Networking Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 7, 2019

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

March 6, 2019

IS-IS Extensions to Support Routing over IPv6 Dataplane
draft-bashandy-isis-srv6-extensions-05.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

Copyright Notice

Table of Contents

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:

```
    0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |   Type        |    Length     |            Flags              |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |   optional sub-sub-TLVs...
```

Type: Suggested value 25, to be assigned by IANA

Length: 2 + length of sub-sub-TLVs

Flags: 2 octets The following flags are defined:

```
       0                   1
        0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       | |O|                           |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

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

o Another locator associated with a different topology/algorithm is the longest match

o 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:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Type        |     Length    |R|R|R|R|    MTID               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Followed by one or more locator entries of the form:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Metric                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Flags       | Algorithm     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Loc Size     | Locator (variable)...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Sub-tlv-len  |         Sub-TLVs (variable) . . .             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

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

```
  0
  0 1 2 3 4 5 6 7
 +-+-+-+-+-+-+-+-+
 |D|A|  Reserved |
 +-+-+-+-+-+-+-+-+
```

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.

A bit: When the Locator is configured as anycast, the A bit SHOULD be set. Otherwise, this bit MUST be clear.

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:

```
       0                   1                   2                   3
       0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |    Type       |     Length    |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |    Flags      | SRv6 Endpoint Function        |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      | SID (128 bits) . . .                                          |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      | SID (cont . . .)                                              |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      | SID (cont . . .)                                              |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      | SID (cont . . .)                                              |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |Sub-sub-tlv-len|         sub-sub-TLVs (variable) . . .         |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

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:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Type        |     Length    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Flags       |  Algorithm    |  Weight       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    SRv6 Endpoint Function     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| SID (128 bits) . . .                                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| SID (cont . . .)                                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| SID (cont . . .)                                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| SID (cont . . .)                                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Sub-sub-tlv-len|         Sub-sub-TLVs (variable) . . .         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

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

Length: variable.

Flags: 1 octet.

```
    0 1 2 3 4 5 6 7
   +-+-+-+-+-+-+-+-+
   |B|S|P|Reserved |
   +-+-+-+-+-+-+-+-+
```

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:

```
       0                   1                   2                   3
       0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |   Type        |     Length    |   System ID (6 octets)        |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               |
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |   Flags       |   Algorithm   |  Weight       |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |    SRv6 Endpoint Function     |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      | SID (128 bits) . . .                                          |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      | SID (cont . . .)                                              |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      | SID (cont . . .)                                              |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      | SID (cont . . .)                                              |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |Sub-sub-tlv-len|      sub-sub-TLVs (variable) . . .            |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

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.

```
          0 1 2 3 4 5 6 7
         +-+-+-+-+-+-+-+-+
         |B|S|P|Reserved |
         +-+-+-+-+-+-+-+-+
```

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 Behaviors

Endpoint behaviors are defined in [I-D.filsfils-spring-srv6-network-programming] and [I-D.ali-spring-srv6-oam]. The numerical identifiers for the Endpoint behaviors are defined in the "SRv6 Endpoint Behaviors" registry defined in [I-D.filsfils-spring-srv6-network-programming]. This section lists the Endpoint behaviors and their identifiers, which MAY be advertised by IS-IS and the SID sub-TLVs in which each type MAY appear.

Endpoint Behavior	Endpoint Behavior Identifier	End SID	End.X SID	Lan End.X SID
End (PSP, USP, USD)	1-4, 28-31	Y	N	N
End.X (PSP, USP, USD)	5-8, 32-35	N	Y	Y
End.T (PSP, USP, USD)	9-12, 36-39	Y	N	N
End.DX6	16	N	Y	Y
End.DX4	17	N	Y	Y
End.DT6	18	Y	N	N
End.DT4	19	Y	N	N
End.DT64	20	Y	N	N
End.OP	40	Y	N	N
End.OTP	41	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-02 (work in progress), October 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-07 (work in progress), February 2019.

[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-16 (work in progress), February 2019.

[I-D.ietf-isis-segment-routing-extensions]
Previdi, S., Ginsberg, L., Filsfils, C., Bashandy, A., Gredler, H., and B. Decraene, "IS-IS Extensions for Segment Routing", draft-ietf-isis-segment-routing-extensions-22 (work in progress), December 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]
Standardization", I. ". O. F., "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 H. Chen
Internet-Draft D. Cheng
Intended status: Standards Track Huawei Technologies
Expires: October 31, 2018 M. Toy
Verizon
Y. Yang
IBM
April 29, 2018

ISIS Flooding Reduction
draft-cc-isis-flooding-reduction-01

Abstract

This document proposes an approach to flood ISIS link state protocol data units on a topology that is a subgraph of the complete ISIS topology per underline physical network, so that the amount of flooding traffic in the network is greatly reduced, and it would reduce convergence time with a more stable and optimized routing environment. The approach can be applied to any network topology in a single ISIS area.

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

Copyright Notice

Table of Contents

1. Introduction

For some networks such as dense Data Center (DC) networks, the existing ISIS Link State PDU (LSP) flooding mechanism is not efficient and may have some issues. The extra LSP flooding consumes network bandwidth. Processing the extra LSP flooding, including receiving, buffering and decoding the extra LSPs, wastes memory space and processor time. This may cause scalability issues and affect the network convergence negatively.

A flooding reduction method between spines and leaves is proposed in [I-D.shen-isis-spine-leaf-ext]. The problem on flooding reduction and an architectural solution are discussed in [I-D.li-dynamic-flooding]. This document proposes an approach to flood ISIS LSPs on a topology that is a subgraph of the entire ISIS topology per underline physical network, so that the amount of flooding traffic in the network is greatly reduced. The workload for processing the extra LSP flooding is decreased significantly. This would improve the scalability and speed up the network convergence, stable and optimize the routing environment.

The approach proposed is applicable to any network topology in a single ISIS area. The approach is backward compatible.

2. Problem Statement

ISIS, like other link-state routing protocols, deploys a so-called reliable flooding mechanism, where a node must transmit a received or self-originated LSP to all its ISIS interfaces (except the interface where a LSP is received) in the defined context. While this mechanism assures each LSP being distributed to every ISIS node in the relevant routing area or domain, the side-effect is that the mechanism often causes redundant LSPs in individual network segments (e.g., on an ISIS point-to-point link or a broadcast subnet), which in turn forces ISIS nodes to process identical LSPs more than once. This results waste of ISIS link bandwidth and ISIS nodes’ computing resources, and the delay of ISIS topology convergence.

The problem explained above becomes more serious in ISIS networks with large number of nodes and links, and in particular, higher degree of interconnection (e.g., meshed topology, spine-leaf topology, etc,). In some environment such as in data centers, the drawback of the existing flooding mechanism has already caused operational problems, including repeated and waves of flooding storms, chock of computing resources, slow convergence, oscillating topology changes, instability of routing environment.

One example is as shown in Figure 1 (a), where Node 1, Node 2 and Node 3 are interconnected in a mesh. When Node 1 receives a new or updated ISIS LSP on its interface I11, it by default would forward to its interface I12 and I13 towards Node 2 and Node 3, respectively, after processing. Node 2 and Node 3 upon reception of the LSP and after processing, would potentially flood the same LSP over their respective interface I23 and I32 toward each other, which is obviously not necessary and at the cost of link bandwidth as well as both nodes’ computing resource.

In example Figure 1 (b), Node 2 and Node 3 both connect to a LAN where Node 4, Node 5 and Node 6 also connect to. When Node 1 receives a LSP as in (a) and floods it to Node 2 and Node 3 respectively, the two nodes would in turn both (instead of one) flood to the LAN, which is unnecessary and at the cost of link bandwidth as well as computing resource of all nodes connected to the LAN.

```
               |                                |
               |I11                             |I11
            +--o---+                         +--o---+
            |Node 1|                         |Node 1|
            +-o--o-+                         +-o--o-+
         I12 /    \ I13                       /    \
            /      \                      I12/      \I13
        I21/        \I31                    /        \
     +----o-+   I32+-o----+           +----o-+      +-o----+
     |Node 2|------|Node 3|           |Node 2|      |Node 3|
     +------+I23   +------+           +--o---+      +---o--+
                                      I2L|      LAN     |I3L
             (a)                    -----o--------o-----o--o------
                                      I4L|     I5L|     I6L|
                                     +---o--+  +--o---+ +--o---+
                                     |Node 4|  |Node 5| |Node 6|
                                     +------+  +------+ +------+

                                                 (b)
```

Figure 1

3. Flooding Topology

It is a norm that an ISIS node sending a received LSP and self-originated LSP to all its ISIS interfaces (except that where an LSP is received), as the reliable-flooding mechanism requires, i.e., any

ISIS LSP would potentially traverses on each ISIS link in a given ISIS network topology, sometimes both directions. As demonstrated in Section 2, dissemination over the entire ISIS network topology has drawbacks.

To change ISIS's aggressive flooding behavior, a flooding topology is introduced. For a given ISIS network topology, a flooding topology is a sub-graph or sub-network of the given network topology that has the same reachability to every node as the given network topology. Thus all the nodes in the given network topology MUST be in the flooding topology. All the nodes MUST be inter-connected directly or indirectly. As a result, ISIS flooding will in most cases occur only on the flooding topology, that includes all ISIS nodes but a subset of ISIS links. Note even the flooding topology is a sub-graph of the original ISIS topology, any single LSP MUST still be disseminated in the entire ISIS network.

There are many different flooding topologies for a given ISIS network topology. A chain connecting all the nodes in the given network topology is a flooding topology. A circle connecting all the nodes is another flooding topology. A tree connecting all the nodes is a flooding topology. In addition, the tree plus the connections between some leaves of the tree and branch nodes of the tree is a flooding topology.

There are many different ways to construct a flooding topology for a given ISIS network topology. A few of them are listed below:

o One node in the network builds a flooding topology and floods the flooding topology to all the other nodes in the network (This seems not very good. Flooding the flooding topology may increase the flooding.);

o Each node in the network automatically calculates a flooding topology by using the same algorithm (No flooding for flooding topology);

o Links on the flooding topology are configured statically.

The minimum requirement for a flooding topology is all ISIS nodes are interconnected (directly or indirectly), but there is only one path from any node to any other node. While this lean-and-mean type of flooding topology degrades ISIS flooding traffic volume to the least, it may introduce some delay of topology convergence in the network with some network topologies. To compensate convergence efficiency, additional ISIS links may be added as part of the flooding topology. There is a trade-off between the density of the flooding topology and the convergence efficiency.

Note that the flooding topology constructed by an ISIS node is dynamic in nature, that means when the ISIS's base topology (the entire topology graph) changes, the flooding topology (the sub-graph) MUST be re-computed/re-constructed to ensure that any node that is reachable on the base topology MUST also be reachable on the flooding topology.

For reference purpose, some algorithms that allow ISIS nodes to automatically compute flooding topology are elaborated in Appendix A. However, this document does not attempt to standardize how a flooding topology is established.

4. Extensions to ISIS

A 1-bit flag F is defined in an ISIS router capability TLV. Flag F set to 1 indicates that the router supports ISIS LSP flood reduction described in this document; and Flag F set to 0 indicates that the router does not do so.

This flag is used for an ISIS node during the process of computing a flooding topology. An ISIS node that advertises its LSP containing a capability TLV with "F" bit set to 1 MUST always be included in the flooding topology computed by other ISIS nodes; but in contrast, the node with "F" bit set to zero may or may not be included in the flooding topology by other nodes, depending on how other nodes construct their flooding topology.

This flag can also be used for an ISIS node to trigger a decision whether it wants to perform LSP flooding to its neighbor.

The format of an ISIS router capability TLV with flag F is shown below.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Type = 242  |Length(5 ~ 255)|           Router ID           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Router ID            |Reserved |F|D|S|    Optional   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  sub-TLVs                                                     ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

5. Flooding Behavior

5.1. Nodes Support Flooding Reduction

This section describes ISIS flooding behavior for ISIS nodes that support flooding reduction described in this document. For these nodes, they MUST set "F" bit to 1 in their LSPs (see Section 4). The flooding behavior for these nodes differs from that as specified in ISIS ([RFC1195]). Section 5.1.1 describes the flooding behavior when an ISIS node receives an ISIS LSP from one of its interface, and Section 5.1.2 describes the flooding behavior for LSP originated by itself.

The revised flooding procedure MUST flood LSPs to every node in the network in any case, as the standard ISIS flooding procedure does.

It assumes that the ISIS node of which the flooding behavior is described below is on the flooding topology, i.e., the node and at least one of its ISIS interface are on the flooding topology, where:

1. When the node has only one interface on the flooding topology, the node is a leaf on the topology.

2. When the node has two interfaces on the flooding topology, the node is a transit node on the topology.

3. A flooding topology with nodes having one or two interfaces on the topology is a lean graph, i.e., there is only one path from any node to any other node on the graph. For flooding efficiency, there could be extra ISIS interfaces that are on the flooding topology, i.e., a node may have more than two interfaces that belong to the flooding topology.

5.1.1. Receiving an ISIS LSP

The flooding behavior when an ISIS node receives a newer ISIS LSP that is not originated by itself from one of its ISIS interface is as follows:

1. The LSP is received on a link that is on the flooding topology. The LSP is flooded only to all the other interfaces that are on the flooding topology.

2. The LSP is received on a link that is not on the flooding topology. This situation can happen when a neighboring node on a point-to-point link newly forms adjacency with the receiving node, or is not currently on the flooding topology; it can happen when the LSP sending neighbor does not support the ISIS flooding

reduction (i.e., with "F" bit set to zero); it can also happen as the receiving link is a broadcast-type interface. The LSP is flooded only to all other interfaces that are on the flooding topology.

3. In both cases above, if there is any neighboring node that is advertising its Router LSP with "F" bit set to zero (see Section 4) but it is not on the flooding topology, the received LSP MUST also be sent to this neighboring node.

In any case, the LSP must not be transmitted back to the receiving interface.

Note before forwarding a received LSP, the ISIS node would do the normal processing as usual.

5.1.2. Originating an ISIS LSP

The flooding behavior when an ISIS node originates an ISIS LSP is as follows:

1. If it is a refresh LSP, i.e., there is no significant change contained in the LSP comparing to the previous LSP, the LSP is transmitted over links on the flooding topology. In addition, if there is any neighboring node that is advertising its Router LSP with "F" bit set to zero (see Section 4) but it is not on the flooding topology, the LSP MUST also be sent to this neighboring node.

2. Otherwise, the LSP is transmitted to all ISIS interfaces. Choosing this action instead of limiting to links on flooding topology would speed up the synchronization around the advertising node’s neighbors, which could then disseminate the new LSP quickly.

5.1.3. An Exception Case

In Section 5.1.1 and Section 5.1.2, there are times when an ISIS node sending out a LSP to an interface on the flooding topology detects a critical interface or node failure. A critical interface is an interface on the flooding topology and is the only connection among some nodes on the flooding topology. When this interface goes down, the flooding topology will be split. Note the flooding topology was pre-computed/pre-constructed; but if at the time a critical interface or a node goes down before a re-newed flooding topology can be computed/constructed, the ISIS node MUST send out the LSP to all interfaces (except where it is received from) as a traditional ISIS node would do. This handling is also taking place if there are more than one interfaces or nodes on the existing flooding topology fail, i.e., if more than one interfaces or nodes on the flooding topology fail, the ISIS node does traditional flooding before the flooding topology is re-built.

5.1.4. One More Note

The destination address that is used when an ISIS node sends out a LSP on an interface on its flooding topology follows the specification in ISIS ([RFC1195]). This means on a local LAN, all other ISIS nodes will receive the LSP.

5.2. Nodes Not Support Flooding Reduction

For ISIS nodes that do not support flooding reduction as described in this document, they MUST set "F" bit to 0 in their Router LSP (see Section 4); note this is also a default setting. These nodes may or may not be on the flooding topology constructed by other nodes that support flooding reduction in the same ISIS area, however that is not a business these nodes need to concern.

The LSP flooding behavior of ISIS nodes that do not support reduction as described in this document MUST follow that as specified in ISIS ([RFC1195]).

6. Security Considerations

This document does not introduce any security issue.

7. IANA Considerations

This document has no request to IANA.

8. Acknowledgements

The authors would like to thank Acee Lindem, Zhibo Hu, Robin Li, Stephane Litkowski and Alvaro Retana for their valuable suggestions and comments on this draft.

9. References

9.1. 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>.

9.2. Informative References

[I-D.li-dynamic-flooding]
Li, T., "Dynamic Flooding on Dense Graphs", draft-li-dynamic-flooding-04 (work in progress), March 2018.

[I-D.shen-isis-spine-leaf-ext]
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.

Appendix A. Algorithms to Build Flooding Topology

There are many algorithms to build a flooding topology. A simple and efficient one is briefed below.

- o Select a node R according to a rule such as the node with the biggest/smallest node ID;

- o Build a tree using R as root of the tree (details below); and then

- o Connect k (k>=0) leaves to the tree to have a flooding topology (details follow).

A.1. Algorithms to Build Tree without Considering Flag F

An algorithm for building a tree from node R as root starts with a candidate queue Cq containing R and an empty flooding topology Ft:

1. Remove the first node A from Cq and add A into Ft

2. If Cq is empty, then return with Ft

3. Suppose that node Xi (i = 1, 2, ..., n) is connected to node A and not in Ft and X1, X2, ..., Xn are in a special order. For example, X1, X2, ..., Xn are ordered by the cost of the link between A and Xi. The cost of the link between A and Xi is less than the cost of the link between A and Xj (j = i + 1). If two costs are the same, Xi's ID is less than Xj's ID. In another example, X1, X2, ..., Xn are ordered by their IDs. If they are not ordered, then make them in the order.

4. Add Xi (i = 1, 2, ..., n) into the end of Cq, goto step 1.

Another algorithm for building a tree from node R as root starts with a candidate queue Cq containing R and an empty flooding topology Ft:

1. Remove the first node A from Cq and add A into Ft

2. If Cq is empty, then return with Ft

3. Suppose that node Xi (i = 1, 2, ..., n) is connected to node A and not in Ft and X1, X2, ..., Xn are in a special order. For example, X1, X2, ..., Xn are ordered by the cost of the link between A and Xi. The cost of the link between A and Xi is less than the cost of the link between A and Xj (j = i + 1). If two costs are the same, Xi's ID is less than Xj's ID. In another example, X1, X2, ..., Xn are ordered by their IDs. If they are not ordered, then make them in the order.

4. Add Xi (i = 1, 2, ..., n) into the front of Cq and goto step 1.

A third algorithm for building a tree from node R as root starts with a candidate list Cq containing R associated with cost 0 and an empty flooding topology Ft:

1. Remove the first node A from Cq and add A into Ft

2. If all the nodes are on Ft, then return with Ft

3. Suppose that node A is associated with a cost Ca which is the cost from root R to node A, node Xi (i = 1, 2, ..., n) is connected to node A and not in Ft and the cost of the link between A and Xi is LCi (i=1, 2, ..., n). Compute Ci = Ca + LCi, check if Xi is in Cq and if Cxi (cost from R to Xi) < Ci. If Xi is not in Cq, then add Xi with cost Ci into Cq; If Xi is in Cq, then If Cxi > Ci then replace Xi with cost Cxi by Xi with Ci in Cq; If Cxi == Ci then add Xi with cost Ci into Cq.

4. Make sure Cq is in a special order. Suppose that Ai (i=1, 2, ..., m) are the nodes in Cq, Cai is the cost associated with Ai,

and IDi is the ID of Ai. One order is that for any k = 1, 2, ..., m-1, Cak < Caj (j = k+1) or Cak = Caj and IDk < IDj. Goto step 1.

A.2. Algorithms to Build Tree Considering Flag F

An algorithm for building a tree from node R as root with consideration of flag F starts with a candidate queue Cq containing R associated with previous hop PH=0 and an empty flooding topology Ft:

1. Remove the first node A with its flag F set to one from the candidate queue Cq if there is such a node A; otherwise (i.e., if there is not such node A in Cq), then remove the first node A from Cq. Add A into the flooding topology Ft.

2. If Cq is empty or all nodes are on Ft, then return with Ft

3. Suppose that node Xi (i = 1, 2, ..., n) is connected to node A and not in the flooding topology Ft and X1, X2, ..., Xn are in a special order considering whether some of them with flag F = 1. For example, X1, X2, ..., Xn are ordered by the cost of the link between A and Xi. The cost of the link between A and Xi is less than that of the link between A and Xj (j = i + 1). If two costs are the same, Xi's ID is less than Xj's ID. The cost of a link is redefined such that 1) the cost of a link between A and Xi both with F = 1 is much less than the cost of any link between A and Xk where Xk with F=0; 2) the real metric of a link between A and Xi and the real metric of a link between A and Xk are used as their costs for determining the order of Xi and Xk if they all (i.e., A, Xi and Xk) with F = 1 or none of Xi and Xk with F = 1.

4. Add Xi (i = 1, 2, ..., n) associated with previous hop PH=A into the end of the candidate queue Cq, and goto step 1.

Another algorithm for building a tree from node R as root with consideration of flag F starts with a candidate queue Cq containing R associated with previous hop PH=0 and an empty flooding topology Ft:

1. Remove the first node A with its flag F set to one from the candidate queue Cq if there is such a node A; otherwise (i.e., if there is not such node A in Cq), then remove the first node A from Cq. Add A into the flooding topology Ft.

2. If Cq is empty or all nodes are on Ft, then return with Ft.

3. Suppose that node Xi (i = 1, 2, ..., n) is connected to node A and not in the flooding topology Ft and X1, X2, ..., Xn are in a special order considering whether some of them with F = 1. For

example, X1, X2, ..., Xn are ordered by the cost of the link between A and Xi. The cost of the link between A and Xi is less than the cost of the link between A and Xj (j = i + 1). If two costs are the same, Xi’s ID is less than Xj’s ID. The cost of a link is redefined such that 1) the cost of a link between A and Xi both with F = 1 is much less than the cost of any link between A and Xk where Xk with F = 0; 2) the real metric of a link between A and Xi and the real metric of a link between A and Xk are used as their costs for determining the order of Xi and Xk if they all (i.e., A, Xi and Xk) have F = 1 or none of Xi and Xk has F = 1.

4. Add Xi (i = 1, 2, ..., n) associated with previous hop PH=A into the front of the candidate queue Cq, and goto step 1.

A third algorithm for building a tree from node R as root with consideration of flag F starts with a candidate list Cq containing R associated with low order cost Lc=0, high order cost Hc=0 and previous hop ID PH=0, and an empty flooding topology Ft:

1. Remove the first node A from Cq and add A into Ft.

2. If all the nodes are on Ft, then return with Ft

3. Suppose that node A is associated with a cost Ca which is the cost from root R to node A, node Xi (i = 1, 2, ..., n) is connected to node A and not in Ft and the cost of the link between A and Xi is LCi (i=1, 2, ..., n). Compute Ci = Ca + LCi, check if Xi is in Cq and if Cxi (cost from R to Xi) < Ci. If Xi is not in Cq, then add Xi with cost Ci into Cq; If Xi is in Cq, then If Cxi > Ci then replace Xi with cost Cxi by Xi with Ci in Cq; If Cxi == Ci then add Xi with cost Ci into Cq.

4. Suppose that node A is associated with a low order cost LCa which is the low order cost from root R to node A and a high order cost HCa which is the high order cost from R to A, node Xi (i = 1, 2, ..., n) is connected to node A and not in the flooding topology Ft and the real cost of the link between A and Xi is Ci (i=1, 2, ..., n). Compute LCxi and HCxi: LCxi = LCa + Ci if both A and Xi have flag F set to one, otherwise LCxi = LCa HCxi = HCa + Ci if A or Xi does not have flag F set to one, otherwise HCxi = HCa If Xi is not in Cq, then add Xi associated with LCxi, HCxi and PH = A into Cq; If Xi associated with LCxi’ and HCxi’ and PHxi’ is in Cq, then If HCxi’ > HCxi then replace Xi with HCxi’, LCxi’ and PHxi’ by Xi with HCxi, LCxi and PH=A in Cq; otherwise (i.e., HCxi’ == HCxi) if LCxi’ > LCxi , then replace Xi with HCxi’, LCxi’ and PHxi’ by Xi with HCxi, LCxi and PH=A in Cq; otherwise (i.e., HCxi’ == HCxi and LCxi’ == LCxi) if PHxi’ > PH, then

replace Xi with HCxi’, LCxi’ and PHxi’ by Xi with HCxi, LCxi and PH=A in Cq.

5. Make sure Cq is in a special order. Suppose that Ai (i=1, 2, ..., m) are the nodes in Cq, HCai and LCai are low order cost and high order cost associated with Ai, and IDi is the ID of Ai. One order is that for any k = 1, 2, ..., m-1, HCak < HCaj (j = k+1) or HCak = HCaj and LCak < LCaj or HCak = HCaj and LCak = LCaj and IDk < IDj. Goto step 1.

A.3. Connecting Leaves

Suppose that we have a flooding topology Ft built by one of the algorithms described above. Ft is like a tree. We may connect k (k >=0) leaves to the tree to have a enhanced flooding topology with more connectivity.

Suppose that there are m (0 < m) leaves directly connected to a node X on the flooding topology Ft. Select k (k <= m) leaves through using a deterministic algorithm or rule. One algorithm or rule is to select k leaves that have smaller or larger IDs (i.e., the IDs of these k leaves are smaller/bigger than the IDs of the other leaves directly connected to node X). Since every node has a unique ID, selecting k leaves with smaller or larger IDs is deterministic.

If k = 1, the leaf selected has the smallest/largest node ID among the IDs of all the leaves directly connected to node X.

For a selected leaf L directly connected to a node N in the flooding topology Ft, select a connection/adjacency to another node from node L in Ft through using a deterministic algorithm or rule.

Suppose that leaf node L is directly connected to nodes Ni (i = 1,2,...,s) in the flooding topology Ft via adjacencies and node Ni is not node N, IDi is the ID of node Ni, and Hi (i = 1,2,...,s) is the number of hops from node L to node Ni in the flooding topology Ft.

One Algorithm or rule is to select the connection to node Nj (1 <= j <= s) such that Hj is the largest among H1, H2, ..., Hs. If there is another node Na (1 <= a <= s) and Hj = Ha, then select the one with smaller (or larger) node ID. That is that if Hj == Ha and IDj < IDa then select the connection to Nj for selecting the one with smaller node ID (or if Hj == Ha and IDj < IDa then select the connection to Na for selecting the one with larger node ID).

Suppose that the number of connections in total between leaves selected and the nodes in the flooding topology Ft to be added is NLc. We may have a limit to NLc.

Authors' Addresses

Huaimo Chen
Huawei Technologies

Email: huaimo.chen@huawei.com

Dean Cheng
Huawei Technologies

Email: dean.cheng@huawei.com

Mehmet Toy
Verizon
USA

Email: mehmet.toy@verizon.com

Yi Yang
IBM
Cary, NC
United States of America

Email: yyietf@gmail.com

Network Working Group H. Chen
Internet-Draft D. Cheng
Intended status: Standards Track Huawei Technologies
Expires: March 24, 2019 M. Toy
Verizon
Y. Yang
IBM
September 20, 2018

LS Flooding Reduction
draft-cc-ospf-flooding-reduction-04

Abstract

This document proposes an approach to flood link states on a topology that is a subgraph of the complete topology per underline physical network, so that the amount of flooding traffic in the network is greatly reduced, and it would reduce convergence time with a more stable and optimized routing environment. The approach can be applied to any network topology in a single area.

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

Copyright Notice

Table of Contents

1. Introduction

For some networks such as dense Data Center (DC) networks, the existing Link State (LS) flooding mechanism is not efficient and may have some issues. The extra LS flooding consumes network bandwidth. Processing the extra LS flooding, including receiving, buffering and decoding the extra LSs, wastes memory space and processor time. This may cause scalability issues and affect the network convergence negatively.

This document proposes an approach to minimize the amount of flooding traffic in the network. Thus the workload for processing the extra LS flooding is decreased significantly. This would improve the scalability, speed up the network convergence, stable and optimize the routing environment.

This approach is also flexible. It has multiple modes for computation of flooding topology. Users can select a mode they prefer, and smoothly switch from one mode to another. The approach is applicable to any network topology in a single area. It is backward compatible.

2. Terminology

Flooding Topology:
A sub-graph or sub-network of a given (physical) network topology that has the same reachability to every node as the given network topology, through which link states are flooded.

critical link or interface on a flooding topology:
A only link or interface among some nodes on the flooding topology. When this link or interface goes down, the flooding topology will be split.

critical node on a flooding topology:

A only node connecting some nodes on the flooding topology. When this node goes down, the flooding topology will be split.

backup path:
A path or a sequence of links, when a critical link or node goes down, providing a connection to connect two parts of a split flooding topology. When a critical node goes down, the flooding topology may be split into more than two parts. In this case, two or more backup paths are needed to connect all the split parts into one.

Remaining Flooding Topology:
A topology from a flooding topology by removing the failed links and nodes from the flooding topology.

LSA:
A Link State Advertisement in OSPF.

LSP:
A Link State Protocol Data Unit (PDU) in IS-IS.

LS:
A Link Sate, which is an LSA or LSP.

3. Conventions Used in This Document

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].

4. Problem Statement

OSPF and IS-IS deploy a so-called reliable flooding mechanism, where a node must transmit a received or self-originated LS to all its interfaces (except the interface where an LS is received). While this mechanism assures each LS being distributed to every node in an area or domain, the side-effect is that the mechanism often causes redundant LS, which in turn forces nodes to process identical LS more than once. This results in the waste of link bandwidth and nodes' computing resources, and the delay of topology convergence.

This becomes more serious in networks with large number of nodes and links, and in particular, higher degree of interconnection (e.g., meshed topology, spine-leaf topology, etc.). In some environments such as in data centers, the drawback of the existing flooding mechanism has already caused operational issues, including repeated and waves of flooding storms, chock of computing resources, slow convergence, oscillating topology changes, instability of routing environment.

One example is as shown in Figure 1, where Node 1, Node 2 and Node 3 are interconnected in a mesh. When Node 1 receives a new or updated LS on its interface I11, it by default would forward the LS to its interface I12 and I13 towards Node 2 and Node 3, respectively, after processing. Node 2 and Node 3 upon reception of the LS and after processing, would potentially flood the same LS over their respective interface I23 and I32 toward each other, which is obviously not necessary and at the cost of link bandwidth as well as both nodes' computing resource.

```
                                   |
                                   |I11
                                +--o---+
                                |Node 1|
                                +-o--o-+
                             I12 /    \ I13
                                /      \
                            I21/        \I31
                         +----o-+   I32+-o----+
                         |Node 2|------|Node 3|
                         +------+I23   +------+
```

Figure 1

5. Flooding Topology

For a given network topology, a flooding topology is a sub-graph or sub-network of the given network topology that has the same reachability to every node as the given network topology. Thus all the nodes in the given network topology MUST be in the flooding topology. All the nodes MUST be inter-connected directly or indirectly. As a result, LS flooding will in most cases occur only on the flooding topology, that includes all nodes but a subset of links. Note even though the flooding topology is a sub-graph of the original topology, any single LS MUST still be disseminated in the entire network.

5.1. Construct Flooding Topology

Many different flooding topologies can be constructed for a given network topology. A chain connecting all the nodes in the given network topology is a flooding topology. A circle connecting all the nodes is another flooding topology. A tree connecting all the nodes is a flooding topology. In addition, the tree plus the connections between some leaves of the tree and branch nodes of the tree is a flooding topology.

The following parameters need to be considered for constructing a flooding topology:

- Number of links: The number of links on the flooding topology is a key factor for reducing the amount of LS flooding. In general, the smaller the number of links, the less the amount of LS flooding.

- Diameter: The shortest distance between the two most distant nodes on the flooding topology is a key factor for reducing the network convergence time. The smaller the diameter, the less the convergence time.

- Redundancy: The redundancy of the flooding topology means a tolerance to the failures of some links and nodes on the flooding topology. If the flooding topology is split by some failures, it is not tolerant to these failures. In general, the larger the number of links on the flooding topology is, the more tolerant the flooding topology to failures.

There are many different ways to construct a flooding topology for a given network topology. A few of them are listed below:

- Central Mode: One node in the network builds a flooding topology and floods the flooding topology to all the other nodes in the network (This seems not good. Flooding the flooding topology may increase the flooding. The amount of traffic for flooding the flooding topology should be minimized.);

- Distributed Mode: Each node in the network automatically calculates a flooding topology by using the same algorithm (No flooding for flooding topology);

- Static Mode: Links on the flooding topology are configured statically.

Note that the flooding topology constructed by a node is dynamic in nature, that means when the base topology (the entire topology graph) changes, the flooding topology (the sub-graph) MUST be re-computed/re-constructed to ensure that any node that is reachable on the base topology MUST also be reachable on the flooding topology.

For reference purpose, some algorithms that allow nodes to automatically compute flooding topology are elaborated in Appendix A.

However, this document does not attempt to standardize how a flooding topology is established.

5.2. Backup for Flooding Topology Split

It is hard to construct a flooding topology that reduces the amount of LS flooding greatly and is tolerant to multiple failures. To get around this, we can compute and use backup paths for a critical link and node on the flooding topology. Using backup paths may also speed up convergence when the link and node fail.

When a critical link on the flooding topology fails, the flooding topology without the critical link (i.e., the remaining flooding topology) is split into two parts. A backup path for the critical link connects the two parts into one. Through the backup path and the remaining flooding topology, an LS can be flooded to every node in the network. The combination of the backup path and the flooding topology is tolerant to the failure of the critical link.

When a critical node on the flooding topology goes down, the flooding topology without the critical node and the links attached to the node (i.e., the remaining flooding topology) is split into two or more parts. One or more backup paths for the critical node connects the split parts into one. Through the backup paths and the remaining flooding topology, an LS can be flooded to every live node in the network. The combination of the backup paths and the flooding topology is tolerant to the failure of the critical node.

In addition to the backup paths for a critical link and node, backup paths for every non critical link and node on the flooding topology can be computed. When the failures of multiple links and nodes on the flooding topology happen, through the remaining flooding topology and the backup paths for these links and nodes, an LS can be flooded to every live node in the network. The combination of the backup paths and the flooding topology is tolerant to the failures of these links and nodes. If there are other failures that break the backup paths, an LS can be flooded to every live node by the traditional flooding procedure.

In a centralized mode, the leader computes the backup paths and floods them to all the other nodes. In a distributed mode, every node computes the backup paths.

6. Extensions to OSPF

The extensions to OSPF comprises two parts: one part is for operations on flooding reduction, the other is specially for centralized mode flooding reduction.

6.1. Extensions for Operations

A new TLV is defined in OSPF RI LSA [RFC7770]. It contains instructions about flooding reduction, which is called Flooding Reduction Instruction TLV or Instruction TLV for short. This TLV is originated from only one node at any time.

The format of a Flooding Reduction Instruction TLV is as follows.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      INST-TLV-Type (TBD1)     |          TLV-Length           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  OP | MOD |   Algorithm   |     Reserved (MUST be zero)   | NL |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~                     sub TLVs (optional)                       ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Flooding Reduction Instruction TLV

A OP field of three bits is defined in the TLV. It may have a value of the followings.

- o 0x001 (R): Perform flooding Reduction, which instructs the nodes in a network to perform flooding reduction.

- o 0x010 (N): Roll back to Normal flooding, which instructs the nodes in a network to roll back to perform normal flooding.

When any of the other values is received, it is ignored.

A MOD field of three bits is defined in the TLV and may have a value of the followings.

- o 0x001 (C): Central Mode, which instructs 1) the nodes in a network to select leaders (primary/designated leader, secondary/backup leader, and so on); 2) the leaders in a network to compute a flooding topology and the primary leader to flood the flooding topology to all the other nodes in the network; 3) every node in the network to receive and use the flooding topology originated by the primary leader.

- o 0x010 (D): Distributed Mode, which instructs every node in a network to compute and use its own flooding topology.

- o 0x011 (S): Static Mode, which instructs every node in a network to use the flooding topology statically configured on the node.

When any of the other values is received, it is ignored.

An Algorithm field of eight bits is defined in the TLV to instruct the leader node in central mode or every node in distributed mode to use the algorithm indicated in this field for computing a flooding topology.

A NL field of three bits is defined in the TLV, which indicates the number of leaders to be selected when Central Mode is used. NL set to 2 means two leaders (a designated/primary leader and a backup/secondary leader) to be selected for an area, and NL set to 3 means three leaders to be selected. When Central Mode is not used, The NL field is not valid.

Some optional sub TLVs may be defined in the future, but none is defined now.

6.2. Extensions for Centralized Mode

6.2.1. Message for Flooding Topology

A flooding topology can be represented by the links in the flooding topology. For the links between a local node and a number of its adjacent (or remote) nodes, we can encode the local node in a way, and encode its adjacent nodes in the same way or another way. After all the links in the flooding topology are encoded, the encoded links can be flooded to every node in the network. After receiving the encoded links, every node decodes the links and creates and/or updates the flooding topology.

For every node in an area, we may use an index to represent it. Every node in an area may order the nodes in a rule, which generates the same sequence of the nodes on every node in the area. The sequence of nodes have the index 0, 1, 2, and so on respectively. For example, every node orders the nodes by their router IDs in ascending order.

6.2.1.1. Links Encoding

A local node can be encoded in two parts: encoded node index size indication (ENSI) and compact node index (CNI). ENSI value plus a number (e.g., 9) gives the size of compact node index. For example, ENSI = 0 indicates that the size of CNIs is 9 bits. In the figure below, Local node LN1 is encoded as ENSI=0 using 3 bits and CNI=LN1’s Index using 9 bits. LN1 is encoded in 12 bits in total.

```
      0 1 2 3 4 5 6 7 8
     +-+-+-+-+-+-+-+-+-+
     |0 0 0|              ENSI (3 bits) [9 bits CNI]
     +-+-+-+-+-+-+-+-+-+
     | LN1 Index Value |  CNI  (9 bits)
     +-+-+-+-+-+-+-+-+-+
```

An Example of Local Node Encoding

The adjacent nodes can be encoded in two parts: Number of Nodes (NN) and compact node indexes (CNIs). The size of CNIs is the same as the local node. For example, three adjacent nodes RN1, RN2 and RN3 are encoded below in 30 bits (i.e., 3.75 bytes).

```
      0 1 2 3 4 5 6 7 8
     +-+-+-+-+-+-+-+-+-+
     |0 1 1|             NN (3 bits)   [3 adjacent nodes]
     +-+-+-+-+-+-+-+-+-+
     |   RN1’s Index   | CNI (9 bits) for RN1
     +-+-+-+-+-+-+-+-+-+
     |   RN2’s Index   | CNI (9 bits) for RN2
     +-+-+-+-+-+-+-+-+-+
     |   RN3’s Index   | CNI (9 bits) for RN3
     +-+-+-+-+-+-+-+-+-+
```

An Example of Adjacent Nodes Encoding

The links between a local node and a number of its adjacent (or remote) nodes can be encoded as the local node followed by the adjacent nodes. For example, three links between local node LN1 and its three adjacent nodes RN1, RN2 and RN3 are encoded below in 42 bits (i.e., 5.25 bytes).

```
     0 1 2 3 4 5 6 7 8
    +-+-+-+-+-+-+-+-+-+                            _
    |0 0 0|             ENSI (3 bits) [9 bits CNI]  |
    +-+-+-+-+-+-+-+-+-+                             } Encoding for
    | LN1 Index Value | CNI (9 bits) for LN1       _| Local Node LN1
    +-+-+-+-+-+-+-+-+-+                            _
    |0 1 1|             NN (3 bits) [3 nodes]       |
    +-+-+-+-+-+-+-+-+-+                             | Encoding for
    |   RN1's Index   | CNI (9 bits) for RN1        | 3 adjacent nodes
    +-+-+-+-+-+-+-+-+-+                             } RN1, RN2, RN3
    |   RN2's Index   | CNI (9 bits) for RN2        | of LN1
    +-+-+-+-+-+-+-+-+-+                             |
    |   RN3's Index   | CNI (9 bits) for RN3       _|
    +-+-+-+-+-+-+-+-+-+
```

An Example of Links Encoding

For a flooding topology computed by a leader of an area, it may be represented by all the links on the flooding topology. A Type-Length-Value (TLV) of the following format for the links encodings can be included in an LSA to represent the flooding topology (FT) and flood the FT to every node in the area.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |      FTLK-TLV-Type (TBD2)     |          TLV-Length           |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    ~           Links Encoding (Node 1 to its adjacent Nodes)       ~
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    ~           Links Encoding (Node 2 to its adjacent Nodes)       ~
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    :                                                               :
    :                                                               :
```

Flooding Topology Links TLV

Note that a link between a local node LN and its adjacent node RN can be encoded once and as a bi-directional link. That is that if it is encoded in a Links Encoding from LN to RN, then the link from RN to LN is implied or assumed.

For OSPFv2, an Opaque LSA of a new opaque type (TBD3) containing a Flooding Topology Links TLV is used to flood the flooding topology from the leader of an area to all the other nodes in the area.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |            LS age             |     Options   | LS Type = 10  |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | FT-Type(TBD3) |                   Instance ID                 |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                      Advertising Router                       |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                      LS Sequence Number                       |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |         LS checksum           |           Length              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   ~                Flooding Topology Links TLV                    ~
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

OSPFv2: Flooding Topology Opaque LSA

For OSPFv3, an area scope LSA of a new LSA function code (TBD4) containing a Flooding Topology Links TLV is used to flood the flooding topology from the leader of an area to all the other nodes in the area.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |            LS age             |1|0|1|       FT-LSA (TBD4)     |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                        Link State ID                          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                      Advertising Router                       |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                      LS Sequence Number                       |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |         LS checksum           |           Length              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   ~                Flooding Topology Links TLV                    ~
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

OSPFv3: Flooding Topology LSA

The U-bit is set to 1, and the scope is set to 01 for area-scoping.

6.2.1.2. Block Encoding

Block encoding uses a single structure to encode a block (or part) of topology, which can be a block of links in a flooding topology. It can also be all the links in the flooding topology. It starts with a local node LN and its adjacent (or remote) nodes RNi (i = 1, 2, ..., n), and can be considered as an extension to the links encoding.

The encoding of links between a local node and its adjacent nodes described in Section 6.2.1.1 is extended to include the links attached to the adjacent nodes.

The encoding for the adjacent nodes is extended to include Extending Flags (E Flags for short) between the NN (Number of Nodes) field and the CNIs (Compact Node Indexes) for the adjacent nodes. The length of the E Flags field is NN bits. The following is an example encoding of the adjacent nodes with E Flags of 3 bits, which is the value of the NN (the number of adjacent nodes).

```
  0 1 2 3 4 5 6 7 8
 +-+-+-+-+-+-+-+-+-+
 |0 1 1|             NN (3 bits)   [3 adjacent nodes]
 +-+-+-+
 |1 0 1|             E Flags [NN=3 bits]
 +-+-+-+-+-+-+-+-+-+
 |   RN1's Index   | CNI (9 bits) for RN1
 +-+-+-+-+-+-+-+-+-+
 |   RN2's Index   | CNI (9 bits) for RN2
 +-+-+-+-+-+-+-+-+-+
 |   RN3's Index   | CNI (9 bits) for RN3
 +-+-+-+-+-+-+-+-+-+
```

An Example of Adjacent Nodes with E Flags Encoding

There is a bit flag (called E flag) in the E Flags field for each adjacent node. The first bit (i.e., the most significant bit) in the E Flags field is for the first adjacent node (e.g., RN1), the second bit is for the second adjacent node (e.g., RN2), and so on. The E flag for an adjacent node RNi set to one indicates that the links attached to the adjacent node RNi are included below. The E flag for an adjacent node RNi set to zero means that no links attached to the adjacent node RNi are included below.

The links attached to the adjacent node RNi are represented by the RNi as a local node and the adjacent nodes of RNi. The encoding for the adjacent nodes of RNi is the same as that for the adjacent nodes of a local node. It consists of an NN field of 3 bits, E Flags field of NN bits, and CNIs for the adjacent nodes of RNi.

The following is an example of a block encoding for a block (or part) of flooding topology below.

```
                             o LN1
                           / | \
                         /       \
                       /     |     \
                     o RN1   o RN2   o RN3
                   /               /   \
                 /               /       \
               /               /           \
              o RN11          o RN31        o RN32
```

An Example Block of Flooding Topology

It represents 6 links: 3 links between local node LN1 and its 3 adjacent nodes RN1, RN2 and RN3; 1 link between RN1 as a local node and its 1 adjacent node RN11; and 2 links between RN3 as a local node and its 2 adjacent nodes RN31 and RN32.

It starts with the encoding of the links between local node LN1 and 3 adjacent nodes RN1, RN2 and RN3 of the local node LN1. The encoding for the local node LN1 is the same as that for a local node described in Section 6.2.1.1. The encoding for 3 adjacent nodes RN1, RN2 and RN3 of local node LN1 comprises an NN field of 3 bits with value of 3, E Flags field of NN = 3 bits, and the indexes of adjacent nodes RN1, RN2 and RN3.

```
     0 1 2 3 4 5 6 7 8
    +-+-+-+-+-+-+-+-+-+                            _
    |0 0 0|             ENSI (3 bits) [9 bits CNI]  |
    +-+-+-+-+-+-+-+-+-+                             } Encoding for
    | LN1 Index Value | CNI  (9 bits)              _| Local Node LN1
    +-+-+-+-+-+-+-+-+-+                            _
    |0 1 1|             NN(3 bits)[3 adjacent nodes]|
    +-+-+-+                                         |
    |1 0 1|             E Flags [NN=3 bits]         | Encoding for
    +-+-+-+-+-+-+-+-+-+                             | 3 adjacent nodes
    |   RN1’s Index   | CNI (9 bits) for RN1        } (RN1, RN2, RN3)
    +-+-+-+-+-+-+-+-+-+                             | of LN1
    |   RN2’s Index   | CNI (9 bits) for RN2        |
    +-+-+-+-+-+-+-+-+-+                             |
    |   RN3’s Index   | CNI (9 bits) for RN3       _|
    +-+-+-+-+-+-+-+-+-+                            _
    |0 0 1|             NN (3 bits)[1 adjacent node]|
    +-+-+-+                                         | Encoding for
    |0|                 E Flags [NN=1 bit]          } 1 adjacent node
    +-+-+-+-+-+-+-+-+-+                             | (RN11)
    |   RN11’s Index  | CNI (9 bits) for RN11      _| of RN1
    +-+-+-+-+-+-+-+-+-+                            _
    |0 1 0|             NN(3 bits)[2 adjacent nodes]|
    +-+-+-+                                         |
    |0 0|               E Flags [NN=2 bits]         | Encoding for
    +-+-+-+-+-+-+-+-+-+                             } 2 adjacent nodes
    |   RN31’s Index  | CNI (9 bits) for RN31       | (RN31, RN32)
    +-+-+-+-+-+-+-+-+-+                             | of RN3 as a
    |   RN32’s Index  | CNI (9 bits) for RN32       | local node
    +-+-+-+-+-+-+-+-+-+                            _|
```

An Example of Block Encoding

The first E flag in the encoding for adjacent nodes RN1, RN2 and RN3 is set to one, which indicates that the links between the first adjacent node RN1 as a local node and its adjacent nodes are included below. In this example, 1 link between RN1 and its adjacent node RN11 is represented by the encoding for the adjacent node RN11 of RN1 as a local node. The encoding for 1 adjacent node RN11 consists of an NN field of 3 bits with value of 1, E Flags field of NN = 1 bits, and the index of adjacent node RN11. The size of the index of RN11 is the same as that of local node LN1 indicated by the ENSI in the encoding for local node LN1.

The second E flag in the encoding for adjacent nodes RN1, RN2 and RN3 is set to zero, which indicates that no links between the second adjacent node RN2 as a local node and its adjacent nodes are included below.

The third E flag in the encoding for adjacent nodes RN1, RN2 and RN3 is set to one, which indicates that the links between the third adjacent node RN3 as a local node and its adjacent nodes are included below. In this example, 2 links between RN3 and its 2 adjacent nodes RN31 and RN32 are represented by the encoding for the adjacent nodes RN31 and RN32 of RN3 as a local node. The encoding for 2 adjacent nodes RN31 and RN32 consists of an NN field of 3 bits with value of 2, E Flags field of NN = 2 bits, and the indexes of adjacent nodes RN31 and RN32. The size of the index of RN31 and RN32 is the same as that of local node LN1 indicated by the ENSI in the encoding for local node LN1.

The block encoding may be used in the place of the links encoding in Section 6.2.1.1 for more efficiency. That is that it may be used in a Flooding Topology Links TLV. Alternatively, a new TLV, which is similar to the Flooding Topology Links TLV, may be defined to contain a number of block encodings.

