberg, L., Previdi, S., and W. Henderickx, "IS-IS Multi-Instance", RFC 8202, DOI 10.17487/RFC8202, June 2017, <https://www.rfc-editor.org/info/rfc8202>.

8.2. Informative References

[DYNAMIC-FLOODING] Li, T., "Dynamic Flooding on Dense Graphs", draft-li-dynamic-flooding (work in progress), 2018.

[RFC4655] Farrel, A., Vasseur, J., and J. Ash, "A Path Computation Element (PCE)-Based Architecture", RFC 4655, DOI 10.17487/RFC4655, August 2006, <https://www.rfc-editor.org/info/rfc4655>.

[RFC5309] Shen, N., Ed. and A. Zinin, Ed., "Point-to-Point Operation over LAN in Link State Routing Protocols", RFC 5309, DOI 10.17487/RFC5309, October 2008, <https://www.rfc-editor.org/info/rfc5309>.

Authors' Addresses

Naiming Shen
Cisco Systems
560 McCarthy Blvd.
Milpitas, CA 95035
US

Email: naiming@cisco.com

Les Ginsberg
Cisco Systems
821 Alder Drive
Milpitas, CA 95035
US

Email: ginsberg@cisco.com

Sanjay Thyamagundalu

Email: tsanjay@gmail.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: July 13, 2018

X. Xu
Huawei
L. Fang
Expedia, Inc
J. Tantsura
Individual
S. Ma
Juniper

January 9, 2018

IS-IS Flooding Reduction in MSDC
draft-xu-isis-flooding-reduction-in-msdc-03

Abstract

IS-IS is commonly used as an underlay routing protocol for MSDC (Massively Scalable Data Center) networks. For a given IS-IS router within the CLOS topology, it would receive multiple copies of exactly the same LSP from multiple IS-IS neighbors. In addition, two IS-IS neighbors may send each other the same LSP simultaneously. The unneccessary link-state information flooding wastes the precious process resource of IS-IS routers greatly due to the fact that there are too many IS-IS neighbors for each IS-IS router within the CLOS topology. This document proposes some extensions to IS-IS so as to reduce the IS-IS flooding within MSDC networks greatly. The reduction of the IS-IS flooding is much beneficial to improve the scalability of MSDC networks.

Status of This Memo

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

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

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

This Internet-Draft will expire on July 13, 2018.

Table of Contents

1. Introduction 2
1.1. Requirements Language 4
2. Terminology 4
3. Modifications to Current IS-IS Behaviors 4
3.1. IS-IS Routers as Non-DIS 4
3.2. Controllers as DIS 5
4. Acknowledgements 5
5. IANA Considerations 5
6. Security Considerations 5
7. References 5
7.1. Normative References 5
7.2. Informative References 6
Authors' Addresses 6

1. Introduction

IS-IS is commonly used as an underlay routing protocol for Massively Scalable Data Center (MSDC) networks where CLOS is the most popular toplogy. For a given IS-IS router within the CLOS topology, it would receive multiple copies of exactly the same LSP from multiple IS-IS neighbors. In addition, two IS-IS neighbors may send each other the same LSP simultaneously. The unnecessary link-state information flooding wastes the precious process resource of IS-IS routers greatly and therefore IS-IS could not scale very well in MSDC networks.

To simplify the network management task, centralized controllers are becoming fundamental network elements in most MSDCs. One or more controllers are usually connected to all routers within the MSDC network via a Local Area Network (LAN) which is dedicated for network management purpose (called management LAN), as shown in Figure 1.