6.2.2. Encodings for Backup Paths

When the leader of an area computes a flooding topology, it may compute a backup path or multiple backup paths for a critical link on the flooding topology. When the critical link fails, a link state can be distributed to every node in the area through one backup path and other links on the flooding topology. In addition, it may compute a backup path or multiple backup paths for a node. When the node fails, a link state can be distributed to the other nodes in the area through the backup paths and the links on the flooding topology.

This section describes two encodings for backup paths: separated encoding and integrated one. In the former, backup paths are encoded in a new message, where the message for the flooding topology described in the previous section is required; In the latter, backup paths are integrated into the flooding topology links encoding, where one message contains the flooding topology and the backup paths.

6.2.2.1. Message for Backup Paths

Backup paths for a node (such as Node1) may be represented by the node index encoding and node backup paths encoding. The former is similar to local node index encoding. The latter has the following format. It comprises a K flag (Key/Critical node flag) of 1 bit, a 3 bits NNBP field (number of node backup paths), and each of the backup paths encoding, which consists of the path length PLEN of 4 bits indicating the length of the path (i.e., the number of nodes), and the encoding of the sequence of nodes along the path such as encodings for nodes PN1, ..., PNn. The encoding of every node may use the encoding of a local node, which comprises encoded node index size indication (ENSI) and compact node index (CNI).

```
 0 1 2 3 4 5 6 7 8
 +-+-+-+-+-+-+-+-+
 |K|                1 bit (K=1: Key/Critical Node, K=0: Normal Node)
 +-+-+-+                                                     _
 |NNBP |            3 bits (number of node backup paths)      |
 +-+-+-+-+                                   _                |
 |PLEN   |          4 bits (backup path len)  |               |
 +-+-+-+-+-+-+-+-+                            |               | Backup
 | PN1 Encoding  |  Variable bits             | One           } paths
 +-+-+-+-+-+-+-+-+                            } backup path   | for Node
 ~               ~                            | for Node      |
 +-+-+-+-+-+-+-+-+                            |               |
 | PNn Encoding  |  Variable bits            _|               |
 +-+-+-+-+-+-+-+-+                                            |
 //              //                                          _|
```

An Example of Node Backup Paths Encoding

Another encoding of the sequence of nodes along the path uses one encoded node index size indication (ENSI) for all the nodes in the path. Thus we have the following Node Backup Paths Encoding.

```
  0 1 2 3 4 5 6 7 8
  +-+-+-+-+-+-+-+-+
  |K|                1 bit (K=1: Key/Critical Node, K=0: Normal Node)
  +-+-+-+                                                    _
  |NNBP |            3 bits (number of node backup paths)     |
  +-+-+-+-+                                   _               |
  |PLEN   |          4 bits (backup path len)  |              |
  +-+-+-+-+                                    |              |
  |ENSI |            3 bits(Ix Bits Indication)|              | Backup
  +-+-+-+-+-+-+-+-+                            | One          } paths
  |   PN1 Index   |  #Bits indicated by ENSI   } backup path  | for Node
  +-+-+-+-+-+-+-+-+                            | for Node     |
  ~               ~                            |              |
  +-+-+-+-+-+-+-+-+                            |              |
  |   PNn Index   |  #Bits indicated by ENSI  _|              |
  +-+-+-+-+-+-+-+-+                                           |
  //              //                                         _|
```

Another Example of Node Backup Paths Encoding

A new TLV called Node Backup Paths TLV is defined below. It may include multiple nodes and their backup paths. Each node is represented by its index encoding, which is followed by its node backup paths encoding.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |      NBP-TLV-Type (TBD5)      |          TLV-Length           |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |Node1 Index Enc|  Variable bits
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    :                 Node1 backup paths encoding                   :
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |Node2 Index Enc|  Variable bits
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    :                 Node2 backup paths encoding                   :
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    //                                                             //
```

Node Backup Paths TLV

The encoding for backup paths for a link (such as Link1) on the flooding topology consists of the link encoding such as Link1 Index Encoding and the link backup paths encoding. The former is similar to local node encoding. It contains encoded link index size indication (ELSI) and compact link index (CLI). The latter has the following format. It comprises a C flag (Critical link flag) of 1 bit, a 2 bits NLB field (number of link backup paths), and each of the backup paths encoding, which consists of the path length PLEN of 3 bits indicating the length of the path (i.e., the number of nodes), and the encoding of the sequence of nodes along the path such as encodings for nodes PN1, ..., PNm. Note that two ends of a link (i.e., the local node and the adjacent/remote node of the link) are not needed in the path. The encoding of every node may use the encoding of a local node, which comprises encoded node index size indication (ENSI) and compact node index (CNI).

```
  0 1 2 3 4 5 6 7 8
  +-+-+-+-+-+-+-+-+-+
  |C|                1 bit (C=1: Critical Link, C=0: Normal Link)
  +-+-+                                                      _
  |NLB|              2 bits (number of link backup paths)     |
  +-+-+-+                                     _               |
  |PLEN |            3 bits (backup path len)  |              |
  +-+-+-+-+-+-+-+-+-+                          |              | Backup
  | PN1 Encoding  |  Variable bits             | One          } paths
  +-+-+-+-+-+-+-+-+-+                          } backup path  | for Link
  ~               ~                            | for Link     |
  +-+-+-+-+-+-+-+-+-+                          |              |
  | PNm Encoding  |  Variable bits            _|              |
  +-+-+-+-+-+-+-+-+-+                                         |
  //              //                                         _|
```

An Example of Link Backup Paths Encoding

Another encoding of the sequence of nodes along the path uses one encoded node index size indication (ENSI) for all the nodes in the path. Thus we have the following Link Backup Paths Encoding.

```
  0 1 2 3 4 5 6 7 8
  +-+-+-+-+-+-+-+-+-+
  |C|                1 bit (C=1: Critical Link, C=0: Normal Link)
  +-+-+                                                      _
  |NLB|              2 bits (number of link backup paths)     |
  +-+-+-+                                     _               |
  |PLEN |            3 bits (backup path len)  |              |
  +-+-+-+                                      |              |
  |ENSI |            3 bits(Ix Bits Indication)|              | Backup
  +-+-+-+-+-+-+-+-+-+                          | One          } paths
  |   PN1 Index   |  #Bits indicated by ENSI   } backup path  | for Link
  +-+-+-+-+-+-+-+-+-+                          | for Link     |
  ~               ~                            |              |
  +-+-+-+-+-+-+-+-+-+                          |              |
  |   PNm Index   |  #Bits indicated by ENSI  _|              |
  +-+-+-+-+-+-+-+-+-+                                         |
  //              //                                         _|
```

Another Example of Link Backup Paths Encoding

A new TLV called Link Backup Paths TLV is defined below. It may include multiple links and their backup paths. Each link is represented by its index encoding, which is followed by its link backup paths encoding.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |      LBP-TLV-Type (TBD6)      |          TLV-Length           |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |Link1 Index Enc|  Variable bits
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    :                 Link1 backup paths encoding                   :
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |Link2 Index Enc|  Variable bits
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    :                 Link2 backup paths encoding                   :
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    //                                                             //
```

Link Backup Paths TLV

For OSPFv2, an Opaque LSA of a new opaque type (TBD7), containing node backup paths TLVs and link backup paths TLVs, is used to flood the backup paths from the leader of an area to all the other nodes in the area.

```
       0                   1                   2                   3
       0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |            LS age             |    Options    | LS Type = 10  |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      | BP-Type(TBD7) |                  Instance ID                  |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                      Advertising Router                       |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                      LS Sequence Number                       |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |         LS checksum           |           Length              |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      ~                    Link Backup Paths TLVs                     ~
      ~                    Node Backup Paths TLVs                     ~
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

OSPFv2: Backup Paths Opaque LSA

For OSPFv3, an area scope LSA of a new LSA function code (TBD8), containing node backup paths TLVs and link backup paths TLVs, is used to flood the backup paths from the leader of an area to all the other nodes in the area.

```
       0                   1                   2                   3
       0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |            LS age             |1|0|1|      BP-LSA (TBD8)      |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                        Link State ID                          |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                      Advertising Router                       |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                      LS Sequence Number                       |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |         LS checksum           |           Length              |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      ~                    Link Backup Paths TLVs                     ~
      ~                    Node Backup Paths TLVs                     ~
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

OSPFv3: Backup Paths LSA

The U-bit is set to 1, and the scope is set to 01 for area-scoping.

6.2.2.2. Backup Paths in Links TLV

A local node and its backup paths can be encoded in the following format. It is the local node (such as local node LN1) encoding followed by the local node backup paths encoding, which is the same as the node backup paths encoding described in Section 6.2.2.1.

```
     0 1 2 3 4 5 6 7 8
    +-+-+-+-+-+-+-+-+-+                             _
    |ENSI |             3 bits(#bits indication)  |
    +-+-+-+-+-+-+-+-+-+                             } Local Node LN1
    | LN1 Index Value | #bits indicated by ENSI _| Encoding
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    :              Local node LN1 backup paths encoding             :
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Local Node with Backup Paths Encoding

A adjacent node and its backup paths can be encoded in the following format. It is the adjacent node (such as adjacent node RN10) index value followed by the adjacent node backup paths encoding, which is the same as the node backup paths encoding described in Section 6.2.2.1.

```
    +-+-+-+-+-+-+-+-+-+-+
    |RN10 Index Value |  (#bits indicated by ENSI)
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    :             adjacent node RN10 backup paths encoding          :
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Adjacent Node with Backup Paths Encoding

The links between a local node and a number of its adjacent nodes, the backup paths for each of the nodes, and the backup paths for each of the links can be encoded in the following format. It is called Links from Node with Backup Paths Encoding.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    :              Local Node with backup paths encoding            :
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    | NN  |   Number of adjacent Nodes (i.e., Number of links)
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    :              Adjacent Node 1 with backup paths encoding       :
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    :                   Link1 backup paths Encoding                 :
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    :              Adjacent Node 2 with backup paths encoding       :
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    :                   Link2 backup paths Encoding                 :
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
```

Links from Node with Backup Paths Encoding

A new TLV called Links with Backup Paths TLV is defined below. It includes a number of Links from Node with Backup Paths Encodings described above. This TLV contains both the flooding topology and the backup paths for the links and nodes on the flooding topology.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |    LNSBP-TLV-Type (TBD9)      |         TLV-Length            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    :          Links from Node 1 with backup paths encoding         :
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    :          Links from Node 2 with backup paths encoding         :
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    :                                                               :
    :                                                               :
                       Links with Backup Paths TLV
```

For OSPFv2, an Opaque LSA of a new opaque type (TBDa), called Flooding Topology with Backup Paths (FTBP) Opaque LSA, containing a Links with Backup Paths TLV, is used to flood the flooding topology with backup paths from the leader of an area to all the other nodes in the area.

```
      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |            LS age             |     Options   | LS Type = 10  |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |FTBP-Type(TBDa)|                  Instance ID                  |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                      Advertising Router                       |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                      LS Sequence Number                       |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |         LS checksum           |             Length            |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     ~                  Links with Backup Paths TLV                  ~
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

        OSPFv2: Flooding Topology with Backup Paths (FTBP) Opaque LSA
```

For OSPFv3, an area scope LSA of a new LSA function code (TBDb), containing a Links with Backup Paths TLV, is used to flood the flooding topology with backup paths from the leader of an area to all the other nodes in the area.

```
        0                   1                   2                   3
        0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |            LS age             |1|0|1|     FTBP-LSA (TBDb)     |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                       Link State ID                           |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                     Advertising Router                        |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                     LS Sequence Number                        |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |         LS checksum           |             Length            |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       ~                  Links with Backup Paths TLV                  ~
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

             OSPFv3: Flooding Topology with Backup Paths (FTBP) LSA
```

6.2.3. Message for Incremental Changes

For adding some links to the flooding topology, we define a new TLV called Add Links TLVs of the following format. When some new links are added to the flooding topology, the leader may not flood the whole flooding topology with the new links to all the other nodes. It may just flood these new links. After receiving these new links, each of the other nodes adds these new links into the existing flooding topology. When the leader floods the whole flooding topology with the new links to all the other nodes, it removes the LSA for the new links. When removing the LSA for these new links, each of the other nodes does not update the flooding topology (i.e., does not remove these links from the flooding topology).

```
      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |      ADDLK-TLV-Type (TBDc)    |          TLV-Length           |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     ~           Links Encoding (Node 1 to its adjacent Nodes)       ~
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     ~           Links Encoding (Node 2 to its adjacent Nodes)       ~
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     :                                                               :
     :                                                               :
                               Add Links TLV
```

For deleting some links from the flooding topology, we define a new TLV called Delete Links TLVs of the following format. When some old links are removed from the flooding topology, the leader may not flood the whole flooding topology without the old links to all the other nodes. It may just flood these old links. After receiving these old links, each of the other nodes deletes these old links from the existing flooding topology. When the leader floods the whole flooding topology without the old links to all the other nodes, it removes the LSA for the old links. When removing the LSA for these old links, each of the other nodes does not update the flooding topology (i.e., does not add these links into the flooding topology).

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |      DELLK-TLV-Type (TBDd)    |          TLV-Length           |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    ~           Links Encoding (Node 1 to its adjacent Nodes)       ~
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    ~           Links Encoding (Node 2 to its adjacent Nodes)       ~
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    :                                                               :
    :                                                               :
```

Delete Links TLV

The Add Links TLVs and Delete Links TLVs should be in a separate LSA instance. The LSA can be a Flooding Topology LSA defined above. Alternatively, we may define a new LSA for these TLVs.

6.2.4. Leaders Selection

The leader or Designated Router (DR) selection for a broadcast link is about selecting two leaders: a DR and Backup DR. This is generalized to select two or more leaders for an area: the primary/ first leader (or leader for short), the secondary leader, the third leader and so on.

A new TLV is defined to include the information on flooding reduction of a node, which is called Flooding Reduction Information TLV or Information TLV for short. This TLV is generated by every node that supports flooding reduction in general. Every node originates a RI LSA with a Flooding Reduction Information TLV containing its priority to become a leader. The format of the TLV is as follows.

```
       0                   1                   2                   3
       0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |      INFO-TLV-Type (TBDe)     |         TLV-Length            |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |    Priority   |             Reserved (MUST be zero)           |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      ~                     sub TLVs (optional)                       ~
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Flooding Reduction Information TLV

A Priority field of eight bits is defined in the TLV to indicate the priority of the node originating the TLV to become the leader node in central mode.

A sub-TLV called leaders sub-TLV is defined. It has the following format.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |      LEADS-TLV-Type (TBDf)    |         TLV-Length            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                    1st Leader Node/Router ID                  |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                    2nd Leader Node/Router ID                  |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    ~                                                               ~
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                    nth Leader Node/Router ID                  |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Leaders sub-TLV

When a node selects itself as a leader, it originates a RI LSA containing the leader in a leaders sub-TLV.

After the first leader node is down, the other leaders will be promoted. The secondary leader becomes the first leader, the third leader becomes the secondary leader, and so on. When a node selects itself as the n-th leader, it originates a RI LSA with a Leaders sub-TLV containing n leaders.

7. Extensions to IS-IS

The extensions to IS-IS is similar to OSPF.

7.1. Extensions for Operations

A new TLV for operations is defined in IS-IS LSP. It has the following format and contains the same contents as the Flooding Reduction Instruction TLV defined in OSPF RI LSA.

```
       0                   1                   2                   3
       0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |INST-Type(TBDi1|    Length     |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |  OP | MOD |   Algorithm   |    Reserved (MUST be zero)  |  NL |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      ~                    sub TLVs (optional)                        ~
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IS-IS Flooding Reduction Instruction TLV

7.2. Extensions for Centralized Mode

7.2.1. TLV for Flooding Topology

A new TLV for the encodings of the links in the flooding topology is defined. It has the following format and contains the same contents as the Flooding Topology Links TLV defined in OSPF Flooding Topology Opaque LSA.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |FTL-Type(TBDi2)|    Length     |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    ~           Links Encoding (Node 1 to its adjacent Nodes)       ~
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    ~           Links Encoding (Node 2 to its adjacent Nodes)       ~
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    :                                                               :
    :                                                               :
```

IS-IS Flooding Topology Links TLV

7.2.2. Encodings for Backup Paths

7.2.2.1. TLVs for Backup Paths

For flooding backup paths separately, we define two TLVs: IS-IS Node Backup Paths TLV and IS-IS Link Backup Path TLV. The former has the following format and contains the same contents as Node Backup Paths TLV in OSPF.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |NBP-Type(TBDi3)|    Length     |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |Node1 Index Enc|  Variable bits
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    :                 Node1 backup paths encoding                   :
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |Node2 Index Enc|  Variable bits
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    :                 Node2 backup paths encoding                   :
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    //                                                             //
                        IS-IS Node Backup Paths TLV
```

The latter has the following format and contains the same contents as Link Backup Paths TLV in OSPF.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |LBP-Type(TBDi4)|    Length     |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |Link1 Index Enc|  Variable bits
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    :                 Link1 backup paths encoding                   :
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |Link2 Index Enc|  Variable bits
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    :                 Link2 backup paths encoding                   :
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    //                                                             //
                        IS-IS Link Backup Paths TLV
```

7.2.2.2. Backup Paths in Links TLV

A new TLV is defined to integrate the backup paths with the links on the flooding topology. It has the following format and contains the same contents as the Links with Backup Paths TLV in OSPF.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |LSB-Type(TBDi5)|    Length    |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    :          Links from Node 1 with backup paths encoding         :
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    :          Links from Node 2 with backup paths encoding         :
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    :                                                               :
    :                                                               :
                      IS-IS Links with Backup Paths TLV
```

7.2.3. TLVs for Incremental Changes

Similar to Add Links TLV in OSPF, a new TLV called IS-IS Add Links TLV is defined. It has the following format and contains the same contents as Add Links TLV in OSPF.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |ADDL-Type(TBDi6|     Length    |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    ~           Links Encoding (Node 1 to its adjacent Nodes)       ~
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    ~           Links Encoding (Node 2 to its adjacent Nodes)       ~
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    :                                                               :
    :                                                               :
                          IS-IS Add Links TLV
```

Similar to Delete Links TLV in OSPF, a new TLV called IS-IS Delete Links TLV is defined. It has the following format and contains the same contents as Delete Links TLV in OSPF.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |DELL-Type(TBDi7|      Length   |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    ~           Links Encoding (Node 1 to its adjacent Nodes)       ~
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    ~           Links Encoding (Node 2 to its adjacent Nodes)       ~
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    :                                                               :
    :                                                               :
                          IS-IS Delete Links TLV
```

7.2.4. Leaders Selection

Similar to Flooding Reduction Information TLV in OSPF, a new TLV called IS-IS Flooding Reduction Information TLV is defined. It has the following format and contains the same contents as Flooding Reduction Information TLV in OSPF.

```
      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |INF-Type(TBDi8)|     Length    |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |    Priority   |             Reserved (MUST be zero)           |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     ~                     sub TLVs (optional)                       ~
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

                  IS-IS Flooding Reduction Information TLV
```

8. Flooding Behavior

This section describes the revised flooding behavior for a node having at least one link on the flooding topology. The revised flooding procedure MUST flood an LS to every node in the network in any case, as the standard flooding procedure does.

8.1. Nodes Perform Flooding Reduction without Failure

8.1.1. Receiving an LS

When a node receives a newer LS that is not originated by itself from one of its interfaces, it floods the LS only to all the other interfaces that are on the flooding topology.

When the LS is received from an interface on the flooding topology, it is flooded only to all the other interfaces that are on the flooding topology. When the LS is received on an interface that is not on the flooding topology, it is also flooded only to all the other interfaces that are on the flooding topology.

In any case, the LS must not be transmitted back to the receiving interface.

Note before forwarding a received LS, the node would do the normal processing as usual.

8.1.2. Originating an LS

When a node originates an LS, it floods the LS to its interfaces on the flooding topology if the LS is a refresh LS (i.e., there is no significant change in the LS comparing to the previous LS); otherwise (i.e., there are significant changes in the LS), it floods the LS to all its interfaces. Choosing flooding the LS with significant changes to all the interfaces instead of limiting to the interfaces on the flooding topology would speed up the distribution of the significant link state changes.

8.1.3. Establishing Adjacencies

Adjacencies being established can be classified into two categories: adjacencies to new nodes and adjacencies to existing nodes.

8.1.3.1. Adjacency to New Node

An adjacency to a new node is an adjacency between a node (say node A) on the flooding topology and the new node (say node Y) which is not on the flooding topology. There is not any adjacency between node Y and a node in the network area.

When new node Y is up and connected to node A, node A assumes that node Y and the link between node Y and node A are on the flooding topology until a new flooding topology is computed and built. Node A may determine whether node Y is a new node through checking if node Y is reachable or on the flooding topology.

The procedure for establishing the adjacency between node A and node Y is the existing normal procedure unchanged. After the status of the adjacency reaches to Exchange or Full, node A sends node Y every new or updated LS that node A receives or originates.

8.1.3.2. Adjacency to Existing Node

An adjacency to an existing node is an adjacency between a node (say node A) on the flooding topology and the existing node (say node X) which exists on the flooding topology. There are some adjacencies between node X and some nodes in the network area.

When existing node X is connected to node A after a link between node X and node A is up, node A assumes that the link connecting node A and node X is not on the flooding topology until a new flooding topology is computed and built. Node A may determine whether node X is an existing node through checking if node X is reachable or on the flooding topology.

The procedure for establishing the adjacency between node A and node X is the existing normal procedure unchanged. Node A does not send node X any new or updated LS that node A receives or originates even after the status of the adjacency reaches to Exchange or Full.

8.2. An Exception Case

During an LS flooding, one or multiple link and node failures may happen. Some failures do not split the flooding topology, thus do not affect the flooding behavior. For example, multiple failures of the links not on the flooding topology do not split the flooding topology and do not affect the flooding behavior. The sections below focus on the failures that may split the flooding topology.

8.2.1. A Critical Failure

For a link failure, if the link is a critical link on the flooding topology, then the LS is flooded through a backup path for the link and the remaining flooding topology until a new flooding topology is computed and built; otherwise, the flooding behavior in Section 8.1 follows.

Similarly, for a node failure, if the node is a critical node on the flooding topology, then the LS is flooded through backup paths for the node and the remaining flooding topology until a new flooding topology is computed and built; otherwise, the flooding behavior in Section 8.1 follows.

8.2.2. Multiple Failures

For multiple link failures, if the number of the failed links on the flooding topology is greater than or equal to two, then the LS is flooded through a backup path for each of the failed links on the flooding topology and the remaining flooding topology until a new flooding topology is computed and built; otherwise, the flooding behavior in Section 8.1 follows.

If all the backup paths for some of the failed links are broken by some failures, the LS is flooded to all interfaces (except where it is received from) until a new flooding topology is computed and built.

For multiple node failures, the LS is flooded through the backup paths for each of the failed nodes and the remaining flooding topology until a new flooding topology is computed and built; otherwise, the flooding behavior in Section 8.1 follows.

If the backup paths for some of the failed nodes are broken by some failures, the LS is flooded to all interfaces (except where it is received from) until a new flooding topology is computed and built.

Note that if it can be quickly determined that the flooding topology is not split by the failures, the flooding behavior in Section 8.1 may follow.

9. Security Considerations

This document does not introduce any security issue.

10. IANA Considerations

10.1. OSPFv2

Under Registry Name: OSPF Router Information (RI) TLVs [RFC7770], IANA is requested to assign two new TLV values for OSPF flooding reduction as follows:

TLV Value	TLV Name	reference
11	Instruction TLV	This document
12	Information TLV	This document

Under the registry name "Opaque Link-State Advertisements (LSA) Option Types" [RFC5250], IANA is requested to assign new Opaque Type registry values for FT LSA, BP LSA, FTBP LSA as follows:

Registry Value	Opaque Type	reference
10	FT LSA	This document
11	BP LSA	This document
12	FTBP LSA	This document

IANA is requested to create and maintain new registries:

- o OSPFv2 FT LSA TLVs

Initial values for the registry are given below. The future assignments are to be made through IETF Review [RFC5226].

Value	OSPFv2 FT LSA TLV Name	Definition
0	Reserved	
1	FT Links TLV	see Section 6.2.1
2-32767	Unassigned	
32768-65535	Reserved	

- o OSPFv2 BP LSA TLVs

Initial values for the registry are given below. The future assignments are to be made through IETF Review [RFC5226].

Value	OSPFv2 TBPLSA TLV Name	Definition
0	Reserved	
1	Node Backup Paths TLV	see Section 6.2.2
2	Link Backup Paths TLV	see Section 6.2.2
3-32767	Unassigned	
32768-65535	Reserved	

- o OSPFv2 FTBP LSA TLVs

Initial values for the registry are given below. The future assignments are to be made through IETF Review [RFC5226].

Value	OSPFv2 FTBP LSA TLV Name	Definition
0	Reserved	
1	Links with Backup Paths TLV	see Section 6.2.2
2-32767	Unassigned	
32768-65535	Reserved	

10.2. OSPFv3

Under the registry name "OSPFv3 LSA Function Codes", IANA is requested to assign new registry values for FT LSA, BP LSA, FTBP LSA as follows:

Value	LSA Function Code Name	reference
16	FT LSA	This document
17	BP LSA	This document
18	FTBP LSA	This document

IANA is requested to create and maintain new registries:

- o OSPFv3 FT LSA TLVs

Initial values for the registry are given below. The future assignments are to be made through IETF Review [RFC5226].

Value	OSPFv3 FT LSA TLV Name	Definition
0	Reserved	
1	FT Links TLV	see Section 6.2.1
2-32767	Unassigned	
32768-65535	Reserved	

- o OSPFv3 BP LSA TLVs

Initial values for the registry are given below. The future assignments are to be made through IETF Review [RFC5226].

Value	OSPFv3 TBPLSA TLV Name	Definition
0	Reserved	
1	Node Backup Paths TLV	see Section 6.2.2
2	Link Backup Paths TLV	see Section 6.2.2
3-32767	Unassigned	
32768-65535	Reserved	

- o OSPFv3 FTBP LSA TLVs

Initial values for the registry are given below. The future assignments are to be made through IETF Review [RFC5226].

Value	OSPFv3 FTBP LSA TLV Name	Definition
0	Reserved	
1	Links with Backup Paths TLV	see Section 6.2.2
2-32767	Unassigned	
32768-65535	Reserved	

10.3. IS-IS

Under Registry Name: IS-IS TLV Codepoints, IANA is requested to assign new TLV values for IS-IS flooding reduction as follows:

Value	TLV Name	Definition
151	FT Links TLV	see Section 7.2.1
152	Node Backup Paths TLV	see Section 7.2.2
153	Link Backup Paths TLV	see Section 7.2.2
154	Links with Backup Paths TLV	see Section 7.2.2
155	Add Links TLV	see Section 7.2.3
156	Delete Links TLV	see Section 7.2.3
157	Instruction TLV	see Section 7.1
158	Information TLV	see Section 7.2.4

11. Acknowledgements

The authors would like to thank Acee Lindem, Zhibo Hu, Robin Li, Stephane Litkowski and Alvaro Retana for their valuable suggestions and comments on this draft.

12. References

12.1. Normative References

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

[RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, DOI 10.17487/RFC2328, April 1998, <https://www.rfc-editor.org/info/rfc2328>.

[RFC5250] Berger, L., Bryskin, I., Zinin, A., and R. Coltun, "The OSPF Opaque LSA Option", RFC 5250, DOI 10.17487/RFC5250, July 2008, <https://www.rfc-editor.org/info/rfc5250>.

[RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, DOI 10.17487/RFC5340, July 2008, <https://www.rfc-editor.org/info/rfc5340>.

[RFC7770] Lindem, A., Ed., Shen, N., Vasseur, JP., Aggarwal, R., and S. Shaffer, "Extensions to OSPF for Advertising Optional Router Capabilities", RFC 7770, DOI 10.17487/RFC7770, February 2016, <https://www.rfc-editor.org/info/rfc7770>.

12.2. Informative References

[I-D.li-dynamic-flooding]
Li, T. and P. Psenak, "Dynamic Flooding on Dense Graphs", draft-li-dynamic-flooding-05 (work in progress), June 2018.

[I-D.shen-isis-spine-leaf-ext]
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.

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

Appendix A. Algorithms to Build Flooding Topology

There are many algorithms to build a flooding topology. A simple and efficient one is briefed below.

- Select a node R according to a rule such as the node with the biggest/smallest node ID;
- Build a tree using R as root of the tree (details below); and then
- Connect k (k>=0) leaves to the tree to have a flooding topology (details follow).

A.1. Algorithms to Build Tree without Considering Others

An algorithm for building a tree from node R as root starts with a candidate queue Cq containing R and an empty flooding topology Ft:

1. Remove the first node A from Cq and add A into Ft

2. If Cq is empty, then return with Ft

3. Suppose that node Xi (i = 1, 2, ..., n) is connected to node A and not in Ft and X1, X2, ..., Xn are in a special order. For example, X1, X2, ..., Xn are ordered by the cost of the link between A and Xi. The cost of the link between A and Xi is less than the cost of the link between A and Xj (j = i + 1). If two costs are the same, Xi's ID is less than Xj's ID. In another example, X1, X2, ..., Xn are ordered by their IDs. If they are not ordered, then make them in the order.

4. Add Xi (i = 1, 2, ..., n) into the end of Cq, goto step 1.

Another algorithm for building a tree from node R as root starts with a candidate queue Cq containing R and an empty flooding topology Ft:

1. Remove the first node A from Cq and add A into Ft

2. If Cq is empty, then return with Ft

3. Suppose that node Xi (i = 1, 2, ..., n) is connected to node A and not in Ft and X1, X2, ..., Xn are in a special order. For example, X1, X2, ..., Xn are ordered by the cost of the link between A and Xi. The cost of the link between A and Xi is less than the cost of the link between A and Xj (j = i + 1). If two costs are the same, Xi's ID is less than Xj's ID. In another example, X1, X2, ..., Xn are ordered by their IDs. If they are not ordered, then make them in the order.

4. Add Xi (i = 1, 2, ..., n) into the front of Cq and goto step 1.

A third algorithm for building a tree from node R as root starts with a candidate list Cq containing R associated with cost 0 and an empty flooding topology Ft:

1. Remove the first node A from Cq and add A into Ft

2. If all the nodes are on Ft, then return with Ft

3. Suppose that node A is associated with a cost Ca which is the cost from root R to node A, node Xi (i = 1, 2, ..., n) is connected to node A and not in Ft and the cost of the link between A and Xi is LCi (i=1, 2, ..., n). Compute Ci = Ca + LCi, check if Xi is in Cq and if Cxi (cost from R to Xi) < Ci. If Xi is not in Cq, then add Xi with cost Ci into Cq; If Xi is in Cq, then If Cxi > Ci then replace Xi with cost Cxi by Xi with Ci in Cq; If Cxi == Ci then add Xi with cost Ci into Cq.

4. Make sure Cq is in a special order. Suppose that Ai (i=1, 2, ..., m) are the nodes in Cq, Cai is the cost associated with Ai,

and IDi is the ID of Ai. One order is that for any k = 1, 2, ..., m-1, Cak < Caj (j = k+1) or Cak = Caj and IDk < IDj. Goto step 1.

A.2. Algorithms to Build Tree Considering Others

An algorithm for building a tree from node R as root with consideration of others's support for flooding reduction starts with a candidate queue Cq containing R associated with previous hop PH=0 and an empty flooding topology Ft:

1. Remove the first node A that supports flooding reduction from the candidate queue Cq if there is such a node A; otherwise (i.e., if there is not such node A in Cq), then remove the first node A from Cq. Add A into the flooding topology Ft.

2. If Cq is empty or all nodes are on Ft, then return with Ft

3. Suppose that node Xi (i = 1, 2, ..., n) is connected to node A and not in the flooding topology Ft and X1, X2, ..., Xn are in a special order considering whether some of them that support flooding reduction (. For example, X1, X2, ..., Xn are ordered by the cost of the link between A and Xi. The cost of the link between A and Xi is less than that of the link between A and Xj (j = i + 1). If two costs are the same, Xi's ID is less than Xj's ID. The cost of a link is redefined such that 1) the cost of a link between A and Xi both support flooding reduction is much less than the cost of any link between A and Xk where Xk with F=0; 2) the real metric of a link between A and Xi and the real metric of a link between A and Xk are used as their costs for determining the order of Xi and Xk if they all (i.e., A, Xi and Xk) support flooding reduction or none of Xi and Xk support flooding reduction.

4. Add Xi (i = 1, 2, ..., n) associated with previous hop PH=A into the end of the candidate queue Cq, and goto step 1.

Another algorithm for building a tree from node R as root with consideration of others' support for flooding reduction starts with a candidate queue Cq containing R associated with previous hop PH=0 and an empty flooding topology Ft:

1. Remove the first node A that supports flooding reduction from the candidate queue Cq if there is such a node A; otherwise (i.e., if there is not such node A in Cq), then remove the first node A from Cq. Add A into the flooding topology Ft.

2. If Cq is empty or all nodes are on Ft, then return with Ft.

3. Suppose that node Xi (i = 1, 2, ..., n) is connected to node A and not in the flooding topology Ft and X1, X2, ..., Xn are in a special order considering whether some of them support flooding reduction. For example, X1, X2, ..., Xn are ordered by the cost of the link between A and Xi. The cost of the link between A and Xi is less than the cost of the link between A and Xj (j = i + 1). If two costs are the same, Xi's ID is less than Xj's ID. The cost of a link is redefined such that 1) the cost of a link between A and Xi both support flooding reduction is much less than the cost of any link between A and Xk where Xk does not support flooding reduction; 2) the real metric of a link between A and Xi and the real metric of a link between A and Xk are used as their costs for determining the order of Xi and Xk if they all (i.e., A, Xi and Xk) support flooding reduction or none of Xi and Xk supports flooding reduction.

4. Add Xi (i = 1, 2, ..., n) associated with previous hop PH=A into the front of the candidate queue Cq, and goto step 1.

A third algorithm for building a tree from node R as root with consideration of others' support for flooding reduction (using flag F = 1 for support, and F = 0 for not support in the following) starts with a candidate list Cq containing R associated with low order cost Lc=0, high order cost Hc=0 and previous hop ID PH=0, and an empty flooding topology Ft:

1. Remove the first node A from Cq and add A into Ft.

2. If all the nodes are on Ft, then return with Ft

3. Suppose that node A is associated with a cost Ca which is the cost from root R to node A, node Xi (i = 1, 2, ..., n) is connected to node A and not in Ft and the cost of the link between A and Xi is LCi (i=1, 2, ..., n). Compute Ci = Ca + LCi, check if Xi is in Cq and if Cxi (cost from R to Xi) < Ci. If Xi is not in Cq, then add Xi with cost Ci into Cq; If Xi is in Cq, then If Cxi > Ci then replace Xi with cost Cxi by Xi with Ci in Cq; If Cxi == Ci then add Xi with cost Ci into Cq.

4. Suppose that node A is associated with a low order cost LCa which is the low order cost from root R to node A and a high order cost HCa which is the high order cost from R to A, node Xi (i = 1, 2, ..., n) is connected to node A and not in the flooding topology Ft and the real cost of the link between A and Xi is Ci (i=1, 2, ..., n). Compute LCxi and HCxi: LCxi = LCa + Ci if both A and Xi have flag F set to one, otherwise LCxi = LCa HCxi = HCa + Ci if A or Xi does not have flag F set to one, otherwise HCxi = HCa If Xi is not in Cq, then add Xi associated with LCxi, HCxi and PH = A

into Cq; If Xi associated with LCxi’ and HCxi’ and PHxi’ is in Cq, then If HCxi’ > HCxi then replace Xi with HCxi’, LCxi’ and PHxi’ by Xi with HCxi, LCxi and PH=A in Cq; otherwise (i.e., HCxi’ == HCxi) if LCxi’ > LCxi , then replace Xi with HCxi’, LCxi’ and PHxi’ by Xi with HCxi, LCxi and PH=A in Cq; otherwise (i.e., HCxi’ == HCxi and LCxi’ == LCxi) if PHxi’ > PH, then replace Xi with HCxi’, LCxi’ and PHxi’ by Xi with HCxi, LCxi and PH=A in Cq.

5. Make sure Cq is in a special order. Suppose that Ai (i=1, 2, ..., m) are the nodes in Cq, HCai and LCai are low order cost and high order cost associated with Ai, and IDi is the ID of Ai. One order is that for any k = 1, 2, ..., m-1, HCak < HCaj (j = k+1) or HCak = HCaj and LCak < LCaj or HCak = HCaj and LCak = LCaj and IDk < IDj. Goto step 1.

A.3. Connecting Leaves

Suppose that we have a flooding topology Ft built by one of the algorithms described above. Ft is like a tree. We may connect k (k >=0) leaves to the tree to have a enhanced flooding topology with more connectivity.

Suppose that there are m (0 < m) leaves directly connected to a node X on the flooding topology Ft. Select k (k <= m) leaves through using a deterministic algorithm or rule. One algorithm or rule is to select k leaves that have smaller or larger IDs (i.e., the IDs of these k leaves are smaller/bigger than the IDs of the other leaves directly connected to node X). Since every node has a unique ID, selecting k leaves with smaller or larger IDs is deterministic.

If k = 1, the leaf selected has the smallest/largest node ID among the IDs of all the leaves directly connected to node X.

For a selected leaf L directly connected to a node N in the flooding topology Ft, select a connection/adjacency to another node from node L in Ft through using a deterministic algorithm or rule.

Suppose that leaf node L is directly connected to nodes Ni (i = 1,2,...,s) in the flooding topology Ft via adjacencies and node Ni is not node N, IDi is the ID of node Ni, and Hi (i = 1,2,...,s) is the number of hops from node L to node Ni in the flooding topology Ft.

One Algorithm or rule is to select the connection to node Nj (1 <= j <= s) such that Hj is the largest among H1, H2, ..., Hs. If there is another node Na (1 <= a <= s) and Hj = Ha, then select the one with smaller (or larger) node ID. That is that if Hj == Ha and IDj < IDa then select the connection to Nj for selecting the one with smaller node ID (or if Hj == Ha and IDj < IDa then select the connection to Na for selecting the one with larger node ID).

Suppose that the number of connections in total between leaves selected and the nodes in the flooding topology Ft to be added is NLc. We may have a limit to NLc.

Authors' Addresses

Huaimo Chen
Huawei Technologies

Email: huaimo.chen@huawei.com

Dean Cheng
Huawei Technologies

Email: dean.cheng@huawei.com

Mehmet Toy
Verizon
USA

Email: mehmet.toy@verizon.com

Yi Yang
IBM
Cary, NC
United States of America

Email: yyietf@gmail.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

Copyright Notice

Table of Contents

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.

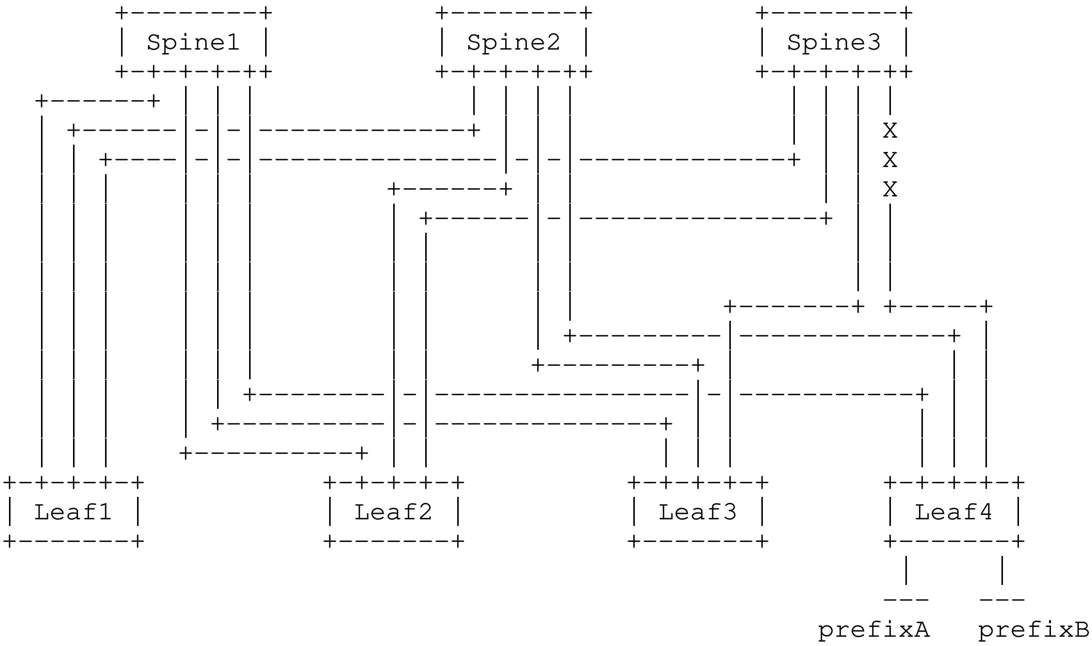

Figure 1: Topology Example

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

LSR Working Group U. Chunduri, Ed.
Internet-Draft Huawei USA
Intended status: Standards Track J. Tantsura
Expires: September 24, 2018 Nuage Networks
Y. Qu
Huawei USA
March 23, 2018

Usage of Non Shortest Path Forwarding (NSPF) IDs in IS-IS
draft-ct-isis-nspfid-for-sr-paths-01

Abstract

This document specifies the advertisement of Non Shortest Path Forwarding IDentifier (NSPF ID) TLV and the computation procedures for the same in IS-IS protocol. NSPF ID allows to simplify the data plane path description of data traffic in SR deployments. This helps mitigate the MTU issues that are caused by additional SR overhead of the packet and allows traffic statistics.

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 [RFC2119].

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

In a network implementing source routing, packets may be transported through the use of segment identifiers (SIDs), where a SID uniquely identifies a segment as defined in [I-D.ietf-spring-segment-routing]. In SR-MPLS, a segment is encoded as a label and an ordered list of segments is encoded as a stack of labels. In SRv6, a segment is encoded as an IPv6 address, with a new type of IPv6 routing header called SRH. An ordered list of segments is encoded as an ordered list of IPv6 addresses in SRH [I-D.ietf-6man-segment-routing-header].

The segment may include one or more nodes, unidirectional adjacencies between two nodes or service instruction by a particular node in the network. A Non Shortest Path (NSP) could be a Traffic Engineered (TE) path or an explicitly provisioned FRR path or a service chained path. NSP can be described using list of segments in SR. However, this creates a problem of having a relatively large stack imposed on the data packet. A path that is encoded with SIDs can be a loose or strict path. In a strict path all the nodes/links on the path are encoded as SIDs, with the expense of number of total SIDs in the stack.

1.1. Mitigation with MSD and RLD

The number of SIDs in the stack a node can impose is referred as Maximum SID Depth (MSD) capability [I-D.ietf-isis-segment-routing-msd], which must be taken into consideration when computing a path to transport a data packet in a network implementing segment routing. [I-D.ietf-isis-mpls-elc] defines Readable Label Depth (RLD) that is used by a head-end to insert Entropy Label pair (ELI/EL) at appropriate depth, so it could be read by transit nodes. There are situations where the source routed path can be excessive as path represented by SR SIDs need to describe all the nodes and ELI/EL based on the readability of the nodes in that path.

While MSD (and RLD) capabilities advertisement help mitigate the problem for a central entity to create the right source routed path per application/operator requirements; actual depth is still limited by the underlying hardware in the data path.

1.2. Issues with Increased SID Depth

Consider the following network where SR-MPLS data plane is in use and with same SRGB (5000-6000) on all nodes i.e., A1 to A7 and B1 to B7 for illustration:

```
        SID:10   SID:20   SID:30   SID:40  SID:50 SID:300(Ax)  SID:60   SID:70
        A1-------A2-------A3-------A4-------A5===============A6----------A7
                 \               /  \5     5/ \   SID:310(Ay) \          /
                  \ 10        10/    +-A10-+   \               \10      /10
                   \           /      SID:100   \               \      /
             SID:80 \A8-----A9/SID:90            \  40           \    /
                    /         \                   +---+           \  /
                   /10 B2x:125 \10                     \           \/
        B1--------B2============B3----B4--------B5-------B6----------B7
        SID:110   SID:120  SID:130   SID:140   SID:150  SID:160  SID:170
```

Figure 1: SR-MPLS Network

Global ADJ SIDs are provisioned between A5 and A6 .All other SIDs shown are nodal SID indices.

All metrics of the links are set to 1, other values as configured.

Shortest Path from A1 to A7: A2-A3-A4-A5-A6-A7

NSP1: From A1 to A7 - A2-A8-B2-B2x-A9-A10-Ax-A7; Pushed Label Stack @A1: 5020:5080:5120:5125:5090:5100:5300:5070 (where B2x is a local ADJ-SID and Ax is a global ADJ-SID)

In the above example NSP1 is represented with a combination of Adjacency and Node SIDs with a stack of 8 labels each. However, this value can be larger, if the use of entropy label is desired and based on the RLD capabilities of each node and additional labels required to insert ELI/EL at appropriate places. Though above network is shown with SR-MPLS data plane, problem is similar if the network were a SR-IPv6 network with all SIDs encoded as IPv6 SIDs in SRH.

In various SR deployments, the following issues may arise:

a. Not all nodes in the path can support MSD or RLD needed to satisfy user/operator requirements, when the number of SIDs increased to describe the source routed path. This problem gets multiplied by four times in SRH compared to MPLS data plane because of the SID size (16 bytes) in SRH.

b. Even if all nodes can support the required MSD or RLD, the bigger label stack/depth can cause potential MTU/fragmentation issues.

c. In some deployments, it is also required reducing the overhead in the network layer, especially for low packet size packets, where the actual data can be way lesser than all encapsulations and SR path overheads.

Apart from the above some deployments need path accounting statistics for path monitoring and traffic re-optimizations. [I-D.hegde-spring-traffic-accounting-for-sr-paths] proposes a solution, however this further increases the depth of SID stack. The approach could be counter productive in the environments, where SID depth is already causing deployment issues as listed above.

To mitigate the above issues, and also to facilitate forwarding plane a mechanism to identify the SR path with a corresponding data plane identifier for accounting of traffic for SR paths, this draft proposes a new IS-IS TLV (Section 2) to advertise the NSPs with Non Shortest Path Forwarding IDentifier (NSPF ID).

This draft lays out procedure for IS-IS nodes to how to use NSPF ID TLV in Section 3. With corresponding data plane, Section 3 mechanism, reduces the SID stack in the data plane from 8 SIDs shown in SR-PATH-1 and SR-PATH-2 with a single NSPF ID. This draft also introduce source routed paths with NSPF ID types defined for native IPv4 and IPv6 data planes as defined in Section 2.2.

1.3. Acronyms

EL	- Entropy Label
ELI	- Entropy Label Indicator
MPLS	- Multi Protocol Label Switching
MSD	- Maximum SID Depth
MTU	- Maximum Transferrable Unit
NSP	- Non Shortest Path
SID	- Segment Identifier
SPF	- Shortest Path First
SR	- Segment Routing
SRH	- Segment Routing Header
SR-MPLS	- Segment Routing with MPLS data plane
SRv6	- Segment Routing with Ipv6 data plane with SRH
SRH	- IPv6 Segment Routing Header

TE   - Traffic Engineering

2. Non Shortest Path Forwarding IDentifier TLV

This section describes the encoding of NSPF ID TLV. This TLV can be seen as having 3 logical section viz., encoding od FEC Prefix, encoding of NSPF-ID with description of ordered path with sub-TLVs and a set of optional non-NSP sub-TLVs which can be used to describe one or more parameters of the path. Multiple instances of this TLV MAY be advertised in IS-IS LSPs with different NSPF-ID Type and with corresponding path description sub-TLVS. The NSPF-ID TLV has Type TBD (suggested value xxx), and has the following format:

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |     Type      |    Length     |    Reserved   |     Flags     |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |          MT-ID                | Prefix Len    |  FEC Prefix   |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    //          FEC Prefix (continued, variable)                    //
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    | NSPF-ID Type  | NSPF-ID Len   | NSPF-ID Flags |  NSPF-ID Algo |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    //          NSPF-ID  (continued, variable)                      //
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |No.of NSP-STs |  NSP sub-TLVs (Variable)                        //
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |No.of Other-STs| Non-NSP sub-TLVs(variable)                    //
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: NSPF ID TLV Format

Type - TBD from IS-IS top level TLV registry.

Length - Total length of the value field in bytes (variable).

Reserved - 1 Octet reserved bits for future use. Reserved bits MUST be reset on transmission and ignored on receive.

Flags - Flags for this TLV are described in Section 2.1.

MT-ID - is the multi-topology identifier defined in [RFC5120] with 4 most significant bits reset on transmission and ignored on receive. The remaining 12-bit field contains the MT-ID.

Prefix Len - contains the length of the prefix in bits. Only the most significant octets of the Prefix are encoded.

FEC Prefix - represents the Forwarding Equivalence Class at the tail-end of the advertised NSP. The "FEC Prefix" corresponds to a routable prefix of the originating node and it MAY have one of the [RFC7794] flags set (X-Flag/R-Flag/N-Flag). Value of this field MUST be 4 octets for IPv4 "FEC Prefix". Value of this field MUST be 16 octets for IPv6 "FEC Prefix". Encoding is similar to TLV 135 and TLV 236 or MT-Capable [RFC5120] IPv4 (TLV 235) and IPv6 Prefixes (TLV 237) respectively.

2.1. Flags

Flags: 1 octet field of NSPD ID TLV has following flags defined. These flags mostly related to applicability of this TLV in an L1 area or entire IS-IS domain:

```
    NSPF ID Flags Format

        0 1 2 3 4 5 6 7
       +-+-+-+-+-+-+-+-+
       |S|D|A| Rsrvd   |
       +-+-+-+-+-+-+-+-+
```

S - If set, the NSPF ID TLV MUST be flooded across the entire routing domain. If the S flag is not set, the NSPF ID TLV MUST NOT be leaked between IS-IS levels. This bit MUST NOT be altered during the TLV leaking

D - when the NSPF ID TLV is leaked from IS-IS level-2 to level-1, the D bit MUST be set. Otherwise, this bit MUST be clear. NSPF ID TLVs with the D bit set MUST NOT be leaked from level-1 to level-2. This is to prevent TLV looping across levels.

A - The originator of the NSPF ID TLV MUST set the A bit in order to signal that the prefixes and NSPF-IDs advertised in the NSPF ID TLV are directly connected to their originators. If this bit is not set, this allows any other node in the network advertise this TLV on behalf of the originating node of the "FEC Prefix". If the NSPF ID TLV is leaked to other areas/levels the A-flag MUST be cleared.

Rsrvd - reserved bits for future use. Reserved bits MUST be reset on transmission and ignored on receive.

2.2. NSPF-ID Fields

This represents the actual data plane identifier in the packet and could be of any data plane as defined in type field. Both "FEC Prefix" and NSPF-ID MUST belong to a same node in the network.

1. NSPF-ID Type: This is a new registry (TBD IANA) for this TLV and the defined types are as follows. Type: 1 - MPLS SID/Label Type: 2 Native IPv4 Address Type: 3 Native IPv6 Address Type 4: IPv6 SID in SRv6 with SRH

2. NSPF-ID Len: Length of the NSPF Identifier field in octets and this depends on the NSPF-ID type. See NSPF-ID below for the length of this field and other considerations.

3. NSPF-ID Flags: 1 Octet field for NSPF-ID flags. Some of the bits could be NSPF-ID type specific and each new type MUST define the flags applicable to the NSPF-ID type. For NSPF-ID Type 1, the flags are same as Section 2.1 definition in [I-D.ietf-isis-segment-routing-extensions]. For NSPF-ID Type 2, 3 and NSPF-ID Type 4 only 'R' flag is applicable. Undefined flags for each NSPF-ID type MUST be considered as reserved. Reserved flag bits in each NSPF-ID type specific flags MUST be reset on transmission and ignored on receive.

4. NSPF-ID Algo: 1 octet value represents the SPF algorithm. Algorithm registry is as defined in [I-D.ietf-isis-segment-routing-extensions].

5. NSPF-ID: This is the NSP forwarding identifier that would be on the data packet. The value of this field is variable and it depends on the NSPF-ID Type. For Type 1, this is and MPLS SID/ Label. For Type 2 this is a 4 byte IPv4 address. For Type 3 and Type 4, it is a 16 byte IPv6 address. For NSPF-ID Type 2, 3 or 4, if the NSPF-ID Len is set to 0, then FEC Prefix would also become the NSPF-ID. In the case when NSPF-ID Len is 0 and NSPF-ID Type is 2, then FEC Prefix length MUST be a 4 byte IPv4 address. Similarly, if NSPF-ID Type is 3 or 4 with NSPF-ID Len is set to 0, then FEC Prefix MUST be of a 16 byte IPv6 Address. For NSPF-ID Type 2, 3 or 4, if the NSPF-ID Len is set to non zero value, then the NSPF-ID MUST not be advertised as a routable prefix in TLV 135, TLV 235, TLV 236 and TLV 237, but that MUST belog to the node where "FEC Prefix" is advertised.

6. No.of NSP-STs: Total number of the NSP sub-TLVs are defined with this 1-octet field. The value MUST NOT be zero.

2.3. NSP sub-TLVs

A new sub-TLV registry is created (TBD IANA) called NSP sub-TLVs. These are used to describe the path in the form of set of contiguous and ordered sub-TLVs, with first sub-TLV representing the top of the stack or first segment. These set of ordered TLVs can have both topological SIDs and non-topological SIDs (e.g., service segments).

Type 1: SID/Label sub-TLV as defined in [I-D.ietf-isis-segment-routing-extensions]. Only Type is defined and Length/Value fields are per Section 2.3 of the referenced document.

Type 2: Prefix SID sub-TLV as defined in [I-D.ietf-isis-segment-routing-extensions]. Only Type is defined and Length/Value fields are per Section 2.1 of the referenced document.

Type 3: Adjacency SID sub-TLV as defined in [I-D.ietf-isis-segment-routing-extensions]. Only Type is defined and Length/Value fields are per Section 2.2 of the referenced document.

Type 4: Length 4 bytes, value is 4 bytes IPv4 address encoded similar to IPv4 FEC Prefix described above.

Type 5: Length 16 bytes; value is 16 bytes IPv6 address encoded similar to IPv6 FEC Prefix described above.

Type 6: SRv6 Node SID TLV as defined in [I-D.bashandy-isis-srv6-extensions]. Only Type is defined and Length/Value fields are per Section 4 of the referenced document.

Type 7: SRv6 Adjacency-SID sub-TLV as defined in [I-D.bashandy-isis-srv6-extensions]. Only Type is defined and Length/Value fields are per Section 6.1 of the referenced document.

Type 8: SRv6 LAN Adjacency-SID sub-TLV as defined in [I-D.bashandy-isis-srv6-extensions]. Only Type is defined and Length/Value fields are per Section 6.2 of the referenced document.

2.4. Non-NSP sub-TLVs

NSPF ID TLV also defines a new sub-TLV registry (TBD IANA) for defining extensible set of sub-TLVs other than describing the path sub-TLVs. Total number of the path sub-TLVs to describe the path are defined in 1-octet field "No.of Other-STs" just before the Non-NSP sub-TLVs. This field serves as a demarcation for set of ordered NSP sub-TLVs and Non-NSP sub-TLVs.

Type 1: Length 0 No value field. Specifies a counter to count number of packets forwarded on this NSPF-ID.

Type 2: Length 0 No value field. Specifies a counter to count number of bytes forwarded on this NSPF-ID specified in the network header (e.g. IPv4, IPv6).

Type 3: Length 4 bytes, and Value is metric of this path represented through the NSPF-ID. Different nodes can advertise the same NSPF-ID for the same FEC-Prefix with a different set of NSP sub-TLVs and the receiving node MUST consider the lowest metric value (TBD more, what happens when metric is same for two different set of NSP sub-TLVs).

3. Elements of Procedure

As specified in Section 1, a NSP can be a TE path, locally provisioned by the operator. Consider the following IS-IS network to describe the operation of NSPF ID TLV as defined in Section 2:

```
                                           1
                                        _______
                                       /   1   \
                                   +---R2-------R3---+
                                  /    \_______/      \
                                 /         1           \
                              1 /       2               \ 1
                               /       ______            \
                              /       /      \            \
                            R1------R6       R7-----------R4
                              \  2    \______/    2       /\
                               \          2              /  \
                              3 \                       / 3  \1
                                 \                 4   /      \
                                  +----R8------R9-----R10------R12
                                                \          1   /
                                               1 \            / 1
                                                  +----R11---+
```

Figure 3: IS-IS Network

In the above diagram (Figure 3) node R1 is an ingress node, or a head-end node, and the node R4 may be an egress node or another head-end node. The numbers shown on each of the links between nodes R1-R11 indicate a IS-IS metric as provisioned by the operator. R1 may be configured to receive TE source routed path information from a central entity (PCE or Controller) that comprise NSP information which relates to sources that are attached to R1. It is also possible to have an NSP provisioned locally by the operator for non-TE needs (FRR or for chaining certain services). The NSP information is encoded as an ordered list of segments from source to a destination node in the network and is represented with an NSPF-ID. The NSP information includes NSP sub-TLVS which represents both topological and non-topological segments and specifies the actual path towards a FEC/Prefix by R4.

The shortest path towards R4 from R1 are through the following sequence of nodes: R1-R2-R3-R4 based on the configured metrics. The central entity may define a few NSPs from R1 to R4 that deviate from the shortest path based on other network characteristic requirements as requested by an application or service. For example, the network characteristics or performance requirements may include bandwidth, jitter, latency, throughput, error rate, etc. A first NSP may be identified by NSPF ID = 2 and may include the path of R1-R6-R7-R4 for a FEC Prefix advertised by R4. A second NSP may be identified by NSPF ID = 3 and may include the path of R1-R8-R9-R10-R4. Though these example shows NSP with all nodal SIDs, it is possible to have an NSP with combination of node and adjacency SIDS (local or global) or with non-topological segments along with these.

Each receiving node, determine whether an advertised NSP includes information regarding the receiving node. This MAY be done, during the end of the SPF computation for MTID that is advertised in this TLV and for the FEC/Prefix. For example, node R9 receives the NSP information, and ignores the first NSP identified by NSPF ID = 2 because this NSP does not include node R9.

However, node R9 may determine that the second NSP identified by NSPF ID = 3 does include the node R9 for the FEC prefix advertised by R4. Therefore, node R9 updates the local forwarding database to include an entry for the destination address of R4 that indicates, that when a data packet comprising a NSPF-ID of 3 is received, forward the data packet to node R10 instead of R11. This is even though from R9 the shortest path cost to reach R4 via R11 is 3 (R9-R11-R12-R4) it chooses the nexthop to R10 to reach R4 as specified in the NSP. Same process happens to all the nodes on the NSP.

In summary, the receiving node checks if this node is on the path, if yes, it adjusts the shortest path nexthop computed towards "FEC Prefix" to the shortest path nexthop towards the next segment as specified in the NSP.

4. NSPF ID Data Plane aspects

Data plane NSPF ID is selected by the entity (e.g., a controller, locally provisioned) which selects a particular NSP in the network. Section 2.2 defines various data plane identifier types and a corresponding data plane identifier type and identifier is selected by the entity which selects the NSP. Other data planes other than described below can also use this TLV to describe the NSP. Further details TBD.

4.1. MPLS Data Plane

If NSPF-ID Type is 1, the NSP belongs to SR-MPLS data plane and the complete NSP stack is represented with a unique SR SID/Label and this gets programmed on the data plane with the appropriate nexthop computed as specified in Section 3 . NSP path description here is a set of ordered SID TLVs and MAY contain both topological and non-topological segments.

4.2. SRv6 Data Plane

If NSPF-ID Type is 4, the NSP belongs to SRv6 with SRH data plane and the complete NSP stack is represented with IPv6 SIDs and this gets programmed on the data plane with the appropriate nexthop computed as specified in Section 3. NSP path description here is a set of ordered SID TLVs and MAY contain both topological and non-topological segments (e.g. network functions, service functions). If NSPF-ID Type is 3 this is the traditional IPv6 mode ([I-D.ietf-dmm-srv6-mobile-uplane]) and this doesn’t include SRH and NSP stack description is similar to NSPF-ID Type 4 as described.

5. NSP Traffic Accounting

As described in Section 2.4, each node described in the NSP optional sub-TLVs can provision the hardware to account the traffic statistics as indicated in the non-NSP sub-TLVs for the actual data traffic. with this more granular and dynamic enablement of traffic statistics for only certain NSPs would be possible. This approach, thus is more safe and secure than any mechanism that involves creating state in the nodes with data traffic itself. This is because creation and deletion of the traffic accounting state for NSPs happen through IS-IS LSP processing and IS-IS security Section 8 options are applicable to this TLV.

How the traffic accounting is distributed to a central entity is out of scope of this document. One can use any method (e.g. gRPC) to extract the NSPF-ID traffic stats from various nodes along the path.

6. Acknowledgements

Thanks to Richard Li, Alex Clemm, Kiran Makhijani and Lin Han for initial discussions on this topic.

Earlier versions of draft-ietf-isis-segment-routing-extensions have a mechanism to advertise EROs through Binding SID.

7. IANA Considerations

This document requests the following new TLVin IANA IS-IS TLV code-point registry.

TLV #	Name
TBD	NSPF ID TLV

This document also requests IANA to create new registries for NSPF-ID Type, NSP sub-TLVs and Non-NSP sub-TLVs in NSPF ID TLV as described in Section 2.

8. Security Considerations

Security concerns for IS-IS are addressed in [RFC5304] and [RFC5310]. Further security analysis for IS-IS protocol is done in [RFC7645]. Advertisement of the additional information defined in this document introduces no new security concerns in IS-IS protocol. However as this extension is related to SR-MPLS and SRH data planes as defined in [I-D.ietf-spring-segment-routing], those particular data plane security considerations does apply here.

9. References

9.1. Normative References

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

9.2. Informative References

[I-D.bashandy-isis-srv6-extensions]
Ginsberg, L., Bashandy, A., Filsfils, C., Decraene, B., and Z. Hu, "IS-IS Extensions to Support Routing over IPv6 Dataplane", draft-bashandy-isis-srv6-extensions-02 (work in progress), March 2018.

[I-D.hegde-spring-traffic-accounting-for-sr-paths]
Hegde, S., "Traffic Accounting for MPLS Segment Routing Paths", draft-hegde-spring-traffic-accounting-for-sr-paths-01 (work in progress), October 2017.

[I-D.ietf-6man-segment-routing-header]
Previdi, S., Filsfils, C., Leddy, J., Matsushima, S., and d. daniel.voyer@bell.ca, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-10 (work in progress), March 2018.

[I-D.ietf-dmm-srv6-mobile-uplane]
Matsushima, S., Filsfils, C., Kohno, M., Camarillo, P., daniel.voyer@bell.ca, d., and C. Perkins, "Segment Routing IPv6 for Mobile User Plane", draft-ietf-dmm-srv6-mobile-uplane-01 (work in progress), March 2018.

[I-D.ietf-isis-mpls-elc]
Xu, X., Kini, S., Sivabalan, S., Filsfils, C., and S. Litkowski, "Signaling Entropy Label Capability and Readable Label-stack Depth Using IS-IS", draft-ietf-isis-mpls-elc-03 (work in progress), January 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-15 (work in progress), December 2017.

[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-09 (work in progress), January 2018.

[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.

[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>.

[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>.

[RFC7413] Cheng, Y., Chu, J., Radhakrishnan, S., and A. Jain, "TCP Fast Open", RFC 7413, DOI 10.17487/RFC7413, December 2014, <https://www.rfc-editor.org/info/rfc7413>.

[RFC7645] Chunduri, U., Tian, A., and W. Lu, "The Keying and Authentication for Routing Protocol (KARP) IS-IS Security Analysis", RFC 7645, DOI 10.17487/RFC7645, September 2015, <https://www.rfc-editor.org/info/rfc7645>.

[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>.

Authors' Addresses

Uma Chunduri (editor)
Huawei USA
2330 Central Expressway
Santa Clara, CA 95050
USA

Email: uma.chunduri@huawei.com

Jeff Tantsura
Nuage Networks
755 Ravendale Drive
Mountain View, CA 94043
USA

Email: jefftant.ietf@gmail.com

Yingzhen Qu
Huawei USA
2330 Central Expressway
Santa Clara, CA 95050
USA

Email: yingzhen.qu@huawei.com

IS-IS for IP Internets L. Ginsberg
Internet-Draft P. Wells
Obsoletes: 5306 (if approved) Cisco Systems, Inc.
Intended status: Standards Track June 28, 2018
Expires: December 30, 2018

Restart Signaling for IS-IS
draft-ginsberg-isis-rfc5306bis-01

Abstract

This document describes a mechanism for a restarting router to signal to its neighbors that it is restarting, allowing them to reestablish their adjacencies without cycling through the down state, while still correctly initiating database synchronization.

This document additionally describes a mechansim for a router to signal its neighbors that it is preparing to initiate a restart while maintaining forwarding plane state. This allows the neighbors to maintain their adjacencies until the router has restarted, but also allows the neighbors to bring the adjacencies down in the event of other topology changes.

This document additionally describes a mechanism for a restarting router to determine when it has achieved Link State Protocol Data Unit (LSP) database synchronization with its neighbors and a mechanism to optimize LSP database synchronization, while minimizing transient routing disruption when a router starts.

This document obsoletes RFC 5306.

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

Copyright Notice

Table of Contents

1. Overview

The Intermediate System to Intermediate System (IS-IS) routing protocol [RFC1195] [ISO10589] is a link state intra-domain routing protocol. Normally, when an IS-IS router is restarted, temporary disruption of routing occurs due to events in both the restarting router and the neighbors of the restarting router.

The router that has been restarted computes its own routes before achieving database synchronization with its neighbors. The results of this computation are likely to be non-convergent with the routes computed by other routers in the area/domain.

Neighbors of the restarting router detect the restart event and cycle their adjacencies with the restarting router through the down state. The cycling of the adjacency state causes the neighbors to regenerate their LSPs describing the adjacency concerned. This in turn causes a temporary disruption of routes passing through the restarting router.

In certain scenarios, the temporary disruption of the routes is highly undesirable. This document describes mechanisms to avoid or minimize the disruption due to both of these causes.

When an adjacency is reinitialized as a result of a neighbor restarting, a router does three things:

1. It causes its own LSP(s) to be regenerated, thus triggering SPF runs throughout the area (or in the case of Level 2, throughout the domain).

2. It sets SRMflags on its own LSP database on the adjacency concerned.

3. In the case of a Point-to-Point link, it transmits a complete set of Complete Sequence Number PDUs (CSNPs), over the adjacency.

In the case of a restarting router process, the first of these is highly undesirable, but the second is essential in order to ensure synchronization of the LSP database.

The third action above minimizes the number of LSPs that must be exchanged and, if made reliable, provides a means of determining when the LSP databases of the neighboring routers have been synchronized. This is desirable whether or not the router is being restarted (so that the overload bit can be cleared in the router’s own LSP, for example).

This document describes a mechanism for a restarting router to signal that it is restarting to its neighbors, and allow them to reestablish their adjacencies without cycling through the down state, while still correctly initiating database synchronization.

This document additionally describes a mechanism for a restarting router to determine when it has achieved LSP database synchronization with its neighbors and a mechanism to optimize LSP database synchronization and minimize transient routing disruption when a router starts.

It is assumed that the three-way handshake [RFC5303] is being used on Point-to-Point circuits.

2. Approach

2.1. Timers

Three additional timers, T1, T2, and T3, are required to support the functionality defined in this document.

An instance of the timer T1 is maintained per interface, and indicates the time after which an unacknowledged (re)start attempt will be repeated. A typical value might be 3 seconds.

An instance of the timer T2 is maintained for each LSP database (LSPDB) present in the system, i.e., for a Level 1/2 system, there will be an instance of the timer T2 for Level 1 and an instance for Level 2. This is the maximum time that the system will wait for LSPDB synchronization. A typical value might be 60 seconds.

A single instance of the timer T3 is maintained for the entire system. It indicates the time after which the router will declare that it has failed to achieve database synchronization (by setting the overload bit in its own LSP). This is initialized to 65535 seconds, but is set to the minimum of the remaining times of received IS-IS Hellos (IIHs) containing a restart TLV with the Restart Acknowledgement (RA) set and an indication that the neighbor has an adjacency in the "UP" state to the restarting router.

NOTE: The timer T3 is only used by a restarting router.

2.2. Restart TLV

A new TLV is defined to be included in IIH PDUs. The presence of this TLV indicates that the sender supports the functionality defined in this document and it carries flags that are used to convey information during a (re)start. All IIHs transmitted by a router that supports this capability MUST include this TLV.

```
 Type 211

      Length: Number of octets in the Value field (1 to (3 + ID Length))
      Value

                                     No. of octets
        +-----------------------+
        |   Flags               |     1
        +-----------------------+
        | Remaining Time        |     2
        +-----------------------+
        | Restarting Neighbor ID|     ID Length
        +-----------------------+

    Flags (1 octet)

         0  1  2  3  4  5  6  7
        +--+--+--+--+--+--+--+--+
        |Reserved|PA|PR|SA|RA|RR|
        +--+--+--+--+--+--+--+--+

        RR - Restart Request
        RA - Restart Acknowledgement
        SA - Suppress adjacency advertisement
        PR - Restart is planned
        PA - Planned restart acknowledgement

   (Note: Remaining fields are required when the RA bit is set.)
     Remaining Time (2 octets)

        Remaining holding time (in seconds)

      Restarting Neighbor System ID (ID Length octets)
```

The System ID of the neighbor to which an RA refers. Note: Implementations based on earlier versions of this document may not include this field in the TLV when the RA is set. In this case, a router that is expecting an RA on a LAN circuit SHOULD assume that the acknowledgement is directed at the local system.

2.2.1. Use of RR and RA Bits

The RR bit is used by a (re)starting router to signal to its neighbors that a (re)start is in progress, that an existing adjacency SHOULD be maintained even under circumstances when the normal operation of the adjacency state machine would require the adjacency to be reinitialized, to request a set of CSNPs, and to request setting of the SRMflags.

The RA bit is sent by the neighbor of a (re)starting router to acknowledge the receipt of a restart TLV with the RR bit set.

When the neighbor of a (re)starting router receives an IIH with the restart TLV having the RR bit set, if there exists on this interface an adjacency in state "UP" with the same System ID, and in the case of a LAN circuit, with the same source LAN address, then, irrespective of the other contents of the "Intermediate System Neighbors" option (LAN circuits) or the "Point-to-Point Three-Way Adjacency" option (Point-to-Point circuits):

a. the state of the adjacency is not changed. If this is the first IIH with the RR bit set that this system has received associated with this adjacency, then the adjacency is marked as being in "Restart mode" and the adjacency holding time is refreshed -- otherwise, the holding time is not refreshed. The "remaining time" transmitted according to (b) below MUST reflect the actual time after which the adjacency will now expire. Receipt of a normal IIH with the RR bit reset will clear the "Restart mode" state. This procedure allows the restarting router to cause the neighbor to maintain the adjacency long enough for restart to successfully complete, while also preventing repetitive restarts from maintaining an adjacency indefinitely. Whether or not an adjacency is marked as being in "Restart mode" has no effect on adjacency state transitions.

b. immediately (i.e., without waiting for any currently running timer interval to expire, but with a small random delay of a few tens of milliseconds on LANs to avoid "storms") transmit over the corresponding interface an IIH including the restart TLV with the RR bit clear and the RA bit set, in the case of Point-to-Point adjacencies having updated the "Point-to-Point Three-Way Adjacency" option to reflect any new values received from the (re)starting router. (This allows a restarting router to quickly acquire the correct information to place in its hellos.) The "Remaining Time" MUST be set to the current time (in seconds) before the holding timer on this adjacency is due to expire. If the corresponding interface is a LAN interface, then the Restarting Neighbor System ID SHOULD be set to the System ID of the router from which the IIH with the RR bit set was received. This is required to correctly associate the acknowledgement and holding time in the case where multiple systems on a LAN restart at approximately the same time. This IIH SHOULD be transmitted before any LSPs or SNPs are transmitted as a result of the receipt of the original IIH.

c. if the corresponding interface is a Point-to-Point interface, or if the receiving router has the highest LnRouterPriority (with the highest source MAC (Media Access Control) address breaking ties) among those routers to which the receiving router has an adjacency in state "UP" on this interface whose IIHs contain the restart TLV, excluding adjacencies to all routers which are considered in "Restart mode" (note the actual DIS is NOT changed by this process), initiate the transmission over the corresponding interface of a complete set of CSNPs, and set SRMflags on the corresponding interface for all LSPs in the local LSP database.

Otherwise (i.e., if there was no adjacency in the "UP" state to the System ID in question), process the IIH as normal by reinitializing the adjacency and setting the RA bit in the returned IIH.

2.2.2. Use of the SA Bit

The SA bit is used by a starting router to request that its neighbor suppress advertisement of the adjacency to the starting router in the neighbor’s LSPs.

A router that is starting has no maintained forwarding function state. This may or may not be the first time the router has started. If this is not the first time the router has started, copies of LSPs generated by this router in its previous incarnation may exist in the LSP databases of other routers in the network. These copies are likely to appear "newer" than LSPs initially generated by the starting router due to the reinitialization of LSP fragment sequence numbers by the starting router. This may cause temporary blackholes to occur until the normal operation of the update process causes the starting router to regenerate and flood copies of its own LSPs with higher sequence numbers. The temporary blackholes can be avoided if the starting router’s neighbors suppress advertising an adjacency to the starting router until the starting router has been able to propagate newer versions of LSPs generated by previous incarnations.

When a router receives an IIH with the restart TLV having the SA bit set, if there exists on this interface an adjacency in state "UP" with the same System ID, and in the case of a LAN circuit, with the same source LAN address, then the router MUST suppress advertisement of the adjacency to the neighbor in its own LSPs. Until an IIH with the SA bit clear has been received, the neighbor advertisement MUST continue to be suppressed. If the adjacency transitions to the "UP" state, the new adjacency MUST NOT be advertised until an IIH with the SA bit clear has been received.

Note that a router that suppresses advertisement of an adjacency MUST NOT use this adjacency when performing its SPF calculation. In particular, if an implementation follows the example guidelines presented in [ISO10589], Annex C.2.5, Step 0:b) "pre-load TENT with the local adjacency database", the suppressed adjacency MUST NOT be loaded into TENT.

2.2.3. Use of PR and PA Bits

The PR bit is used by a router which is planning to initiate a restart to signal to its neighbors that it will be restarting.

The PA bit is sent by the neighbor of a router planning to restart to acknowledge receipt of a restart TLV with the PR bit set.

When the neighbor of a router planning a restart receives an IIH with the restart TLV having the PR bit set, if there exists on this interface an adjacency in state "UP" with the same System ID, and in the case of a LAN circuit, with the same source LAN address, then:

a. if this is the first IIH with the PR bit set that this system has received associated with this adjacency, then the adjacency is marked as being in "Planned Restart state" and the adjacency holding time is refreshed -- otherwise, the holding time is not refreshed. The "remaining time" transmitted according to (b) below MUST reflect the actual time after which the adjacency will now expire. Receipt of a normal IIH with the PR bit reset will clear the "Planned Restart mode" state. This procedure allows the router planning a restart to cause the neighbor to maintain the adjacency long enough for restart to successfully complete. Whether or not an adjacency is marked as being in "Planned Restart mode" has no effect on adjacency state transitions.

b. immediately (i.e., without waiting for any currently running timer interval to expire, but with a small random delay of a few tens of milliseconds on LANs to avoid "storms") transmit over the corresponding interface an IIH including the restart TLV with the PR bit clear and the PA bit set. The "Remaining Time" MUST be set to the current time (in seconds) before the holding timer on this adjacency is due to expire. If the corresponding interface is a LAN interface, then the Restarting Neighbor System ID SHOULD be set to the System ID of the router from which the IIH with the

PR bit set was received. This is required to correctly associate the acknowledgement and holding time in the case where multiple systems on a LAN are planning a restart at approximately the same time.

While a control plane restart is in progress it is expected that the restarting router will be unable to respond to topology changes. It is therefore useful to signal a planned restart (if the forwarding plane on the restarting router is maintained) so that the neighbors of the restarting router can determine whether it is safe to maintain the adjacency if other topology changes occur prior to the completion of the restart. Signalling a planned restart in the absence of maintained forwarding plane state is likely to lead to significant traffic loss and MUST NOT be done.

Neighbors of the router which has signaled planned restart SHOULD maintain the adjacency in a planned restart state until it receives an IIH with the RR bit set, receives an IIH with both PR and RR bits clear, or the adjacency holding time expires - whichever occurs first.

While the adjacency is in planned restart state the following actions MAY be taken:

a. If additional topology changes occur, the adjacency which is in planned restart state MAY be brought down even though the hold time has not yet expired. Given that the neighbor which has signaled a planned restart is not expected to update its forwarding plane in response to signaling of the topology changes (since it is restarting) traffic which transits that node is at risk of being improperly forwarded. On a LAN circuit, if the router in planned restart state is the DIS at any supported level, the adjacency(ies) SHOULD be brought down whenever any LSP update is either generated or received so as to trigger a new DIS election. Failure to do so will compromise the reliability of the Update Process on that circuit. What other criteria are used to determine what topology changes will trigger bringing the adjacency down is a local implementation decision.

b. If a BFD session to the neighbor which signals a planned restart is in the UP state and subsequently goes DOWN, the event MAY be ignored since it is possible this is an expected side effect of the restart. Use of the Control Plane Independent state as signalled in BFD control packets [RFC5880] SHOULD be considered in the decision to ignore a BFD Session DOWN event

c. On a Point-to-Point circuit, transmission of LSPs, CSNPs, and PSNPs MAY be suppressed. It is expected that the PDUs will not be received.

2.3. Adjacency (Re)Acquisition

Adjacency (re)acquisition is the first step in (re)initialization. Restarting and starting routers will make use of the RR bit in the restart TLV, though each will use it at different stages of the (re)start procedure.

2.3.1. Adjacency Reacquisition during Restart

The restarting router explicitly notifies its neighbor that the adjacency is being reacquired, and hence that it SHOULD NOT reinitialize the adjacency. This is achieved by setting the RR bit in the restart TLV. When the neighbor of a restarting router receives an IIH with the restart TLV having the RR bit set, if there exists on this interface an adjacency in state "UP" with the same System ID, and in the case of a LAN circuit, with the same source LAN address, then the procedures described in Section 3.2.1 are followed.

A router that does not support the restart capability will ignore the restart TLV and reinitialize the adjacency as normal, returning an IIH without the restart TLV.

On restarting, a router initializes the timer T3, starts the timer T2 for each LSPDB, and for each interface (and in the case of a LAN circuit, for each level) starts the timer T1 and transmits an IIH containing the restart TLV with the RR bit set.

On a Point-to-Point circuit, the restarting router SHOULD set the "Adjacency Three-Way State" to "Init", because the receipt of the acknowledging IIH (with RA set) MUST cause the adjacency to enter the "UP" state immediately.

On a LAN circuit, the LAN-ID assigned to the circuit SHOULD be the same as that used prior to the restart. In particular, for any circuits for which the restarting router was previously DIS, the use of a different LAN-ID would necessitate the generation of a new set of pseudonode LSPs, and corresponding changes in all the LSPs referencing them from other routers on the LAN. By preserving the LAN-ID across the restart, this churn can be prevented. To enable a restarting router to learn the LAN-ID used prior to restart, the LAN-ID specified in an IIH with RR set MUST be ignored.

Transmission of "normal" IIHs is inhibited until the conditions described below are met (in order to avoid causing an unnecessary adjacency initialization). Upon expiry of the timer T1, it is restarted and the IIH is retransmitted as above.

When a restarting router receives an IIH a local adjacency is established as usual, and if the IIH contains a restart TLV with the RA bit set (and on LAN circuits with a Restart Neighbor System ID that matches that of the local system), the receipt of the acknowledgement over that interface is noted. When the RA bit is set and the state of the remote adjacency is "UP", then the timer T3 is set to the minimum of its current value and the value of the "Remaining Time" field in the received IIH.

On a Point-to-Point link, receipt of an IIH not containing the restart TLV is also treated as an acknowledgement, since it indicates that the neighbor is not restart capable. However, since no CSNP is guaranteed to be received over this interface, the timer T1 is cancelled immediately without waiting for a complete set of CSNPs. Synchronization may therefore be deemed complete even though there are some LSPs which are held (only) by this neighbor (see Section 3.4). In this case, we also want to be certain that the neighbor will reinitialize the adjacency in order to guarantee that the SRMflags have been set on its database, thus ensuring eventual LSPDB synchronization. This is guaranteed to happen except in the case where the Adjacency Three-Way State in the received IIH is "UP" and the Neighbor Extended Local Circuit ID matches the extended local circuit ID assigned by the restarting router. In this case, the restarting router MUST force the adjacency to reinitialize by setting the local Adjacency Three-Way State to "DOWN" and sending a normal IIH.

In the case of a LAN interface, receipt of an IIH not containing the restart TLV is unremarkable since synchronization can still occur so long as at least one of the non-restarting neighboring routers on the LAN supports restart. Therefore, T1 continues to run in this case. If none of the neighbors on the LAN are restart capable, T1 will eventually expire after the locally defined number of retries.

In the case of a Point-to-Point circuit, the "LocalCircuitID" and "Extended Local Circuit ID" information contained in the IIH can be used immediately to generate an IIH containing the correct three-way handshake information. The presence of "Neighbor Extended Local Circuit ID" information that does not match the value currently in use by the local system is ignored (since the IIH may have been transmitted before the neighbor had received the new value from the restarting router), but the adjacency remains in the initializing state until the correct information is received.

In the case of a LAN circuit, the source neighbor information (e.g., SNPAAddress) is recorded and used for adjacency establishment and maintenance as normal.

When BOTH a complete set of CSNPs (for each active level, in the case of a Point-to-Point circuit) and an acknowledgement have been received over the interface, the timer T1 is cancelled.

Once the timer T1 has been cancelled, subsequent IIHs are transmitted according to the normal algorithms, but including the restart TLV with both RR and RA clear.

If a LAN contains a mixture of systems, only some of which support the new algorithm, database synchronization is still guaranteed, but the "old" systems will have reinitialized their adjacencies.

If an interface is active, but does not have any neighboring router reachable over that interface, the timer T1 would never be cancelled, and according to Section 3.4.1.1, the SPF would never be run. Therefore, timer T1 is cancelled after some predetermined number of expirations (which MAY be 1).

2.3.2. Adjacency Acquisition during Start

The starting router wants to ensure that in the event that a neighboring router has an adjacency to the starting router in the "UP" state (from a previous incarnation of the starting router), this adjacency is reinitialized. The starting router also wants neighboring routers to suppress advertisement of an adjacency to the starting router until LSP database synchronization is achieved. This is achieved by sending IIHs with the RR bit clear and the SA bit set in the restart TLV. The RR bit remains clear and the SA bit remains set in subsequent transmissions of IIHs until the adjacency has reached the "UP" state and the initial T1 timer interval (see below) has expired.

Receipt of an IIH with the RR bit clear will result in the neighboring router utilizing normal operation of the adjacency state machine. This will ensure that any old adjacency on the neighboring router will be reinitialized.

Upon receipt of an IIH with the SA bit set, the behavior described in Section 3.2.2 is followed.

Upon starting, a router starts timer T2 for each LSPDB.

For each interface (and in the case of a LAN circuit, for each level), when an adjacency reaches the "UP" state, the starting router starts a timer T1 and transmits an IIH containing the restart TLV with the RR bit clear and SA bit set. Upon expiry of the timer T1, it is restarted and the IIH is retransmitted with both RR and SA bits set (only the RR bit has changed state from earlier IIHs).

Upon receipt of an IIH with the RR bit set (regardless of whether or not the SA bit is set), the behavior described in Section 2.2.1 is followed.

When an IIH is received by the starting router and the IIH contains a restart TLV with the RA bit set (and on LAN circuits with a Restart Neighbor System ID that matches that of the local system), the receipt of the acknowledgement over that interface is noted.

On a Point-to-Point link, receipt of an IIH not containing the restart TLV is also treated as an acknowledgement, since it indicates that the neighbor is not restart capable. Since the neighbor will have reinitialized the adjacency, this guarantees that SRMflags have been set on its database, thus ensuring eventual LSPDB synchronization. However, since no CSNP is guaranteed to be received over this interface, the timer T1 is cancelled immediately without waiting for a complete set of CSNPs. Synchronization may therefore be deemed complete even though there are some LSPs that are held (only) by this neighbor (see Section 2.4).

In the case of a LAN interface, receipt of an IIH not containing the restart TLV is unremarkable since synchronization can still occur so long as at least one of the non-restarting neighboring routers on the LAN supports restart. Therefore, T1 continues to run in this case. If none of the neighbors on the LAN are restart capable, T1 will eventually expire after the locally defined number of retries. The usual operation of the update process will ensure that synchronization is eventually achieved.

When BOTH a complete set of CSNPs (for each active level, in the case of a Point-to-Point circuit) and an acknowledgement have been received over the interface, the timer T1 is cancelled. Subsequent IIHs sent by the starting router have the RR and RA bits clear and the SA bit set in the restart TLV.

Timer T1 is cancelled after some predetermined number of expirations (which MAY be 1).

When the T2 timer(s) are cancelled or expire, transmission of "normal" IIHs (with RR, RA, and SA bits clear) will begin.

2.3.3. Multiple Levels

A router that is operating as both a Level 1 and a Level 2 router on a particular interface MUST perform the above operations for each level.

On a LAN interface, it MUST send and receive both Level 1 and Level 2 IIHs and perform the CSNP synchronizations independently for each level.

On a Point-to-Point interface, only a single IIH (indicating support for both levels) is required, but it MUST perform the CSNP synchronizations independently for each level.

2.4. Database Synchronization

When a router is started or restarted, it can expect to receive a complete set of CSNPs over each interface. The arrival of the CSNP(s) is now guaranteed, since an IIH with the RR bit set will be retransmitted until the CSNP(s) are correctly received.

The CSNPs describe the set of LSPs that are currently held by each neighbor. Synchronization will be complete when all these LSPs have been received.

When (re)starting, a router starts an instance of timer T2 for each LSPDB as described in Section 3.3.1 or Section 3.3.2. In addition to normal processing of the CSNPs, the set of LSPIDs contained in the first complete set of CSNPs received over each interface is recorded, together with their remaining lifetime. In the case of a LAN interface, a complete set of CSNPs MUST consist of CSNPs received from neighbors that are not restarting. If there are multiple interfaces on the (re)starting router, the recorded set of LSPIDs is the union of those received over each interface. LSPs with a remaining lifetime of zero are NOT so recorded.

As LSPs are received (by the normal operation of the update process) over any interface, the corresponding LSPID entry is removed (it is also removed if an LSP arrives before the CSNP containing the reference). When an LSPID has been held in the list for its indicated remaining lifetime, it is removed from the list. When the list of LSPIDs is empty and the timer T1 has been cancelled for all the interfaces that have an adjacency at this level, the timer T2 is cancelled.

At this point, the local database is guaranteed to contain all the LSP(s) (either the same sequence number or a more recent sequence number) that were present in the neighbors' databases at the time of (re)starting. LSPs that arrived in a neighbor's database after the time of (re)starting may or may not be present, but the normal operation of the update process will guarantee that they will eventually be received. At this point, the local database is deemed to be "synchronized".

Since LSPs mentioned in the CSNP(s) with a zero remaining lifetime are not recorded, and those with a short remaining lifetime are deleted from the list when the lifetime expires, cancellation of the timer T2 will not be prevented by waiting for an LSP that will never arrive.

2.4.1. LSP Generation and Flooding and SPF Computation

The operation of a router starting, as opposed to restarting, is somewhat different. These two cases are dealt with separately below.

2.4.1.1. Restarting

In order to avoid causing unnecessary routing churn in other routers, it is highly desirable that the router's own LSPs generated by the restarting system are the same as those previously present in the network (assuming no other changes have taken place). It is important therefore not to regenerate and flood the LSPs until all the adjacencies have been re-established and any information required for propagation into the local LSPs is fully available. Ideally, the information is loaded into the LSPs in a deterministic way, such that the same information occurs in the same place in the same LSP (and hence the LSPs are identical to their previous versions). If this can be achieved, the new versions may not even cause SPF to be run in other systems. However, provided the same information is included in the set of LSPs (albeit in a different order, and possibly different LSPs), the result of running the SPF will be the same and will not cause churn to the forwarding tables.

In the case of a restarting router, none of the router's own LSPs are transmitted, nor are the router's own forwarding tables updated while the timer T3 is running.

Redistribution of inter-level information MUST be regenerated before this router's LSP is flooded to other nodes. Therefore, the Level-n non-pseudonode LSP(s) MUST NOT be flooded until the other level's T2 timer has expired and its SPF has been run. This ensures that any inter-level information that is to be propagated can be included in the Level-n LSP(s).

During this period, if one of the router's own (including pseudonodes) LSPs is received, which the local router does not currently have in its own database, it is NOT purged. Under normal operation, such an LSP would be purged, since the LSP clearly should not be present in the global LSP database. However, in the present circumstances, this would be highly undesirable, because it could cause premature removal of a router's own LSP -- and hence churn in remote routers. Even if the local system has one or more of the router's own LSPs (which it has generated, but not yet transmitted), it is still not valid to compare the received LSP against this set, since it may be that as a result of propagation between Level 1 and Level 2 (or vice versa), a further router's own LSP will need to be generated when the LSP databases have synchronized.

During this period, a restarting router SHOULD send CSNPs as it normally would. Information about the router's own LSPs MAY be included, but if it is included it MUST be based on LSPs that have been received, not on versions that have been generated (but not yet transmitted). This restriction is necessary to prevent premature removal of an LSP from the global LSP database.

When the timer T2 expires or is cancelled indicating that synchronization for that level is complete, the SPF for that level is run in order to derive any information that is required to be propagated to another level, but the forwarding tables are not yet updated.

Once the other level's SPF has run and any inter-level propagation has been resolved, the router's own LSPs can be generated and flooded. Any own LSPs that were previously ignored, but that are not part of the current set of own LSPs (including pseudonodes), MUST then be purged. Note that it is possible that a Designated Router change may have taken place, and consequently the router SHOULD purge those pseudonode LSPs that it previously owned, but that are now no longer part of its set of pseudonode LSPs.

When all the T2 timers have expired or been cancelled, the timer T3 is cancelled and the local forwarding tables are updated.

If the timer T3 expires before all the T2 timers have expired or been cancelled, this indicates that the synchronization process is taking longer than the minimum holding time of the neighbors. The router's own LSP(s) for levels that have not yet completed their first SPF computation are then flooded with the overload bit set to indicate that the router's LSPDB is not yet synchronized (and therefore other routers MUST NOT compute routes through this router). Normal operation of the update process resumes, and the local forwarding tables are updated. In order to prevent the neighbor's adjacencies from expiring, IIHs with the normal interface value for the holding time are transmitted over all interfaces with neither RR nor RA set in the restart TLV. This will cause the neighbors to refresh their adjacencies. The router's own LSP(s) will continue to have the overload bit set until timer T2 has expired or been cancelled.

2.4.1.2. Starting

In the case of a starting router, as soon as each adjacency is established, and before any CSNP exchanges, the router's own zeroth LSP is transmitted with the overload bit set. This prevents other routers from computing routes through the router until it has reliably acquired the complete set of LSPs. The overload bit remains set in subsequent transmissions of the zeroth LSP (such as will occur if a previous copy of the router's own zeroth LSP is still present in the network) while any timer T2 is running.

When all the T2 timers have been cancelled, the router's own LSP(s) MAY be regenerated with the overload bit clear (assuming the router is not in fact overloaded, and there is no other reason, such as incomplete BGP convergence, to keep the overload bit set) and flooded as normal.

Other LSPs owned by this router (including pseudonodes) are generated and flooded as normal, irrespective of the timer T2. The SPF is also run as normal and the Routing Information Base (RIB) and Forwarding Information Base (FIB) updated as routes become available.

To avoid the possible formation of temporary blackholes, the starting router sets the SA bit in the restart TLV (as described in Section 3.3.2) in all IIHs that it sends.

When all T2 timers have been cancelled, the starting router MUST transmit IIHs with the SA bit clear.

3. State Tables

This section presents state tables that summarize the behaviors described in this document. Other behaviors, in particular adjacency state transitions and LSP database update operation, are NOT included in the state tables except where this document modifies the behaviors described in [ISO10589] and [RFC5303].

The states named in the columns of the tables below are a mixture of states that are specific to a single adjacency (ADJ suppressed, ADJ Seen RA, ADJ Seen CSNP) and states that are indicative of the state of the protocol instance (Running, Restarting, Starting, SPF Wait).

Three state tables are presented from the point of view of a running router, a restarting router, and a starting router.

3.1. Running Router

Event	Running	ADJ suppressed
RX PR	Set Planned Restart state. Send PA	
RX PR clr and RX RR clr	Clear Planned Restart State	
RX RR	Maintain ADJ State Send RA Set SRM,send CSNP (Note 1) Update Hold Time, set Restart Mode (Note 2)	
RX RR clr	Clr Restart mode	
RX SA	Suppress IS neighbor TLV in LSP(s) Goto ADJ Suppressed	
RX SA clr		Unsuppress IS neighbor TLV in LSP(s) Goto Running

Note 1: CSNPs are sent by routers in accordance with Section 2.2.1c

Note 2: If Restart Mode clear

3.2. Restarting Router

Event	Restarting	ADJ Seen RA	ADJ Seen CSNP	SPF Wait
Restart planned	Send PR			
Planned restart canceled	Send PR clr			
Router	Send IIH/RR			

restarts	ADJ Init Start T1,T2,T3			
RX RR	Send RA			
RX RA	Adjust T3 Goto ADJ Seen RA		Cancel T1 Adjust T3	
RX CSNP set	Goto ADJ Seen CSNP	Cancel T1		
RX IIH w/o Restart TLV	Cancel T1 (Point-to-point only)			
T1 expires	Send IIH/RR Restart T1	Send IIH/RR Restart T1	Send IIH/RR Restart T1	
T1 expires nth time	Send IIH/ normal	Send IIH/ normal	Send IIH/ normal	
T2 expires	Trigger SPF Goto SPF Wait			
T3 expires	Set overload bit Flood local LSPs Update fwd plane			
LSP DB Sync	Cancel T2, and T3 Trigger SPF Goto SPF wait			
All SPF done				Clear overload bit Update fwd plane Flood local LSPs Goto Running

3.3. Starting Router

Event	Starting	ADJ Seen RA	ADJ Seen CSNP
Router starts	Send IIH/SA Start T1,T2		
RX RR	Send RA		
RX RA	Goto ADJ Seen RA		Cancel T1
RX CSNP Set	Goto ADJ Seen CSNP	Cancel T1	
RX IIH w no Restart TLV	Cancel T1 (Point-to-Point only)		
ADJ UP	Start T1 Send local LSPs with overload bit set		
T1 expires	Send IIH/RR and SA Restart T1	Send IIH/RR and SA Restart T1	Send IIH/RR and SA Restart T1
T1 expires nth time	Send IIH/SA	Send IIH/SA	Send IIH/SA
T2 expires	Clear overload bit Send IIH normal Goto Running		
LSP DB Sync	Cancel T2 Clear overload bit Send IIH normal		

4. IANA Considerations

This document defines the following IS-IS TLV that is listed in the IS-IS TLV codepoint registry:

Type	Description	IIH	LSP	SNP
211	Restart TLV	y	n	n

5. Security Considerations

Any new security issues raised by the procedures in this document depend upon the ability of an attacker to inject a false but apparently valid IIH, the ease/difficulty of which has not been altered.

If the RR bit is set in a false IIH, neighbors who receive such an IIH will continue to maintain an existing adjacency in the "UP" state and may (re)send a complete set of CSNPs. While the latter action is wasteful, neither action causes any disruption in correct protocol operation.

If the RA bit is set in a false IIH, a (re)starting router that receives such an IIH may falsely believe that there is a neighbor on the corresponding interface that supports the procedures described in this document. In the absence of receipt of a complete set of CSNPs on that interface, this could delay the completion of (re)start procedures by requiring the timer T1 to time out the locally defined maximum number of retries. This behavior is the same as would occur on a LAN where none of the (re)starting router's neighbors support the procedures in this document and is covered in Sections 2.3.1 and 2.3.2.

If an SA bit is set in a false IIH, this could cause suppression of the advertisement of an IS neighbor, which could either continue for an indefinite period or occur intermittently with the result being a possible loss of reachability to some destinations in the network and/or increased frequency of LSP flooding and SPF calculation.

The possibility of IS-IS PDU spoofing can be reduced by the use of authentication as described in [RFC1195] and [ISO10589], and especially the use of cryptographic authentication as described in [RFC5304] and [RFC5310].

6. Manageability Considerations

These extensions that have been designed, developed, and deployed for many years do not have any new impact on management and operation of the IS-IS protocol via this standardization process.

7. Acknowledgements

For RFC 5306 the authors acknowledged contributions made by Jeff Parker, Radia Perlman, Mark Schaefer, Naiming Shen, Nischal Sheth, Russ White, and Rena Yang.

The authors of this updated version acknowledge the contribution of Mike Shand, co-auther of RFC 5306.

8. Normative References

[ISO10589]
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.

[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>.

[RFC5303] Katz, D., Saluja, R., and D. Eastlake 3rd, "Three-Way Handshake for IS-IS Point-to-Point Adjacencies", RFC 5303, DOI 10.17487/RFC5303, October 2008, <https://www.rfc-editor.org/info/rfc5303>.

[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>.

[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>.

[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>.

[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>.

Appendix A. Summary of Changes from RFC 5306

This document extends RFC 5306 by introducing support for signalling the neighbors of a restarting router that a planned restart is about to occur. This allows the neighbors to be aware of the state of the restarting router so that appropriate action may be taken if other topology changes occur while the planned restart is in progress. Since the forwarding plane of the restarting router is maintained based upon the pre-restart state of the network, additional topology changes introduce the possibility that traffic may be lost if paths via the restarting router continue to be used while the restart is in progress.

In support of this new functionality two new flags have been introduced:

PR - Restart is planned
PA - Planned restart acknowledgement

No changes to the post restart exchange between the restarting router and its neighbors have been introduced.

Authors' Addresses

Les Ginsberg
Cisco Systems, Inc.

Email: ginsberg@cisco.com

Paul Wells
Cisco Systems, Inc.

Email: pauwells@cisco.com

Network Working Group P. Psenak, Ed.
Internet-Draft Cisco Systems
Intended status: Standards Track S. Hegde, Ed.
Expires: August 20, 2018 Juniper Networks, Inc.
C. Filsfils
Cisco Systems, Inc.
A. Gulko
Thomson Reuters
February 16, 2018

ISIS Segment Routing Flexible Algorithm
draft-hegdeppsenak-isis-sr-flex-algo-02.txt

Abstract

IGP protocols traditionally compute best paths over the network based on the IGP metric assigned to the links. Many network deployments use RSVP-TE based or Segment Routing based Traffic Engineering to enforce traffic over a path that is computed using different metrics or constraints than the shortest IGP path. Various mechanisms are used to steer the traffic towards such traffic engineered paths. This document proposes a solution that allows IGPs themselves to compute constraint based paths over the network without the use of the above mentioned traffic engineering technologies.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

IGP computed path based on the shortest IGP metric must often be replaced by traffic engineered path due to the traffic requirements which are not reflected in the IGP metric. Some networks engineer the IGP metric assignments in a way that the IGP Metric reflects the link bandwidth or delay. If, for example, the IGP metric is reflecting the bandwidth on the link and the application traffic is delay sensitive, the best IGP path may not reflect the best path from such application's perspective.

To overcome such IGP limitation, various sorts of traffic engineering has been deployed, including RSVP-TE or SR-TE, in which case the TE component is responsible for computing the path based on additional metrics and/or constraints. Such paths need to be installed in the forwarding and replace the original paths computed by IGPs. Tunnels are often used to represent the engineered paths and mechanisms like one described in [RFC3906] are used to replace the native IGP paths with such tunnel paths.

Segment Routing (SR) allows a flexible definition of end-to-end paths within IGP topologies by encoding paths as sequences of topological sub-paths, called segments. It also defines an algorithm that defines how the paths are computed. It also provides a way to associate Prefix-SID with an algorithm. This allows IGPs to compute paths based on various algorithms and cause traffic to be forwarded on such paths using the algorithm specific segments.

This document describes the IS-IS extension to support Segment Routing Flexible Algorithm on an MPLS data-plane.

1.1. Requirements notation

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].

2. Flexible Algorithm

Many possible constraints may be used to compute a path over a network. Some networks are deployed as multiple planes. A simple form of constraint may be to use a particular plane. A more sophisticated form of constraint can include some extended metric as described in [RFC7810]. Constraints which restrict paths to links with specific affinities or avoid links with specific affinities are also possible. Combinations of these are also possible.

To provide maximum flexibility we do not want to provide a strict mapping between the set of constraints and the algorithm that is associated with it. We want the mapping between the algorithm value and it’s meaning to be flexible and defined by the user. As far as all routers in the domain have the common understanding what the particular algorithm value represents, the computation for such algorithm is consistent and traffic is not subject to any looping.

Because the meaning of the algorithm is not defined by any standard, but is defined by the user, we call it Flex-Algorithm.

3. Flexible Algorithm Advertisement

[I-D.ietf-isis-segment-routing-extensions] defines an SR-Algorithm. This algorithm defines how the best path is computed by the IGP. Routers advertise the support for the algorithm as a node capability.

Prefix SIDs are also advertised with an algorithm value and as such are tightly coupled with the algorithm.

Existing advertisement of the SR-Algorithm is used for the Flex-Algorithm advertisements as defined in [I-D.ietf-isis-segment-routing-extensions].

SR-Algorithm is a one octet value. We propose to split the range of values as follows:

0-127 - standardised values assigned by IANA

128-255 - user defined values.

4. Flexible Algorithm Definition Advertisement

To guarantee the loop free forwarding for paths computed for a particular Flex-Algorithm, all routers in the flooding scope of the algorithm definition MUST agree on the definition of the Flex-Algorithm.

4.1. Flexible Algorithm Definition Sub-TLV

Flexible Algorithm Definition Sub-TLV (FAD Sub-TLV) is used to advertise the definition of the Flex-Algorithm.

FAD Sub-TLV is advertised as Sub-TLV of the IS-IS Router Capability TLV-242 that is defined in [RFC7981].

When the definition of the Flex Algorithm is advertised, it is applicable to all topologies supported on the receiving node.

FAD Sub-TLV has the following format:

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |      Type     |    Length     |   Algorithm   |  Metric Type  |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |  Alg. Type    |    Priority   |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                          Sub-TLVs                             |
    +                                                               +
    |                            ...                                |

    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

Type: TBD1

Length: variable, dependent on the included Sub-TLVs

Algorithm: Flex-Algorithm number. Value between 128 and 255 inclusive.

Metric Type: Type of metric to be used during the calculation. Following values are defined:

0: IGP Metric

1: Min Unidirectional Link Delay as defined in [RFC7810].

2: TE default metric as defined in [RFC5305].

Algorithm Type: Single octet identifying the algorithm type used to compute paths for the Flex-Algoritm. Values are defined in "IGP Algorithm Types" registry defined under "Interior Gateway Protocol (IGP) Parameters" IANA registries.

Priority: Single octet that specifies the priority of the advertisement.

Sub-TLVs - optional sub-TLVs.

When the router is configured with the local definition of the Flex-Algorithm, the router MUST advertise its local definition in the FAD Sub-TLV. If the local definition of the Flex-Algorithm is not advertised, the inconsistency in the configuration of the Flex-Algorithm on various nodes cannot be detected and traffic routed based on a Flex-Algorithm path may loop permanently.

Every router, that is configured to support a particular Flex-Algorithm, MUST select the Flex-Algorithm definition based on the following rules:

From the received advertisements of the FAD, select the one(s) with the highest priority.

If there are multiple advertisements of the FAD with the same highest priority, select the one that is originated from the router with the highest Router ID. Router ID is required to be advertised in every Router Capability TLV [RFC7981].

If the router has a local definition of the Flex-Algorithm, compare it with the received FAD advertisements using the same rules as have been used to pick the best FAD advertisement, e.g., priority and Router ID.

A router that is not configured to support a particular Flex-Algorithm MUST ignore FAD Sub-TLVs advertisements for such Flex-Algorithm.

Having a deterministic way that always produces a valid Flex-Algorithm definition avoids conflicts and maximizes the availability of the forwarding for the traffic that is using the Flex-Algorithm paths.

Any change in the Flex-Algorithm definition may result in temporary disruption of traffic that is forwarded based on such Flex-Algorithm paths. The impact is similar to any other event that requires network wide convergence

The FAD Sub-TLV of the IS-IS Router Capability TLV-242 MUST be propagated throughout the level. It MAY be advertised across level boundaries, if the S-flag in the Router Capability TLV is set. The S-Flag SHOULD not be set by default unless local configuration policy on the originating router indicates domain wide flooding.

Flex-Algorithm definition is topology independent. A node which advertises support for a given Flex-Algorithm may support that Flex-Algorithm on any subset of the topologies it supports. Enabling of a supported Flex-Algorithm on a given topology is a matter of local configuration. For a given topology, if out of the set of nodes supporting that topology AND advertising support for a given Flex-Algorithm only a subset of the nodes actually compute/install Flex-Algorithm specific paths in the forwarding plane for that topology, some traffic intended for such topology/Flex-Algorithm could be dropped if forwarded to a node on which the Flex-Algorithm is not enabled on that topology.

4.2. Flexible Algorithm Exclude Admin Group Sub-TLV

The Flexible-Algorithm definition can specify 'colors' that are used by the operator to exclude links during the Flex-Algorithm path computation.

Flexible Algorithm Exclude Admin Group Sub-TLV (FAEAG Sub-TLV) is a Sub-TLV of the FAD Sub-TLV. It has the following format:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |      Type       |    Length     |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                      Extended Admin Group                     |
   +-                                                             -+
   |                            ...                                |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

Type: 1

Length: variable, dependent on the size of the Extended Admin Group. MUST be a multiple of 4 octets.

Extended Administrative Group: Extended Administrative Group as defined in [RFC7308].

FAEAG Sub-TLV SHOULD only appear once in FAD Sub-TLV. If it appears more then once, FAD Sub-TLV MUST be ignored by the receiver.

4.3. Flexible Algorithm Include Admin Group Sub-TLVs

The Flexible-Algorithm definition can specify 'colors' that are used by the operator to include links during the Flex-Algorithm path computation.

The format of the include Sub-TLVs is identical to the format of the FAEAG Sub-TLV in Section 4.2.

Two forms of inclusion are available - include-any and include-all.

Flexible Algorithm Include-Any Admin Group Sub-TLV - Type 2.

Flexible Algorithm Include-All Admin Group Sub-TLV - Type 3.

Flexible Algorithm Include Admin Group Sub-TLVs SHOULD only appear once in FAD Sub-TLV. If any of these Sub-TLVs appear more then once, FAD Sub-TLV MUST be ignored by the receiver.

5. Calculation of Flexible Algorithm Paths

A router may compute path for multiple Flex-Algorithms.

A router MUST be configured to support Flex-Algorithm K before it can compute any path for Flex-Algorithm K.

A router MUST either be configured with a local definition of Flex-Algorithm K or receive the definition via the FAD Sub-TLV, as described in Section 4.1, before it can compute any path for Flex-Algorithm K.

When computing the path for Flex-Algorithm K, all nodes that do not advertise support for Flex-Algorithm K in SR-Algorithm Sub-TLV ([I-D.ietf-isis-segment-routing-extensions]), MUST be pruned from the topology.

When computing the path for Flex-Algorithm K, the metric that is part of the Flex-Algorithm definition (Section 4.1) MUST be used.

Various link include or exclude rules can be part of the Flex-Algorithm definition. These rules use Extended Administrative Groups (EAG) as defined in [RFC7308]. [RFC7308] uses term 'colors' as a shorthand to refer to particular bits with an EAG. Link advertisement CAN also include EAG, which describe which color is set on the link.

Link advertisement CAN also include Administrative Group (AG) TLV ([RFC5305]). The coexistence of EAG and AG is described in the section 2.3.1 of [RFC7308].

Rules, in the order as specified below, MUST be used to prune link from the topology during the Flex-Algorithm computation.

For all links in the topology:

1. Check if any exclude rule is part of the Flex-Algorithm definition. If such exclude rule exists, check if any color that is part of the exclude rule is also set on the link. If such a color exist, the link MUST be pruned from the computation.

2. Check if any include-any rule is part of the Flex-Algorithm definition. if such include-any rule exists, check if any color that is part of the include-any rule is also set on the link. If

such color does not exist, the link MUST be pruned from the computation.

3. Check if any include-all rule is part of the Flex-Algorithm definition. If such include-all rule exists, check if all colors that are part of the include-all rule are also set on the link. If not all such colors are set on the link, the link MUST be pruned from the computation.

4. If the Flex-Algorithm definition uses other than IGP metric (Section 4.1), and such metric is not advertised for the particular link in a topology for which the computation is done, such link MUST be pruned from the computation. A metric of value 0 MUST NOT be assumed in such case.

Flex-Algorithm K path MUST be installed in the MPLS forwarding plane using the MPLS label that corresponds to the Prefix-SID that was advertised for algorithm K. If the Prefix SID for algorithm K is not known, the Flex-Algorithm K path to such prefix MUST NOT be installed in the MPLS forwarding plane.

Loop Free Alternate (LFA) paths for Flex-Algorithm K path MUST be computed using the same constraints as the calculation of the primary paths for Flex-Algorithm K. LFA path MUST only use Prefix-SIDs advertised specifically for algorithm K to enforce the traffic over such path. LFA path MUST NOT use Adjacency-SID that belong to the link that has been pruned from the computation.

If LFA protection is being used to protect Flex-Algorithm K paths, all routers in the area SHOULD advertise at least one Flex-Algorithm K specific Prefix-SID. These Prefix-SIDs are used to enforce traffic over the LFA computed backup path.

Flex-Algorithm paths MAY be used by other applications, that do not utilize MPLS forwarding plane. It is outside of the sope of this specification, how these application learn and use the Flex-Algorithm specific paths.

Any Shortest Path Tree calculation is limited to a single area. Same applies to Flex-Algorithm calculations. Given that the computing router may not have the visibility to the topology of remote areas, the Flex-Algorithm K path to an inter-area prefix will only be computed for the local area. The egress L1/L2 router will be selected based on the best path for the Flex-Algorithm K in the local area and such egress L1/L2 router will be responsible to compute the best Flex-Algorithm K path over the next area. This may produce end-to-end path, which is not the best from the Flex-Algorithm K perspective. If the best end-to-end path for Flex-Algorithm K needs to be used for inter-area destinations, paths for such destinations need to be computed by the entity that has the topological information about all areas.

6. Backward Compatibility

This extension brings no new backward compatibility issues.

7. Security Considerations

This extension adds no new security considerations.

8. IANA Considerations

This documents request allocation for the following ISIS TLVs and subTLVs.

8.1. Sub TLVs for Type 242

This document makes the following registrations in the "sub-TLVs for TLV 242" registry.

Type: TBD1 (suggested value 24).

Description: Flexible Algorithm Definition Sub-TLV.

Reference: This document (Section 4.1).

8.2. New Sub-Sub-TLV registry

This document creates the following Sub-TLV Registry:

Registry: Sub-TLVs for Flexible Algorithm Definition Sub-TLV

Registration Procedure: Expert review

Reference: This document (Section 4.1)

This document resisters following Sub-TLVs in the "Sub-TLVs for Flexible Algorithm Definition Sub-TLV" registry:

Type: 1

Description: Flexible Algorithm Exclude Admin Group Sub-TLV

Reference: This document (Section 4.2).

Type: 2

Description: Flexible Algorithm Include-Any Admin Group Sub-TLV

Reference: This document (Section 4.3).

Type: 3

Description: Flexible Algorithm Include-All Admin Group Sub-TLV

Reference: This document (Section 4.3).

8.2.1. Flexible Algorithm Definition TLV Metric Registry

This document creates the following Registry:

Registry: Flexible Algorithm Definition TLV Metric Registry

Registration Procedure: Expert review

Reference: This document (Section 4.1)

This document registers following values in the "Flexible Algorithm Definition TLV Metric Registry":

Type: TBD, suggested value 0

Description: IGP metric

Reference: This document (Section 4.1)

Type: TBD, suggested value 1

Description: Min Unidirectional Link Delay [RFC7810]

Reference: This document (Section 4.1)

Type: TBD, suggested value 2

Description: TE Default Metric [RFC5305]

Reference: This document (Section 4.1)

9. Acknowledgments

This draft, among other things, is also addressing the problem that the [I-D.gulkohegde-routing-planes-using-sr] was trying to solve. All authors of that draft agreed to join this draft.

Thanks to Les Ginsberg and Ketan Talaulikar for review and useful comments.

Thanks to Cengiz Halit for his review and feedback during initial phase of the solution definition.

10. References

10.1. 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.

[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, DOI 10.17487/RFC5305, October 2008, <https://www.rfc-editor.org/info/rfc5305>.

[RFC7308] Osborne, E., "Extended Administrative Groups in MPLS Traffic Engineering (MPLS-TE)", RFC 7308, DOI 10.17487/RFC7308, July 2014, <https://www.rfc-editor.org/info/rfc7308>.

[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>.

[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>.

10.2. Informative References

[I-D.gulkohegde-routing-planes-using-sr]
Hegde, S. and a. arkadiy.gulko@thomsonreuters.com, "Separating Routing Planes using Segment Routing", draft-gulkohegde-routing-planes-using-sr-00 (work in progress), March 2017.

[RFC3906] Shen, N. and H. Smit, "Calculating Interior Gateway Protocol (IGP) Routes Over Traffic Engineering Tunnels", RFC 3906, DOI 10.17487/RFC3906, October 2004, <https://www.rfc-editor.org/info/rfc3906>.

Authors' Addresses

Peter Psenak (editor)
Cisco Systems
Apollo Business Center
Mlynske nivy 43
Bratislava, 82109
Slovakia

Email: ppsenak@cisco.com

Shraddha Hegde (editor)
Juniper Networks, Inc.
Embassy Business Park
Bangalore, KA, 560093
India

Email: shraddha@juniper.net

Clarence Filsfils
Cisco Systems, Inc.
Brussels
Belgium

Email: cfilsfil@cisco.com

Arkadiy Gulko
Thomson Reuters

Email: arkadiy.gulko@thomsonreuters.com

Networking Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 31, 2020

L. Ginsberg
P. Psenak
Cisco Systems
S. Previdi
Huawei
W. Henderickx
Nokia
J. Drake
Juniper Networks
June 29, 2020

IS-IS Application-Specific Link Attributes
draft-ietf-isis-te-app-19

Abstract

Existing traffic engineering related link attribute advertisements have been defined and are used in RSVP-TE deployments. Since the original RSVP-TE use case was defined, additional applications (e.g., Segment Routing Policy, Loop Free Alternate) that also make use of the link attribute advertisements have been defined . 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 document introduces new link attribute advertisements that address both of these shortcomings.

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

Copyright Notice

Table of Contents

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], [RFC7308], and [RFC8570]. Use of these extensions has been associated with deployments supporting Traffic Engineering over Multiprotocol Label Switching (MPLS) in the presence of the Resource Reservation Protocol (RSVP) - more succinctly referred to as RSVP-TE [RFC3209].

For the purposes of this document an application is a technology that makes use of link attribute advertisements - examples of which are listed in Section 3.

In recent years new applications that have use cases for many of the link attributes historically used by RSVP-TE have been introduced. Such applications include Segment Routing Policy (SR Policy) [I-D.ietf-spring-segment-routing-policy] and Loop Free Alternates (LFA) [RFC5286]. This has introduced ambiguity in that if a deployment includes a mix of RSVP-TE support and SR Policy 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 SR Policy. If the topologies are fully congruent this may not be an issue, but any incongruence leads to ambiguity.

An example where this ambiguity causes a problem is a network where RSVP-TE is enabled only on a subset of its links. A link attribute is advertised for the purpose of another application (e.g. SR Policy) for a link that is not enabled for RSVP-TE. As soon as the router that is an RSVP-TE head-end sees the link attribute being advertised for that link, it assumes RSVP-TE is enabled on that link, even though it is not. If such RSVP-TE head-end router tries to setup an RSVP-TE path via that link, it will result in a path setup failure.

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 that 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 that 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. Therefore, any discussion of existing use cases is limited to requirements that 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 that 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 Segment Routing (SR). Included among these use cases is SR Policy which is defined in [I-D.ietf-spring-segment-routing-policy]. If both RSVP-TE and SR Policy 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 SR Policy 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 that 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, 25, 141, 222, and 223 and TLVs for Shared Risk Link Group (SRLG) advertisement.

Sub-TLV values are defined in the Sub-TLVs for TLVs 22, 23, 25, 141, 222, and 223 registry.

TLVs are defined in the TLV Codepoints Registry.

3.1. Legacy sub-TLVs

Sub-TLVs for TLVs 22, 23, 25, 141, 222, and 223

Type	Description
3	Administrative group (color)
9	Maximum link bandwidth
10	Maximum reservable link bandwidth
11	Unreserved bandwidth
14	Extended Administrative Group
18	TE Default Metric
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 (ASLA) Advertisements:

1) ASLA sub-TLV for TLVs 22, 23, 25, 141, 222, and 223 (defined in Section 4.2).

2)Application-Specific Shared Risk Link Group (SRLG) TLV (defined in Section 4.3).

In support of these new advertisements, an application identifier bit mask is defined that identifies the application(s) associated with a given advertisement (defined in Section 4.1).

In addition to supporting the advertisement of link attributes used by standardized applications, link attributes can also be advertised for use by user defined applications. Such applications are not subject to standardization and are outside the scope of this document.

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.