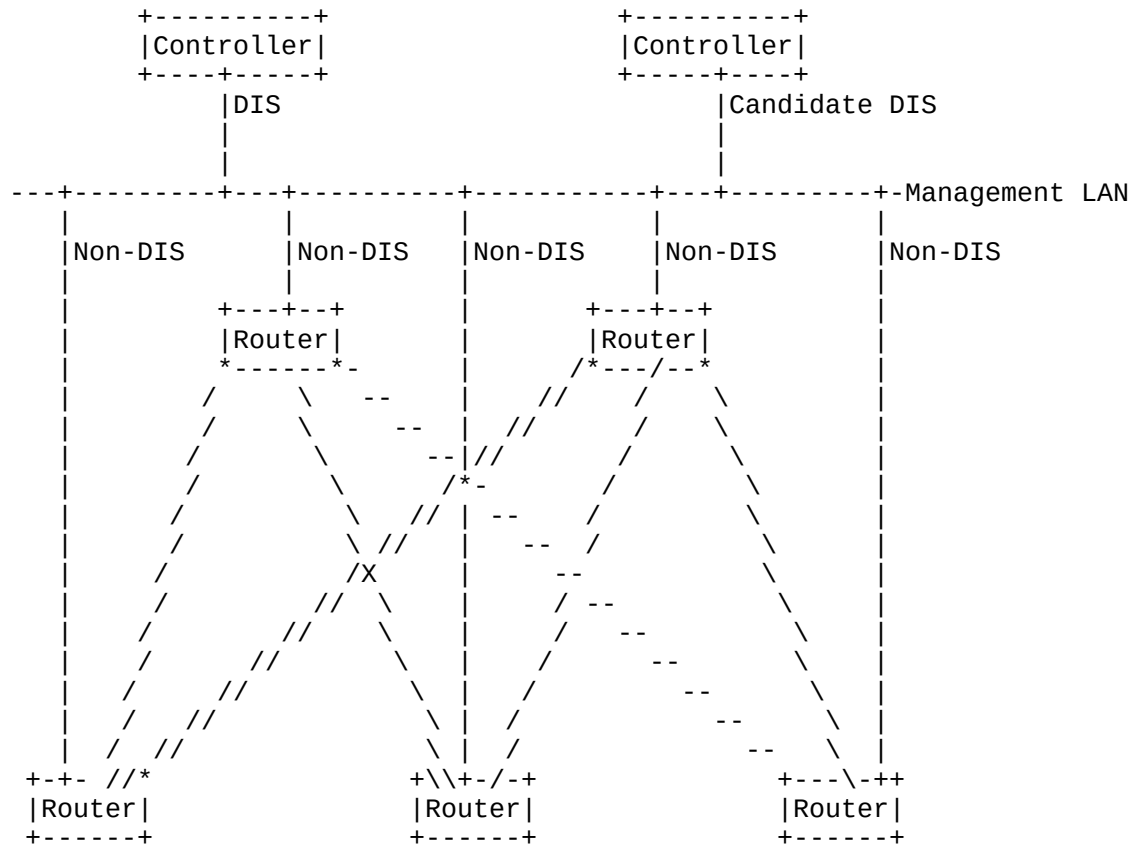


Figure 1

With the assistance of a controller acting as IS-IS Designated Intermediate System (DIS) for the management LAN, IS-IS routers within the MSDC network don't need to exchange any IS-IS Protocl Datagram Units (PDUs) other than Hello packets among them. In order to obtain the full topology information (i.e., the fully synchronized link-state database) of the MSDC's network, these IS-IS routers would exchange the link-state information with the controller being elected as IS-IS DIS for the management LAN instead.

To further suppress the flooding of multicast IS-IS PDUs originated from IS-IS routers over the management LAN, IS-IS routers would not send multicast IS-IS Hello packets over the management LAN. Insteads, they just wait for IS-IS Hello packets originated from the controller being elected as IS-IS DIS initially. Once an IS-IS DIS for the management LAN has been discovered, they start to send IS-IS Hello packets directly (as unicasts) to the IS-IS DIS periodically.

In addition, IS-IS routers would send IS-IS PDUs to the IS-IS DIS for the management LAN as unicasts as well. In contrast, the controller being elected as IS-IS DIS would send IS-IS PDUs as before. As a result, IS-IS routers would not receive IS-IS PDUs from one another unless these IS-IS PDUs are forwarded as unknown unicasts over the management LAN. Through the above modifications to the current IS-IS router behaviors, the IS-IS flooding is greatly reduced, which is much beneficial to improve the scalability of MSDC networks.

1.1. Requirements Language

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

2. Terminology

This memo makes use of the terms defined in [RFC1195].

3. Modifications to Current IS-IS Behaviors

3.1. IS-IS Routers as Non-DIS

After the bidirectional exchange of IS-IS Hello packets among IS-IS routers, IS-IS routers would originate Link State PDUs (LSPs) accordingly. However, these self-originated LSPs need not to be exchanged directly among them anymore. Instead, these LSPs just need to be sent solely to the controller being elected as IS-IS DIS for the management LAN.

To further reduce the flood of multicast IS-IS PDUs over the management LAN, IS-IS routers SHOULD send IS-IS PDUs as unicasts. More specifically, IS-IS routers SHOULD send unicast IS-IS Hello packets periodically to the controller being elected as IS-IS DIS. In other words, IS-IS routers would not send any IS-IS Hello packet over the management LAN until they have found an IS-IS DIS for the management LAN. Note that IS-IS routers SHOULD NOT be elected as IS-IS DIS for the management LAN (This is done by setting the DIS Priority of those IS-IS routers to zero). As a result, IS-IS routers would not see each other over the management LAN. In other word, IS-IS routers would not establish adjacencies with one other. Furthermore, IS-IS routers SHOULD send all the types of IS-IS PDUs to the controller being elected as IS-IS DIS as unicasts as well.

To advoid the data traffic from being forwarded across the management LAN, the cost of all IS-IS routers' interfaces to the management LAN SHOULD be set to the maximum value.

When a given IS-IS router lost its connection to the management LAN, it SHOULD actively establish adjacency with all of its IS-IS neighbors within the CLOS network. As such, it could obtain the full LSDB of the CLOS network while flooding its self-originated LSPs to the remaining part of the whole CLOS network through these IS-IS neighbor.

3.2. Controllers as DIS

The controller being elected as IS-IS DIS would send IS-IS PDUs as multicasts or unicasts as before. And it SHOULD accept and process those unicast IS-IS PDUs originated from IS-IS routers. Upon receiving any new LSP from a given IS-IS router, the controller being elected as DIS MUST flood it immediately to the management LAN for two purposes: 1) implicitly acknowledging the receipt of that LSP; 2) synchronizing that LSP to all the other IS-IS routers.

Furthermore, to decrease the frequency of advertising Complete Sequence Number PDU (CSNP) on the controller being elected as DIS, it’s RECOMMENDED that IS-IS routers SHOULD send an explicit acknowledgement with a Partial Sequence Number PDU (PSNP) upon receiving a new LSP from the controller being elected as DIS.

4. Acknowledgements

The authors would like to thank Peter Lothberg and Erik Auerswald for his valuable comments and suggestions on this document.

5. IANA Considerations

TBD.

6. Security Considerations

TBD.

7. References

7.1. Normative References

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

[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>.

7.2. Informative References

[RFC4136] Pillay-Esnault, P., "OSPF Refresh and Flooding Reduction in Stable Topologies", RFC 4136, DOI 10.17487/RFC4136, July 2005, <https://www.rfc-editor.org/info/rfc4136>.

Authors' Addresses

Xiaohu Xu
Huawei

Email: xuxh.mail@gmail.com

Luyuan Fang
Expedia, Inc

Email: luyuanf@gmail.com

Jeff Tantsura
Individual

Email: jefftant@gmail.com

Shaowen Ma
Juniper

Email: mashao@juniper.net