```
 0  1  2  3  4  5  6  7
+--+--+--+--+--+--+--+--+
| SABM Length + Flag    |  1 octet
+--+--+--+--+--+--+--+--+
| UDABM Length + Flag   |  1 octet
+--+--+--+--+--+--+--+--+
|   SABM         ...       0 - 8 octets
+--+--+--+--+--+--+--+--+
|   UDABM        ...       0 - 8 octets
+--+--+--+--+--+--+--+--+
```

SABM Length + Flag (1 octet)
    Standard Application Identifier Bit Mask
     Length + Flag

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

L-flag: Legacy Flag.
See Section 4.2 for a description of how
this flag is used.

SABM Length: Indicates the length in octets (0-8) of the
Standard Application Identifier Bit Mask. The length SHOULD
be the minimum required to send all bits that are set.

UDABM Length + Flag (1 octet)
User Defined Application Identifier Bit Mask
Length + Flag

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

R: Reserved. SHOULD be transmitted as 0 and
MUST be ignored on receipt

UDABM Length: Indicates the length in octets (0-8) of the
User Defined Application Identifier Bit Mask. The length SHOULD
be the minimum required to send all bits that are set.

SABM (variable length)
Standard Application Identifier Bit Mask

(SABM Length * 8) bits

This field is omitted if SABM Length is 0.

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

R-bit: Set to specify RSVP-TE

S-bit: Set to specify Segment Routing Policy

F-bit: Set to specify Loop Free Alternate (LFA)

(includes all LFA types)

UDABM (variable length)
User Defined Application Identifier Bit Mask

(UDABM Length * 8) bits

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

This field is omitted if UDABM Length is 0.

NOTE: SABM/UDABM Length is arbitrarily limited to 8 octets in order to insure that sufficient space is left to advertise link attributes without overrunning the maximum length of a sub-TLV.

Standard Application Identifier Bits are defined/sent starting with Bit 0.

User Defined Application Identifier Bits have no relationship to Standard Application Identifier 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.

In the case of both SABM and UDABM, the following rules apply:

- Undefined bits that are transmitted 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.

- Bits that are not supported by an implementation MUST be ignored on receipt.

.

4.2. Application-Specific Link Attributes sub-TLV

A new sub-TLV for TLVs 22, 23, 25, 141, 222, and 223 is defined that supports specification of the applications and application-specific attribute values.

```
      Type: 16 (temporarily assigned by IANA)
      Length: Variable (1 octet)
      Value:

        Application Identifier Bit Mask
        (as defined in Section 4.1)

        Link Attribute sub-sub-TLVs - format matches the
        existing formats defined in [RFC5305], [RFC7308],
        and [RFC8570]
```

If the SABM or UDABM length in the Application Identifier Bit Mask is greater than 8, the entire sub-TLV MUST be ignored.

When the L-flag is set in the Application Identifier Bit Mask, all of the applications specified in the bit mask MUST use the legacy advertisements for the corresponding link found in TLVs 22, 23, 25, 141, 222, and 223 or TLV 138 or TLV 139 as appropriate. 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 Identifier Bit Masks and all such advertisements MUST be ignored on receipt.

Multiple Application-Specific Link Attribute 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 for the same link attribute for a given link. In cases where conflicting values for the same application/attribute/link are advertised the first advertisement received in the lowest numbered LSP SHOULD be used and subsequent advertisements of the same attribute SHOULD 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.

If link attributes are advertised associated with zero length Application Identifier Bit Masks for both standard applications and user defined applications, then any Standard Application and/or any User Defined Application is permitted to use that set of link attributes so long as there is not another set of attributes advertised on that same link that is associated with a non-zero length Application Identifier Bit Mask with a matching Application Identifier Bit set.

A new registry of sub-sub-TLVs is to be created by IANA that defines the link attribute sub-sub-TLV code points. This document defines a sub-sub-TLV for each of the existing sub-TLVs listed in Section 3.1 except as noted below. The 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 identifier to the corresponding sub-sub-TLV.

4.2.1. Special Considerations for Maximum Link Bandwidth

Maximum link bandwidth is an application independent attribute of the link. When advertised using the Application-Specific Link Attributes sub-TLV, multiple values for the same link MUST NOT be advertised. This can be accomplished most efficiently by having a single advertisement for a given link where the Application Identifier Bit Mask identifies all the applications that are making use of the value for that link.

It is also possible to advertise the same value for a given link multiple times with disjoint sets of applications specified in the Application Identifier Bit Mask. This is less efficient but still valid.

It is also possible to advertise a single advertisement with zero length SABM and UDABM so long as the constraints discussed in Section 4.2 and Section 6.2 are acceptable.

If different values for Maximum Link Bandwidth for a given link are advertised, all values MUST be ignored.

4.2.2. Special Considerations for Reservable/Unreserved Bandwidth

Maximum Reservable Link Bandwidth and Unreserved Bandwidth are attributes specific to RSVP-TE. When advertised using the Application-Specific Link Attributes sub-TLV, bits other than the RSVP-TE (R-bit) MUST NOT be set in the Application Identifier Bit Mask. If an advertisement of Maximum Reservable Link Bandwidth or Unreserved Bandwidth is received with bits other than the RSVP-TE bit set, the advertisement MUST be ignored.

4.2.3. Considerations for Extended TE Metrics

[RFC8570] defines a number of dynamic performance metrics associated with a link. It is conceivable that such metrics could be measured specific to traffic associated with a specific application. Therefore this document includes support for advertising these link attributes specific to a given application. However, in practice it may well be more practical to have these metrics reflect the performance of all traffic on the link regardless of application. In such cases, advertisements for these attributes will be associated with all of the applications utilizing that link. This can be done either by explicitly specifying the applications in the Application Identifier Bit Mask or by using a zero length Application Identifier Bit Mask.

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 [RFC5307] and TLV 139 [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 (Temporarily assigned by IANA)
Length: Number of octets in the value field (1 octet)
Value:
  Neighbor System-ID + pseudo-node ID (7 octets)
  Application Identifier Bit Mask
   (as defined in Section 4.1)
  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 values chosen are intentionally matching the equivalent sub-TLVs from [RFC5305], [RFC5307], and [RFC6119].

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

At least one set of link identifiers (IPv4, IPv6, or Link Local/ Remote) MUST be present. Multiple occurrences of the same identifier type MUST NOT be present. TLVs that 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 Identifier Bit Mask, SRLG values MUST NOT be included in the TLV. Any SRLG values that 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 Identifier 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. Attribute Advertisements and Enablement

This document defines extensions to support the advertisement of application-specific link attributes.

Whether the presence of link attribute advertisements for a given application indicates that the application is enabled on that link depends upon the application. Similarly, whether the absence of link attribute advertisements indicates that the application is not enabled depends upon the application.

In the case of RSVP-TE, the advertisement of application-specific link attributes implies that RSVP is enabled on that link. The absence of RSVP-TE application-specific link attributes in combination with the absence of legacy advertisements implies that RSVP is not enabled on that link.

In the case of SR Policy, advertisement of application-specific link attributes does not indicate enablement of SR Policy on that link. The advertisements are only used to support constraints that may be applied when specifying an explicit path. SR Policy is implicitly enabled on all links that are part of the Segment Routing enabled topology independent of the existence of link attribute advertisements.

In the case of LFA, advertisement of application-specific link attributes does not indicate enablement of LFA on that link. Enablement is controlled by local configuration.

If, in the future, additional standard applications are defined to use this mechanism, the specification defining this use MUST define the relationship between application-specific link attribute advertisements and enablement for that application.

This document allows the advertisement of application-specific link attributes with no application identifiers i.e., both the Standard Application Identifier Bit Mask and the User Defined Application Identifier Bit Mask are not present (See Section 4.1). This supports the use of the link attribute by any application. In the presence of an application where the advertisement of link attribute advertisements is used to infer the enablement of an application on that link (e.g., RSVP-TE), the absence of the application identifier leaves ambiguous whether that application is enabled on such a link. This needs to be considered when making use of the "any application" encoding.

6. Deployment Considerations

This section discuss deployment considerations associated with the use of application-specific link attribute advertisements.

6.1. Use of Legacy Advertisements

Bit Identifiers for Standard Applications are defined in Section 4.1. All of the identifiers defined in this document are associated with applications that were already deployed in some networks prior to the writing of this document. Therefore, such applications have been deployed using the legacy advertisements. The Standard Applications defined in this document may continue to use legacy advertisements for a given link so long as at least one of the following conditions is true:

- The application is RSVP-TE

- The application is SR Policy or LFA and RSVP-TE is not deployed anywhere in the network

- The application is SR Policy or LFA, RSVP-TE is deployed in the network, and both the set of links on which SR Policy and/or LFA advertisements are required and the attribute values used by SR Policy and/or LFA on all such links is fully congruent with the links and attribute values used by RSVP-TE

Under the conditions defined above, implementations that support the extensions defined in this document have the choice of using legacy advertisements or application-specific advertisements in support of SR Policy and/or LFA. This will require implementations to provide controls specifying which type of advertisements are to be sent/ processed on receive for these applications. Further discussion of the associated issues can be found in Section 6.3.

New applications that future documents define to make use of the advertisements defined in this document MUST NOT make use of legacy advertisements. This simplifies deployment of new applications by eliminating the need to support multiple ways to advertise attributes for the new applications.

6.2. Use of Zero Length Application Identifier Bit Masks

Link attribute advertisements associated with zero length Application Identifier Bit Masks for both standard applications and user defined applications are usable by any application, subject to the restrictions specified in Section 4.2. If support for a new application is introduced on any node in a network in the presence of such advertisements, these advertisements are permitted to 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.

6.3. Interoperability, Backwards Compatibility and Migration Concerns

Existing deployments of RSVP-TE, SR Policy, and/or LFA utilize the legacy advertisements listed in Section 3. Routers that 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 in the presence of routers that do not support these extensions need to be able to interoperate with the use of legacy advertisements by the legacy routers. The following sub-sections discuss interoperability and backwards compatibility concerns for a number of deployment scenarios.

6.3.1. 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-flag set and no link attribute values. This avoids duplication of link attribute advertisements.

6.3.2. Multiple Applications: All Attributes Not Shared with 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 that have the L-flag clear. In cases where some link attributes are shared with RSVP-TE, this requires duplicate advertisements for those attributes.

These guidelines apply 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.

6.3.3. Interoperability with Legacy Routers

For the applications defined in this document, routers that do not support the extensions defined in this document will send and receive only legacy link attribute advertisements. So long as there is any legacy router in the network that has any of the applications enabled, all routers MUST continue to advertise link attributes using legacy advertisements. In addition, the link attribute values associated with the set of applications supported by legacy routers (RSVP-TE, SR Policy, and/or LFA) are always shared since legacy routers have no way of advertising or processing application-specific values. Once all legacy routers have been upgraded, migration from legacy advertisements to ASLA advertisements can be achieved via the following steps:

1)Send ASLA advertisements while continuing to advertise using legacy (all advertisements are then duplicated). Receiving routers continue to use legacy advertisements.

2)Enable the use of the ASLA advertisements on all routers

3)Remove legacy advertisements

When the migration is complete, it then becomes possible to advertise incongruent values per application on a given link.

Note that the use of the L-flag is of no value in the migration.

Documents defining new applications that make use of the application-specific advertisements defined in this document MUST discuss interoperability and backwards compatibility issues that could occur in the presence of routers that do not support the new application.

6.3.4. Use of Application-Specific Advertisements for RSVP-TE

The extensions defined in this document support RSVP-TE as one of the supported applications. This allows that RSVP-TE could eventually utilize the application-specific advertisements. This can be done in the following step-wise manner:

1)Upgrade all routers to support the extensions in this document

2)Advertise all legacy link attributes using ASLA advertisements with L-flag clear and R-bit set. At this point both legacy and application-specific advertisements are being sent.

3)Remove legacy advertisements

7. IANA Considerations

This section lists the protocol code point changes introduced by this document and the related IANA changes required.

For new registries defined under IS-IS TLV Codepoints Registry with registration procedure "Expert Review", guidance for designated experts can be found in [RFC7370].

7.1. Application-Specific Link Attributes sub-TLV

This document defines a new sub-TLV in the Sub-TLVs for TLVs 22, 23, 25, 141, 222, and 223 registry. See Section 4.2

Type	Description	22	23	25	141	222	223
16	Application-Specific Link Attributes	y	y	y(s)	y	y	y

7.2. Application-Specific SRLG TLV

This document defines one new TLV in the IS-IS TLV Codepoints Registry. See Section 4.3

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

7.3. Application-Specific Link Attributes sub-sub-TLV Registry

This document requests a new IANA registry under the IS-IS TLV Codepoints Registry be created to control the assignment of sub-sub-TLV codepoints for the Application-Specific Link Attributes sub-TLV defined in Section 7.1. The suggested name of the new registry is "sub-sub-TLV code points for application-specific link attributes". The registration procedure is "Expert Review" as defined in [RFC8126]. The following assignments are made by this document:

Type	Description	Encoding Reference
0-2	Unassigned	
3	Administrative group (color)	RFC5305
4-8	Unassigned	
9	Maximum link bandwidth	RFC5305
10	Maximum reservable link bandwidth	RFC5305
11	Unreserved bandwidth	RFC5305
12-13	Unassigned	
14	Extended Administrative Group	RFC7308
15-17	Unassigned	
18	TE Default Metric	RFC5305
19-32	Unassigned	
33	Unidirectional Link Delay	RFC8570
34	Min/Max Unidirectional Link Delay	RFC8570
35	Unidirectional Delay Variation	RFC8570
36	Unidirectional Link Loss	RFC8570
37	Unidirectional Residual Bandwidth	RFC8570
38	Unidirectional Available Bandwidth	RFC8570
39	Unidirectional Utilized Bandwidth	RFC8570
40-255	Unassigned	

Note to IANA: For future codepoints, in cases where the document that defines the encoding is different from the document that assigns the codepoint, the encoding reference MUST be to the document that defines the encoding.

Note to designated experts: If a link attribute can be advertised both as a sub-TLV of TLVs 22, 23, 25, 141, 222, and 223 and as a sub-sub-TLV of the Application-Specific Link Attributes sub-TLV defined in this document, then the same numerical code should be assigned to the link attribute whenever possible.

7.4. Link Attribute Application Identifier Registry

This document requests a new IANA registry be created, under the category of "Interior Gateway Protocol (IGP) Parameters", to control the assignment of Application Identifier Bits. The suggested name of the new registry is "Link Attribute Applications". The registration policy for this registry is "Expert Review" [RFC8126]. Bit definitions SHOULD be assigned such that all bits in the lowest available octet are allocated before assigning bits in the next octet. This minimizes the number of octets that will need to be transmitted. The following assignments are made by this document:

Bit #	Name
0	RSVP-TE (R-bit)
1	Segment Routing Policy (S-bit)
2	Loop Free Alternate (F-bit)
3-63	Unassigned

7.5. SRLG sub-TLVs

This document requests a new IANA registry be created under the IS-IS TLV Codepoints Registry 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 [RFC8126]. The following assignments are made by this document:

Value	Description	Encoding Reference
0-3	Unassigned	
4	Link Local/Remote Identifiers	[RFC5307]
5	Unassigned	
6	IPv4 interface address	[RFC5305]
7	Unassigned	
8	IPv4 neighbor address	[RFC5305]
9-11	Unassigned	
12	IPv6 Interface Address	[RFC6119]
13	IPv6 Neighbor Address	[RFC6119]
14-255	Unassigned	

Note to IANA: For future codepoints, in cases where the document that defines the encoding is different from the document that assigns the codepoint, the encoding reference MUST be to the document that defines the encoding.

8. Security Considerations

Security concerns for IS-IS are addressed in [ISO10589], [RFC5304], and [RFC5310]. While IS-IS is deployed under a single administrative domain, there can be deployments where potential attackers have access to one or more networks in the IS-IS routing domain. In these deployments, the stronger authentication mechanisms defined in the aforementioned documents SHOULD be used.

This document defines a new way to advertise link attributes. Tampering with the information defined in this document may have an effect on applications using it, including impacting Traffic Engineering as discussed in [RFC8570]. As the advertisements defined in this document limit the scope to specific applications, the impact of tampering is similarly limited in scope.

9. Acknowledgements

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

10. References

10.1. Normative References

[ISO10589]
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.

[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>.

[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>.

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

[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>.

[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>.

[RFC7308] Osborne, E., "Extended Administrative Groups in MPLS Traffic Engineering (MPLS-TE)", RFC 7308, DOI 10.17487/RFC7308, July 2014, <https://www.rfc-editor.org/info/rfc7308>.

[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>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

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

[RFC8570] Ginsberg, L., Ed., Previdi, S., Ed., Giacalone, S., Ward, D., Drake, J., and Q. Wu, "IS-IS Traffic Engineering (TE) Metric Extensions", RFC 8570, DOI 10.17487/RFC8570, March 2019, <https://www.rfc-editor.org/info/rfc8570>.

10.2. Informative References

[I-D.ietf-spring-segment-routing-policy]
Filsfils, C., Sivabalan, S., Voyer, D., Bogdanov, A., and P. Mattes, "Segment Routing Policy Architecture", draft-ietf-spring-segment-routing-policy-07 (work in progress), May 2020.

[RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, DOI 10.17487/RFC3209, December 2001, <https://www.rfc-editor.org/info/rfc3209>.

[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>.

[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, <https://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
Huawei

Email: stefano@previdi.net

Wim Henderickx
Nokia
Copernicuslaan 50
Antwerp 2018 94089
Belgium

Email: wim.henderickx@nokia.com

John Drake
Juniper Networks

Email: jdrake@juniper.net

IS-IS Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 17, 2020

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

October 15, 2019

YANG Data Model for IS-IS Protocol
draft-ietf-isis-yang-isis-cfg-42

Abstract

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

Copyright Notice

Table of Contents

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, the retrieval of IS-IS 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 the NMDA (Network Management Datastore Architecture) [RFC8342].

2. Design of the Data Model

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

The figure below describes the overall structure of the ietf-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 {osi-interface}?
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 prefixes* [address-family prefix]
     |        ...
     +--ro protection-statistics* [frr-protection-method]
     |  +--ro frr-protection-method    identityref
     |  +--ro address-family-stats* [address-family]
     |        ...
     +--rw discontinuity-time?       yang:date-and-time
     +--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 into:

- Global parameters.

- Per-interface configuration (see Section 2.4).

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

The model includes optional features, for which the corresponding configuration data nodes are also 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 shutdown a particular IS-IS instance.

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

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 the MT extensions.

The "name" used in the topology list should refer to an existing Routing Information Base (RIB) defined for the device [RFC8349].

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

Multiple address families (such as, IPv4 or IPv6) can also be enabled within the default topology. This can be achieved using the address-families container (requiring the "nlpid-control" feature to be supported).

2.3. Per-Level Parameters

Some parameters allow a per-level configuration. For such parameters, 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>
        </priority>
```

An implementation MUST prefer a level-specific parameter over a top-level parameter. For example, if the priority is 100 for the level-1 and 250 for the top-level configuration, the implementation must use 100 for the level-1 priority and 250 for the level-2 priority.

Some parameters, such as, "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 MUST augment the current model by adding both level-1 and level-2 containers and MUST 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 the previous section. An interface-specific parameter MUST be preferred over an IS-IS global parameter.

Some parameters, such as, 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 enable?                     boolean {admin-control}?
       +--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 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
          +--rw discontinuity-time?        yang:date-and-time
          +--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 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 functionality. The "mpls/ldp/igp-sync" leaf under "interface" allows activation of the functionality on a per-interface basis. The "mpls/ldp/igp-sync" container in the global configuration is intentionally empty and is not required for feature activation. The goal of this empty container is to facilitate augmentation with additional parameters, e.g., timers.

2.7. ISO parameters

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

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

The clns-mtu can be configured for an interface.

2.8. IP FRR

This YANG module supports LFA (Loop Free Alternates) [RFC5286] and remote LFA [RFC7490] as IP Fast Re-Route (FRR) techniques. The "fast-reroute" container may be augmented by other models to support other IP FRR flavors (MRT as defined in [RFC7812], TI-LFA as defined in [I-D.ietf-rtgwg-segment-routing-ti-lfa], etc.).

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

Remote LFA is considered as an extension of LFA. Remote LFA cannot be enabled if LFA is not enabled.

The "candidate-enable" data leaf designates that an interface can be used as a backup.

2.9. Operational States

Operational state is defined in module in various containers at various levels:

- o system-counters: Provides statistical information about the global system.

- o interface: Provides configuration state information for each interface.

- o adjacencies: Provides state information about current IS-IS adjacencies.

- spf-log: Provides information about SPF events for an IS-IS instance. This SHOULD be implemented as a wrapping buffer.

- lsp-log: Provides information about LSP events for an IS-IS instance (reception of an LSP or modification of a local LSP). This SHOULD be implemented as a wrapping buffer and the implementation MAY optionally log LSP refreshes.

- local-rib: Provides the IS-IS internal routing table.

- database: Provides contents of the current Link State Database.

- hostnames: Provides the system-id to hostname mappings [RFC5301].

- fast-reroute: Provides IP FRR state information.

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 all neighbors.

- clear-adjacency: Restart a particular set of IS-IS adjacencies.

4. Notifications

The "ietf-isis" module defines the following notifications:

database-overload: This notification is sent when the IS-IS Node overload condition changes.

lsp-too-large: This notification is sent when the system tries to propagate a PDU that is too large.

if-state-change: This notification is sent when an interface’s state changes.

corrupted-lsp-detected: This notification is sent when the IS-IS node discovers that an LSP that was previously stored in the Link State Database, i.e., local 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 with 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 with IS-IS-specific operational states.

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

The modules defined in this document uses 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], [RFC4090],[RFC5029], [RFC5130], [RFC5302], [RFC5305], [RFC5306], [RFC5307], [RFC5308], [RFC5880], [RFC5881], [RFC6119], [RFC6232], [RFC7794], [RFC7981], [RFC8570], [RFC7917], [RFC8405].

```
<CODE BEGINS> file "ietf-isis@2019-10-15.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 Web:   <https://datatracker.ietf.org/group/lsr/>
     WG List:  <mailto:lsr@ietf.org>

     Editor:   Stephane Litkowski
               <mailto:slitkows.ietf@gmail.com>
     Author:   Derek Yeung
               <mailto:derek@arrcus.com>
     Author:   Acee Lindem
               <mailto:acee@cisco.com>
     Author:   Jeffrey Zhang
               <mailto:zzhang@juniper.net>
     Author:   Ladislav Lhotka
               <mailto:llhotka@nic.cz>";

  description
   "This YANG module defines the generic configuration and
    operational state for the IS-IS protocol common to all
    vendor implementations. 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 model conforms to the Network Management
    Datastore Architecture (NMDA) 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
    (https://trustee.ietf.org/license-info).

    This version of this YANG module is part of RFC XXXX
    (https://www.rfc-editor.org/info/rfcXXXX); 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 (RFC 2119) (RFC 8174) when, and only when,
    they appear in all capitals, as shown here.

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

  revision 2019-10-15 {
    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.";
  }

  identity frr-protection-method {
    description
      "Base identity for a Fast Reroute protection method.";
  }
  identity frr-protection-method-lfa {
    base frr-protection-method;
    description "Loop Free Alternate as defined in RFC5286.";
  }
  identity frr-protection-method-rlfa {
    base frr-protection-method;
    description "Remote Loop Free Alternate as defined in RFC7490.";
  }
  identity frr-protection-method-rsvpte {
    base frr-protection-method;
    description "RSVP-TE as defined in RFC4090.";
  }

  identity frr-protection-available-type {
    description "Base identity for Fast Reroute protection types
                provided by an alternate path.";
  }
  identity frr-protection-available-node-type {
    base frr-protection-available-type;
    description "Node protection is provided by the alternate.";
  }
  identity frr-protection-available-link-type {
    base frr-protection-available-type;
    description "Link protection is provided by the alternate.";
  }
  identity frr-protection-available-srlg-type {
    base frr-protection-available-type;
    description "SRLG protection is provided by the alternate.";
  }
  identity frr-protection-available-downstream-type {
    base frr-protection-available-type;
    description "The alternate is downstream of node in the path.";
  }
  identity frr-protection-available-other-type {
    base frr-protection-available-type;
    description "The level of protection is unknown.";
  }

  identity frr-alternate-type {
    description "Base identity for IP Fast Reroute alternate type.";
  }
  identity frr-alternate-type-equal-cost {
    base frr-alternate-type;
        description "ECMP alternate.";
  }
  identity frr-alternate-type-lfa {
    base frr-alternate-type;
        description "LFA alternate.";
  }
  identity frr-alternate-type-remote-lfa {
    base frr-alternate-type;
        description "Remote LFA alternate.";
  }
  identity frr-alternate-type-tunnel {
    base frr-alternate-type;
        description "Tunnel based alternate (such as,
                 RSVP-TE or GRE).";
  }
  identity frr-alternate-mrt {
    base frr-alternate-type;
        description "MRT alternate.";
  }
  identity frr-alternate-tilfa {
    base frr-alternate-type;
        description "TILFA alternate.";
  }
  identity frr-alternate-other {
    base frr-alternate-type;
        description "Other alternate.";
  }


  identity unidirectional-link-delay-subtlv-flag {
      description "Base identity for unidirectional-link-delay
                  subTLV flags. Flags are defined in RFC8570.";
  }
  identity unidirectional-link-delay-subtlv-a-flag {
      base unidirectional-link-delay-subtlv-flag;
      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.";
  }
  identity min-max-unidirectional-link-delay-subtlv-flag {
      description
        "Base identity for min-max-unidirectional-link-delay
         subTLV flags. Flags are defined in RFC8570.";
  }
  identity min-max-unidirectional-link-delay-subtlv-a-flag {
      base min-max-unidirectional-link-delay-subtlv-flag;
      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.";
  }
  identity unidirectional-link-loss-subtlv-flag {
      description "Base identity for unidirectional-link-loss
                  subTLV flags. Flags are defined in RFC8570.";
  }

  identity unidirectional-link-loss-subtlv-a-flag {
      base unidirectional-link-loss-subtlv-flag;
      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.";
  }
  identity tlv229-flag {
      description "Base identity for TLV229 flags. Flags are defined
                  in RFC5120.";
  }
  identity tlv229-overload-flag {
      base tlv229-flag;
      description
          "If set, the originator is overloaded,
          and must be avoided in path calculation.";
  }
  identity tlv229-attached-flag {
      base tlv229-flag;
      description
          "If set, the originator is attached to
          another area using the referred metric.";
  }
  identity router-capability-flag {
      description "Base identity for router capability flags.
        Flags are defined in RFC7981.";
  }
  identity router-capability-flooding-flag {
      base router-capability-flag;
      description
          "Quote from RFC7981: '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'.";
  }
  identity router-capability-down-flag {
      base router-capability-flag;
      description
          "Quote from RFC7981: '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'.";
  }

  identity lsp-flag {
      description "Base identity for LSP attributes.
```

```
                  Attributes are defined in ISO 10589";
     }
     identity lsp-partitioned-flag {
         base lsp-flag;
         description "Originator partition repair supported";
     }
     identity lsp-attached-error-metric-flag {
         base lsp-flag;
         description "Set when originator is attached to
              another area using the error metric.";
     }
     identity lsp-attached-delay-metric-flag {
         base lsp-flag;
         description "Set when originator is attached to
              another area using the delay metric.";
     }
     identity lsp-attached-expense-metric-flag {
         base lsp-flag;
         description "Set when originator is attached to
              another area using the expense metric.";
     }
     identity lsp-attached-default-metric-flag {
         base lsp-flag;
         description "Set when originator is attached to
              another area using the default metric.";
     }
     identity lsp-overload-flag {
         base lsp-flag;
         description
             "If set, the originator is overloaded,
              and must be avoided in path calculation.";
     }
     identity lsp-l1system-flag {
         base lsp-flag;
         description
             "Set when the Intermediate System has an L1 type.";
     }
     identity lsp-l2system-flag {
         base lsp-flag;
         description
             "Set when the Intermediate System has an L2 type.";
     }


     /* Feature definitions */

     feature osi-interface {
       description "Support of OSI specific parameters on an
```

```
                   interface.";
     }
     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
         "Support for LDP IGP synchronization.";
       reference "RFC5443 - LDP IGP Synchronization.";
     }
     feature fast-reroute {
       description
         "Support for IP Fast Reroute (IP-FRR).";
     }
     feature nsr {
       description
```

```
         "Support for Non-Stop-Routing (NSR). The IS-IS NSR feature
             allows  a router with redundant control-plane capability
             (e.g., dual Route-Processor (RP) cards) to maintain its
             state and adjacencies during planned and unplanned
             IS-IS instance restarts. It differs from graceful-restart
             or Non-Stop Forwarding (NSF) in that no protocol signaling
             or assistance from adjacent IS-IS neighbors is required to
             recover control-plane state.";
     }
     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.
           In IS-IS, the overload bit is usually used to signal that
           a node cannot be used as a transit. The overload-max-metric
           feature brings a similar behavior leveraging on setting all
           the link metrics 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
         "Support for IS-IS interface metric computation
          according to a 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
         "Support for the advertisement
          of a Network Layer Protocol Identifier 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 for maximum-area-addresses configuration.";
     }

     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 for 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, such as,
          '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 identityref {
                       base frr-alternate-type;
                     }
             description
               "Type of alternate.";
           }
           leaf best {
             type boolean;
             description
               "Is set when the alternate is the preferred one,
                is clear 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.";
           }
           container protection-available {
             leaf-list protection-types {
               type identityref {
                 base frr-protection-available-type;
               }
               description "This list contains a set of protection
                           types defined as identities.
                           An identity must be added for each type of
                           protection provided by the alternate.
                           As an example, if an alternate provides
                           SRLG, node and link protection, three
                           identities must be added in this list:
                           one for SRLG protection, one for node
                           protection, one for link protection.";
             }
             description "Protection types 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 prefixes {
           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 identityref {
             base frr-protection-method;
           }
           description "Protection method used.";
         }
         list address-family-stats {
           key address-family;

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

             description "Address-family";
           }
           leaf total-routes {
             type yang:gauge32;
             description "Total prefixes.";
           }
           leaf unprotected-routes {
             type yang:gauge32;
             description
               "Total prefixes that are not protected.";
           }
           leaf protected-routes {
             type yang:gauge32;
             description
               "Total prefixes that are protected.";
           }
           leaf link-protected-routes {
             type yang:gauge32;
             description
               "Total prefixes that are link protected.";
           }
           leaf node-protected-routes {
             type yang:gauge32;
             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.
            This list provides a consolidated view of both
            32-bit and 64-bit tags (RFC5130) available for the prefix.";
       }
       leaf route-type {
         type enumeration {
           enum l2-intra-area {
             description "Level 2 internal route. As per RFC5302,
                          the prefix is directly connected to the
                          advertising router. It cannot be
                          distinguished from an L1->L2 inter-area
                          route.";
           }
           enum l1-intra-area {
             description "Level 1 internal route. As per RFC5302,
                          the prefix is directly connected to the
                          advertising router.";
           }
           enum l2-external {
             description "Level 2 external route. As per RFC5302,
                          such a route is learned from other IGPs.
                          It cannot be distinguished from an L1->L2
                          inter-area external route.";
           }
           enum l1-external {
             description "Level 1 external route. As per RFC5302,
                          such a route is learned from other IGPs.";
           }
           enum l1-inter-area {
             description "These prefixes are learned via L2 routing.";
           }
           enum l1-inter-area-external {
             description "These prefixes are learned via L2 routing
                          towards an l2-external route.";
           }
         }
         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 for the adjacency.
                On a p2p link this might be level 1 and 2,
                but on a LAN, the usage will be level 1
                between neighbors at level 1 or level 2 between
                neighbors 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.";
           }
           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";
           }
         }
         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 metric-type-global-cfg-with-default {
       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;
         description  "Value of the metric";
       }
       description
         "Global default metric config grouping.";
     }
     grouping default-metric-global-cfg-with-default {
       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.";
           }
           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";
         description
           "Interval (in seconds) between successive hello
            messages.";
       }

       description "Interval between hello messages.";
     }
     grouping hello-interval-cfg-with-default {
       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;
         description
           "Number of missed hello messages prior to
            declaring the adjacency down.";
       }
       description
           "Number of missed hello messages prior to
            adjacency down grouping.";
     }
     grouping hello-multiplier-cfg-with-default {
       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";
         }
         description
           "Priority of interface for DIS election.";
       }

       description "Interface DIS election priority grouping";
     }
     grouping priority-cfg-with-default {
       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;
         description "Metric value.";
       }
       description "Interface metric grouping";
     }
     grouping metric-cfg-with-default {
       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-with-default;
         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-with-default;
         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 Originator
                   Identification TLV.";
       }
       description "Grouping for LSP global parameters.";
     }
     grouping spf-parameters {
       container spf-control {
           leaf paths {
             if-feature max-ecmp;
             type uint16 {
               range "1..65535";
             }
             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;
       leaf discontinuity-time {
         type yang:date-and-time;
         description
           "The time of the most recent occasion at which any one
            or more of this IS-IS instance's counters suffered a
            discontinuity.  If no such discontinuities have occurred
            since the IS-IS instance was last re-initialized, then
            this node contains the time the IS-IS instance was
            re-initialized which normally occurs when it was
            created.";
       }
     }

     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";

       uses admin-control;

       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 for the interface. This
            dictates the type of hello messages that are used.";
       }

       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-with-default;
         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-with-default;
          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-with-default;
          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-with-default;
          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;
        leaf discontinuity-time {
          type yang:date-and-time;
          description
            "The time of the most recent occasion at which any one
             or more of this IS-IS interface's counters suffered a
             discontinuity.  If no such discontinuities have occurred
             since the IS-IS interface was last re-initialized, then
             this node contains the time the IS-IS interface was
             re-initialized which normally occurs when it was
             created.";
          }
      }

     /* 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
          "Grouping for IS-IS system counters";
      }
      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.";
              }
              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 yang:counter32;
              description
                "Event identifier -  purely internal value.
                 It is expected the most recent events to have the bigger
                 id number.";
            }
            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 yang:counter32;
              description
                "Event identifier -  purely internal value.
                 It is expected the most recent events to have the bigger
                 id number.";
            }
            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.
            Set to 'false' to indicate an internal metric.";
       }
       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 expense 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
              "Indicates whether IS-IS error metric is supported.";
          }
          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.
              Set to true when the prefix has been advertised down
              the hierarchy.";
        }
        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 64-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.
               Set to true when the prefix has been advertised down
               the hierarchy.";
        }
        leaf ip-prefix {
          type inet:ipv6-address;
          description "IPv6 prefix address";
        }
        leaf prefix-len {
          type uint8;
          description  "IPv6 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 64-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 RFC8570";

         container unidirectional-link-delay {
           description
             "Container for the average delay
             from the local neighbor to the remote one.";
           container flags {
             leaf-list unidirectional-link-delay-subtlv-flags {
                 type identityref {
                     base unidirectional-link-delay-subtlv-flag;
                 }
                description
                      "This list contains identities for the bits
                       which are set.";
             }
             description
               "unidirectional-link-delay subTLV 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.";
           container flags {
             leaf-list min-max-unidirectional-link-delay-subtlv-flags {
                 type identityref {
                     base min-max-unidirectional-link-delay-subtlv-flag;
                 }
                 description
                     "This list contains identities for the bits which are
                     set.";
             }
             description
               "min-max-unidirectional-link-delay subTLV 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.";
           container flags {
             leaf-list unidirectional-link-loss-subtlv-flags {
                 type identityref {
                     base unidirectional-link-loss-subtlv-flag;
                 }
                 description
                     "This list contains identities for the bits which are
                     set.";
             }
             description
               "unidirectional-link-loss subTLV 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.
                  Set to 'false' to indicate an internal metric.";
             }
             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 expense metric value";
            }
            leaf supported {
              type boolean;
              default "false";
              description "IS-IS expense metric supported";
            }
            description "IS-IS 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.";
           }
           container attributes {
               leaf-list flags {
                   type identityref {
                       base tlv229-flag;
                   }
                   description
                   "This list contains identities for the bits which are
                   set.";
               }
               description
                   "TLV 229 flags.";
           }
         }
         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 {
             container flags {
               leaf-list router-capability-flags {
                   type identityref {
                       base router-capability-flag;
                   }
                   description
                   "This list contains identities for the bits which are
                   set.";
               }
               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;

             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.";
       }
       container attributes {
           leaf-list lsp-flags {
               type identityref {
                   base lsp-flag;
               }
               description
                   "This list contains identities for the bits which are
                   set.";
           }
```

```
            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 {
        if-feature osi-interface;
        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 (ISO10589)
         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 "0..255";
         }
         description
           "The system may provide a reason to reject the
            adjacency. If the reason is not available,
            the reason string will not 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 modules specified in this document define a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8446].

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

There are a number of data nodes defined in ietf-isis.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. Writable data node represent configuration of each instance and interface. These correspond to the following schema nodes:

/isis

/isis/interfaces/interface[name]

For IS-IS, the ability to modify IS-IS configuration will allow the entire IS-IS domain to be compromised including forming adjacencies with unauthorized routers to misroute traffic or mount a massive

Denial-of-Service (DoS) attack. For example, adding IS-IS on any unprotected interface could allow an IS-IS adjacency to be formed with an unauthorized and malicious neighbor. Once an adjacency is formed, traffic could be hijacked. As a simpler example, a Denial-Of-Service attack could be mounted by changing the cost of an IS-IS interface to be asymmetric such that a hard routing loop ensues. In general, unauthorized modification of most IS-IS features will pose their own set of security risks and the "Security Considerations" in the respective reference RFCs should be consulted.

Some of the readable data nodes in the ietf-isis.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. Similarly, the IS-IS local RIB exposes the reachable prefixes in the IS-IS routing domain. The Link State Database (LSDB) and local RIB are represented by the following schema nodes:

```
/isis/database

/isis/local-rib
```

Exposure of the Link State Database and local RIB include information beyond the scope of the IS-IS router and this may be undesirable since exposure may facilitate other attacks. Additionally, the complete IP network topology and, if deployed, the traffic engineering topology of the IS-IS domain can be reconstructed from the Link State Database. Though not as straightforward, the IS-IS local RIB can also be discover topological information. Network operators may consider their topologies to be sensitive confidential data.

For IS-IS authentication, configuration is supported via the specification of key-chain [RFC8177] or the direct 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 IS-IS 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.

The actual authentication key data (whether locally specified or part of a key-chain) is sensitive and needs to be kept secret from unauthorized parties; compromise of the key data would allow an attacker to forge IS-IS traffic that would be accepted as authentic, potentially compromising the entirety IS-IS domain.

The model describes several notifications, implementations must rate-limit the generation of these notifications to avoid creating significant notification load. Otherwise, this notification load may have some side effects on the system stability and may be exploited as an attack vector.

8. Contributors

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

9. Acknowledgements

The authors would like to thank Tom Petch, Alvaro Retana, Stewart Bryant, Barry Leiba, Benjamin Kaduk and Adam Roach, and Roman Danyliw for their review and comments.

10. 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
```

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]
"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)", 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>.

[RFC4090] Pan, P., Ed., Swallow, G., Ed., and A. Atlas, Ed., "Fast Reroute Extensions to RSVP-TE for LSP Tunnels", RFC 4090, DOI 10.17487/RFC4090, May 2005, <https://www.rfc-editor.org/info/rfc4090>.

[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>.

[RFC5302] Li, T., Smit, H., and T. Przygienda, "Domain-Wide Prefix Distribution with Two-Level IS-IS", RFC 5302, DOI 10.17487/RFC5302, October 2008, <https://www.rfc-editor.org/info/rfc5302>.

[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>.

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

[RFC5308] Hopps, C., "Routing IPv6 with IS-IS", RFC 5308, DOI 10.17487/RFC5308, October 2008, <https://www.rfc-editor.org/info/rfc5308>.

[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>.

[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>.

[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>.

[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>.

[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>.

[RFC8570] Ginsberg, L., Ed., Previdi, S., Ed., Giacalone, S., Ward, D., Drake, J., and Q. Wu, "IS-IS Traffic Engineering (TE) Metric Extensions", RFC 8570, DOI 10.17487/RFC8570, March 2019, <https://www.rfc-editor.org/info/rfc8570>.

11.2. Informative References

[I-D.ietf-rtgwg-segment-routing-ti-lfa]
Litkowski, S., Bashandy, A., Filsfils, C., Decraene, B., Francois, P., daniel.voyer@bell.ca, d., Clad, F., and P. Camarillo, "Topology Independent Fast Reroute using Segment Routing", draft-ietf-rtgwg-segment-routing-ti-lfa-01 (work in progress), March 2019.

[RFC7812] Atlas, A., Bowers, C., and G. Enyedi, "An Architecture for IP/LDP Fast Reroute Using Maximally Redundant Trees (MRT-FRR)", RFC 7812, DOI 10.17487/RFC7812, June 2016, <https://www.rfc-editor.org/info/rfc7812>.

[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>192.0.2.1</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
Cisco Systems

Email: slitkows.ietf@gmail.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

Open Shortest Path First IGP
Internet-Draft
Intended status: Standards Track
Expires: May 8, 2019

P. Psenak, Ed.
K. Talaulikar
Cisco Systems, Inc.
W. Henderickx
Nokia
P. Pillay-Esnault
Huawei

November 4, 2018

OSPF LLS Extensions for Local Interface ID Advertisement
draft-ietf-ospf-lls-interface-id-09

Abstract

Every OSPF interface is assigned an identifier, Interface ID, which uniquely identifies the interface on the router. In some cases it is useful to know the assigned Interface ID on the remote side of the adjacency (Remote Interface ID).

This draft describes the extensions to OSPF link-local signalling (LLS) to advertise the Local Interface Identifier.

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 BCP14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Every OSPF interface is assigned an Interface ID, which uniquely identifies the interface on the router. [RFC2328] uses this Interface ID in the Router-LSA Link Data for unnumbered links and uses the value of the MIB-II IfIndex [RFC2863]. [RFC4203] refers to these Interface IDs as the Link Local/Remote Identifiers and defines a way to advertise and use them for Generalized Multi-Protocol Label Switching (GMPLS) purposes. [RFC7684] defines a way to advertise Local/Remote Interface IDs in the OSPFv2 Extended Link LSA.

There is a known OSPFv2 protocol problem in verifying the bi-directional connectivity with parallel unnumbered links. If there are two parallel unnumbered links between a pair of routers and each link is only advertised from single direction, such two unidirectional parallel links could be considered as a valid single bidirectional link during the OSPF route computation on some other router. If each link is advertised with both its Local and Remote

Interface IDs, the advertisement of each link from both sides of adjacency can be verified by cross-checking the Local and Remote Interface IDs of both advertisements.

From the perspective of the advertising router, the Local Interface Identifier is a known value, however the Remote Interface Identifier needs to be learnt before it can be advertised. [RFC4203] suggests to use TE Link Local LSA [RFC3630] to communicate the Local Interface Identifier to neighbors on the link. Though such mechanism works, it has some drawbacks.

This draft proposes an extension to OSPF link-local signalling (LLS) [RFC5613] to advertise the Local Interface Identifier.

1.1. Interface ID Exchange using TE Opaque LSA

Usage of the Link Local TE Opaque LSA to propagate the Local Interface Identifier to the neighbors on the link is described in [RFC4203]. This mechanism has the following problems:

> LSAs can only be flooded over an existing adjacency that is in Exchange state or greater. The adjacency state machine progresses independently on each side of the adjacency and, as such, may reach the Full state on one side before the TE Link Opaque LSA arrives. The consequence is that link can be initially advertised without the Remote Interface Identifier. Later, when the TE Link Opaque LSA arrives, the link must be advertised again, this time with the valid Remote Interface Identifier. Implementations may choose to wait before advertising the link, but there is no guarantee that the neighbor will ever advertise the TE Link Opaque LSA with the Interface Identifier. In summary, the existing mechanism does not guarantee that the Remote Interface Identifier is known at the time the link is advertised.
>
> The TE Opaque LSA is defined for MPLS Traffic Engineering, but the knowledge of the Remote Interface Identifier is useful also for cases where MPLS TE is not used. One example is the mentioned lack of a valid 2-way connectivity check for parallel point-to-point links between OSPF routers.

2. Interface ID Exchange using OSPF LLS

To address the problems described earlier and to allow the Interface Identifier exchange to be part of the neighbor discovery process, we propose to extend OSPF link-local signalling to advertise the Local Interface Identifier in OSPF Hello and Database Description (DD) packets.

2.1. Local Interface Identifier TLV

The Local Interface Identifier TLV is a LLS TLV. It has following format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Type             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Local Interface Identifier                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

Type: TBD

Length: 4 octets

Local Interface Identifier: The value of the local Interface Identifier.

Local Interface Identifier TLV signalling using LLS is applicable to all OSPF interface types other than virtual links.

3. Backward Compatibility with RFC 4203

If the Local Interface ID signaling via Link Local TE Opaque LSA is supported in addition to the new LLS mechanism, implementations which support Local Interface ID signalling using LLS MUST prefer the Local Interface ID value received through LLS over the value received through the Link Local TE Opaque LSA if both are received from the same OSPF router.

Implementations which support Local Interface ID signalling via Link Local TE Opaque LSA MAY continue to do so to ensure backward compatibility. If they also support Local Interface ID signalling using LLS as described herein, they MUST signal the same Local Interface ID via both mechanisms.

During the rare conditions, when the Local Interface ID changes, a timing interval may exist, where the received values of the Local Interface ID advertised through LLS and Link Local TE Opaque LSA may differ. Such situation is temporary and received values via both mechanisms should become equal as soon as the next Hello and/or Link Local TE Opaque LSA is re-generated by the originator.

4. IANA Considerations

This specification allocates a single code point from the "Open Shortest Path First (OSPF) Link Local Signalling (LLS) - Type/Length/ Value Identifiers (TLV)" registry.

Following value is allocated:

o TBD - Local Interface Identifier TLV

5. Security Considerations

The security considerations for "OSPF Link-Local Signaling" [RFC5613] also apply to the Local Interface Identifier TLV described herein. The current usage of a neighbor's Local Interface Identifier is to disambiguate parallel links between OSPF routers. Hence, modification of the advertised Local Interface Identifier TLV may result in the wrong neighbor interface identifier being advertised in the OSPFv2 Extended Link LSA [RFC7684] and could prevent the link from being used. If authentication is being used in the OSPF routing domain [RFC5709], then the Cryptographic Authentication TLV [RFC5613] SHOULD also be used to protect that contents of the Link-Local Signaling (LLS) block.

Receiving a malformed LLS Interface Identifier TLV MUST NOT result in a hard router or OSPF process failure. The reception of malformed LLS TLVs or Sub-TLVs SHOULD be logged but such logging MUST be rate-limited to prevent Denial-of-Service (DoS) attacks.

The interface ID is assigned by the advertising OSPF router as a locally unique identifier and need not be unique in any broader context; it is not expected to contain any information about the device owner or traffic transiting the device, so there are no privacy oncerns associated with its advertisement.

6. Acknowledgments

Thanks to Tony Przygienda for his extensive review and useful comments.

7. References

7.1. Normative References

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

[RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, DOI 10.17487/RFC2328, April 1998, <https://www.rfc-editor.org/info/rfc2328>.

[RFC3630] Katz, D., Kompella, K., and D. Yeung, "Traffic Engineering (TE) Extensions to OSPF Version 2", RFC 3630, DOI 10.17487/RFC3630, September 2003, <https://www.rfc-editor.org/info/rfc3630>.

[RFC4203] Kompella, K., Ed. and Y. Rekhter, Ed., "OSPF Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)", RFC 4203, DOI 10.17487/RFC4203, October 2005, <https://www.rfc-editor.org/info/rfc4203>.

[RFC5613] Zinin, A., Roy, A., Nguyen, L., Friedman, B., and D. Yeung, "OSPF Link-Local Signaling", RFC 5613, DOI 10.17487/RFC5613, August 2009, <https://www.rfc-editor.org/info/rfc5613>.

[RFC7684] Psenak, P., Gredler, H., Shakir, R., Henderickx, W., Tantsura, J., and A. Lindem, "OSPFv2 Prefix/Link Attribute Advertisement", RFC 7684, DOI 10.17487/RFC7684, November 2015, <https://www.rfc-editor.org/info/rfc7684>.

[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>.

7.2. Informative References

[RFC2863] McCloghrie, K. and F. Kastenholz, "The Interfaces Group MIB", RFC 2863, DOI 10.17487/RFC2863, June 2000, <https://www.rfc-editor.org/info/rfc2863>.

[RFC5709] Bhatia, M., Manral, V., Fanto, M., White, R., Barnes, M., Li, T., and R. Atkinson, "OSPFv2 HMAC-SHA Cryptographic Authentication", RFC 5709, DOI 10.17487/RFC5709, October 2009, <https://www.rfc-editor.org/info/rfc5709>.

Authors’ Addresses

Peter Psenak (editor)
Cisco Systems, Inc.
Apollo Business Center
Mlynske nivy 43
Bratislava 821 09
Slovakia

Email: ppsenak@cisco.com

Ketan Jivan Talaulikar
Cisco Systems, Inc.
S.No. 154/6, Phase I, Hinjawadi
PUNE, MAHARASHTRA   411 057
India

Email: ketant@cisco.com

Wim Henderickx
Nokia
Copernicuslaan 50
Antwerp 2018
Belgium

Email: wim.henderickx@nokia.com

Padma Pillay-Esnault
Huawei
2330 Central Expressway
Santa Clara, CA 95050
USA

Email: padma@huawei.com

Open Shortest Path First IGP
Internet-Draft
Intended status: Standards Track
Expires: July 13, 2019

P. Psenak, Ed.
Cisco Systems, Inc.
S. Previdi, Ed.
Individual

January 9, 2019

OSPFv3 Extensions for Segment Routing
draft-ietf-ospf-ospfv3-segment-routing-extensions-23

Abstract

Segment Routing (SR) allows a flexible definition of end-to-end paths within IGP topologies by encoding paths as sequences of topological sub-paths, called "segments". These segments are advertised by the link-state routing protocols (IS-IS and OSPF).

This draft describes the OSPFv3 extensions required for Segment Routing with MPLS 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 BCP14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Segment Routing (SR) allows a flexible definition of end-to-end paths within IGP topologies by encoding paths as sequences of topological sub-paths, called "segments". These segments are advertised by the link-state routing protocols (IS-IS and OSPF). Prefix segments represent an ECMP-aware shortest-path to a prefix (or a node), as per the state of the IGP topology. Adjacency segments represent a hop over a specific adjacency between two nodes in the IGP. A prefix segment is typically a multi-hop path while an adjacency segment, in most cases, is a one-hop path. SR's control-plane can be applied to both IPv6 and MPLS data-planes, and does not require any additional signalling (other than IGP extensions). The IPv6 data plane is out of the scope of this specification - OSPFv3 extension for SR with IPv6 data plane will be specified in a separate document. When used in MPLS networks, SR paths do not require any LDP or RSVP-TE signalling. However, SR can interoperate in the presence of LSPs established with RSVP or LDP.

This draft describes the OSPFv3 extensions required for Segment Routing with MPLS data plane.

Segment Routing architecture is described in [RFC8402].

Segment Routing use cases are described in [RFC7855].

2. Terminology

This section lists some of the terminology used in this document:

ABR - Area Border Router

Adj-SID - Adjacency Segment Identifier

AS - Autonomous System

ASBR - Autonomous System Boundary Router

DR - Designated Router

IS-IS - Intermediate System to Intermediate System

LDP - Label Distribution Protocol

LSP - Label Switched Path

MPLS - Multi Protocol Label Switching

OSPF - Open Shortest Path First

SPF - Shortest Path First

RSVP - Resource Reservation Protocol

SID - Segment Identifier

SR - Segment Routing

SRGB - Segment Routing Global Block

SRLB - Segment Routing Local Block

SRMS - Segment Routing Mapping Server

TLV - Type Length Value

3. Segment Routing Identifiers

Segment Routing defines various types of Segment Identifiers (SIDs): Prefix-SID, Adjacency-SID, and LAN Adjacency SID.

3.1. SID/Label Sub-TLV

The SID/Label Sub-TLV appears in multiple TLVs or Sub-TLVs defined later in this document. It is used to advertise the SID or label associated with a prefix or adjacency. The SID/Label Sub-TLV has following format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Type             |               Length          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      SID/Label (variable)                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

Type: 7

Length: Either 3 or 4 octets

SID/Label: If length is set to 3, then the 20 rightmost bits represent a label. If length is set to 4, then the value represents a 32-bit SID.

The receiving router MUST ignore the SID/Label Sub-TLV if the length is other than 3 or 4.

4. Segment Routing Capabilities

Segment Routing requires some additional router capabilities to be advertised to other routers in the area.

These SR capabilities are advertised in the OSPFv3 Router Information Opaque LSA (defined in [RFC7770]) and specified in [I-D.ietf-ospf-segment-routing-extensions].

5. OSPFv3 Extended Prefix Range TLV

In some cases it is useful to advertise attributes for a range of prefixes in a single advertisement. The Segment Routing Mapping Server, which is described in [I-D.ietf-spring-segment-routing-ldp-interop], is an example of where SIDs for multiple prefixes can be advertised. To optimize such advertisement in case of multiple prefixes from a contiguous address range, OSPFv3 Extended Prefix Range TLV is defined."

The OSPFv3 Extended Prefix Range TLV is a top-level TLV of the following LSAs defined in [RFC8362]:

E-Intra-Area-Prefix-LSA

E-Inter-Area-Prefix-LSA

E-AS-External-LSA

E-Type-7-LSA

Multiple OSPFv3 Extended Prefix Range TLVs MAY be advertised in each LSA mentioned above. The OSPFv3 Extended Prefix Range TLV has the following format:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |              Type             |             Length            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | Prefix Length |      AF       |         Range Size            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |    Flags      |                 Reserved                      |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                      Address Prefix (variable)                |
   |                            ...                                |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                       Sub-TLVs (variable)                     |
   +-                                                             -+
   |                                                               |
```

where:

Type: 9

Length: Variable, in octets, dependent on Sub-TLVs.

Prefix length: Length of prefix in bits.

AF: Address family for the prefix.

AF: 0 - IPv4 unicast

AF: 1 - IPv6 unicast

Range size: Represents the number of prefixes that are covered by the advertisement. The Range Size MUST NOT exceed the number of prefixes that could be satisfied by the prefix length without including:

Addresses from the IPv4 multicast address range (224.0.0.0/3), if the AF is IPv4 unicast

Addresses other than the IPv6 unicast addresses, if the AF is IPv6 unicast

Flags: Reserved. MUST be zero when sent and are ignored when received.

Reserved: SHOULD be set to 0 on transmission and MUST be ignored on reception.

Address Prefix:

For the address family IPv4 unicast, the prefix itself is encoded as a 32-bit value. The default route is represented by a prefix of length 0.

For the address family IPv6 unicast, the prefix, encoded as an even multiple of 32-bit words, padded with zeroed bits as necessary. This encoding consumes ((PrefixLength + 31) / 32) 32-bit words.

Prefix encoding for other address families is beyond the scope of this specification. Prefix encoding for other address families can be defined in the future standard-track IETF specifications.

The range represents the contiguous set of prefixes with the same prefix length as specified by the Prefix Length field. The set starts with the prefix that is specified by the Address Prefix field. The number of prefixes in the range is equal to the Range size.

If the OSPFv3 Extended Prefix Range TLVs advertising the exact same range appears in multiple LSAs of the same type, originated by the same OSPFv3 router, the LSA with the numerically smallest Instance ID MUST be used and subsequent instances of the OSPFv3 Extended Prefix Range TLVs MUST be ignored.

6. Prefix SID Sub-TLV

The Prefix SID Sub-TLV is a Sub-TLV of the following OSPFv3 TLVs as defined in [RFC8362] and in Section 5:

Intra-Area Prefix TLV

Inter-Area Prefix TLV

External Prefix TLV

OSPFv3 Extended Prefix Range TLV

It MAY appear more than once in the parent TLV and has the following format:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |             Type              |             Length            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     Flags     |   Algorithm   |           Reserved            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                  SID/Index/Label (variable)                   |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

Type: 4

Length: 7 or 8 octets, dependent on the V-flag

Flags: Single octet field. The following flags are defined:

```
     0  1  2  3  4  5  6  7
   +--+--+--+--+--+--+--+--+
   |  |NP|M |E |V |L |  |  |
   +--+--+--+--+--+--+--+--+
```

where:

NP-Flag: No-PHP flag. If set, then the penultimate hop MUST NOT pop the Prefix-SID before delivering packets to the node that advertised the Prefix-SID.

M-Flag: Mapping Server Flag. If set, the SID was advertised by a Segment Routing Mapping Server as described in [I-D.ietf-spring-segment-routing-ldp-interop].

E-Flag: Explicit-Null Flag. If set, any upstream neighbor of the Prefix-SID originator MUST replace the Prefix-SID with the Explicit-NULL label (0 for IPv4, 2 for IPv6) before forwarding the packet.

V-Flag: Value/Index Flag. If set, then the Prefix-SID carries an absolute value. If not set, then the Prefix-SID carries an index.

L-Flag: Local/Global Flag. If set, then the value/index carried by the Prefix-SID has local significance. If not set, then the value/index carried by this Sub-TLV has global significance.

Other bits: Reserved. These MUST be zero when sent and are ignored when received.

Reserved: SHOULD be set to 0 on transmission and MUST be ignored on reception.

Algorithm: Single octet identifying the algorithm the Prefix-SID is associated with as defined in [I-D.ietf-ospf-segment-routing-extensions].

A router receiving a Prefix-SID from a remote node and with an algorithm value that such remote node has not advertised in the SR-Algorithm Sub-TLV [I-D.ietf-ospf-segment-routing-extensions] MUST ignore the Prefix-SID Sub-TLV.

SID/Index/Label: According to the V-Flag and L-Flag, it contains:

V-flag is set to 0 and L-flag is set to 0: The SID/Index/Label field is a 4 octet index defining the offset in the SID/Label space advertised by this router

V-flag is set to 1 and L-flag is set to 1: The SID/Index/Label field is a 3 octet local label where the 20 rightmost bits are used for encoding the label value.

All other combinations of V-flag and L-flag are invalid and any SID advertisement received with an invalid setting for V and L flags MUST be ignored.

If an OSPFv3 router advertises multiple Prefix-SIDs for the same prefix, topology, and algorithm, all of them MUST be ignored.

When calculating the outgoing label for the prefix, the router MUST take into account, as described below, the E, NP, and M flags advertised by the next-hop router if that router advertised the SID for the prefix. This MUST be done regardless of whether the next-hop router contributes to the best path to the prefix.

The NP-Flag (No-PHP) MUST be set and the E-flag MUST be clear for Prefix-SIDs allocated to prefixes that are propagated between areas by an ABR based on intra-area or inter-area reachability, unless the advertised prefix is directly attached to such ABR.

The NP-Flag (No-PHP) MUST be set and the E-flag MUST be clear for Prefix-SIDs allocated to redistributed prefixes, unless the redistributed prefix is directly attached to the advertising ASBR.

If the NP-Flag is not set, then any upstream neighbor of the Prefix-SID originator MUST pop the Prefix-SID. This is equivalent to the penultimate hop popping mechanism used in the MPLS dataplane. If the NP-flag is not set, then the received E-flag is ignored.

If the NP-flag is set then:

If the E-flag is not set, then any upstream neighbor of the Prefix-SID originator MUST keep the Prefix-SID on top of the stack. This is useful when the originator of the Prefix-SID needs to stitch the incoming packet into a continuing MPLS LSP to the final destination. This could occur at an ABR (prefix propagation from one area to another) or at an ASBR (prefix propagation from one domain to another).

If the E-flag is set, then any upstream neighbor of the Prefix-SID originator MUST replace the Prefix-SID with an Explicit-NULL label. This is useful, e.g., when the originator of the Prefix-SID is the final destination for the related prefix and the originator wishes to receive the packet with the original Traffic Class field [RFC5462].

When the M-Flag is set, the NP-flag and the E-flag MUST be ignored on reception.

As the Mapping Server does not specify the originator of a prefix advertisement, it is not possible to determine PHP behavior solely based on the Mapping Server advertisement. However, PHP behavior SHOULD be done in following cases:

The Prefix is intra-area type and the downstream neighbor is the originator of the prefix.

The Prefix is inter-area type and the downstream neighbor is an ABR, which is advertising prefix reachability and is setting the LA-bit in the Prefix Options as described in [RFC8362].

The Prefix is external type and the downstream neighbor is an ASBR, which is advertising prefix reachability and is setting the LA-bit in the Prefix Options as described in [RFC8362].

When a Prefix-SID is advertised in the OSPFv3 Extended Prefix Range TLV, then the value advertised in the Prefix SID Sub-TLV is interpreted as a starting SID/Label value.

Example 1: If the following router addresses (loopback addresses) need to be mapped into the corresponding Prefix SID indexes:

```
          Router-A: 2001:DB8::1/128, Prefix-SID: Index 1
          Router-B: 2001:DB8::2/128, Prefix-SID: Index 2
          Router-C: 2001:DB8::3/128, Prefix-SID: Index 3
          Router-D: 2001:DB8::4/128, Prefix-SID: Index 4
```

then the Address Prefix field in the OSPFv3 Extended Prefix Range TLV would be set to 2001:DB8::1, the Prefix Length would be set to 128, the Range Size would be set to 4, and the Index value in the Prefix-SID Sub-TLV would be set to 1.

Example 2: If the following prefixes need to be mapped into the corresponding Prefix-SID indexes:

```
          2001:DB8:1::0/120,   Prefix-SID: Index 51
          2001:DB8:1::100/120, Prefix-SID: Index 52
          2001:DB8:1::200/120, Prefix-SID: Index 53
          2001:DB8:1::300/120, Prefix-SID: Index 54
          2001:DB8:1::400/120, Prefix-SID: Index 55
          2001:DB8:1::500/120, Prefix-SID: Index 56
          2001:DB8:1::600/120, Prefix-SID: Index 57
```

then the Prefix field in the OSPFv3 Extended Prefix Range TLV would be set to 2001:DB8:1::0, the Prefix Length would be set to 120, the Range Size would be set to 7, and the Index value in the Prefix-SID Sub-TLV would be set to 51.

7. Adjacency Segment Identifier (Adj-SID)

An Adjacency Segment Identifier (Adj-SID) represents a router adjacency in Segment Routing.

7.1. Adj-SID Sub-TLV

The Adj-SID Sub-TLV is an optional Sub-TLV of the Router-Link TLV as defined in [RFC8362]. It MAY appear multiple times in the Router-Link TLV. The Adj-SID Sub-TLV has the following format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Type            |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Flags         |     Weight    |             Reserved          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   SID/Label/Index (variable)                  |
+---------------------------------------------------------------+
```

where:

Type: 5

Length: 7 or 8 octets, dependent on the V flag.

Flags: Single octet field containing the following flags:

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|B|V|L|G|P|     |
+-+-+-+-+-+-+-+-+
```

where:

B-Flag: Backup Flag. If set, the Adj-SID refers to an adjacency that is eligible for protection (e.g., using IPFRR or MPLS-FRR) as described in section 3.5 of [RFC8402].

The V-Flag: Value/Index Flag. If set, then the Adj-SID carries an absolute value. If not set, then the Adj-SID carries an index.

The L-Flag: Local/Global Flag. If set, then the value/index carried by the Adj-SID has local significance. If not set, then the value/index carried by this Sub-TLV has global significance.

The G-Flag: Group Flag. When set, the G-Flag indicates that the Adj-SID refers to a group of adjacencies (and therefore MAY be assigned to other adjacencies as well).

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

Other bits: Reserved. These MUST be zero when sent and are ignored when received.

Reserved: SHOULD be set to 0 on transmission and MUST be ignored on reception.

Weight: Weight used for load-balancing purposes. The use of the weight is defined in [RFC8402].

SID/Index/Label: as described in Section 6.

An SR-capable router MAY allocate an Adj-SID for each of its adjacencies and set the B-Flag when the adjacency is eligible for protection by an FRR mechanism (IP or MPLS) as described in [RFC8402].

An SR-capable router MAY allocate more than one Adj-SID to an adjacency.

An SR-capable router MAY allocate the same Adj-SID to different adjacencies.

When the P-flag is not set, the Adj-SID MAY be persistent. When the P-flag is set, the Adj-SID MUST be persistent.

7.2. LAN Adj-SID Sub-TLV

The LAN Adj-SID Sub-TLV is an optional Sub-TLV of the Router-Link TLV. It MAY appear multiple times in the Router-Link TLV. It is used to advertise a SID/Label for an adjacency to a non-DR router on a broadcast, NBMA, or hybrid [RFC6845] network.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |              Type             |            Length             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     Flags     |     Weight    |            Reserved           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                           Neighbor ID                         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                    SID/Label/Index (variable)                 |
   +---------------------------------------------------------------+
```

where:

Type: 6

Length: 11 or 12 octets, dependent on V-flag.

Flags: same as in Section 7.1

Weight: Weight used for load-balancing purposes. The use of the weight is defined in [RFC8402].

Reserved: SHOULD be set to 0 on transmission and MUST be ignored on reception.

Neighbor ID: The Router ID of the neighbor for which the LAN-Adj-SID is advertised.

SID/Index/Label: as described in Section 6.

When the P-flag is not set, the LAN Adj-SID MAY be persistent. When the P-flag is set, the LAN Adj-SID MUST be persistent.

8. Elements of Procedure

8.1. Intra-area Segment routing in OSPFv3

An OSPFv3 router that supports segment routing MAY advertise Prefix-SIDs for any prefix to which it is advertising reachability (e.g., a loopback IP address as described in Section 6).

A Prefix-SID can also be advertised by SR Mapping Servers (as described in [I-D.ietf-spring-segment-routing-ldp-interop]). A Mapping Server advertises Prefix-SIDs for remote prefixes that exist in the OSPFv3 routing domain. Multiple Mapping Servers can advertise Prefix-SIDs for the same prefix, in which case the same Prefix-SID MUST be advertised by all of them. The SR Mapping Server could use either area flooding scope or autonomous system flooding scope when advertising Prefix SIDs for prefixes, based on the configuration of the SR Mapping Server. Depending on the flooding scope used, the SR Mapping Server chooses the OSPFv3 LSA type that will be used. If the area flooding scope is needed, an E-Intra-Area-Prefix-LSA [RFC8362] is used. If autonomous system flooding scope is needed, an E-AS-External-LSA [RFC8362] is used.

When a Prefix-SID is advertised by the Mapping Server, which is indicated by the M-flag in the Prefix-SID Sub-TLV (Section 6), the route type as implied by the LSA type is ignored and the Prefix-SID is bound to the corresponding prefix independent of the route type.

Advertisement of the Prefix-SID by the Mapping Server using an Inter-Area Prefix TLV, External-Prefix TLV, or Intra-Area-Prefix TLV [RFC8362] does not itself contribute to the prefix reachability. The NU-bit [RFC5340] MUST be set in the PrefixOptions field of the LSA which is used by the Mapping Server to advertise SID or SID Range, which prevents the advertisement from contributing to prefix reachability.

An SR Mapping Server MUST use the OSPFv3 Extended Prefix Range TLVs when advertising SIDs for prefixes. Prefixes of different route-types can be combined in a single OSPFv3 Extended Prefix Range TLV advertised by an SR Mapping Server.

Area-scoped OSPFv3 Extended Prefix Range TLVs are propagated between areas, similar to propagation of prefixes between areas. Same rules that are used for propagating prefixes between areas [RFC5340] are used for the propagation of the prefix ranges.

8.2. Inter-area Segment routing in OSPFv3

In order to support SR in a multi-area environment, OSPFv3 MUST propagate Prefix-SID information between areas. The following procedure is used to propagate Prefix SIDs between areas.

When an OSPFv3 ABR advertises an Inter-Area-Prefix-LSA from an intra-area prefix to all its connected areas, it will also include the Prefix-SID Sub-TLV, as described in Section 6. The Prefix-SID value will be set as follows:

> The ABR will look at its best path to the prefix in the source area and find the advertising router associated with the best path to that prefix.
>
> The ABR will then determine if such router advertised a Prefix-SID for the prefix and use it when advertising the Prefix-SID to other connected areas.
>
> If no Prefix-SID was advertised for the prefix in the source area by the router that contributes to the best path to the prefix, the originating ABR will use the Prefix-SID advertised by any other router when propagating the Prefix-SID for the prefix to other areas.

When an OSPFv3 ABR advertises Inter-Area-Prefix-LSA LSAs from an inter-area route to all its connected areas, it will also include the Prefix-SID Sub-TLV, as described in Section 6. The Prefix-SID value will be set as follows:

> The ABR will look at its best path to the prefix in the backbone area and find the advertising router associated with the best path to that prefix.
>
> The ABR will then determine if such router advertised a Prefix-SID for the prefix and use it when advertising the Prefix-SID to other connected areas.
>
> If no Prefix-SID was advertised for the prefix in the backbone area by the ABR that contributes to the best path to the prefix, the originating ABR will use the Prefix-SID advertised by any other router when propagating the Prefix-SID for the prefix to other areas.

8.3. Segment Routing for External Prefixes

AS-External-LSAs are flooded domain wide. When an ASBR, which supports SR, originates an E-AS-External-LSA, it SHOULD also include a Prefix-SID Sub-TLV, as described in Section 6. The Prefix-SID value will be set to the SID that has been reserved for that prefix.

When an NSSA [RFC3101] ABR translates an E-NSSA-LSA into an E-AS-External-LSA, it SHOULD also advertise the Prefix-SID for the prefix. The NSSA ABR determines its best path to the prefix advertised in the translated E-NSSA-LSA and finds the advertising router associated with that path. If the advertising router has advertised a Prefix-SID for the prefix, then the NSSA ABR uses it when advertising the Prefix-SID for the E-AS-External-LSA. Otherwise, the Prefix-SID advertised by any other router will be used.

8.4. Advertisement of Adj-SID

The Adjacency Segment Routing Identifier (Adj-SID) is advertised using the Adj-SID Sub-TLV as described in Section 7.

8.4.1. Advertisement of Adj-SID on Point-to-Point Links

An Adj-SID MAY be advertised for any adjacency on a P2P link that is in neighbor state 2-Way or higher. If the adjacency on a P2P link transitions from the FULL state, then the Adj-SID for that adjacency MAY be removed from the area. If the adjacency transitions to a state lower than 2-Way, then the Adj-SID advertisement MUST be withdrawn from the area.

8.4.2. Adjacency SID on Broadcast or NBMA Interfaces

Broadcast, NBMA, or hybrid [RFC6845] networks in OSPFv3 are represented by a star topology where the DR is the central point to which all other routers on the broadcast, NBMA, or hybrid network connect. As a result, routers on the broadcast, NBMA, or hybrid network advertise only their adjacency to the DR. Routers that do not act as DR do not form or advertise adjacencies with each other. They do, however, maintain 2-Way adjacency state with each other and are directly reachable.

When Segment Routing is used, each router on the broadcast, NBMA, or hybrid network MAY advertise the Adj-SID for its adjacency to the DR using the Adj-SID Sub-TLV as described in Section 7.1.

SR-capable routers MAY also advertise a LAN-Adj-SID for other neighbors (e.g., BDR, DR-OTHER) on the broadcast, NBMA, or hybrid network using the LAN-Adj-SID Sub-TLV as described in Section 7.2.

9. IANA Considerations

This specification updates several existing OSPFv3 registries.

9.1. OSPFv3 Extended-LSA TLV Registry

Following values are allocated:

- 9 - OSPFv3 Extended Prefix Range TLV

9.2. OSPFv3 Extended-LSA Sub-TLV registry

- 4 - Prefix SID Sub-TLV

- 5 - Adj-SID Sub-TLV

- 6 - LAN Adj-SID Sub-TLV

- 7 - SID/Label Sub-TLV

10. Security Considerations

With the OSPFv3 segment routing extensions defined herein, OSPFv3 will now program the MPLS data plane [RFC3031]. Previously, LDP [RFC5036] or another label distribution mechanism was required to advertise MPLS labels and program the MPLS data plane.

In general, the same types of attacks that can be carried out on the IP control plane can be carried out on the MPLS control plane resulting in traffic being misrouted in the respective data planes. However, the latter can be more difficult to detect and isolate.

Existing security extensions as described in [RFC5340] and [RFC8362] apply to these segment routing extensions. While OSPFv3 is under a single administrative domain, there can be deployments where potential attackers have access to one or more networks in the OSPFv3 routing domain. In these deployments, stronger authentication mechanisms such as those specified in [RFC4552] or [RFC7166] SHOULD be used.

Implementations MUST assure that malformed TLV and Sub-TLV defined in this document are detected and do not provide a vulnerability for attackers to crash the OSPFv3 router or routing process. Reception of a malformed TLV or Sub-TLV SHOULD be counted and/or logged for further analysis. Logging of malformed TLVs and Sub-TLVs SHOULD be rate-limited to prevent a Denial of Service (DoS) attack (distributed or otherwise) from overloading the OSPFv3 control plane.

11. Contributors

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

Clarence Filsfils
Cisco Systems, Inc.
Brussels
Belgium

Email: cfilsfil@cisco.com

Hannes Gredler
RtBrick Inc.
Austria

Email: hannes@rtbrick.com

Rob Shakir
Google, Inc.
1600 Amphitheatre Parkway
Mountain View, CA 94043
US

Email: robjs@google.com

Wim Henderickx
Nokia
Copernicuslaan 50
Antwerp 2018
BE

Email: wim.henderickx@nokia.com

Jeff Tantsura
Nuage Networks
US

Email: jefftant.ietf@gmail.com

Thanks to Acee Lindem for his substantial contribution to the content of this document.

We would like to thank Anton Smirnov for his contribution as well.

12. References

12.1. Normative References

[ALGOREG] "IGP Algorithm Types", <https://www.iana.org/assignments/igp-parameters/igp-parameters.xhtml#igp-algorithm-types>.

[I-D.ietf-ospf-segment-routing-extensions]
Psenak, P., Previdi, S., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPF Extensions for Segment Routing", draft-ietf-ospf-segment-routing-extensions-27 (work in progress), December 2018.

[I-D.ietf-spring-segment-routing-ldp-interop]
Bashandy, A., Filsfils, C., Previdi, S., Decraene, B., and S. Litkowski, "Segment Routing interworking with LDP", draft-ietf-spring-segment-routing-ldp-interop-15 (work in progress), September 2018.

[I-D.ietf-spring-segment-routing-mpls]
Bashandy, A., Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls-18 (work in progress), December 2018.

[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>.

[RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, DOI 10.17487/RFC3031, January 2001, <https://www.rfc-editor.org/info/rfc3031>.

[RFC3101] Murphy, P., "The OSPF Not-So-Stubby Area (NSSA) Option", RFC 3101, DOI 10.17487/RFC3101, January 2003, <https://www.rfc-editor.org/info/rfc3101>.

[RFC5036] Andersson, L., Ed., Minei, I., Ed., and B. Thomas, Ed., "LDP Specification", RFC 5036, DOI 10.17487/RFC5036, October 2007, <https://www.rfc-editor.org/info/rfc5036>.

[RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, DOI 10.17487/RFC5340, July 2008, <https://www.rfc-editor.org/info/rfc5340>.

[RFC5462] Andersson, L. and R. Asati, "Multiprotocol Label Switching (MPLS) Label Stack Entry: "EXP" Field Renamed to "Traffic Class" Field", RFC 5462, DOI 10.17487/RFC5462, February 2009, <https://www.rfc-editor.org/info/rfc5462>.

[RFC6845] Sheth, N., Wang, L., and J. Zhang, "OSPF Hybrid Broadcast and Point-to-Multipoint Interface Type", RFC 6845, DOI 10.17487/RFC6845, January 2013, <https://www.rfc-editor.org/info/rfc6845>.

[RFC7770] Lindem, A., Ed., Shen, N., Vasseur, JP., Aggarwal, R., and S. Shaffer, "Extensions to OSPF for Advertising Optional Router Capabilities", RFC 7770, DOI 10.17487/RFC7770, February 2016, <https://www.rfc-editor.org/info/rfc7770>.

[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>.

[RFC8362] Lindem, A., Roy, A., Goethals, D., Reddy Vallem, V., and F. Baker, "OSPFv3 Link State Advertisement (LSA) Extensibility", RFC 8362, DOI 10.17487/RFC8362, April 2018, <https://www.rfc-editor.org/info/rfc8362>.

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

12.2. Informative References

[RFC4552] Gupta, M. and N. Melam, "Authentication/Confidentiality for OSPFv3", RFC 4552, DOI 10.17487/RFC4552, June 2006, <https://www.rfc-editor.org/info/rfc4552>.

[RFC7166] Bhatia, M., Manral, V., and A. Lindem, "Supporting Authentication Trailer for OSPFv3", RFC 7166, DOI 10.17487/RFC7166, March 2014, <https://www.rfc-editor.org/info/rfc7166>.

[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, <https://www.rfc-editor.org/info/rfc7855>.

Authors' Addresses

Peter Psenak (editor)
Cisco Systems, Inc.
Eurovea Centre, Central 3
Pribinova Street 10
Bratislava 81109
Slovakia

Email: ppsenak@cisco.com

Stefano Previdi (editor)
Individual

Email: stefano.previdi@net

LSR Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 1, 2021

P. Psenak, Ed.
L. Ginsberg
Cisco Systems
W. Henderickx
Nokia
J. Tantsura
Apstra
J. Drake
Juniper Networks

June 30, 2020

OSPF Application-Specific Link Attributes
draft-ietf-ospf-te-link-attr-reuse-16.txt

Abstract

Existing traffic engineering related link attribute advertisements have been defined and are used in RSVP-TE deployments. Since the original RSVP-TE use case was defined, additional applications (e.g., Segment Routing Policy, Loop Free Alternate) have been defined that also make use of the link attribute advertisements. 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 document introduces new link attribute advertisements in OSPFv2 and OSPFv3 that address both of these shortcomings.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Advertisement of link attributes by the OSPFv2 [RFC2328] and OSPFv3 [RFC5340] protocols in support of traffic engineering (TE) was introduced by [RFC3630] and [RFC5329] respectively. It has been extended by [RFC4203], [RFC7308] and [RFC7471]. Use of these extensions has been associated with deployments supporting Traffic Engineering over Multiprotocol Label Switching (MPLS) in the presence of the Resource Reservation Protocol (RSVP) - more succinctly referred to as RSVP-TE [RFC3209].

For the purposes of this document an application is a technology that makes use of link attribute advertisements, examples of which are listed in Section 6.

In recent years new applications have been introduced that have use cases for many of the link attributes historically used by RSVP-TE. Such applications include Segment Routing (SR) Policy [I-D.ietf-spring-segment-routing-policy] and Loop Free Alternates (LFA) [RFC5286]. This has introduced ambiguity in that if a deployment includes a mix of RSVP-TE support and SR Policy 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 SR Policy. If the topologies are fully congruent this may not be an issue, but any incongruence leads to ambiguity.

An example where this ambiguity causes a problem is a network in that RSVP-TE is enabled only on a subset of its links. A link attribute is advertised for the purpose of another application (e.g. SR Policy) for a link that is not enabled for RSVP-TE. As soon as the router that is an RSVP-TE head-end sees the link attribute being advertised for that link, it assumes RSVP-TE is enabled on that link, even though it is not. If such RSVP-TE head-end router tries to setup an RSVP-TE path via that link, it will result in the path setup failure.

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 that 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 that is easily extensible for the introduction of new applications and new use cases.

2. 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.

3. Requirements Discussion

As stated previously, evolution of use cases for link attributes can be expected to continue. Therefore, any discussion of existing use cases is limited to requirements that are known at the time of this writing. However, in order to determine the functionality required beyond what already exists in OSPF, it is only necessary to discuss use cases that 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 Segment Routing (SR). Included among these use cases is SR Policy which is defined in [I-D.ietf-spring-segment-routing-policy]. If both RSVP-TE and SR Policy 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 SR Policy 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 that 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.

4. Existing Advertisement of Link Attributes

There are existing advertisements used in support of RSVP-TE. These advertisements are carried in the OSPFv2 TE Opaque LSA [RFC3630] and OSPFv3 Intra-Area-TE-LSA [RFC5329]. Additional RSVP-TE link attributes have been defined by [RFC4203], [RFC7308] and [RFC7471].

Extended Link Opaque LSAs as defined in [RFC7684] for OSPFv2 and Extended Router-LSAs [RFC8362] for OSPFv3 are used to advertise link attributes that are used by applications other than RSVP-TE or GMPLS [RFC4203]. These LSAs were defined as a generic containers for distribution of the extended link attributes.

5. Advertisement of Link Attributes

This section outlines the solution for advertising link attributes originally defined for RSVP-TE or GMPLS when they are used for other applications.

5.1. OSPFv2 Extended Link Opaque LSA and OSPFv3 E-Router-LSA

Advantages of Extended Link Opaque LSAs as defined in [RFC7684] for OSPFv2 and Extended Router-LSAs [RFC8362] for OSPFv3 with respect to advertisement of link attributes originally defined for RSVP-TE when used in packet networks and in GMPLS:

1. Advertisement of the link attributes does not make the link part of the RSVP-TE topology. It avoids any conflicts and is fully compatible with [RFC3630] and [RFC5329].

2. The OSPFv2 TE Opaque LSA and OSPFv3 Intra-Area-TE-LSA remains truly opaque to OSPFv2 and OSPFv3 as originally defined in [RFC3630] and [RFC5329] respectively. Their contents are not inspected by OSPF, which instead acts as a pure transport.

3. There is a clear distinction between link attributes used by RSVP-TE and link attributes used by other OSPFv2 or OSPFv3 applications.

4. All link attributes that are used by other applications are advertised in a single LSA, the Extended Link Opaque LSA in OSPFv2 or the OSPFv3 E-Router-LSA [RFC8362] in OSPFv3.

The disadvantage of this approach is that in rare cases, the same link attribute is advertised in both the TE Opaque and Extended Link Attribute LSAs in OSPFv2 or the Intra-Area-TE-LSA and E-Router-LSA in OSPFv3.

Extended Link Opaque LSA [RFC7684] and E-Router-LSA [RFC8362] are used to advertise any link attributes used for non-RSVP-TE applications in OSPFv2 or OSPFv3 respectively, including those that have been originally defined for RSVP-TE applications (See Section 7).

TE link attributes used for RSVP-TE/GMPLS continue to use OSPFv2 TE Opaque LSA [RFC3630] and OSPFv3 Intra-Area-TE-LSA [RFC5329].

The format of the link attribute TLVs that have been defined for RSVP-TE applications will be kept unchanged even when they are used for non-RSVP-TE applications. Unique code points are allocated for these link attribute TLVs from the OSPFv2 Extended Link TLV Sub-TLV Registry [RFC7684] and from the OSPFv3 Extended-LSA Sub-TLV Registry [RFC8362], as specified in Section 15.

6. Advertisement of Application-Specific Values

To allow advertisement of the application-specific values of the link attribute, a new Application-Specific Link Attributes (ASLA) sub-TLV is defined. The ASLA sub-TLV is a sub-TLV of the OSPFv2 Extended Link TLV [RFC7684] and OSPFv3 Router-Link TLV [RFC8362].

On top of advertising the link attributes for standardized applications, link attributes can be advertised for the purpose of applications that are not standardized. We call such an application a "User Defined Application" or "UDA". These applications are not subject to standardization and are outside of the scope of this specification.

The ASLA sub-TLV is an optional sub-TLV of OSPFv2 Extended Link TLV and OSPFv3 Router-Link TLV. Multiple ASLA sub-TLVs can be present in its parent TLV when different applications want to control different link attributes or when different value of the same attribute needs to be advertised by multiple applications. The ASLA sub-TLV MUST be used for advertisement of the link attributes listed at the end on this section if these are advertised inside OSPFv2 Extended Link TLV and OSPFv3 Router-Link TLV. It has the following format:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |              Type             |             Length            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  SABM Length  | UDABM Length  |            Reserved           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |           Standard Application Identifier Bit Mask             |
   +-                                                             -+
   |                              ...                              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |           User Defined Application Identifier Bit Mask        |
   +-                                                             -+
   |                              ...                              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                      Link Attribute sub-sub-TLVs              |
   +-                                                             -+
   |                              ...                              |
```

where:

Type: 10 (OSPFv2), 11 (OSPFv3)

Length: variable

SABM Length: Standard Application Identifier Bit Mask Length in octets. The value MUST be 0, 4 or 8. If the Standard Application Bit Mask is not present, the Standard Application Bit Mask Length MUST be set to 0.

UDABM Length: User Defined Application Identifier Bit Mask Length in octets. The value MUST be 0, 4 or 8. If the User Defined Application Bit Mask is not present, the User Defined Application Bit Mask Length MUST be set to 0.

Standard Application Identifier Bit Mask: Optional set of bits, where each bit represents a single standard application. Bits are defined in the Link Attribute Application Identifier Registry, which has been defined in [I-D.ietf-isis-te-app]. Current assignments are repeated here for informational purpose:

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

Bit-0 (R-bit): RSVP-TE

Bit-1 (S-bit): Segment Routing Policy

Bit-2 (F-bit): Loop Free Alternate (LFA). Includes all LFA types

User Defined Application Identifier Bit Mask: Optional set of bits, where each bit represents a single user defined application.

If the SABM or UDABM length is other than 0, 4, or 8, the ASLA sub-TLV MUST be ignored by the receiver.

Standard Application Identifier Bits are defined/sent starting with Bit 0. Undefined bits that are transmitted 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. Bits that are not supported by an implementation MUST be ignored on receipt.

User Defined Application Identifier Bits have no relationship to Standard Application Identifier 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. Undefined bits which are transmitted 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. Bits that are not supported by an implementation MUST be ignored on receipt.

If the link attribute advertisement is intended to be only used by a specific set of applications, corresponding Bit Masks MUST be present and application-specific bit(s) MUST be set for all applications that use the link attributes advertised in the ASLA sub-TLV.

Application Bit Masks apply to all link attributes that support application-specific values and are advertised in the ASLA sub-TLV.

The advantage of not making the Application Bit Masks part of the attribute advertisement itself is that the format of any previously defined link attributes can be kept and reused when advertising them in the ASLA sub-TLV.

If the same attribute is advertised in more than one ASLA sub-TLVs with the application listed in the Application Bit Masks, the application SHOULD use the first instance of advertisement and ignore any subsequent advertisements of that attribute.

If link attributes are advertised with zero length Application Identifier Bit Masks for both standard applications and user defined applications, then any Standard Application and/or any User Defined

Application is permitted to use that set of link attributes. If support for a new application is introduced on any node in a network in the presence of such advertisements, these advertisements are permitted to 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.

An application-specific advertisement (Application Identifier Bit Mask with a matching Application Identifier Bit set) for an attribute MUST always be preferred over the advertisement of the same attribute with the zero length Application Identifier Bit Masks for both standard applications and user defined applications on the same link.

This document defines the initial set of link attributes that MUST use the ASLA sub-TLV if advertised in the OSPFv2 Extended Link TLV or in the OSPFv3 Router-Link TLV. Documents which define new link attributes MUST state whether the new attributes support application-specific values and as such are advertised in an ASLA sub-TLV. The standard link attributes that are advertised in ASLA sub-TLVs are:

- Shared Risk Link Group [RFC4203]

- Unidirectional Link Delay [RFC7471]

- Min/Max Unidirectional Link Delay [RFC7471]

- Unidirectional Delay Variation [RFC7471]

- Unidirectional Link Loss [RFC7471]

- Unidirectional Residual Bandwidth [RFC7471]

- Unidirectional Available Bandwidth [RFC7471]

- Unidirectional Utilized Bandwidth [RFC7471]

- Administrative Group [RFC3630]

- Extended Administrative Group [RFC7308]

- TE Metric [RFC3630]

7. Reused TE link attributes

This section defines the use case and indicates the code points (Section 15) from the OSPFv2 Extended Link TLV Sub-TLV Registry and

OSPFv3 Extended-LSA Sub-TLV Registry for some of the link attributes that have been originally defined for RSVP-TE or GMPLS.

7.1. Shared Risk Link Group (SRLG)

The SRLG of a link can be used in OSPF calculated IPFRR (IP Fast Reroute) [RFC5714] to compute a backup path that does not share any SRLG group with the protected link.

To advertise the SRLG of the link in the OSPFv2 Extended Link TLV, the same format for the sub-TLV defined in section 1.3 of [RFC4203] is used and TLV type 11 is used. Similarly, for OSPFv3 to advertise the SRLG in the OSPFv3 Router-Link TLV, TLV type 12 is used.

7.2. Extended Metrics

[RFC3630] defines several link bandwidth types. [RFC7471] defines extended link metrics that are based on link bandwidth, delay and loss characteristics. All of these can be used to compute primary and backup paths within an OSPF area to satisfy requirements for bandwidth, delay (nominal or worst case) or loss.

To advertise extended link metrics in the OSPFv2 Extended Link TLV, the same format for the sub-TLVs defined in [RFC7471] is used with the following TLV types:

12 - Unidirectional Link Delay

13 - Min/Max Unidirectional Link Delay

14 - Unidirectional Delay Variation

15 - Unidirectional Link Loss

16 - Unidirectional Residual Bandwidth

17 - Unidirectional Available Bandwidth

18 - Unidirectional Utilized Bandwidth

To advertise extended link metrics in the OSPFv3 Extended-LSA Router-Link TLV, the same format for the sub-TLVs defined in [RFC7471] is used with the following TLV types:

13 - Unidirectional Link Delay

14 - Min/Max Unidirectional Link Delay

15 - Unidirectional Delay Variation

16 - Unidirectional Link Loss

17 - Unidirectional Residual Bandwidth

18 - Unidirectional Available Bandwidth

19 - Unidirectional Utilized Bandwidth

7.3. Administrative Group

[RFC3630] and [RFC7308] define the Administrative Group and Extended Administrative Group sub-TLVs respectively.

To advertise the Administrative Group and Extended Administrative Group in the OSPFv2 Extended Link TLV, the same format for the sub-TLVs defined in [RFC3630] and [RFC7308] is used with the following TLV types:

19 - Administrative Group

20 - Extended Administrative Group

To advertise Administrative Group and Extended Administrative Group in the OSPFv3 Router-Link TLV, the same format for the sub-TLVs defined in [RFC3630] and [RFC7308] is used with the following TLV types:

20 - Administrative Group

21 - Extended Administrative Group

7.4. Traffic Engineering Metric

[RFC3630] defines Traffic Engineering Metric.

To advertise the Traffic Engineering Metric in the OSPFv2 Extended Link TLV, the same format for the sub-TLV defined in section 2.5.5 of [RFC3630] is used and TLV type 22 is used. Similarly, for OSPFv3 to advertise the Traffic Engineering Metric in the OSPFv3 Router-Link TLV, TLV type 22 is used.

8. Maximum Link Bandwidth

Maximum link bandwidth is an application independent attribute of the link that is defined in [RFC3630]. Because it is an application independent attribute, it MUST NOT be advertised in ASLA sub-TLV.

Instead, it MAY be advertised as a sub-TLV of the Extended Link Opaque LSA Extended Link TLV in OSPFv2 [RFC7684] or sub-TLV of OSPFv3 E-Router-LSA Router-Link TLV in OSPFv3 [RFC8362].

To advertise the Maximum link bandwidth in the OSPFv2 Extended Link TLV, the same format for sub-TLV defined in [RFC3630] is used with TLV type 23.

To advertise the Maximum link bandwidth in the OSPFv3 Router-Link TLV, the same format for sub-TLV defined in [RFC3630] is used with TLV type 23.

9. Considerations for Extended TE Metrics

[RFC7471] defines a number of dynamic performance metrics associated with a link. It is conceivable that such metrics could be measured specific to traffic associated with a specific application. Therefore this document includes support for advertising these link attributes specific to a given application. However, in practice it may well be more practical to have these metrics reflect the performance of all traffic on the link regardless of application. In such cases, advertisements for these attributes can be associated with all of the applications utilizing that link. This can be done either by explicitly specifying the applications in the Application Identifier Bit Mask or by using a zero length Application Identifier Bit Mask.

10. Local Interface IPv6 Address Sub-TLV

The Local Interface IPv6 Address Sub-TLV is an application independent attribute of the link that is defined in [RFC5329]. Because it is an application independent attribute, it MUST NOT be advertised in the ASLA sub-TLV. Instead, it MAY be advertised as a sub-TLV of the OSPFv3 E-Router-LSA Router-Link TLV [RFC8362].

To advertise the Local Interface IPv6 Address Sub-TLV in the OSPFv3 Router-Link TLV, the same format for sub-TLV defined in [RFC5329] is used with TLV type 24.

11. Remote Interface IPv6 Address Sub-TLV

The Remote Interface IPv6 Address Sub-TLV is an application independent attribute of the link that is defined in [RFC5329]. Because it is an application independent attribute, it MUST NOT be advertised in the ASLA sub-TLV. Instead, it MAY be advertised as a sub-TLV of the OSPFv3 E-Router-LSA Router-Link TLV [RFC8362].

To advertise the Remote Interface IPv6 Address Sub-TLV in the OSPFv3 Router-Link TLV, the same format for sub-TLV defined in [RFC5329] is used with TLV type 25.

12. Attribute Advertisements and Enablement

This document defines extensions to support the advertisement of application-specific link attributes.

There are applications where the application enablement on the link is relevant - e.g., RSVP-TE - one needs to make sure that RSVP is enabled on the link before sending a RSVP-TE signaling message over it.

There are applications where the enablement of the application on the link is irrelevant and has nothing to do with the fact that some link attributes are advertised for the purpose of such application. An example of this is LFA.

Whether the presence of link attribute advertisements for a given application indicates that the application is enabled on that link depends upon the application. Similarly, whether the absence of link attribute advertisements indicates that the application is not enabled depends upon the application.

In the case of RSVP-TE, the advertisement of application-specific link attributes has no implication of RSVP-TE being enabled on that link. The RSVP-TE enablement is solely derived from the information carried in the OSPFv2 TE Opaque LSA [RFC3630] and OSPFv3 Intra-Area-TE-LSA [RFC5329].

In the case of SR Policy, advertisement of application-specific link attributes does not indicate enablement of SR Policy. The advertisements are only used to support constraints that may be applied when specifying an explicit path. SR Policy is implicitly enabled on all links that are part of the Segment Routing enabled topology independent of the existence of link attribute advertisements

In the case of LFA, advertisement of application-specific link attributes does not indicate enablement of LFA on that link. Enablement is controlled by local configuration.

If, in the future, additional standard applications are defined to use this mechanism, the specification defining this use MUST define the relationship between application-specific link attribute advertisements and enablement for that application.

This document allows the advertisement of application-specific link attributes with no application identifiers i.e., both the Standard Application Identifier Bit Mask and the User Defined Application Identifier Bit Mask are not present (See Section 6). This supports the use of the link attribute by any application. In the presence of an application where the advertisement of link attribute advertisements is used to infer the enablement of an application on that link (e.g., RSVP-TE), the absence of the application identifier leaves ambiguous whether that application is enabled on such a link. This needs to be considered when making use of the "any application" encoding.

13. Deployment Considerations

13.1. Use of Legacy RSVP-TE LSA Advertisements

Bit Identifiers for Standard Applications are defined in Section 6. All of the identifiers defined in this document are associated with applications that were already deployed in some networks prior to the writing of this document. Therefore, such applications have been deployed using the RSVP-TE LSA advertisements. The Standard Applications defined in this document may continue to use RSVP-TE LSA advertisements for a given link so long as at least one of the following conditions is true:

> The application is RSVP-TE
>
> The application is SR Policy or LFA and RSVP-TE is not deployed anywhere in the network
>
> The application is SR Policy or LFA, RSVP-TE is deployed in the network, and both the set of links on which SR Policy and/or LFA advertisements are required and the attribute values used by SR Policy and/or LFA on all such links is fully congruent with the links and attribute values used by RSVP-TE

Under the conditions defined above, implementations that support the extensions defined in this document have the choice of using RSVP-TE LSA advertisements or application-specific advertisements in support of SR Policy and/or LFA. This will require implementations to provide controls specifying which type of advertisements are to be sent/ processed on receive for these applications. Further discussion of the associated issues can be found in Section 13.2.

New applications that future documents define to make use of the advertisements defined in this document MUST NOT make use of RSVP-TE LSA advertisements. This simplifies deployment of new applications by eliminating the need to support multiple ways to advertise attributes for the new applications.

13.2. Interoperability, Backwards Compatibility and Migration Concerns

Existing deployments of RSVP-TE, SR Policy, and/or LFA utilize the legacy advertisements listed in Section 4. 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 in the presence of routers that do not support these extensions need to be able to interoperate with the use of legacy advertisements by the legacy routers. The following sub-sections discuss interoperability and backwards compatibility concerns for a number of deployment scenarios.

13.2.1. 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 for RSVP-TE. Attributes for applications other than RSVP-TE MUST be advertised using application-specific advertisements. This results in duplicate advertisements for those attributes.

13.2.2. Multiple Applications: Some Attributes Not Shared with 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, interoperability is achieved by using legacy advertisements for RSVP-TE. Attributes for applications other than RSVP-TE MUST be advertised using application-specific advertisements. In cases where some link attributes are shared with RSVP-TE, this requires duplicate advertisements for those attributes

13.2.3. Interoperability with Legacy Routers

For the applications defined in this document, routers that do not support the extensions defined in this document will send and receive only legacy link attribute advertisements. So long as there is any legacy router in the network that has any of the applications enabled, all routers MUST continue to advertise link attributes using legacy advertisements. In addition, the link attribute values associated with the set of applications supported by legacy routers (RSVP-TE, SR Policy, and/or LFA) are always shared since legacy routers have no way of advertising or processing application-specific values. Once all legacy routers have been upgraded, migration from legacy advertisements to application specific advertisements can be achieved via the following steps:

1)Send new application-specific advertisements while continuing to advertise using the legacy advertisement (all advertisements are then duplicated). Receiving routers continue to use legacy advertisements.

2)Enable the use of the application-specific advertisements on all routers

3)Keep legacy advertisements if needed for RSVP-TE purposes.

When the migration is complete, it then becomes possible to advertise incongruent values per application on a given link.

Documents defining new applications that make use of the application-specific advertisements defined in this document MUST discuss interoperability and backwards compatibility issues that could occur in the presence of routers that do not support the new application.

13.2.4. Use of Application-Specific Advertisements for RSVP-TE

The extensions defined in this document support RSVP-TE as one of the supported applications. It is however RECOMMENDED to advertise all link-attributes for RSVP-TE in the existing OSPFv2 TE Opaque LSA [RFC3630] and OSPFv3 Intra-Area-TE-LSA [RFC5329] to maintain backward compatibility. RSVP-TE can eventually utilize the application-specific advertisements for newly defined link attributes, that are defined as application-specific.

Link attributes that are not allowed to be advertised in the ASLA Sub-TLV, such as Maximum Reservable Link Bandwidth and Unreserved Bandwidth MUST use the OSPFv2 TE Opaque LSA [RFC3630] and OSPFv3 Intra-Area-TE-LSA [RFC5329] and MUST NOT be advertised in ASLA Sub-TLV.

14. Security Considerations

Existing security extensions as described in [RFC2328], [RFC5340] and [RFC8362] apply to extensions defined in this document. While OSPF is under a single administrative domain, there can be deployments where potential attackers have access to one or more networks in the OSPF routing domain. In these deployments, stronger authentication mechanisms such as those specified in [RFC5709], [RFC7474], [RFC4552] or [RFC7166] SHOULD be used.

Implementations must assure that malformed TLV and Sub-TLV defined in this document are detected and do not provide a vulnerability for attackers to crash the OSPF router or routing process. Reception of a malformed TLV or Sub-TLV SHOULD be counted and/or logged for further analysis. Logging of malformed TLVs and Sub-TLVs SHOULD be rate-limited to prevent a Denial of Service (DoS) attack (distributed or otherwise) from overloading the OSPF control plane.

This document defines a new way to advertise link attributes. Tampering with the information defined in this document may have an effect on applications using it, including impacting Traffic Engineering that uses various link attributes for its path computation. This is similar in nature to the impacts associated with (for example) [RFC3630]. As the advertisements defined in this document limit the scope to specific applications, the impact of tampering is similarly limited in scope.

15. IANA Considerations

This specifications updates two existing registries:

- OSPFv2 Extended Link TLV Sub-TLVs Registry

- OSPFv3 Extended-LSA Sub-TLV Registry

New values are allocated using the IETF Review procedure as described in [RFC5226].

15.1. OSPFv2

The OSPFv2 Extended Link TLV Sub-TLVs Registry [RFC7684] defines sub-TLVs at any level of nesting for OSPFv2 Extended Link TLVs. IANA has assigned the following Sub-TLV types from the OSPFv2 Extended Link TLV Sub-TLVs Registry:

10 - Application-Specific Link Attributes

11 - Shared Risk Link Group

12 - Unidirectional Link Delay

13 - Min/Max Unidirectional Link Delay

14 - Unidirectional Delay Variation

15 - Unidirectional Link Loss

16 - Unidirectional Residual Bandwidth

17 - Unidirectional Available Bandwidth

18 - Unidirectional Utilized Bandwidth

19 - Administrative Group

20 - Extended Administrative Group

22 - TE Metric

23 - Maximum Link Bandwidth

15.2. OSPFv3

The OSPFv3 Extended-LSA Sub-TLV Registry [RFC8362] defines sub-TLVs at any level of nesting for OSPFv3 Extended LSAs. IANA has assigned the following Sub-TLV types from the OSPFv3 Extended-LSA Sub-TLV Registry:

11 - Application-Specific Link Attributes

12 - Shared Risk Link Group

13 - Unidirectional Link Delay

14 - Min/Max Unidirectional Link Delay

15 - Unidirectional Delay Variation

16 - Unidirectional Link Loss

17 - Unidirectional Residual Bandwidth

18 - Unidirectional Available Bandwidth

19 - Unidirectional Utilized Bandwidth

20 - Administrative Group

21 - Extended Administrative Group

22 - TE Metric

23 - Maximum Link Bandwidth

24 - Local Interface IPv6 Address Sub-TLV

25 - Remote Interface IPv6 Address Sub-TLV

16. Contributors

The following people contributed to the content of this document and should be considered as co-authors:

Acee Lindem
Cisco Systems
301 Midenhall Way
Cary, NC 27513
USA

Email: acee@cisco.com

Ketan Talaulikar
Cisco Systems, Inc.
India

Email: ketant@cisco.com

Hannes Gredler
RtBrick Inc.
Austria

Email: hannes@rtbrick.com

17. Acknowledgments

Thanks to Chris Bowers for his review and comments.

Thanks to Alvaro Retana for his detailed review and comments.

18. References

18.1. Normative References

[I-D.ietf-isis-te-app]
Ginsberg, L., Psenak, P., Previdi, S., Henderickx, W., and J. Drake, "IS-IS Application-Specific Link Attributes", draft-ietf-isis-te-app-19 (work in progress), June 2020.

[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>.

[RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, DOI 10.17487/RFC2328, April 1998, <https://www.rfc-editor.org/info/rfc2328>.

[RFC3630] Katz, D., Kompella, K., and D. Yeung, "Traffic Engineering (TE) Extensions to OSPF Version 2", RFC 3630, DOI 10.17487/RFC3630, September 2003, <https://www.rfc-editor.org/info/rfc3630>.

[RFC4203] Kompella, K., Ed. and Y. Rekhter, Ed., "OSPF Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)", RFC 4203, DOI 10.17487/RFC4203, October 2005, <https://www.rfc-editor.org/info/rfc4203>.

[RFC5329] Ishiguro, K., Manral, V., Davey, A., and A. Lindem, Ed., "Traffic Engineering Extensions to OSPF Version 3", RFC 5329, DOI 10.17487/RFC5329, September 2008, <https://www.rfc-editor.org/info/rfc5329>.

[RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, DOI 10.17487/RFC5340, July 2008, <https://www.rfc-editor.org/info/rfc5340>.

[RFC7308] Osborne, E., "Extended Administrative Groups in MPLS Traffic Engineering (MPLS-TE)", RFC 7308, DOI 10.17487/RFC7308, July 2014, <https://www.rfc-editor.org/info/rfc7308>.

[RFC7471] Giacalone, S., Ward, D., Drake, J., Atlas, A., and S. Previdi, "OSPF Traffic Engineering (TE) Metric Extensions", RFC 7471, DOI 10.17487/RFC7471, March 2015, <https://www.rfc-editor.org/info/rfc7471>.

[RFC7684] Psenak, P., Gredler, H., Shakir, R., Henderickx, W., Tantsura, J., and A. Lindem, "OSPFv2 Prefix/Link Attribute Advertisement", RFC 7684, DOI 10.17487/RFC7684, November 2015, <https://www.rfc-editor.org/info/rfc7684>.

[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>.

[RFC8362] Lindem, A., Roy, A., Goethals, D., Reddy Vallem, V., and F. Baker, "OSPFv3 Link State Advertisement (LSA) Extensibility", RFC 8362, DOI 10.17487/RFC8362, April 2018, <https://www.rfc-editor.org/info/rfc8362>.

18.2. Informative References

[I-D.ietf-spring-segment-routing-policy]
Filsfils, C., Sivabalan, S., Voyer, D., Bogdanov, A., and P. Mattes, "Segment Routing Policy Architecture", draft-ietf-spring-segment-routing-policy-07 (work in progress), May 2020.

[RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, DOI 10.17487/RFC3209, December 2001, <https://www.rfc-editor.org/info/rfc3209>.

[RFC4552] Gupta, M. and N. Melam, "Authentication/Confidentiality for OSPFv3", RFC 4552, DOI 10.17487/RFC4552, June 2006, <https://www.rfc-editor.org/info/rfc4552>.

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

[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>.

[RFC5709] Bhatia, M., Manral, V., Fanto, M., White, R., Barnes, M., Li, T., and R. Atkinson, "OSPFv2 HMAC-SHA Cryptographic Authentication", RFC 5709, DOI 10.17487/RFC5709, October 2009, <https://www.rfc-editor.org/info/rfc5709>.

[RFC5714] Shand, M. and S. Bryant, "IP Fast Reroute Framework", RFC 5714, DOI 10.17487/RFC5714, January 2010, <https://www.rfc-editor.org/info/rfc5714>.

[RFC7166] Bhatia, M., Manral, V., and A. Lindem, "Supporting Authentication Trailer for OSPFv3", RFC 7166, DOI 10.17487/RFC7166, March 2014, <https://www.rfc-editor.org/info/rfc7166>.

[RFC7474] Bhatia, M., Hartman, S., Zhang, D., and A. Lindem, Ed., "Security Extension for OSPFv2 When Using Manual Key Management", RFC 7474, DOI 10.17487/RFC7474, April 2015, <https://www.rfc-editor.org/info/rfc7474>.

Authors' Addresses

Peter Psenak (editor)
Cisco Systems
Eurovea Centre, Central 3
Pribinova Street 10
Bratislava 81109
Slovakia

Email: ppsenak@cisco.com

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

Email: ginsberg@cisco.com

Wim Henderickx
Nokia
Copernicuslaan 50
Antwerp, 2018 94089
Belgium

Email: wim.henderickx@nokia.com

Jeff Tantsura
Apstra
US

Email: jefftant.ietf@gmail.com

John Drake
Juniper Networks
1194 N. Mathilda Ave
Sunnyvale, California 94089
USA

Email: jdrake@juniper.net

Internet
Internet-Draft
Intended status: Standards Track
Expires: April 19, 2020

D. Yeung
Arrcus
Y. Qu
Futurewei
J. Zhang
Juniper Networks
I. Chen
The MITRE Corporation
A. Lindem
Cisco Systems

October 17, 2019

YANG Data Model for OSPF Protocol
draft-ietf-ospf-yang-29

Abstract

This document defines a YANG data model that can be used to configure and manage OSPF. The model is based on YANG 1.1 as defined in RFC 7950 and conforms to the Network Management Datastore Architecture (NMDA) as described in RFC 8342.

Status of This Memo

Copyright Notice

Table of Contents

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], RESTCONF [RFC8040], and other Network Management protocols. 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 OSPF and it is an augmentation to the core routing data model. It fully conforms to the Network Management Datastore Architecture (NMDA) [RFC8342]. A core routing data model is defined in [RFC8349], and it provides the basis for the development of data models for routing protocols. The interface data model is defined in [RFC8343] and is used for referencing interfaces from the routing protocol. The key-chain data model used for OSPF authentication is defined in [RFC8177] and provides both a reference to configured key-chains and an enumeration of cryptographic algorithms.

Both OSPFv2 [RFC2328] and OSPFv3 [RFC5340] are supported. In addition to the core OSPF protocol, features described in other OSPF RFCs are also supported. These includes demand circuit [RFC1793], traffic engineering [RFC3630], multiple address family [RFC5838], graceful restart [RFC3623] [RFC5187], NSSA [RFC3101], and OSPFv2 or OSPFv3 as a PE-CE Protocol [RFC4577], [RFC6565]. These non-core features are optional in the OSPF data model.

1.1. Requirements Language

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

1.2. Tree Diagrams

This document uses the graphical representation of data models defined in [RFC8340].

2. Design of Data Model

Although the basis of OSPF configuration elements like routers, areas, and interfaces remains the same, the detailed configuration model varies among router vendors. Differences are observed in terms of how the protocol instance is tied to the routing domain and how multiple protocol instances are be instantiated among others.

The goal of this document is to define a data model that provides a common user interface to the OSPFv2 and OSPFv3 protocols. There is very little information that is designated as "mandatory", providing freedom for vendors to adapt this data model to their respective product implementations.

2.1. OSPF Operational State

The OSPF operational state is included in the same tree as OSPF configuration consistent with the Network Management Datastore Architecture [RFC8342]. Consequently, only the routing container in the ietf-routing model [RFC8349] is augmented. The routing-state container is not augmented.

2.2. Overview

The OSPF YANG module defined in this document has all the common building blocks for the OSPF protocol.

The OSPF YANG module augments the /routing/control-plane-protocols/ control-plane-protocol path defined in the ietf-routing module. The ietf-ospf model defines a single instance of OSPF which may be instantiated as an OSPFv2 or OSPFv3 instance. Multiple instances are instantiated as multiple control-plane protocols instances.

```
module: ietf-ospf
  augment /rt:routing/rt:control-plane-protocols/
           rt:control-plane-protocol:
    +--rw ospf
          .
          .
       +--rw af?                      identityref
          .
          .
       +--rw areas
       |  +--rw area* [area-id]
       |     +--rw area-id                  area-id-type
       |        .
       |        .
       |     +--rw virtual-links
       |     |  +--rw virtual-link* [transit-area-id router-id]
       |     |     .
       |     |     .
       |     +--rw sham-links {pe-ce-protocol}?
       |     |  +--rw sham-link* [local-id remote-id]
       |     |     .
       |     |     .
       |     +--rw interfaces
       |        +--rw interface* [name]
       |           .
       |           .
       +--rw topologies {multi-topology}?
          +--rw topology* [name]
             .
             .
```

The ospf container includes one OSPF protocol instance. The instance includes OSPF router level configuration and operational state. Each OSPF instance maps to a control-plane-protcol instance as defined in [RFC8349].

The area and area/interface containers define the OSPF configuration and operational state for OSPF areas and interfaces respectively.

The topologies container defines the OSPF configuration and operational state for OSPF topologies when the multi-topology feature is supported.

2.3. OSPFv2 and OSPFv3

The data model defined herein supports both OSPFv2 and OSPFv3.

The field 'version' is used to indicate the OSPF version and is mandatory. Based on the configured version, the data model varies to accommodate the differences between OSPFv2 and OSPFv3.

2.4. Optional Features

Optional features are beyond the basic OSPF configuration and it is the responsibility of each vendor to decide whether to support a given feature on a particular device.

This model defines the following optional features:

1. multi-topology: Support Multi-Topology Routing (MTR) [RFC4915].

2. multi-area-adj: Support OSPF multi-area adjacency [RFC5185].

3. explicit-router-id: Support explicit per-instance Router-ID specification.

4. demand-circuit: Support OSPF demand circuits [RFC1793].

5. mtu-ignore: Support disabling OSPF Database Description packet MTU mismatch checking specified in section 10.6 of [RFC2328].

6. lls: Support OSPF link-local signaling (LLS) [RFC5613].

7. prefix-suppression: Support OSPF prefix advertisement suppression [RFC6860].

8. ttl-security: Support OSPF Time to Live (TTL) security check support [RFC5082].

9. nsr: Support OSPF Non-Stop Routing (NSR). The OSPF NSR feature allows a router with redundant control-plane capability (e.g., dual Route-Processor (RP) cards) to maintain its state and adjacencies during planned and unplanned control-plane processing restarts. It differs from graceful-restart or Non-

Stop Forwarding (NSF) in that no protocol signaling or assistance from adjacent OSPF neighbors is required to recover control-plane state.

10. graceful-restart: Support Graceful OSPF Restart [RFC3623], [RFC5187].

11. auto-cost: Support OSPF interface cost calculation according to reference bandwidth [RFC2328].

12. max-ecmp: Support configuration of the maximum number of Equal-Cost Multi-Path (ECMP) paths.

13. max-lsa: Support configuration of the maximum number of LSAs the OSPF instance will accept [RFC1765].

14. te-rid: Support configuration of the Traffic Engineering (TE) Router-ID, i.e., the Router Address described in Section 2.4.1 of [RFC3630] or the Router IPv6 Address TLV described in Section 3 of [RFC5329].

15. ldp-igp-sync: Support LDP IGP synchronization [RFC5443].

16. ospfv2-authentication-trailer: Support OSPFv2 Authentication trailer as specified in [RFC5709] or [RFC7474].

17. ospfv3-authentication-ipsec: Support IPsec for OSPFv3 authentication [RFC4552].

18. ospfv3-authentication-trailer: Support OSPFv3 Authentication trailer as specified in [RFC7166].

19. fast-reroute: Support IP Fast Reroute (IP-FRR) [RFC5714].

20. node-flag: Support node-flag for OSPF prefixes. [RFC7684].

21. node-tag: Support node admin tag for OSPF instances [RFC7777].

22. lfa: Support Loop-Free Alternates (LFAs) [RFC5286].

23. remote-lfa: Support Remote Loop-Free Alternates (R-LFA) [RFC7490].

24. stub-router: Support RFC 6987 OSPF Stub Router advertisement [RFC6987].

25. pe-ce-protocol: Support OSPF as a PE-CE protocol [RFC4577], [RFC6565].

26. ietf-spf-delay: Support IETF SPF delay algorithm [RFC8405].

27. bfd: Support BFD detection of OSPF neighbor reachability [RFC5880], [RFC5881], and [I-D.ietf-bfd-yang].

28. hybrid-interface: Support OSPF Hybrid Broadcast and Point-to-Point Interfaces [RFC6845].

It is expected that vendors will support additional features through vendor-specific augmentations.

2.5. OSPF Router Configuration/Operational State

The ospf container is the top-level container in this data model. It represents an OSPF protocol instance and contains the router level configuration and operational state. The operational state includes the instance statistics, IETF SPF delay statistics, AS-Scoped Link State Database, local RIB, SPF Log, and the LSA log.

```
   module: ietf-ospf
     augment /rt:routing/rt:control-plane-protocols/
               rt:control-plane-protocol:
       +--rw ospf
          .
          .
          +--rw af                     iana-rt-types:address-family
          +--rw enable?                boolean
          +--rw explicit-router-id?   rt-types:router-id
          |                            {explicit-router-id}?
          +--rw preference
          |  +--rw (scope)?
          |     +--:(single-value)
          |     |  +--rw all?          uint8
          |     +--:(multi-values)
          |        +--rw (granularity)?
          |        |  +--:(detail)
          |        |  |  +--rw intra-area?   uint8
          |        |  |  +--rw inter-area?   uint8
          |        |  +--:(coarse)
          |        |     +--rw internal?     uint8
          |        +--rw external?     uint8
          +--rw nsr {nsr}?
          |  +--rw enable?   boolean
          +--rw graceful-restart {graceful-restart}?
          |  +--rw enable?                        boolean
          |  +--rw helper-enable?                 boolean
          |  +--rw restart-interval?              uint16
          |  +--rw helper-strict-lsa-checking?   boolean
          +--rw auto-cost {auto-cost}?
          |  +--rw enable?                 boolean
          |  +--rw reference-bandwidth?   uint32
          +--rw spf-control
          |  +--rw paths?            uint16 {max-ecmp}?
          |  +--rw ietf-spf-delay {ietf-spf-delay}?
          |     +--rw initial-delay?   uint16
          |     +--rw short-delay?     uint16
          |     +--rw long-delay?      uint16
          |     +--rw hold-down?       uint16
          |     +--rw time-to-learn?   uint16
          |     +--ro current-state?             enumeration
          |     +--ro remaining-time-to-learn?   uint16
          |     +--ro remaining-hold-down?       uint16
          |     +--ro last-event-received?       yang:timestamp
          |     +--ro next-spf-time?             yang:timestamp
          |     +--ro last-spf-time?             yang:timestamp
          +--rw database-control
          |  +--rw max-lsa?   uint32 {max-lsa}?
          +--rw stub-router {stub-router}?
          |  +--rw (trigger)?
          |     +--:(always)
          |        +--rw always!
          +--rw mpls
          |  +--rw te-rid {te-rid}?
          |  |  +--rw ipv4-router-id?   inet:ipv4-address
          |  |  +--rw ipv6-router-id?   inet:ipv6-address
          |  +--rw ldp
          |     +--rw igp-sync?   boolean {ldp-igp-sync}?
          +--rw fast-reroute {fast-reroute}?
          |  +--rw lfa {lfa}?
          +--ro protected-routes
          |  +--ro af-stats* [af prefix alternate]
          |     +--ro af                    iana-rt-types:address-family
          |     +--ro prefix                string
          |     +--ro alternate             string
          |     +--ro alternate-type?       enumeration
          |     +--ro best?                 boolean
          |     +--ro non-best-reason?      string
          |     +--ro protection-available? bits
          |     +--ro alternate-metric1?    uint32
          |     +--ro alternate-metric2?    uint32
          |     +--ro alternate-metric3?    uint32
          +--ro unprotected-routes
          |  +--ro af-stats* [af prefix]
          |     +--ro af                    iana-rt-types:address-family
          |     +--ro prefix                string
          +--ro protection-statistics* [frr-protection-method]
          |  +--ro frr-protection-method string
          |  +--ro af-stats* [af]
          |     +--ro af                   iana-rt-types:address-family
          |     +--ro total-routes?         uint32
          |     +--ro unprotected-routes?   uint32
          |     +--ro protected-routes?     uint32
          |     +--ro linkprotected-routes? uint32
          |     +--ro nodeprotected-routes? uint32
          +--rw node-tags {node-tag}?
          |  +--rw node-tag* [tag]
          |     +--rw tag      uint32
          +--ro router-id?
          +--ro local-rib
          |  +--ro route* [prefix]
          |     +--ro prefix        inet:ip-prefix
          |     +--ro next-hops
          |     |  +--ro next-hop* [next-hop]
          |     |     +--ro outgoing-interface?   if:interface-ref
          |     |     +--ro next-hop              inet:ip-address
          |     +--ro metric?       uint32
          |     +--ro route-type?   route-type
          |     +--ro route-tag?    uint32
          +--ro statistics
          |  +--ro discontinuity-time         yang:date-and-time
          |  +--ro originate-new-lsa-count?   yang:counter32
          |  +--ro rx-new-lsas-count?         yang:counter32
          |  +--ro as-scope-lsa-count?        yang:gauge32
          |  +--ro as-scope-lsa-chksum-sum?   uint32
          |  +--ro database
          |     +--ro as-scope-lsa-type*
          |        +--ro lsa-type?        uint16
          |        +--ro lsa-count?       yang:gauge32
          |        +--ro lsa-cksum-sum?   int32
          +--ro database
          |  +--ro as-scope-lsa-type* [lsa-type]
          |     +--ro as-scope-lsas
          |        +--ro as-scope-lsa* [lsa-id adv-router]
          |           +--ro lsa-id               union
          |           +--ro adv-router           inet:ipv4-address
          |           +--ro decoded-completed?   boolean
          |           +--ro raw-data?            yang:hex-string
          |           +--ro (version)?
          |              +--:(ospfv2)
          |              |  +--ro ospfv2
          .              .
          .              .
          |              +--:(ospfv3)
          |                 +--ro ospfv3
          .
          .
          +--ro spf-log
          |  +--ro event* [id]
          |     +--ro id                      uint32
          |     +--ro spf-type?               enumeration
          |     +--ro schedule-timestamp?     yang:timestamp
          |     +--ro start-timestamp?        yang:timestamp
          |     +--ro end-timestamp?          yang:timestamp
          |     +--ro trigger-lsa*
          |        +--ro area-id?      area-id-type
          |        +--ro link-id?      union
          |        +--ro type?         uint16
          |        +--ro lsa-id?       yang:dotted-quad
          |        +--ro adv-router?   yang:dotted-quad
          |        +--ro seq-num?      uint32
          +--ro lsa-log
          |  +--ro event* [id]
          |     +--ro id                      uint32
          |     +--ro lsa
          |     |  +--ro area-id?      area-id-type
          |     |  +--ro link-id?      union
          |     |  +--ro type?         uint16
          |     |  +--ro lsa-id?       yang:dotted-quad
          |     |  +--ro adv-router?   yang:dotted-quad
          |     |  +--ro seq-num?      uint32
          |     +--ro received-timestamp?     yang:timestamp
          |     +--ro reason?                 identityref
          .
          .
```

2.6. OSPF Area Configuration/Operational State

The area container contains OSPF area configuration and the list of interface containers representing all the OSPF interfaces in the area. The area operational state includes the area statistics and the Area Link State Database (LSDB).

```
   module: ietf-ospf
     augment /rt:routing/rt:control-plane-protocols/
             rt:control-plane-protocol:
       +--rw ospf
          .
          .
          +--rw areas
          |  +--rw area* [area-id]
          |     +--rw area-id                     area-id-type
          |     +--rw area-type?                  identityref
          |     +--rw summary?                  boolean
          |     +--rw default-cost?             uint32
          |     +--rw ranges
          |     |  +--rw range* [prefix]
          |     |     +--rw prefix       inet:ip-prefix
          |     |     +--rw advertise?   boolean
          |     |     +--rw cost?        uint24
          |     +--rw topologies {ospf:multi-topology}?
          |     |  +--rw topology* [name]
          |     |     +--rw name  -> ../../../../../../../../
          |     |                    ../../../rt:ribs/rib/name
          |     |     +--rw summary?        boolean
          |     |     +--rw default-cost?   ospf-metric
          |     |     +--rw ranges
          |     |         +--rw range* [prefix]
          |     |            +--rw prefix       inet:ip-prefix
          |     |            +--rw advertise?   boolean
          |     |            +--rw cost?        ospf-metric
          |     +--ro statistics
          |     |  +--ro discontinuity-time           yang:date-and-time
          |     |  +--ro spf-runs-count?              yang:counter32
          |     |  +--ro abr-count?                   yang:gauge32
          |     |  +--ro asbr-count?                  yang:gauge32
          |     |  +--ro ar-nssa-translator-event-count?
          |     |                                     yang:counter32
          |     |  +--ro area-scope-lsa-count?        yang:gauge32
          |     |  +--ro area-scope-lsa-cksum-sum?    int32
          |     |  +--ro database
          |     |     +--ro area-scope-lsa-type*
          |     |        +--ro lsa-type?        uint16
          |     |        +--ro lsa-count?       yang:gauge32
          |     |        +--ro lsa-cksum-sum?   int32
          |     +--ro database
          |     |  +--ro area-scope-lsa-type* [lsa-type]
          |     |     +--ro lsa-type           uint16
          |     |     +--ro area-scope-lsas
          |     |        +--ro area-scope-lsa* [lsa-id adv-router]
          |     |           +--ro lsa-id               union
          .     .           .
          .     .           .
          |     |           +--ro (version)?
          |     |              +--:(ospfv2)
          |     |              |  +--ro ospfv2
          |     |              |     +--ro header
          .     .              .     .
          .     .              .     .
          |     |              |     +--ro body
          |     |              |        +--ro router
          .     .              .        .
          .     .              .        .
          |     |              |        +--ro network
          .     .              .        .
          .     .              .        .
          |     |              |        +--ro summary
          .     .              .        .
          .     .              .        .
          |     |              |        +--ro external
          .     .              .        .
          .     .              .        .
          |     |              |        +--ro opaque
          .     .              .        .
          .     .              .        .
          |     |              +--:(ospfv3)
          |     |                 +--ro ospfv3
          |     |                    +--ro header
          .     .                    .
          .     .                    .
          |     |                    +--ro body
          |     |                       +--ro router
          .     .                       .
          .     .                       .
          |     |                       +--ro network
          .     .                       .
          .     .                       .
          |     |                       +--ro inter-area-prefix
          .     .                       .
          .     .                       .
          |     |                       +--ro inter-area-router
          .     .                       .
          .     .                       .
          |     |                       +--ro as-external
          .     .                       .
          .     .                       .
          |     |                       +--ro nssa
          .     .                       .
          .     .                       .
          |     |                       +--ro link
          .     .                       .
          .     .                       .
          |     |                       +--ro intra-area-prefix
          .     .                       .
          .     .                       .
          |     |                       +--ro router-information
          .     .                       .
          .     .                       .
          |     +--rw virtual-links
          |     |  +--rw virtual-link* [transit-area-id router-id]
          |     |     +--rw transit-area-id       -> ../../../../
          |     |                                    area/area-id
          |     |     +--rw router-id             rt-types:router-id
          |     |     +--rw hello-interval?       uint16
          |     |     +--rw dead-interval?        uint32
          |     |     +--rw retransmit-interval?  uint16
          |     |     +--rw transmit-delay?       uint16
          |     |     +--rw lls?                  boolean {lls}?
          |     |     +--rw ttl-security {ttl-security}?
          |     |     |  +--rw enable?   boolean
          |     |     |  +--rw hops?     uint8
          |     |     +--rw enable?               boolean
          |     |     +--rw authentication
          |     |     |  +--rw (auth-type-selection)?
          |     |     |     +--:(ospfv2-auth)
          |     |     |     |  +--rw ospfv2-auth-trailer-rfc?
          |     |     |     |  |       ospfv2-auth-trailer-rfc-version
          |     |     |     |  |        {ospfv2-authentication-trailer}?
          |     |     |     |  +--rw (ospfv2-auth-specification)?
          |     |     |     |     +--:(auth-key-chain) {key-chain}?
          |     |     |     |     |  +--rw ospfv2-key-chain?
          |     |     |     |     |         key-chain:key-chain-ref
          |     |     |     |     +--:(auth-key-explicit)
          |     |     |     |        +--rw ospfv2-key-id?     uint32
          |     |     |     |        +--rw ospfv2-key?        string
          |     |     |     |        +--rw ospfv2-crypto-algorithm?
          |     |     |     |                identityref
          |     |     |     +--:(ospfv3-auth-ipsec)
          |     |     |     |      {ospfv3-authentication-ipsec}?
          |     |     |     |  +--rw sa?                       string
          |     |     |     +--:(ospfv3-auth-trailer)
          |     |     |        |  {ospfv3-authentication-trailer}?
          |     |     |        +--rw (ospfv3-auth-specification)?
          |     |     |           +--:(auth-key-chain) {key-chain}?
          |     |     |           |  +--rw ospfv3-key-chain?
          |     |     |           |          key-chain:key-chain-ref
          |     |     |           +--:(auth-key-explicit)
          |     |     |              +--rw ospfv3-sa-id?        uint16
          |     |     |              +--rw ospfv3-key?          string
          |     |     |              +--rw ospfv3-crypto-algorithm?
          |     |     |                      identityref
          |     |     +--ro cost?              uint16
          |     |     +--ro state?             if-state-type
          |     |     +--ro hello-timer?       rt-types:
          |     |     |                         rtimer-value-seconds16
          |     |     +--ro wait-timer?        rt-types:
          |     |     |                         rtimer-value-seconds16
          |     |     +--ro dr-router-id?      rt-types:router-id
          |     |     +--ro dr-ip-addr?        inet:ip-address
          |     |     +--ro bdr-router-id?     rt-types:router-id
          |     |     +--ro bdr-ip-addr?       inet:ip-address
          |     |     +--ro statistics
          |     |     |  +--ro discontinuity-time     yang:date-and-time
          |     |     |  +--ro if-event-count?        yang:counter32
          |     |     |  +--ro link-scope-lsa-count?  yang:gauge32
          |     |     |  +--ro link-scope-lsa-cksum-sum?
          |     |     |                               uint32
          |     |     |  +--ro database
          |     |     |     +--ro link-scope-lsa-type*
          |     |     |        +--ro lsa-type?        uint16
          |     |     |        +--ro lsa-count?       yang:gauge32
          |     |     |        +--ro lsa-cksum-sum?   int32
          |     |     +--ro neighbors
          |     |     |  +--ro neighbor* [neighbor-router-id]
          |     |     |     +--ro neighbor-router-id
          |     |     |                           rt-types:router-id
          |     |     |     +--ro address?        inet:ip-address
          |     |     |     +--ro dr-router-id?   rt-types:router-id
          |     |     |     +--ro dr-ip-addr?     inet:ip-address
          |     |     |     +--ro bdr-router-id?  rt-types:router-id
          |     |     |     +--ro bdr-ip-addr?    inet:ip-address
          |     |     |     +--ro state?          nbr-state-type
          |     |     |     +--ro dead-timer? rt-types:
          |     |     |     |                  rtimer-value-seconds16
          |     |     |     +--ro statistics
          |     |     |        +--ro discontinuity-time
          |     |     |                           yang:date-and-time
          |     |     |        +--ro nbr-event-count?
          |     |     |                           yang:counter32
          |     |     |        +--ro nbr-retrans-qlen?
          |     |     |                           yang:gauge32
          |     |     +--ro database
          |     |        +--ro link-scope-lsa-type* [lsa-type]
          |     |           +--ro lsa-type           uint16
          |     |           +--ro link-scope-lsas
          .     .
          .     .
          |     +--rw sham-links {pe-ce-protocol}?
          |     |  +--rw sham-link* [local-id remote-id]
          |     |     +--rw local-id               inet:ip-address
          |     |     +--rw remote-id              inet:ip-address
          |     |     +--rw hello-interval?        uint16
          |     |     +--rw dead-interval?         uint32
          |     |     +--rw retransmit-interval?   uint16
          |     |     +--rw transmit-delay?        uint16
          |     |     +--rw lls?                   boolean {lls}?
          |     |     +--rw ttl-security {ttl-security}?
          |     |     |  +--rw enable?   boolean
          |     |     |  +--rw hops?     uint8
          |     |     +--rw enable?             boolean
          |     |     +--rw authentication
          |     |     |  +--rw (auth-type-selection)?
          |     |     |     +--:(ospfv2-auth)
          |     |     |     |  +--rw ospfv2-auth-trailer-rfc?
          |     |     |     |  |       ospfv2-auth-trailer-rfc-version
          |     |     |     |  |        {ospfv2-authentication-trailer}?
          |     |     |     |  +--rw (ospfv2-auth-specification)?
          |     |     |     |     +--:(auth-key-chain) {key-chain}?
          |     |     |     |     |  +--rw ospfv2-key-chain?
          |     |     |     |     |         key-chain:key-chain-ref
          |     |     |     |     +--:(auth-key-explicit)
          |     |     |     |        +--rw ospfv2-key-id?     uint32
          |     |     |     |        +--rw ospfv2-key?        string
          |     |     |     |        +--rw ospfv2-crypto-algorithm?
          |     |     |     |                identityref
          |     |     |     +--:(ospfv3-auth-ipsec)
          |     |     |     |      {ospfv3-authentication-ipsec}?
          |     |     |     |  +--rw sa?                       string
          |     |     |     +--:(ospfv3-auth-trailer)
          |     |     |        |  {ospfv3-authentication-trailer}?
          |     |     |        +--rw (ospfv3-auth-specification)?
          |     |     |           +--:(auth-key-chain) {key-chain}?
          |     |     |           |  +--rw ospfv3-key-chain?
          |     |     |           |          key-chain:key-chain-ref
          |     |     |           +--:(auth-key-explicit)
          |     |     |              +--rw ospfv3-sa-id?        uint16
          |     |     |              +--rw ospfv3-key?          string
          |     |     |              +--rw ospfv3-crypto-algorithm?
          |     |     |                      identityref
          |     |     +--rw cost?               uint16
          |     |     +--rw mtu-ignore?         boolean
          |     |                               {mtu-ignore}?
          |     |     +--rw prefix-suppression? boolean
          |     |                               {prefix-suppression}?
          |     |     +--ro state?              if-state-type
          |     |     +--ro hello-timer?       rt-types:
          |     |     |                         rtimer-value-seconds16
          |     |     +--ro wait-timer?        rt-types:
          |     |     |                         rtimer-value-seconds16
          |     |     +--ro dr-router-id?       rt-types:router-id
          |     |     +--ro dr-ip-addr?         inet:ip-address
          |     |     +--ro bdr-router-id?      rt-types:router-id
          |     |     +--ro bdr-ip-addr?        inet:ip-address
          |     |     +--ro statistics
          |     |     |  +--ro discontinuity-time     yang:date-and-time
          |     |     |  +--ro if-event-count?        yang:counter32
          |     |     |  +--ro link-scope-lsa-count?  yang:gauge32
          |     |     |  +--ro link-scope-lsa-cksum-sum?
          |     |     |                               uint32
          |     |     |  +--ro database
          |     |     |     +--ro link-scope-lsa-type*
          |     |     |        +--ro lsa-type?        uint16
          |     |     |        +--ro lsa-count?       yang:gauge32
          |     |     |        +--ro lsa-cksum-sum?   int32
          |     |     +--ro neighbors
          |     |     |  +--ro neighbor* [neighbor-router-id]
          |     |     |     +--ro neighbor-router-id
          |     |     |                           rt-types:router-id
          |     |     |     +--ro address?        inet:ip-address
          |     |     |     +--ro dr-router-id?   rt-types:router-id
          |     |     |     +--ro dr-ip-addr?     inet:ip-address
          |     |     |     +--ro bdr-router-id?  rt-types:router-id
          |     |     |     +--ro bdr-ip-addr?    inet:ip-address
          |     |     |     +--ro state?          nbr-state-type
          |     |     |     +--ro cost?           uint32
          |     |     |     +--ro dead-timer? rt-types:
          |     |     |     |                  rtimer-value-seconds16
          |     |     |     +--ro statistics
          |     |     |        +--ro nbr-event-count?
          |     |     |                           yang:counter32
          |     |     |        +--ro nbr-retrans-qlen?
          |     |     |                           yang:gauge32
          |     |     +--ro database
          |     |        +--ro link-scope-lsa-type* [lsa-type]
          |     |           +--ro lsa-type           uint16
          |     |           +--ro link-scope-lsas
          .     .
          .     .
```

2.7. OSPF Interface Configuration/Operational State

The interface container contains OSPF interface configuration and operational state. The interface operational state includes the statistics, list of neighbors, and Link-Local Link State Database (LSDB).

```
   module: ietf-ospf
     augment /rt:routing/rt:control-plane-protocols/
              rt:control-plane-protocol:
       +--rw ospf
          .
          .
          +--rw areas
          |  +--rw area* [area-id]
          |     .
          |     .
          |     +--rw interfaces
          |        +--rw interface* [name]
          |           +--rw name                    if:interface-ref
          |           +--rw interface-type?        enumeration
          |           +--rw passive?               boolean
          |           +--rw demand-circuit?        boolean
          |                                        {demand-circuit}?
          |           +--rw priority?              uint8
          |           +--rw multi-areas {multi-area-adj}?
          |           |  +--rw multi-area* [multi-area-id]
          |           |     +--rw multi-area-id      area-id-type
          |           |     +--rw cost?              uint16
          |           +--rw static-neighbors
          |           |  +--rw neighbor* [identifier]
          |           |     +--rw identifier       inet:ip-address
          |           |     +--rw cost?            uint16
          |           |     +--rw poll-interval?   uint16
          |           |     +--rw priority?        uint8
          |           +--rw node-flag?             boolean
          |                                        {node-flag}?
          |           +--rw bfd {bfd}?
          |           |  +--rw enable?   boolean
          |           +--rw fast-reroute {fast-reroute}?
          |           |  +--rw lfa {lfa}?
          |           |     +--rw candidate-enable?   boolean
          |           |     +--rw enable?             boolean
          |           |     +--rw remote-lfa {remote-lfa}?
          |           |        +--rw enable?   boolean
          |           +--rw hello-interval?        uint16
          |           +--rw dead-interval?         uint32
          |           +--rw retransmit-interval?   uint16
          |           +--rw transmit-delay?        uint16
          |           +--rw lls?                   boolean {lls}?
          |           +--rw ttl-security {ttl-security}?
          |           |  +--rw enable?   boolean
          |           |  +--rw hops?     uint8
          |           +--rw enable?                boolean
          |           +--rw authentication
          |           |  +--rw (auth-type-selection)?
          |           |     +--:(ospfv2-auth)
          |           |     |  +--rw ospfv2-auth-trailer-rfc?
          |           |     |  |       ospfv2-auth-trailer-rfc-version
          |           |     |  |        {ospfv2-authentication-trailer}?
          |           |     |  +--rw (ospfv2-auth-specification)?
          |           |     |     +--:(auth-key-chain) {key-chain}?
          |           |     |     |  +--rw ospfv2-key-chain?
          |           |     |     |         key-chain:key-chain-ref
          |           |     |     +--:(auth-key-explicit)
          |           |     |        +--rw ospfv2-key-id?     uint32
          |           |     |        +--rw ospfv2-key?        string
          |           |     |        +--rw ospfv2-crypto-algorithm?
          |           |     |                identityref
          |           |     +--:(ospfv3-auth-ipsec)
          |           |     |      {ospfv3-authentication-ipsec}?
          |           |     |  +--rw sa?                       string
          |           |     +--:(ospfv3-auth-trailer)
          |           |        |  {ospfv3-authentication-trailer}?
          |           |        +--rw (ospfv3-auth-specification)?
          |           |           +--:(auth-key-chain) {key-chain}?
          |           |           |  +--rw ospfv3-key-chain?
          |           |           |          key-chain:key-chain-ref
          |           |           +--:(auth-key-explicit)
          |           |              +--rw ospfv3-sa-id?        uint16
          |           |              +--rw ospfv3-key?          string
          |           |              +--rw ospfv3-crypto-algorithm?
          |           |                      identityref
          |           +--rw cost?               uint16
          |           +--rw mtu-ignore?         boolean
          |           |                         {mtu-ignore}?
          |           +--rw prefix-suppression? boolean
          |           |                         {prefix-suppression}?
          |           +--ro state?                 if-state-type
          |           +--ro hello-timer?       rt-types:
          |           |                         rtimer-value-seconds16
          |           +--ro wait-timer?        rt-types:
          |           |                         rtimer-value-seconds16
          |           +--ro dr-router-id?       rt-types:router-id
          |           +--ro dr-ip-addr?         inet:ip-address
          |           +--ro bdr-router-id?      rt-types:router-id
          |           +--ro bdr-ip-addr?        inet:ip-address
          |           +--ro statistics
          |           |  +--ro if-event-count?        yang:counter32
          |           |  +--ro link-scope-lsa-count?  yang:gauge32
          |           |  +--ro link-scope-lsa-cksum-sum?
          |           |                               uint32
          |           |  +--ro database
          |           |     +--ro link-scope-lsa-type*
          |           |        +--ro lsa-type?        uint16
          |           |        +--ro lsa-count?       yang:gauge32
          |           |        +--ro lsa-cksum-sum?   int32
          |           +--ro neighbors
          |           |  +--ro neighbor* [neighbor-router-id]
          |           |     +--ro neighbor-router-id
          |           |                           rt-types:router-id
          |           |     +--ro address?        inet:ip-address
          |           |     +--ro dr-router-id?   rt-types:router-id
          |           |     +--ro dr-ip-addr?     inet:ip-address
          |           |     +--ro bdr-router-id?  rt-types:router-id
          |           |     +--ro bdr-ip-addr?    inet:ip-address
          |           |     +--ro state?          nbr-state-type
          |           |     +--ro dead-timer? rt-types:
          |           |     |                  rtimer-value-seconds16
          |           |     +--ro statistics
          |           |        +--ro nbr-event-count?
          |           |                           yang:counter32
          |           |        +--ro nbr-retrans-qlen?
          |           |                           yang:gauge32
          |           +--ro database
          |           .  +--ro link-scope-lsa-type* [lsa-type]
          |           .     +--ro lsa-type           uint16
          |           .     +--ro link-scope-lsas
          .           .
          .           .
          |           +--rw topologies {ospf:multi-topology}?
          |           |  +--rw topology* [name]
          |           |     +--rw name  -> ../../../../../../../../
          |           |                    ../../../rt:ribs/rib/name
          |           |     +--rw cost? uint32
          |           +--rw instance-id?           uint8
          .
          .
```

2.8. OSPF Notifications

This YANG model defines a list of notifications that inform YANG clients of important events detected during protocol operation. The defined notifications cover the common set of traps from the OSPFv2 MIB [RFC4750] and OSPFv3 MIB [RFC5643].

```
     notifications:
       +---n if-state-change
       |  +--ro routing-protocol-name?
       |  +     -> /rt:routing/control-plane-protocols/
       |  +         control-plane-protocol/name
       |  +--ro af?
       |  +     -> /rt:routing/control-plane-protocols/
       |  +         control-plane-protocol
       |  +         [rt:name=current()/../routing-protocol-name]/
       |  +         ospf:ospf/af
       |  +--ro (if-link-type-selection)?
       |  |  +--:(interface)
       |  |  |  +--ro interface
       |  |  |     +--ro interface?   if:interface-ref
       |  |  +--:(virtual-link)
       |  |  |  +--ro virtual-link
       |  |  |     +--ro transit-area-id?      area-id-type
       |  |  |     +--ro neighbor-router-id?   rt-types:router-id
       |  |  +--:(sham-link)
       |  |     +--ro sham-link
       |  |        +--ro area-id?          area-id-type
       |  |        +--ro local-ip-addr?    inet:ip-address
       |  |        +--ro remote-ip-addr?   inet:ip-address
       |  +--ro state?                   if-state-type
       +---n if-config-error
       |  +--ro routing-protocol-name?
       |  +     -> /rt:routing/control-plane-protocols/
       |  +         control-plane-protocol/name
       |  +--ro af?
       |  +     -> /rt:routing/control-plane-protocols/
       |  +         control-plane-protocol
       |  +         [rt:name=current()/../routing-protocol-name]/
       |  +         ospf:ospf/af
       |  +--ro (if-link-type-selection)?
       |  |  +--:(interface)
       |  |  |  +--ro interface
       |  |  |     +--ro interface?   if:interface-ref
       |  |  +--:(virtual-link)
       |  |  |  +--ro virtual-link
       |  |  |     +--ro transit-area-id?      area-id-type
       |  |  |     +--ro neighbor-router-id?   rt-types:router-id
       |  |  +--:(sham-link)
       |  |     +--ro sham-link
       |  |        +--ro area-id?          area-id-type
       |  |        +--ro local-ip-addr?    inet:ip-address
       |  |        +--ro remote-ip-addr?   inet:ip-address
       |  +--ro packet-source?           yang:dotted-quad
       |  +--ro packet-type?             packet-type
       |  +--ro error?                   enumeration
       +---n nbr-state-change
       |  +--ro routing-protocol-name?
       |  +     -> /rt:routing/control-plane-protocols/
       |  +         control-plane-protocol/name
       |  +--ro af?
       |  +     -> /rt:routing/control-plane-protocols/
       |  +         control-plane-protocol
       |  +         [rt:name=current()/../routing-protocol-name]/
       |  +         ospf:ospf/af
       |  +--ro (if-link-type-selection)?
       |  |  +--:(interface)
       |  |  |  +--ro interface
       |  |  |     +--ro interface?   if:interface-ref
       |  |  +--:(virtual-link)
       |  |  |  +--ro virtual-link
       |  |  |     +--ro transit-area-id?      area-id-type
       |  |  |     +--ro neighbor-router-id?   rt-types:router-id
       |  |  +--:(sham-link)
       |  |     +--ro sham-link
       |  |        +--ro area-id?          area-id-type
       |  |        +--ro local-ip-addr?    inet:ip-address
       |  |        +--ro remote-ip-addr?   inet:ip-address
       |  +--ro neighbor-router-id?      rt-types:router-id
       |  +--ro neighbor-ip-addr?        yang:dotted-quad
       |  +--ro state?                   nbr-state-type
       +---n nbr-restart-helper-status-change
       |  +--ro routing-protocol-name?
       |  +     -> /rt:routing/control-plane-protocols/
       |  +         control-plane-protocol/name
       |  +--ro af?
       |  +     -> /rt:routing/control-plane-protocols/
       |  +         control-plane-protocol
       |  +         [rt:name=current()/../routing-protocol-name]/
       |  +         ospf:ospf/af
       |  +--ro (if-link-type-selection)?
       |  |  +--:(interface)
       |  |  |  +--ro interface
       |  |  |     +--ro interface?   if:interface-ref
       |  |  +--:(virtual-link)
       |  |  |  +--ro virtual-link
       |  |  |     +--ro transit-area-id?      area-id-type
       |  |  |     +--ro neighbor-router-id?   rt-types:router-id
       |  |  +--:(sham-link)
       |  |     +--ro sham-link
       |  |        +--ro area-id?          area-id-type
       |  |        +--ro local-ip-addr?    inet:ip-address
       |  |        +--ro remote-ip-addr?   inet:ip-address
       |  +--ro neighbor-router-id?      rt-types:router-id
       |  +--ro neighbor-ip-addr?        yang:dotted-quad
       |  +--ro status?                  restart-helper-status-type
       |  +--ro age?                     uint32
       |  +--ro exit-reason?             restart-exit-reason-type
       +---n if-rx-bad-packet
       |  +--ro routing-protocol-name?
       |  +     -> /rt:routing/control-plane-protocols/
       |  +         control-plane-protocol/name
       |  +--ro af?
       |  +     -> /rt:routing/control-plane-protocols/
       |  +         control-plane-protocol
       |  +         [rt:name=current()/../routing-protocol-name]/
       |  +         ospf:ospf/af
       |  +--ro (if-link-type-selection)?
       |  |  +--:(interface)
       |  |  |  +--ro interface
       |  |  |     +--ro interface?   if:interface-ref
       |  |  +--:(virtual-link)
       |  |  |  +--ro virtual-link
       |  |  |     +--ro transit-area-id?      area-id-type
       |  |  |     +--ro neighbor-router-id?   rt-types:router-id
       |  |  +--:(sham-link)
       |  |     +--ro sham-link
       |  |        +--ro area-id?          area-id-type
       |  |        +--ro local-ip-addr?    inet:ip-address
       |  |        +--ro remote-ip-addr?   inet:ip-address
       |  +--ro packet-source?           yang:dotted-quad
       |  +--ro packet-type?             packet-type
       +---n lsdb-approaching-overflow
       |  +--ro routing-protocol-name?
       |  +     -> /rt:routing/control-plane-protocols/
       |  +         control-plane-protocol/name
       |  +--ro af?
       |  +     -> /rt:routing/control-plane-protocols/
       |  +         control-plane-protocol
       |  +         [rt:name=current()/../routing-protocol-name]/
       |  +         ospf:ospf/af
       |  +--ro ext-lsdb-limit?          uint32
       +---n lsdb-overflow
       |  +--ro routing-protocol-name?
       |  +     -> /rt:routing/control-plane-protocols/
       |  +         control-plane-protocol/name
       |  +--ro af?
       |  +     -> /rt:routing/control-plane-protocols/
       |  +         control-plane-protocol
       |  +         [rt:name=current()/../routing-protocol-name]/
       |  +         ospf:ospf/af
       |  +--ro ext-lsdb-limit?          uint32
       +---n nssa-translator-status-change
       |  +--ro routing-protocol-name?
       |  +     -> /rt:routing/control-plane-protocols/
       |  +         control-plane-protocol/name
       |  +--ro af?
       |  +     -> /rt:routing/control-plane-protocols/
       |  +         control-plane-protocol
       |  +         [rt:name=current()/../routing-protocol-name]/
       |  +         ospf:ospf/af
       |  +--ro area-id?                 area-id-type
       |  +--ro status?                  nssa-translator-state-type
       +---n restart-status-change
          +--ro routing-protocol-name?
          +     -> /rt:routing/control-plane-protocols/
          +         control-plane-protocol/name
          +--ro af?
          +     -> /rt:routing/control-plane-protocols/
          +         control-plane-protocol
          +         [rt:name=current()/../routing-protocol-name]/
          +         ospf:ospf/af
          +--ro status?                  restart-status-type
          +--ro restart-interval?        uint16
          +--ro exit-reason?             restart-exit-reason-type
```

2.9. OSPF RPC Operations

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

- clear-database: reset the content of a particular OSPF Link State Database.

- clear-neighbor: Reset a particular OSPF neighbor or group of neighbors associated with an OSPF interface.

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

3. OSPF YANG Module

The following RFCs and drafts are not referenced in the document text but are referenced in the ietf-ospf.yang module: [RFC0905], [RFC4576], [RFC4973], [RFC5250], [RFC5309], [RFC5642], [RFC5881], [RFC6991], [RFC7770], [RFC7884], [RFC8294], and [RFC8476].

```
   <CODE BEGINS> file "ietf-ospf@2019-10-17.yang"
   module ietf-ospf {
     yang-version 1.1;
     namespace "urn:ietf:params:xml:ns:yang:ietf-ospf";

     prefix ospf;

     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 (NMDA Version)";
     }

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

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

     import ietf-bfd-types {
       prefix "bfd-types";
       reference "RFC YYYY: YANG Data Model for Bidirectional
         Forwarding Detection (BFD). Please replace YYYY with
         published RFC number for draft-ietf-bfd-yang.";
     }

     organization
         "IETF LSR - Link State Routing Working Group";

     contact
       "WG Web:   <https://datatracker.ietf.org/group/lsr/>
        WG List:  <mailto:lsr@ietf.org>

        Editor:   Derek Yeung
                  <mailto:derek@arrcus.com>
        Author:   Acee Lindem
                  <mailto:acee@cisco.com>
        Author:   Yingzhen Qu
                  <mailto:yingzhen.qu@futurewei.com>
        Author:   Salih K A
                  <mailto:salih@juniper.net>
        Author:   Ing-Wher Chen
                  <mailto:ingwherchen@mitre.org>";

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

        This YANG model conforms to the Network Management
        Datastore Architecture (NMDA) 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
        (https://trustee.ietf.org/license-info).

        This version of this YANG module is part of RFC XXXX
        (https://www.rfc-editor.org/info/rfcXXXX); 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 (RFC 2119) (RFC 8174) when, and only when,
        they appear in all capitals, as shown here.

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

     revision 2019-10-17 {
       description
         "Initial revision.";
       reference
         "RFC XXXX: A YANG Data Model for OSPF.";
     }

     feature multi-topology {
       description
         "Support Multiple-Topology Routing (MTR).";
       reference "RFC 4915: Multi-Topology Routing";
     }

     feature multi-area-adj {
       description
         "OSPF multi-area adjacency support as in RFC 5185.";
       reference "RFC 5185: Multi-Area Adjacency";
     }
     feature explicit-router-id {
       description
         "Set Router-ID per instance explicitly.";
     }

     feature demand-circuit {
       description
         "OSPF demand circuit support as in RFC 1793.";
       reference "RFC 1793: OSPF Demand Circuits";
     }

     feature mtu-ignore {
       description
         "Disable OSPF Database Description packet MTU
          mismatch checking specified in the OSPF
          protocol specification.";
       reference "RFC 2328: OSPF Version 2, section 10.6";
     }

     feature lls {
       description
         "OSPF link-local signaling (LLS) as in RFC 5613.";
       reference "RFC 5613: OSPF Link-Local Signaling";
     }
     feature prefix-suppression {
       description
         "OSPF prefix suppression support as in RFC 6860.";
       reference "RFC 6860: Hide Transit-Only Networks in OSPF";
     }

     feature ttl-security {
       description
         "OSPF Time to Live (TTL) security check support.";
       reference "RFC 5082: The Generalized TTL Security
                  Mechanism (GTSM)";
     }

     feature nsr {
       description
         "Non-Stop-Routing (NSR) support. The OSPF NSR feature
          allows  a router with redundant control-plane capability
          (e.g., dual Route-Processor (RP) cards) to maintain its
          state and adjacencies during planned and unplanned
          OSPF instance restarts. It differs from graceful-restart
          or Non-Stop Forwarding (NSF) in that no protocol signaling
          or assistance from adjacent OSPF neighbors is required to
          recover control-plane state.";
     }

     feature graceful-restart {
       description
         "Graceful OSPF Restart as defined in RFC 3623 and
          RFC 5187.";
       reference "RFC 3623: Graceful OSPF Restart
                  RFC 5187: OSPFv3 Graceful Restart";
     }

     feature auto-cost {
       description
         "Calculate OSPF interface cost according to
          reference bandwidth.";
        reference "RFC 2328: OSPF Version 2";
     }

     feature max-ecmp {
       description
         "Setting maximum number of ECMP paths.";
     }

     feature max-lsa {
       description
         "Setting the maximum number of LSAs the OSPF instance
```

```
          will accept.";
       reference "RFC 1765: OSPF Database Overload";
     }

     feature te-rid {
       description
         "Support configuration of the Traffic Engineering (TE)
          Router-ID, i.e., the Router Address described in Section
          2.4.1 of RFC3630 or the Router IPv6 Address TLV described
          in Section 3 of RFC5329.";
       reference "RFC 3630: Traffic Engineering (TE) Extensions
                  to OSPF Version 2
                  RFC 5329: Traffic Engineering (TE) Extensions
                  to OSPF Version 3";
     }

     feature ldp-igp-sync {
       description
         "LDP IGP synchronization.";
       reference "RFC 5443: LDP IGP Synchronization";
     }

     feature ospfv2-authentication-trailer {
       description
         "Support OSPFv2 authentication trailer for OSPFv2
          authentication.";
       reference "RFC 5709: Supporting Authentication
                  Trailer for OSPFv2
                  RFC 7474: Security Extension for OSPFv2 When
                  Using Manual Key Management";
     }

     feature ospfv3-authentication-ipsec {
       description
         "Support IPsec for OSPFv3 authentication.";
       reference "RFC 4552: Authentication/Confidentiality
                 for OSPFv3";
     }

     feature ospfv3-authentication-trailer {
       description
         "Support OSPFv3 authentication trailer for OSPFv3
          authentication.";
       reference "RFC 7166: Supporting Authentication
                  Trailer for OSPFv3";
     }

     feature fast-reroute {
      description
        "Support for IP Fast Reroute (IP-FRR).";
      reference "RFC 5714: IP Fast Reroute Framework";
     }

     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 OSPF prefixes.";
      reference "RFC 7684: OSPFv2 Prefix/Link Advertisement";
     }

     feature node-tag {
       description
         "Support for node admin tag for OSPF routing instances.";
       reference "RFC 7777: Advertising Node Administrative
                  Tags in OSPF";
     }

     feature lfa {
      description
        "Support for Loop-Free Alternates (LFAs).";
      reference "RFC 5286: Basic Specification for IP Fast
                 Reroute: Loop-Free Alternates";
     }

     feature remote-lfa {
      description
        "Support for Remote Loop-Free Alternates (R-LFA).";
      reference "RFC 7490: Remote Loop-Free Alternate (LFA)
                 Fast Reroute (FRR)";
     }

     feature stub-router {
       description
         "Support for RFC 6987 OSPF Stub Router Advertisement.";
       reference "RFC 6987: OSPF Stub Router Advertisement";
     }

     feature pe-ce-protocol {
       description
         "Support for OSPF as a PE-CE protocol";
       reference "RFC 4577: OSPF as the Provider/Customer Edge
```

```
                  Protocol for BGP/MPLS IP Virtual Private
                  Networks (VPNs)
                  RFC 6565: OSPFv3 as a Provider Edge to Customer
                  Edge (PE-CE) Routing Protocol";
     }

     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 OSPF neighbor reachability.";
       reference "RFC 5880: Bidirectional Forwarding Detection (BFD)
                  RFC 5881: Bidirectional Forwarding Detection
                  (BFD) for IPv4 and IPv6 (Single Hop)";
     }

     feature hybrid-interface {
       description
         "Support for OSPF Hybrid interface type.";
       reference "RFC 6845: OSPF Hybrid Broadcast and
                  Point-to-Multipoint Interface Type";
     }

     identity ospf {
       base "rt:routing-protocol";
       description "Any OSPF protocol version";
     }

     identity ospfv2 {
       base "ospf";
       description "OSPFv2 protocol";
     }

     identity ospfv3 {
       base "ospf";
       description "OSPFv3 protocol";
     }

     identity area-type {
       description "Base identity for OSPF area type.";
     }

     identity normal-area {
       base area-type;
       description "OSPF normal area.";
     }

     identity stub-nssa-area {
       base area-type;
       description "OSPF stub or NSSA area.";
     }

     identity stub-area {
       base stub-nssa-area;
       description "OSPF stub area.";
     }

     identity nssa-area {
       base stub-nssa-area;
       description "OSPF Not-So-Stubby Area (NSSA).";
       reference "RFC 3101: The OSPF Not-So-Stubby Area
                  (NSSA) Option";
     }

     identity ospf-lsa-type {
       description
         "Base identity for OSPFv2 and OSPFv3
          Link State Advertisement (LSA) types";
     }

     identity ospfv2-lsa-type {
       base ospf-lsa-type;
       description
         "OSPFv2 LSA types";
     }

     identity ospfv2-router-lsa {
       base ospfv2-lsa-type;
       description
         "OSPFv2 Router LSA - Type 1";
     }

     identity ospfv2-network-lsa {
       base ospfv2-lsa-type;
       description
         "OSPFv2 Network LSA - Type 2";
     }

     identity ospfv2-summary-lsa-type {
       base ospfv2-lsa-type;
       description
```

```
         "OSPFv2 Summary LSA types";
     }

     identity ospfv2-network-summary-lsa {
       base ospfv2-summary-lsa-type;
       description
         "OSPFv2 Network Summary LSA - Type 3";
     }

     identity ospfv2-asbr-summary-lsa {
       base ospfv2-summary-lsa-type;
       description
         "OSPFv2 AS Boundary Router (ASBR) Summary LSA - Type 4";
     }

     identity ospfv2-external-lsa-type {
       base ospfv2-lsa-type;
       description
         "OSPFv2 External LSA types";
     }

     identity ospfv2-as-external-lsa {
       base ospfv2-external-lsa-type;
       description
         "OSPFv2 AS External LSA - Type 5";
     }

     identity ospfv2-nssa-lsa {
       base ospfv2-external-lsa-type;
       description
         "OSPFv2 Not-So-Stubby-Area (NSSA) LSA - Type 7";
     }

     identity ospfv2-opaque-lsa-type {
       base ospfv2-lsa-type;
       description
         "OSPFv2 Opaque LSA types";
     }

     identity ospfv2-link-scope-opaque-lsa {
       base ospfv2-opaque-lsa-type;
       description
         "OSPFv2 Link-Scoped Opaque LSA - Type 9";
     }

     identity ospfv2-area-scope-opaque-lsa {
       base ospfv2-opaque-lsa-type;
       description
```

```
         "OSPFv2 Area-Scoped Opaque LSA - Type 10";
     }

     identity ospfv2-as-scope-opaque-lsa {
       base ospfv2-opaque-lsa-type;
       description
         "OSPFv2 AS-Scoped Opaque LSA - Type 11";
     }

     identity ospfv2-unknown-lsa-type {
       base ospfv2-lsa-type;
       description
         "OSPFv2 Unknown LSA type";
     }

     identity ospfv3-lsa-type {
       base ospf-lsa-type;
       description
         "OSPFv3 LSA types.";
     }

     identity ospfv3-router-lsa {
       base ospfv3-lsa-type;
       description
         "OSPFv3 Router LSA - Type 0x2001";
     }

     identity ospfv3-network-lsa {
       base ospfv3-lsa-type;
       description
         "OSPFv3 Network LSA - Type 0x2002";
     }

     identity ospfv3-summary-lsa-type {
       base ospfv3-lsa-type;
       description
         "OSPFv3 Summary LSA types";
     }

     identity ospfv3-inter-area-prefix-lsa {
       base ospfv3-summary-lsa-type;
       description
         "OSPFv3 Inter-area Prefix LSA - Type 0x2003";
     }

     identity ospfv3-inter-area-router-lsa {
       base ospfv3-summary-lsa-type;
       description
```

```
           "OSPFv3 Inter-area Router LSA - Type 0x2004";
       }

       identity ospfv3-external-lsa-type {
         base ospfv3-lsa-type;
         description
           "OSPFv3 External LSA types";
       }

       identity ospfv3-as-external-lsa {
         base ospfv3-external-lsa-type;
         description
           "OSPFv3 AS-External LSA - Type 0x4005";
       }

       identity ospfv3-nssa-lsa {
         base ospfv3-external-lsa-type;
         description
           "OSPFv3 Not-So-Stubby-Area (NSSA) LSA - Type 0x2007";
       }

       identity ospfv3-link-lsa {
         base ospfv3-lsa-type;
         description
           "OSPFv3 Link LSA - Type 0x0008";
       }

       identity ospfv3-intra-area-prefix-lsa {
         base ospfv3-lsa-type;
         description
           "OSPFv3 Intra-area Prefix LSA - Type 0x2009";
       }

       identity ospfv3-router-information-lsa {
         base ospfv3-lsa-type;
         description
           "OSPFv3 Router Information LSA - Types 0x800C,
            0xA00C, and 0xC00C";
       }

       identity ospfv3-unknown-lsa-type {
         base ospfv3-lsa-type;
         description
           "OSPFv3 Unknown LSA type";
       }

       identity lsa-log-reason {
         description
           "Base identity for an LSA log reason.";
       }

       identity lsa-refresh {
         base lsa-log-reason;
         description
           "Identity used when the LSA is logged
            as a result of receiving a refresh LSA.";
       }

       identity lsa-content-change {
         base lsa-log-reason;
         description
           "Identity used when the LSA is logged
            as a result of a change in the content
            of the LSA.";
       }

       identity lsa-purge {
         base lsa-log-reason;
         description
           "Identity used when the LSA is logged
            as a result of being purged.";
       }

       identity informational-capability {
         description
           "Base identity for router informational capabilities.";
       }

       identity graceful-restart {
         base informational-capability;
         description
           "When set, the router is capable of restarting
            gracefully.";
         reference "RFC 3623: Graceful OSPF Restart
                    RFC 5187: OSPFv3 Graceful Restart";
       }

       identity graceful-restart-helper {
         base informational-capability;
         description
           "When set, the router is capable of acting as
            a graceful restart helper.";
         reference "RFC 3623: Graceful OSPF Restart
                    RFC 5187: OSPFv3 Graceful Restart";
       }
       identity stub-router {
         base informational-capability;
         description
           "When set, the router is capable of acting as
            an OSPF Stub Router.";
         reference "RFC 6987: OSPF Stub Router Advertisement";
       }

       identity traffic-engineering {
         base informational-capability;
         description
           "When set, the router is capable of OSPF traffic
            engineering.";
         reference "RFC 3630: Traffic Engineering (TE) Extensions
                    to OSPF Version 2
                    RFC 5329: Traffic Engineering (TE) Extensions
                    to OSPF Version 3";
       }

       identity p2p-over-lan {
         base informational-capability;
         description
           "When set, the router is capable of OSPF Point-to-Point
            over LAN.";
         reference "RFC 5309: Point-to-Point Operation over LAN
                    in Link State Routing Protocols";
       }

       identity experimental-te {
         base informational-capability;
         description
           "When set, the router is capable of OSPF experimental
            traffic engineering.";
         reference
           "RFC 4973: OSPF-xTE OSPF Experimental Traffic
            Engineering";
       }

       identity router-lsa-bit {
         description
           "Base identity for Router-LSA bits.";
       }

       identity vlink-end-bit {
         base router-lsa-bit;
         description
           "V bit, when set, the router is an endpoint of one or
            more virtual links.";
       }

       identity asbr-bit {
         base router-lsa-bit;
         description
           "E bit, when set, the router is an AS Boundary
            Router (ASBR).";
       }

       identity abr-bit {
         base router-lsa-bit;
         description
           "B bit, when set, the router is an Area Border
            Router (ABR).";
       }

       identity nssa-bit {
         base router-lsa-bit;
         description
           "Nt bit, when set, the router is an NSSA border router
            that is unconditionally translating NSSA LSAs into
            AS-external LSAs.";
       }

       identity ospfv3-lsa-option {
         description
           "Base identity for OSPF LSA options flags.";
       }

       identity af-bit {
         base ospfv3-lsa-option;
         description
           "AF bit, when set, the router supports OSPFv3 Address
            Families as in RFC5838.";
       }

       identity dc-bit {
         base ospfv3-lsa-option;
         description
           "DC bit, when set, the router supports demand circuits.";
       }

       identity r-bit {
         base ospfv3-lsa-option;
         description
           "R bit, when set, the originator is an active router.";
       }
       identity n-bit {
         base ospfv3-lsa-option;
         description
           "N bit, when set, the router is attached to an NSSA";
       }

       identity e-bit {
         base ospfv3-lsa-option;
         description
           "E bit, this bit describes the way AS-external LSAs
            are flooded";
       }

       identity v6-bit {
         base ospfv3-lsa-option;
         description
           "V6 bit, if clear, the router/link should be excluded
            from IPv6 routing calculation";
       }

       identity ospfv3-prefix-option {
         description
           "Base identity for OSPFv3 Prefix Options.";
       }

       identity nu-bit {
         base ospfv3-prefix-option;
         description
           "NU Bit, when set, the prefix should be excluded
            from IPv6 unicast calculations.";
       }

       identity la-bit {
         base ospfv3-prefix-option;
         description
           "LA bit, when set, the prefix is actually an IPv6
            interface address of the Advertising Router.";
       }

       identity p-bit {
         base ospfv3-prefix-option;
         description
           "P bit, when set, the NSSA area prefix should be
            translated to an AS External LSA and advertised
            by the translating NSSA Border Router.";
       }

       identity dn-bit {
         base ospfv3-prefix-option;
         description
           "DN bit, when set, the inter-area-prefix LSA or
            AS-external LSA prefix has been advertised as an
            L3VPN prefix.";
       }

       identity ospfv2-lsa-option {
         description
           "Base identity for OSPFv2 LSA option flags.";
       }

       identity mt-bit {
         base ospfv2-lsa-option;
         description
           "MT bit, When set, the router supports multi-topology as
            in RFC 4915.";
       }

       identity v2-dc-bit {
         base ospfv2-lsa-option;
         description
           "DC bit, When set, the router supports demand circuits.";
       }

       identity v2-p-bit {
         base ospfv2-lsa-option;
         description
           "P bit, wnly used in type-7 LSA. When set, an NSSA
            border router should translate the type-7 LSA
            to a type-5 LSA.";
       }

       identity mc-flag {
         base ospfv2-lsa-option;
         description
           "MC Bit, when set, the router supports MOSPF.";
       }

       identity v2-e-flag {
         base ospfv2-lsa-option;
         description
           "E Bit, this bit describes the way AS-external LSAs
            are flooded.";
       }

       identity o-bit {
         base ospfv2-lsa-option;
         description
           "O bit, when set, the router is opaque-capable as in
            RFC 5250.";
       }

       identity v2-dn-bit {
         base ospfv2-lsa-option;
         description
           "DN bit, when a type 3, 5 or 7 LSA is sent from a PE
            to a CE, the DN bit must be set. See RFC 4576.";
       }

       identity ospfv2-extended-prefix-flag {
         description
           "Base identity for extended prefix TLV flag.";
       }

       identity a-flag {
         base ospfv2-extended-prefix-flag;
         description
           "Attach flag, when set it indicates that the prefix
            corresponds and a route what is directly connected to
            the advertising router..";
       }

       identity node-flag {
         base ospfv2-extended-prefix-flag;
         description
           "Node flag, when set, it indicates that the prefix is
            used to represent the advertising node, e.g., a loopback
            address.";
       }

       typedef ospf-metric {
         type uint32 {
           range "0 .. 16777215";
         }
         description
           "OSPF Metric - 24-bit unsigned integer.";
       }

       typedef ospf-link-metric {
         type uint16 {
           range "0 .. 65535";
         }
         description
           "OSPF Link Metric - 16-bit unsigned integer.";
       }
       typedef opaque-id {
         type uint32 {
           range "0 .. 16777215";
         }
         description
           "Opaque ID - 24-bit unsigned integer.";
       }

       typedef area-id-type {
         type yang:dotted-quad;
         description
           "Area ID type.";
       }

       typedef route-type {
         type enumeration {
           enum intra-area {
             description "OSPF intra-area route.";
           }
           enum inter-area {
             description "OSPF inter-area route.";
           }
           enum external-1 {
             description "OSPF type 1 external route.";
           }
           enum external-2 {
             description "OSPF type 2 external route.";
           }
           enum nssa-1 {
             description "OSPF type 1 NSSA route.";
           }
           enum nssa-2 {
             description "OSPF type 2 NSSA route.";
           }
         }
         description "OSPF route type.";
       }

       typedef if-state-type {
         type enumeration {
           enum down {
             value "1";
             description
               "Interface down state.";
           }
           enum loopback {
             value "2";
             description
```

```
            "Interface loopback state.";
        }
        enum waiting {
          value "3";
          description
            "Interface waiting state.";
        }
        enum point-to-point {
          value "4";
          description
            "Interface point-to-point state.";
        }
        enum dr {
          value "5";
          description
            "Interface Designated Router (DR) state.";
        }
        enum bdr {
          value "6";
          description
            "Interface Backup Designated Router (BDR) state.";
        }
        enum dr-other {
          value "7";
          description
            "Interface Other Designated Router state.";
        }
      }
      description
        "OSPF interface state type.";
    }

    typedef router-link-type {
      type enumeration {
        enum point-to-point-link {
          value "1";
          description
            "Point-to-Point link to Router";
        }
        enum transit-network-link {
          value "2";
          description
            "Link to transit network identified by
             Designated-Router (DR)";
        }
        enum stub-network-link {
          value "3";
          description
```

```
             "Link to stub network identified by subnet";
         }
         enum virtual-link {
           value "4";
           description
             "Virtual link across transit area";
         }
       }
       description
         "OSPF Router Link Type.";
     }

     typedef nbr-state-type {
       type enumeration {
         enum down {
           value "1";
           description
             "Neighbor down state.";
         }
         enum attempt {
           value "2";
           description
             "Neighbor attempt state.";
         }
         enum init {
           value "3";
           description
             "Neighbor init state.";
         }
         enum 2-way {
           value "4";
           description
             "Neighbor 2-Way state.";
         }
         enum exstart {
           value "5";
           description
             "Neighbor exchange start state.";
         }
         enum exchange {
           value "6";
           description
             "Neighbor exchange state.";
         }
         enum loading {
           value "7";
           description
             "Neighbor loading state.";
         }
         enum full {
           value "8";
           description
             "Neighbor full state.";
         }
       }
       description
         "OSPF neighbor state type.";
     }

     typedef restart-helper-status-type {
       type enumeration {
         enum not-helping {
           value "1";
           description
             "Restart helper status not helping.";
         }
         enum helping {
           value "2";
           description
             "Restart helper status helping.";
         }
       }
       description
         "Restart helper status type.";
     }

     typedef restart-exit-reason-type {
       type enumeration {
         enum none {
           value "1";
           description
             "Restart not attempted.";
         }
         enum in-progress {
           value "2";
           description
             "Restart in progress.";
         }
         enum completed {
           value "3";
           description
             "Restart successfully completed.";
         }
         enum timed-out {
           value "4";
           description
```

```
             "Restart timed out.";
         }
         enum topology-changed {
           value "5";
           description
             "Restart aborted due to topology change.";
         }
       }
       description
         "Describes the outcome of the last attempt at a
          graceful restart, either by itself or acting
          as a helper.";
     }

     typedef packet-type {
       type enumeration {
         enum hello {
           value "1";
           description
             "OSPF Hello packet.";
         }
         enum database-description {
           value "2";
           description
             "OSPF Database Description packet.";
         }
         enum link-state-request {
           value "3";
           description
             "OSPF Link State Request packet.";
         }
         enum link-state-update {
           value "4";
           description
             "OSPF Link State Update packet.";
         }
         enum link-state-ack {
           value "5";
           description
             "OSPF Link State Acknowledgement packet.";
         }
       }
       description
         "OSPF packet type.";
     }

     typedef nssa-translator-state-type {
       type enumeration {
         enum enabled {
           value "1";
           description
             "NSSA translator enabled state.";
         }
         enum elected {
           value "2";
           description
             "NSSA translator elected state.";
         }
         enum disabled {
           value "3";
           description
             "NSSA translator disabled state.";
         }
       }
       description
         "OSPF NSSA translator state type.";
     }

     typedef restart-status-type {
       type enumeration {
         enum not-restarting {
           value "1";
           description
             "Router is not restarting.";
         }
         enum planned-restart {
           value "2";
           description
             "Router is going through planned restart.";
         }
         enum unplanned-restart {
           value "3";
           description
             "Router is going through unplanned restart.";
         }
       }
       description
         "OSPF graceful restart status type.";
     }

     typedef fletcher-checksum16-type {
       type string {
          pattern ’(0x)?[0-9a-fA-F]{4}’;
       }
        description
         "Fletcher 16-bit checksum in hex-string format 0xXXXX.";
        reference "RFC 905: ISO Transport Protocol specification
                   ISO DP 8073";
     }

     typedef ospfv2-auth-trailer-rfc-version {
       type enumeration {
         enum rfc5709 {
           description
             "Support OSPF Authentication Trailer as
              described in RFC 5709";
           reference "RFC 5709: OSPFv2 HMAC-SHA Cryptographic
                      Authentication";

         }
         enum rfc7474 {
           description
             "Support OSPF Authentication Trailer as
              described in RFC 7474";
           reference
             "RFC 7474: Security Extension for OSPFv2
              When Using Manual Key Management Authentication";

         }
       }
       description
         "OSPFv2 Authentication Trailer Support";
     }

     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;
          }
        }
      }

      grouping node-tag-tlv {
        description "OSPF Node Admin Tag TLV grouping.";
        list node-tag {
          leaf tag {
            type uint32;
            description
              "Node admin tag value.";
          }
          description
            "List of tags.";
        }
      }

      grouping router-capabilities-tlv {
        description "OSPF Router Capabilities TLV grouping.";
        reference "RFC 7770: OSPF Router Capabilities";
        container router-informational-capabilities {
          leaf-list informational-capabilities {
            type identityref {
              base informational-capability;
            }
            description
              "Informational capability list. This list will
               contains the identities for the informational
               capabilities supported by router.";
          }
          description
             "OSPF Router Informational Flag Definitions.";
        }
        list informational-capabilities-flags {
          leaf informational-flag {
            type uint32;
            description
              "Individual informational capability flag.";
          }
          description
            "List of informational capability flags. This will
             return all the 32-bit informational flags irrespective
             of whether or not they are known to the device.";
        }
        list functional-capabilities {
          leaf functional-flag {
            type uint32;
            description
              "Individual functional capability flag.";
          }
          description
            "List of functional capability flags. This will
             return all the 32-bit functional flags irrespective
             of whether or not they are known to the device.";
        }
      }

      grouping dynamic-hostname-tlv {
        description "Dynamic Hostname TLV";
        reference "RFC 5642: Dynamic Hostnames for OSPF";
        leaf hostname {
          type string {
            length "1..255";
          }
          description "Dynamic Hostname";
        }
      }

      grouping sbfd-discriminator-tlv {
        description "Seamless BFD Discriminator TLV";
        reference "RFC 7884: S-BFD Discriminators in OSPF";
        list sbfd-discriminators {
          leaf sbfd-discriminator {
            type uint32;
            description "Individual S-BFD Discriminator.";
          }
          description
            "List of S-BFD Discriminators";
        }
      }

      grouping maximum-sid-depth-tlv {
        description "Maximum SID Depth (MSD) TLV";
        reference
          "RFC 8476: Signaling Maximum Segment Depth (MSD)
           using OSPF";
        list msd-type {
          leaf msd-type {
            type uint8;
            description "Maximum Segment Depth (MSD) type";
          }
          leaf msd-value {
            type uint8;
            description
              "Maximum Segment Depth (MSD) value for the type";
          }
          description
            "List of Maximum Segment Depth (MSD) tuples";
        }
      }

      grouping ospf-router-lsa-bits {
        container router-bits {
          leaf-list rtr-lsa-bits {
            type identityref {
              base router-lsa-bit;
            }
            description
              "Router LSA bits list. This list will contain
               identities for the bits which are set in the
               Router-LSA bits.";
          }
          description "Router LSA Bits.";
        }
        description
          "Router LSA Bits - Currently common for OSPFv2 and
           OSPFv3 but it may diverge with future augmentations.";
      }

      grouping ospfv2-router-link {
        description "OSPFv2 router link.";
        leaf link-id {
          type union {
            type inet:ipv4-address;
            type yang:dotted-quad;
          }
          description "Router-LSA Link ID";
        }
        leaf link-data {
          type union {
            type inet:ipv4-address;
            type uint32;
          }
          description "Router-LSA Link data.";
        }
        leaf type {
          type router-link-type;
          description "Router-LSA Link type.";
        }
      }

      grouping ospfv2-lsa-body {
        description "OSPFv2 LSA body.";
        container router {
          when "derived-from-or-self(../../header/type, "
              + "'ospfv2-router-lsa')" {
            description
              "Only applies to Router-LSAs.";
          }
          description
            "Router LSA.";
          uses ospf-router-lsa-bits;
          leaf num-of-links {
            type uint16;
            description "Number of links in Router LSA.";
          }
          container links {
            description "All router Links.";
            list link {
              description "Router LSA link.";
              uses ospfv2-router-link;
              container topologies {
                description "All topologies for the link.";
                list topology {
                  description
                    "Topology specific information.";
                  leaf mt-id {
                    type uint8;
                    description
                      "The MT-ID for the topology enabled on
                       the link.";
                  }
                  leaf metric {
                    type uint16;
                    description "Metric for the topology.";
                  }
                }
              }
            }
          }
        }
        container network {
          when "derived-from-or-self(../../header/type, "
              + "'ospfv2-network-lsa')" {
            description
              "Only applies to Network LSAs.";
          }
          description
            "Network LSA.";
          leaf network-mask {
            type yang:dotted-quad;
            description
              "The IP address mask for the network.";
          }
          container attached-routers {
            description "All attached routers.";
            leaf-list attached-router {
              type inet:ipv4-address;
              description
                "List of the routers attached to the network.";
            }
          }
        }
        container summary {
          when "derived-from(../../header/type, "
             + "'ospfv2-summary-lsa-type')" {
            description
              "Only applies to Summary LSAs.";
          }
          description
            "Summary LSA.";
          leaf network-mask {
            type inet:ipv4-address;
            description
              "The IP address mask for the network";
          }
          container topologies {
            description "All topologies for the summary LSA.";
            list topology {
              description
                "Topology specific information.";
              leaf mt-id {
                type uint8;
                description
                  "The MT-ID for the topology enabled for
                   the summary.";
              }
              leaf metric {
                type ospf-metric;
                description "Metric for the topology.";
              }
            }
          }
        }
        container external {
          when "derived-from(../../header/type, "
             + "'ospfv2-external-lsa-type')" {
            description
              "Only applies to AS-external LSAs and NSSA LSAs.";
          }
          description
            "External LSA.";
          leaf network-mask {
            type inet:ipv4-address;
            description
              "The IP address mask for the network";
          }
          container topologies {
            description "All topologies for the external.";
            list topology {
              description
                "Topology specific information.";
              leaf mt-id {
                type uint8;
                description
                  "The MT-ID for the topology enabled for the
                   external or NSSA prefix.";
              }
              leaf flags {
                type bits {
                  bit E {
                    description
                      "When set, the metric specified is a Type 2
                       external metric.";
                  }
                }
                description "Flags.";
              }
              leaf metric {
                type ospf-metric;
                description "Metric for the topology.";
              }
              leaf forwarding-address {
                type inet:ipv4-address;
                description
                  "Forwarding address.";
              }
              leaf external-route-tag {
                type uint32;
                description
                  "Route tag for the topology.";
              }
            }
          }
        }
        container opaque {
          when "derived-from(../../header/type, "
             + "'ospfv2-opaque-lsa-type')" {
            description
              "Only applies to Opaque LSAs.";
          }
          description
            "Opaque LSA.";

          container ri-opaque {
            description "OSPF Router Information (RI) opaque LSA.";
            reference "RFC 7770: OSPF Router Capabilities";

            container router-capabilities-tlv {
              description
                "Informational and functional router capabilities";
              uses router-capabilities-tlv;
            }

            container node-tag-tlvs {
              description
                "All node tag TLVs.";
              list node-tag-tlv {
                description
                  "Node tag TLV.";
                uses node-tag-tlv;
              }
            }

            container dynamic-hostname-tlv {
              description "OSPF Dynamic Hostname";
              uses dynamic-hostname-tlv;
            }

            container sbfd-discriminator-tlv {
              description "OSPF S-BFD Discriminators";
              uses sbfd-discriminator-tlv;
            }

            container maximum-sid-depth-tlv {
              description "OSPF Maximum SID Depth (MSD) values";
              uses maximum-sid-depth-tlv;
            }
            uses unknown-tlvs;
          }
          container te-opaque {
            description "OSPFv2 Traffic Engineering (TE) opaque LSA.";
            reference "RFC 3630: Traffic Engineering (TE)
                       Extensions to OSPFv2";

            container router-address-tlv {
              description
                "Router address TLV.";
              leaf router-address {
                type inet:ipv4-address;
                description
                  "Router address.";
              }
            }

            container link-tlv {
              description "Describes a single link, and it is constructed
              of a set of Sub-TLVs.";
              leaf link-type {
                type router-link-type;
                mandatory true;
                description "Link type.";
              }
              leaf link-id {
                type union {
                  type inet:ipv4-address;
                  type yang:dotted-quad;
                }
                mandatory true;
                description "Link ID.";
              }
              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.";
                }
              }
              leaf admin-group {
                type uint32;
                description
                  "Administrative group/Resource Class/Color.";
              }
              uses unknown-tlvs;
            }
          }

          container extended-prefix-opaque {
            description "All extended prefix TLVs in the LSA.";
            list extended-prefix-tlv {
              description "Extended prefix TLV.";
              leaf route-type {
                type enumeration {
                  enum unspecified {
                    value "0";
                    description "Unspecified.";
                  }
                  enum intra-area {
                    value "1";
                    description "OSPF intra-area route.";
                  }
                  enum inter-area {
                    value "3";
                    description "OSPF inter-area route.";
                  }
                  enum external {
                    value "5";
                    description "OSPF External route.";
                  }
                  enum nssa {
                    value "7";
                    description "OSPF NSSA external route.";
                  }
                }
                description "Route type.";
              }
              container flags {
                leaf-list extended-prefix-flags {
                  type identityref {
                    base ospfv2-extended-prefix-flag;
                  }
                  description
                    "Extended prefix TLV flags list. This list will
                     contain identities for the prefix flags that
                     are set in the extended prefix flags.";
                }
                description "Prefix Flags.";
              }
              leaf prefix {
                type inet:ip-prefix;
                description "Address prefix.";
              }
              uses unknown-tlvs;
            }
          }

          container extended-link-opaque {
            description "All extended link TLVs in the LSA.";
            container extended-link-tlv {
              description "Extended link TLV.";
              uses ospfv2-router-link;
              container maximum-sid-depth-tlv {
                description "OSPF Maximum SID Depth (MSD) values";
                uses maximum-sid-depth-tlv;
              }
              uses unknown-tlvs;
            }
          }
        }
      }

      grouping ospfv3-lsa-options {
        description "OSPFv3 LSA options";
        container lsa-options {
          leaf-list lsa-options {
            type identityref {
              base ospfv3-lsa-option;
            }
            description
              "OSPFv3 LSA Option flags list. This list will contain
               the identities for the OSPFv3 LSA options that are
               set for the LSA.";
          }
          description "OSPFv3 LSA options.";
        }
      }

      grouping ospfv3-lsa-prefix {
        description
          "OSPFv3 LSA prefix.";

        leaf prefix {
          type inet:ip-prefix;
          description
            "LSA Prefix.";
        }
        container prefix-options {
          leaf-list prefix-options {
            type identityref {
              base ospfv3-prefix-option;
            }
            description
              "OSPFv3 prefix option flag list. This list will
               contain the identities for the OSPFv3 options
               that are set for the OSPFv3 prefix.";
          }
          description "Prefix options.";
        }
      }

      grouping ospfv3-lsa-external {
        description
          "AS-External and NSSA LSA.";
        leaf metric {
          type ospf-metric;
          description "Metric";
        }
        leaf flags {
          type bits {
            bit E {
              description
                "When set, the metric specified is a Type 2
                 external metric.";
            }
            bit F {
              description
                "When set, a Forwarding Address is included
                 in the LSA.";
            }
            bit T {
              description
                "When set, an External Route Tag is included
                 in the LSA.";
            }
          }
          description "Flags.";
        }

        leaf referenced-ls-type {
          type identityref {
            base ospfv3-lsa-type;
          }
          description "Referenced Link State type.";
        }
        leaf unknown-referenced-ls-type {
          type uint16;
          description
            "Value for an unknown Referenced Link State type.";
        }

        uses ospfv3-lsa-prefix;

        leaf forwarding-address {
          type inet:ipv6-address;
          description
            "Forwarding address.";
        }

        leaf external-route-tag {
          type uint32;
          description
```

```
            "Route tag.";
        }
        leaf referenced-link-state-id {
          type uint32;
          description
            "Referenced Link State ID.";
        }
      }

      grouping ospfv3-lsa-body {
        description "OSPFv3 LSA body.";
        container router {
          when "derived-from-or-self(../../header/type, "
             + "'ospfv3-router-lsa')" {
            description
              "Only applies to Router LSAs.";
          }
          description "Router LSA.";
          uses ospf-router-lsa-bits;
          uses ospfv3-lsa-options;

          container links {
            description "All router link.";
            list link {
              description "Router LSA link.";
              leaf interface-id {
                type uint32;
                description "Interface ID for link.";
              }
              leaf neighbor-interface-id {
                type uint32;
                description "Neighbor's Interface ID for link.";
              }
              leaf neighbor-router-id {
                type rt-types:router-id;
                description "Neighbor's Router ID for link.";
              }
              leaf type {
                type router-link-type;
                description "Link type: 1 - Point-to-Point Link
                                        2 - Transit Network Link
                                        3 - Stub Network Link
                                        4 - Virtual Link";
              }
              leaf metric {
                type uint16;
                  description "Link Metric.";
              }
            }
          }
        }
        container network {
          when "derived-from-or-self(../../header/type, "
             + "'ospfv3-network-lsa')" {
            description
              "Only applies to Network LSAs.";
          }
          description "Network LSA.";

          uses ospfv3-lsa-options;

          container attached-routers {
            description "All attached routers.";
            leaf-list attached-router {
              type rt-types:router-id;
              description
                "List of the routers attached to the network.";
            }
          }
        }
        container inter-area-prefix {
          when "derived-from-or-self(../../header/type, "
             + "'ospfv3-inter-area-prefix-lsa')" {
            description
              "Only applies to Inter-Area-Prefix LSAs.";
          }
          leaf metric {
            type ospf-metric;
            description "Inter-Area Prefix Metric";
          }
          uses ospfv3-lsa-prefix;
          description "Prefix LSA.";
        }
        container inter-area-router {
          when "derived-from-or-self(../../header/type, "
             + "'ospfv3-inter-area-router-lsa')" {
            description
              "Only applies to Inter-Area-Router LSAs.";
          }
          uses ospfv3-lsa-options;
          leaf metric {
            type ospf-metric;
            description "AS Boundary Router (ASBR) Metric.";
          }
          leaf destination-router-id {
            type rt-types:router-id;
```

```
            description
              "The Router ID of the ASBR described by the LSA.";
          }
          description "Inter-Area-Router LSA.";
        }
        container as-external {
          when "derived-from-or-self(../../header/type, "
             + "'ospfv3-as-external-lsa')" {
            description
              "Only applies to AS-external LSAs.";
          }

          uses ospfv3-lsa-external;

          description "AS-External LSA.";
        }
        container nssa {
          when "derived-from-or-self(../../header/type, "
             + "'ospfv3-nssa-lsa')" {
            description
              "Only applies to NSSA LSAs.";
          }
          uses ospfv3-lsa-external;

          description "NSSA LSA.";
        }
        container link {
          when "derived-from-or-self(../../header/type, "
             + "'ospfv3-link-lsa')" {
            description
              "Only applies to Link LSAs.";
          }
          leaf rtr-priority {
            type uint8;
            description
              "Router priority for DR election. A router with a
               higher priority will be preferred in the election
               and a value of 0 indicates the router is not
               eligible to become Designated Router or Backup
               Designated Router (BDR).";
          }
          uses ospfv3-lsa-options;

          leaf link-local-interface-address {
            type inet:ipv6-address;
            description
              "The originating router's link-local
               interface address for the link.";
```

```
         }

         leaf num-of-prefixes {
           type uint32;
           description "Number of prefixes.";
         }

         container prefixes {
           description "All prefixes for the link.";
           list prefix {
             description
               "List of prefixes associated with the link.";
             uses ospfv3-lsa-prefix;
           }
         }
         description "Link LSA.";
       }
       container intra-area-prefix {
         when "derived-from-or-self(../../header/type, "
            + "'ospfv3-intra-area-prefix-lsa')" {
           description
             "Only applies to Intra-Area-Prefix LSAs.";
         }
         description "Intra-Area-Prefix LSA.";

         leaf referenced-ls-type {
           type identityref {
             base ospfv3-lsa-type;
           }
           description "Referenced Link State type.";
         }
         leaf unknown-referenced-ls-type {
           type uint16;
           description
             "Value for an unknown Referenced Link State type.";
         }
         leaf referenced-link-state-id {
           type uint32;
           description
             "Referenced Link State ID.";
         }
         leaf referenced-adv-router {
           type rt-types:router-id;
           description
             "Referenced Advertising Router.";
         }

         leaf num-of-prefixes {
           type uint16;
           description "Number of prefixes.";
         }
         container prefixes {
           description "All prefixes in this LSA.";
           list prefix {
             description "List of prefixes in this LSA.";
             uses ospfv3-lsa-prefix;
             leaf metric {
               type ospf-metric;
               description "Prefix Metric.";
             }
           }
         }
       }
       container router-information {
         when "derived-from-or-self(../../header/type, "
            + "’ospfv3-router-information-lsa’)" {
           description
             "Only applies to Router Information LSAs (RFC7770).";
         }
         container router-capabilities-tlv {
           description
             "Informational and functional router capabilities";
           uses router-capabilities-tlv;
         }
         container node-tag-tlvs {
           description
             "All node tag tlvs.";
           list node-tag-tlv {
             description
               "Node tag tlv.";
             uses node-tag-tlv;
           }
         }
         container dynamic-hostname-tlv {
           description "OSPF Dynamic Hostname";
           uses dynamic-hostname-tlv;
         }
         container sbfd-discriminator-tlv {
           description "OSPF S-BFD Discriminators";
           uses sbfd-discriminator-tlv;
         }
         description "Router Information LSA.";
         reference "RFC 7770: Extensions for Advertising Router
                    Capabilities";
       }
     }
     grouping lsa-header {
       description
           "Common LSA for OSPFv2 and OSPFv3";
       leaf age {
         type uint16;
         mandatory true;
         description "LSA age.";
       }
       leaf type {
         type identityref {
           base ospf-lsa-type;
         }
         mandatory true;
         description "LSA type";
       }
       leaf adv-router {
         type rt-types:router-id;
         mandatory true;
         description "LSA advertising router.";
       }
       leaf seq-num {
         type uint32;
         mandatory true;
         description "LSA sequence number.";
       }
       leaf checksum {
         type fletcher-checksum16-type;
         mandatory true;
         description "LSA checksum.";
       }
       leaf length {
         type uint16;
         mandatory true;
         description "LSA length including the header.";
       }
     }

     grouping ospfv2-lsa {
       description
           "OSPFv2 LSA - LSAs are uniquely identified by
            the <LSA Type, Link-State ID, Advertising Router>
            tuple with the sequence number differentiating
            LSA instances.";
       container header {
         must "(derived-from(type, "
            + "'ospfv2-opaque-lsa-type') and "
            + "opaque-id and opaque-type) or "
            + "(not(derived-from(type, "
            + "'ospfv2-opaque-lsa-type')) "
            + "and not(opaque-id) and not(opaque-type))" {
           description
             "Opaque type and ID only apply to Opaque LSAs.";
         }
         description
           "Decoded OSPFv2 LSA header data.";

         container lsa-options {
           leaf-list lsa-options {
             type identityref {
               base ospfv2-lsa-option;
             }
             description
               "LSA option flags list. This list will contain
                the identities for the identities for the OSPFv2
                LSA options that are set.";
           }
           description
             "LSA options.";
         }

         leaf lsa-id {
           type yang:dotted-quad;
           mandatory true;
           description "Link-State ID.";
         }

         leaf opaque-type {
           type uint8;
           description "Opaque type.";
         }

         leaf opaque-id {
           type opaque-id;
           description "Opaque ID.";
         }

         uses lsa-header;
       }
       container body {
         description
           "Decoded OSPFv2 LSA body data.";
         uses ospfv2-lsa-body;
       }
     }

     grouping ospfv3-lsa {
       description
           "Decoded OSPFv3 LSA.";
       container header {
         description
           "Decoded OSPFv3 LSA header data.";
         leaf lsa-id {
           type uint32;
           mandatory true;
           description "OSPFv3 LSA ID.";
         }
         uses lsa-header;
       }
       container body {
         description
           "Decoded OSPF LSA body data.";
         uses ospfv3-lsa-body;
       }
     }
     grouping lsa-common {
       description
           "Common fields for OSPF LSA representation.";
       leaf decode-completed {
         type boolean;
         description
           "The OSPF LSA body was successfully decoded other than
            unknown TLVs. Unknown LSAs types and OSPFv2 unknown
            opaque LSA types are not decoded. Additionally,
            malformed LSAs are generally not accepted and will
            not be in the Link State Database.";
       }
       leaf raw-data {
         type yang:hex-string;
         description
           "The complete LSA in network byte
            order hexadecimal as received or originated.";
       }
     }

     grouping lsa {
       description
           "OSPF LSA.";
       uses lsa-common;
       choice version {
         description
           "OSPFv2 or OSPFv3 LSA body.";
         container ospfv2 {
           description "OSPFv2 LSA";
           uses ospfv2-lsa;
```

```
          }
          container ospfv3 {
            description "OSPFv3 LSA";
            uses ospfv3-lsa;
          }
        }
      }

      grouping lsa-key {
        description
          "OSPF LSA key - the database key for each LSA of a given
           type in the Link State DataBase (LSDB).";
        leaf lsa-id {
          type union {
            type yang:dotted-quad;
            type uint32;
          }
          description
            "Link-State ID.";
        }
        leaf adv-router {
          type rt-types:router-id;
          description
            "Advertising router.";
        }
      }

      grouping instance-stat {
        description "Per-instance statistics";
        leaf discontinuity-time {
          type yang:date-and-time;
          description
            "The time on the most recent occasion at which any one or
             more of this OSPF instance’s counters suffered a
             discontinuity.  If no such discontinuities have occurred
             since the OSPF instance was last re-initialized, then
             this node contains the time the OSPF instance was
             re-initialized which normally occurs when it was
             created.";
        }
        leaf originate-new-lsa-count {
          type yang:counter32;
          description
            "The number of new LSAs originated. Discontinuities in the
             value of this counter can occur when the OSPF instance is
             re-initialized.";
        }
        leaf rx-new-lsas-count {
          type yang:counter32;
          description
            "The number of new LSAs received. Discontinuities in the
             value of this counter can occur when the OSPF instance is
             re-initialized.";
        }
        leaf as-scope-lsa-count {
          type yang:gauge32;
          description "The number of AS-scope LSAs.";
        }
        leaf as-scope-lsa-chksum-sum {
          type uint32;
          description
            "The module 2**32 sum of the LSA checksums
             for AS-scope LSAs. The value should be treated as
             unsigned when comparing two sums of checksums. While
             differing checksums indicate a different combination
             of LSAs, equivalent checksums don’t guarantee that the
             LSAs are the same given that multiple combinations of
             LSAs can result in the same checksum.";
        }
        container database {
          description "Container for per AS-scope LSA statistics.";
          list as-scope-lsa-type {
            description "List of AS-scope LSA statistics";
            leaf lsa-type {
              type uint16;
              description "AS-Scope LSA type.";
            }
            leaf lsa-count {
              type yang:gauge32;
              description "The number of LSAs of the LSA type.";
            }
            leaf lsa-cksum-sum {
              type uint32;
              description
                "The module 2**32 sum of the LSA checksums
                 for the LSAs of this type. The value should be
                 treated as unsigned when comparing two sums of
                 checksums. While differing checksums indicate a
                 different combination of LSAs, equivalent checksums
                 don’t guarantee that the LSAs are the same given that
                 multiple combinations of LSAs can result in the same
                 checksum.";
            }
          }
        }
        uses instance-fast-reroute-state;
```

```
     }

     grouping area-stat {
       description "Per-area statistics.";
       leaf discontinuity-time {
         type yang:date-and-time;
         description
           "The time on the most recent occasion at which any one or
             more of this OSPF area’s counters suffered a
             discontinuity.  If no such discontinuities have occurred
             since the OSPF area was last re-initialized, then
             this node contains the time the OSPF area was
             re-initialized which normally occurs when it was
             created.";
       }
       leaf spf-runs-count {
         type yang:counter32;
         description
           "The number of times the intra-area SPF has run.
            Discontinuities in the value of this counter can occur
            when the OSPF area is re-initialized.";
       }
       leaf abr-count {
         type yang:gauge32;
         description
           "The total number of Area Border Routers (ABRs)
            reachable within this area.";
       }
       leaf asbr-count {
         type yang:gauge32;
         description
           "The total number of AS Boundary Routers (ASBRs).";
       }
       leaf ar-nssa-translator-event-count {
         type yang:counter32;
         description
           "The number of NSSA translator-state changes.
            Discontinuities in the value of this counter can occur
            when the OSPF area is re-initialized.";
       }
       leaf area-scope-lsa-count {
         type yang:gauge32;
         description
           "The number of area-scope LSAs in the area.";
       }
       leaf area-scope-lsa-cksum-sum {
         type uint32;
         description
```

```
            "The module 2**32 sum of the LSA checksums
             for area-scope LSAs. The value should be treated as
             unsigned when comparing two sums of checksums. While
             differing checksums indicate a different combination
             of LSAs, equivalent checksums don't guarantee that the
             LSAs are the same given that multiple combinations of
             LSAs can result in the same checksum.";
        }
        container database {
          description "Container for area-scope LSA type statistics.";
          list area-scope-lsa-type {
            description "List of area-scope LSA statistics";
            leaf lsa-type {
              type uint16;
              description "Area-scope LSA type.";
            }
            leaf lsa-count {
              type yang:gauge32;
              description "The number of LSAs of the LSA type.";
            }
            leaf lsa-cksum-sum {
              type uint32;
              description
                "The module 2**32 sum of the LSA checksums
                 for the LSAs of this type. The value should be
                 treated as unsigned when comparing two sums of
                 checksums. While differing checksums indicate a
                 different combination of LSAs, equivalent checksums
                 don't guarantee that the LSAs are the same given that
                 multiple combinations of LSAs can result in the same
                 checksum.";
            }
          }
        }
      }

      grouping interface-stat {
        description "Per-interface statistics";
        leaf discontinuity-time {
          type yang:date-and-time;
          description
            "The time on the most recent occasion at which any one or
             more of this OSPF interface's counters suffered a
             discontinuity.  If no such discontinuities have occurred
             since the OSPF interface was last re-initialized, then
             this node contains the time the OSPF interface was
             re-initialized which normally occurs when it was
             created.";
```

```
       }
       leaf if-event-count {
         type yang:counter32;
         description
           "The number of times this interface has changed its
            state or an error has occurred. Discontinuities in the
            value of this counter can occur when the OSPF interface
            is re-initialized.";
       }
       leaf link-scope-lsa-count {
         type yang:gauge32;
         description "The number of link-scope LSAs.";
       }
       leaf link-scope-lsa-cksum-sum {
         type uint32;
         description
           "The module 2**32 sum of the LSA checksums
            for link-scope LSAs. The value should be treated as
            unsigned when comparing two sums of checksums. While
            differing checksums indicate a different combination
            of LSAs, equivalent checksums don't guarantee that the
            LSAs are the same given that multiple combinations of
            LSAs can result in the same checksum.";
       }
       container database {
         description "Container for link-scope LSA type statistics.";
         list link-scope-lsa-type {
           description "List of link-scope LSA statistics";
           leaf lsa-type {
             type uint16;
             description "Link scope LSA type.";
           }
           leaf lsa-count {
             type yang:gauge32;
             description "The number of LSAs of the LSA type.";
           }
           leaf lsa-cksum-sum {
             type uint32;
             description
               "The module 2**32 sum of the LSA checksums
                for the LSAs of this type. The value should be
                treated as unsigned when comparing two sums of
                checksums. While differing checksums indicate a
                different combination of LSAs, equivalent checksums
                don't guarantee that the LSAs are the same given that
                multiple combinations of LSAs can result in the same
                checksum.";
           }
         }
       }
     }

     grouping neighbor-stat {
       description "Per-neighbor statistics.";
       leaf discontinuity-time {
         type yang:date-and-time;
         description
           "The time on the most recent occasion at which any one or
            more of this OSPF neighbor's counters suffered a
            discontinuity.  If no such discontinuities have occurred
            since the OSPF neighbor was last re-initialized, then
            this node contains the time the OSPF neighbor was
            re-initialized which normally occurs when the neighbor
            is dynamically discovered andcreated.";
       }
       leaf nbr-event-count {
         type yang:counter32;
         description
           "The number of times this neighbor has changed
            state or an error has occurred. Discontinuities in the
            value of this counter can occur when the OSPF neighbor
            is re-initialized.";
       }
       leaf nbr-retrans-qlen {
         type yang:gauge32;
         description
           "The current length of the retransmission queue.";
       }
     }

     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 instance-fast-reroute-state {
       description "IP-FRR state data grouping";

       container protected-routes {
         if-feature fast-reroute;
         config false;
         description "Instance protection statistics";

         list address-family-stats {
           key "address-family prefix alternate";
           description
             "Per Address Family protected prefix information";

           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
               "Indicates that this alternate is preferred.";
           }
           leaf non-best-reason {
             type string {
               length "1..255";
             }
             description
               "Information field to describe why the alternate
                is not best.";
           }
           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";
           }
         }
       }

       container unprotected-routes {
         if-feature fast-reroute;
         config false;
         description "List of prefixes that are not protected";

         list address-family-stats {
           key "address-family prefix";
           description
             "Per Address Family (AF) unprotected prefix statistics.";

           leaf address-family {
             type iana-rt-types:address-family;
             description "Address-family";
           }
           leaf prefix {
             type inet:ip-prefix;
             description "Unprotected prefix.";
           }
         }
       }

       list protection-statistics {
         key frr-protection-method;
         config false;
         description "List protection method statistics";

         leaf frr-protection-method {
           type string;
           description "Protection method used.";
         }
         list address-family-stats {
           key address-family;
           description "Per Address Family protection statistics.";

           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.";
           }
         }
       }
```

```
     }

     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;
           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
             "LFA configuration.";
         }
         description
           "Interface IP Fast-reroute configuration.";
       }
     }

     grouping interface-physical-link-config {
       description
         "Interface cost configuration that only applies to
          physical interfaces (non-virtual) and sham links.";
       leaf cost {
         type ospf-link-metric;
         description
```

```
           "Interface cost.";
       }
       leaf mtu-ignore {
         if-feature mtu-ignore;
         type boolean;
         description
           "Enable/Disable bypassing the MTU mismatch check in
            Database Description packets specified in RFC 2328,
            section 10.6.";
       }
       leaf prefix-suppression {
         if-feature prefix-suppression;
         type boolean;
         description
           "Suppress advertisement of the prefixes associated
            with the interface.";
       }
     }

     grouping interface-common-config {
       description
         "Common configuration for all types of interfaces,
          including virtual links and sham links.";

       leaf hello-interval {
         type uint16;
         units seconds;
         description
           "Interval between hello packets (seconds). It must
            be the same for all routers on the same network.
            Different networks, implementations, and deployments
            will use different hello-intervals. A sample value
            for a LAN network would be 10 seconds.";
         reference "RFC 2328: OSPF Version 2, Appendix C.3";
       }

       leaf dead-interval {
         type uint16;
         units seconds;
         must "../dead-interval > ../hello-interval" {
           error-message "The dead interval must be "
                       + "larger than the hello interval";
           description
             "The value must be greater than the 'hello-interval'.";
         }
         description
           "Interval after which a neighbor is declared down
            (seconds) if hello packets are not received. It is
```

```
              typically 3 or 4 times the hello-interval. A typical
              value for LAN networks is 40 seconds.";
            reference "RFC 2328: OSPF Version 2, Appendix C.3";
          }

          leaf retransmit-interval {
            type uint16 {
              range "1..3600";
            }
            units seconds;
            description
              "Interval between retransmitting unacknowledged Link
               State Advertisements (LSAs) (seconds). This should
               be well over the round-trip transmit delay for
               any two routers on the network. A sample value
               would be 5 seconds.";
            reference "RFC 2328: OSPF Version 2, Appendix C.3";
          }

          leaf transmit-delay {
            type uint16;
            units seconds;
            description
              "Estimated time needed to transmit Link State Update
               (LSU) packets on the interface (seconds). LSAs have
               their age incremented by this amount when advertised
               on the interface. A sample value would be 1 second.";
            reference "RFC 2328: OSPF Version 2, Appendix C.3";
          }

          leaf lls {
            if-feature lls;
            type boolean;
            description
              "Enable/Disable link-local signaling (LLS) support.";
          }

          container ttl-security {
            if-feature ttl-security;
            description "Time to Live (TTL) security check.";
            leaf enable {
              type boolean;
              description
                "Enable/Disable TTL security check.";
            }
            leaf hops {
              type uint8 {
                range "1..254";
              }
              default 1;
              description
                "Maximum number of hops that an OSPF packet may
                 have traversed before reception.";
            }
          }
          leaf enable {
            type boolean;
            default true;
            description
              "Enable/disable OSPF protocol on the interface.";
          }

          container authentication {
            description "Authentication configuration.";
            choice auth-type-selection {
              description
                "Options for OSPFv2/OSPFv3 authentication
                 configuration.";
              case ospfv2-auth {
                when "derived-from-or-self(../../../../../../rt:type, "
                  + "'ospfv2')" {
                  description "Applied to OSPFv2 only.";
                }
                leaf ospfv2-auth-trailer-rfc {
                  if-feature ospfv2-authentication-trailer;
                  type ospfv2-auth-trailer-rfc-version;
                  description
                    "Version of OSFPv2 authentication trailer support -
                     RFC 5709 or RFC 7474";
                }
                choice ospfv2-auth-specification {
                  description
                    "Key chain or explicit key parameter specification";
                  case auth-key-chain {
                    if-feature key-chain;
                    leaf ospfv2-key-chain {
                      type key-chain:key-chain-ref;
                      description
                        "key-chain name.";
                    }
                  }
                  case auth-key-explicit {
                    leaf ospfv2-key-id {
                      type uint32;
                      description
                        "Key Identifier";
```

```
                 }
                 leaf ospfv2-key {
                   type string;
                   description
                     "OSPFv2 authentication key. The
                      length of the key may be dependent on the
                      cryptographic algorithm.";
                 }
                 leaf ospfv2-crypto-algorithm {
                   type identityref {
                     base key-chain:crypto-algorithm;
                   }
                   description
                     "Cryptographic algorithm associated with key.";
                 }
               }
             }
           }
           case ospfv3-auth-ipsec {
             when "derived-from-or-self(../../../../../../rt:type, "
               +  "’ospfv3’)" {
               description "Applied to OSPFv3 only.";
             }
             if-feature ospfv3-authentication-ipsec;
             leaf sa {
                 type string;
                 description
                   "Security Association (SA) name.";
             }
           }
           case ospfv3-auth-trailer {
             when "derived-from-or-self(../../../../../../rt:type, "
               +  "’ospfv3’)" {
               description "Applied to OSPFv3 only.";
             }
             if-feature ospfv3-authentication-trailer;
             choice ospfv3-auth-specification {
               description
                 "Key chain or explicit key parameter specification";
               case auth-key-chain {
                 if-feature key-chain;
                 leaf ospfv3-key-chain {
                   type key-chain:key-chain-ref;
                   description
                     "key-chain name.";
                 }
               }
               case auth-key-explicit {
                 leaf ospfv3-sa-id {
                   type uint16;
                   description
                     "Security Association (SA) Identifier";
                 }
                 leaf ospfv3-key {
                   type string;
                   description
                     "OSPFv3 authentication key. The
                      length of the key may be dependent on the
                      cryptographic algorithm.";
                 }
                 leaf ospfv3-crypto-algorithm {
                   type identityref {
                     base key-chain:crypto-algorithm;
                   }
                   description
                     "Cryptographic algorithm associated with key.";
                 }
               }
             }
           }
         }
       }
     }

     grouping interface-config {
       description "Configuration for real interfaces.";

       leaf interface-type {
         type enumeration {
           enum "broadcast" {
             description
               "Specify OSPF broadcast multi-access network.";
           }
           enum "non-broadcast" {
             description
               "Specify OSPF Non-Broadcast Multi-Access
                (NBMA) network.";
           }
           enum "point-to-multipoint" {
             description
               "Specify OSPF point-to-multipoint network.";
           }
           enum "point-to-point" {
             description
               "Specify OSPF point-to-point network.";
           }
           enum "hybrid" {
             if-feature hybrid-interface;
             description
               "Specify OSPF hybrid broadcast/P2MP network.";
           }
         }
         description
           "Interface type.";
       }

       leaf passive {
         type boolean;
         description
           "Enable/Disable passive interface - a passive interface's
            prefix will be advertised but no neighbor adjacencies
            will be formed on the interface.";
       }

       leaf demand-circuit {
         if-feature demand-circuit;
         type boolean;
         description
           "Enable/Disable demand circuit.";
       }

       leaf priority {
         type uint8;
         description
           "Configure OSPF router priority. On multi-access network
            this value is for Designated Router (DR) election. The
            priority is ignored on other interface types. A router
            with a higher priority will be preferred in the election
            and a value of 0 indicates the router is not eligible to
            become Designated Router or Backup Designated Router
            (BDR).";
       }

       container multi-areas {
         if-feature multi-area-adj;
         description "Container for multi-area config.";
         list multi-area {
           key multi-area-id;
           description
             "Configure OSPF multi-area adjacency.";
           leaf multi-area-id {
             type area-id-type;
             description
               "Multi-area adjacency area ID.";
           }
           leaf cost {
             type ospf-link-metric;
             description
               "Interface cost for multi-area adjacency.";
           }
         }
       }

       container static-neighbors {
         description "Statically configured neighbors.";

         list neighbor {
           key "identifier";
           description
             "Specify a static OSPF neighbor.";

           leaf identifier {
             type inet:ip-address;
             description
               "Neighbor Router ID, IPv4 address, or IPv6 address.";
           }

           leaf cost {
             type ospf-link-metric;
             description
               "Neighbor cost. Different implementations have different
                default costs with some defaulting to a cost inversely
                proportional to the interface speed. Others will
                default to 1 equating the cost to a hop count." ;
           }
           leaf poll-interval {
             type uint16;
             units seconds;
             description
               "Neighbor poll interval (seconds) for sending OSPF
                hello packets to discover the neighbor on NBMA
                networks. This interval dictates the granularity for
                discovery of new neighbors. A sample would be
                120 seconds (2 minutes) for a legacy Packet Data
                Network (PDN) X.25 network.";
             reference "RFC 2328: OSPF Version 2, Appendix C.5";
           }
           leaf priority {
             type uint8;
             description
                "Neighbor priority for DR election. A router with a
                 higher priority will be preferred in the election
                 and a value of 0 indicates the router is not
                 eligible to become Designated Router or Backup
                 Designated Router (BDR).";
           }
         }
       }

       leaf node-flag {
         if-feature node-flag;
         type boolean;
         default false;
         description
           "Set prefix as identifying the advertising router.";
         reference "RFC 7684: OSPFv2 Prefix/Link Attribute
                    Advertisement";
       }

       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). Please replace YYYY with
           published RFC number for draft-ietf-bfd-yang.";
       }

       uses interface-fast-reroute-config;
       uses interface-common-config;
       uses interface-physical-link-config;
     }

     grouping neighbor-state {
       description
         "OSPF neighbor operational state.";

       leaf address {
         type inet:ip-address;
         config false;
         description
           "Neighbor address.";
       }
       leaf dr-router-id {
         type rt-types:router-id;
         config false;
         description "Neighbor’s Designated Router (DR) Router ID.";
       }

       leaf dr-ip-addr {
         type inet:ip-address;
         config false;
         description "Neighbor’s Designated Router (DR) IP address.";
       }

       leaf bdr-router-id {
         type rt-types:router-id;
         config false;
         description
           "Neighbor’s Backup Designated Router (BDR) Router ID.";
       }

       leaf bdr-ip-addr {
         type inet:ip-address;
         config false;
         description
           "Neighbor’s Backup Designated Router (BDR) IP Address.";
       }
       leaf state {
         type nbr-state-type;
         config false;
         description
           "OSPF neighbor state.";
       }
       leaf cost {
         type ospf-link-metric;
         config false;
         description "Cost to reach neighbor for Point-to-Multipoint
                      and Hybrid networks";
       }
       leaf dead-timer {
         type rt-types:timer-value-seconds16;
         config false;
         description "This timer tracks the remaining time before
                      the neighbor is declared dead.";
       }
       container statistics {
         config false;
         description "Per-neighbor statistics";
         uses neighbor-stat;
       }
     }

     grouping interface-common-state {
       description
         "OSPF interface common operational state.";
       reference "RFC2328 Section 9: OSPF Version2 -
                  The Interface Data Structure";
       leaf state {
         type if-state-type;
         config false;
         description "Interface state.";
       }

       leaf hello-timer {
         type rt-types:timer-value-seconds16;
         config false;
         description "This timer tracks the remaining time before
                      the next hello packet is sent on the
                      interface.";
       }

       leaf wait-timer {
         type rt-types:timer-value-seconds16;
         config false;
         description "This timer tracks the remaining time before
                      the interface exits the Waiting state.";
       }

       leaf dr-router-id {
         type rt-types:router-id;
         config false;
         description "Designated Router (DR) Router ID.";
       }

       leaf dr-ip-addr {
         type inet:ip-address;
         config false;
         description "Designated Router (DR) IP address.";
       }

       leaf bdr-router-id {
         type rt-types:router-id;
         config false;
         description "Backup Designated Router (BDR) Router ID.";
       }

       leaf bdr-ip-addr {
         type inet:ip-address;
         config false;
         description "Backup Designated Router (BDR) IP Address.";
       }

       container statistics {
         config false;
         description "Per-interface statistics";
         uses interface-stat;
       }

       container neighbors {
         config false;
         description "All neighbors for the interface.";
         list neighbor {
           key "neighbor-router-id";
           description
             "List of interface OSPF neighbors.";
           leaf neighbor-router-id {
             type rt-types:router-id;
             description
               "Neighbor Router ID.";
           }
           uses neighbor-state;
         }
       }
       container database {
         config false;
         description "Link-scope Link State Database.";
         list link-scope-lsa-type {
           key "lsa-type";
           description
             "List OSPF link-scope LSAs.";
           leaf lsa-type {
             type uint16;
             description "OSPF link-scope LSA type.";
           }
           container link-scope-lsas {
             description
               "All link-scope LSAs of this LSA type.";
             list link-scope-lsa {
               key "lsa-id adv-router";
               description "List of OSPF link-scope LSAs";
               uses lsa-key;
               uses lsa {
                 refine "version/ospfv2/ospfv2" {
                   must "derived-from-or-self( "
                      + "../../../../../../../../../../"
                      + "rt:type, 'ospfv2')" {
                     description "OSPFv2 LSA.";
                   }
                 }
                 refine "version/ospfv3/ospfv3" {
                   must "derived-from-or-self( "
                      + "../../../../../../../../../../"
                      + "rt:type, 'ospfv3')" {
```

```
                     description "OSPFv3 LSA.";
                   }
                 }
               }
             }
           }
         }
       }
     }

     grouping interface-state {
       description
         "OSPF interface operational state.";
       reference "RFC2328 Section 9: OSPF Version2 -
                  The Interface Data Structure";

       uses interface-common-state;
     }

     grouping virtual-link-config {
       description
         "OSPF virtual link configuration state.";

       uses interface-common-config;
     }

     grouping virtual-link-state {
       description
         "OSPF virtual link operational state.";

       leaf cost {
         type ospf-link-metric;
         config false;
         description
           "Virtual link interface cost.";
       }
       uses interface-common-state;
     }

     grouping sham-link-config {
       description
         "OSPF sham link configuration state.";

       uses interface-common-config;
       uses interface-physical-link-config;
     }

     grouping sham-link-state {
       description
         "OSPF sham link operational state.";
       uses interface-common-state;
     }

     grouping address-family-area-config {
       description
         "OSPF address-family specific area config state.";

       container ranges {
         description "Container for summary ranges";

         list range {
           key "prefix";
           description
             "Summarize routes matching address/mask -
              Applicable to Area Border Routers (ABRs) only.";
           leaf prefix {
             type inet:ip-prefix;
             description
               "IPv4 or IPv6 prefix";
           }
           leaf advertise {
             type boolean;
             description
               "Advertise or hide.";
           }
           leaf cost {
             type ospf-metric;
             description
              "Advertised cost of summary route.";
           }
         }
       }
     }

     grouping area-common-config {
       description
         "OSPF area common configuration state.";

       leaf summary {
         when "derived-from(../area-type,'stub-nssa-area')" {
           description
             "Summary advertisement into the stub/NSSA area.";
         }
         type boolean;
         description
           "Enable/Disable summary advertisement into the stub or
```

```
            NSSA area.";
       }
       leaf default-cost {
         when "derived-from(../area-type,'stub-nssa-area')" {
             description
               "Cost for LSA default route advertised into the
                stub or NSSA area.";
         }
         type ospf-metric;
         description
           "Set the summary default route cost for a
            stub or NSSA area.";
       }
     }

     grouping area-config {
       description
         "OSPF area configuration state.";

       leaf area-type {
         type identityref {
           base area-type;
         }
         default normal-area;
         description
           "Area type.";
       }

       uses area-common-config;
       uses address-family-area-config;
     }

     grouping area-state {
       description
         "OSPF area operational state.";

       container statistics {
         config false;
         description "Per-area statistics";
         uses area-stat;
       }

       container database {
         config false;
         description "Area-scope Link State Database.";
         list area-scope-lsa-type {
           key "lsa-type";
           description "List OSPF area-scope LSAs.";
           leaf lsa-type {
             type uint16;
             description "OSPF area-scope LSA type.";
           }
           container area-scope-lsas {
             description
               "All area-scope LSAs of an area-scope
                LSA type.";
             list area-scope-lsa {
               key "lsa-id adv-router";
               description "List of OSPF area-scope LSAs";
               uses lsa-key;
               uses lsa {
                 refine "version/ospfv2/ospfv2" {
                   must "derived-from-or-self( "
                      + "../../../../../../../../"
                      + "rt:type, 'ospfv2')" {
                     description "OSPFv2 LSA.";
                   }
                 }
                 refine "version/ospfv3/ospfv3" {
                   must "derived-from-or-self( "
                      + "../../../../../../../../"
                      + "rt:type, 'ospfv3')" {
                     description "OSPFv3 LSA.";
                   }
                 }
               }
             }
           }
         }
       }
     }

     grouping local-rib {
       description "Local-rib - RIB for Routes computed by the local
                    OSPF 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 route-type {
             type route-type;
             description "Route type for this route.";
           }
           leaf route-tag {
             type uint32;
             description "Route tag for this route.";
           }
         }
       }
     }

     grouping ietf-spf-delay {
       leaf initial-delay {
         type uint32;
         units milliseconds;
         description
           "Delay used while in QUIET state (milliseconds).";
       }
       leaf short-delay {
         type uint32;
         units milliseconds;
         description
           "Delay used while in SHORT_WAIT state (milliseconds).";
       }
       leaf long-delay {
         type uint32;
         units milliseconds;
         description
```

```
            "Delay used while in LONG_WAIT state (milliseconds).";
        }
        leaf hold-down {
          type uint32;
          units milliseconds;
          description
            "Timer used to consider an IGP stability period
             (milliseconds).";
        }
        leaf time-to-learn {
          type uint32;
          units milliseconds;
          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;
          config false;
          description
            "Remaining time until time-to-learn timer fires.";
        }
        leaf remaining-hold-down {
          type rt-types:timer-value-milliseconds;
          config false;
          description
            "Remaining time until hold-down timer fires.";
        }
        leaf last-event-received {
          type yang:timestamp;
          config false;
          description
```

```
             "Time of last SPF triggering event.";
         }
         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
           "OSPF 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 instance-config {
         description
           "OSPF instance config state.";

         leaf enable {
           type boolean;
           default true;
           description
             "Enable/Disable the protocol.";
         }
         leaf explicit-router-id {
           if-feature explicit-router-id;
           type rt-types:router-id;
           description
             "Defined in RFC 2328. A 32-bit number
              that uniquely identifies the router.";
         }

         container preference {
           description
             "Route preference configuration. In many
              implementations, preference is referred to as
              administrative distance.";
           reference
             "RFC 8349: A YANG Data Model for Routing Management
              (NMDA Version)";
           choice scope {
             description
               "Options for expressing preference
                as single or multiple values.";
             case single-value {
               leaf all {
                 type uint8;
                 description
                   "Preference for intra-area, inter-area, and
                    external routes.";
               }
             }
             case multi-values {
               choice granularity {
                 description
                   "Options for expressing preference
                    for intra-area and inter-area routes.";
                 case detail {
                   leaf intra-area {
                     type uint8;
                     description
                       "Preference for intra-area routes.";
                   }
                   leaf inter-area {
                     type uint8;
                     description
                       "Preference for inter-area routes.";
                   }
                 }
                 case coarse {
                   leaf internal {
                     type uint8;
                     description
                       "Preference for both intra-area and
                        inter-area routes.";
                   }
                 }
               }
               leaf external {
                 type uint8;
                 description
                   "Preference for AS external routes.";
               }
             }
           }
         }

         container nsr {
           if-feature nsr;
           description
             "Non-Stop Routing (NSR) config state.";
           leaf enable {
             type boolean;
             description
               "Enable/Disable NSR.";
           }
         }

         container graceful-restart {
           if-feature graceful-restart;
           description
             "Graceful restart config state.";
           reference "RFC 3623: OSPF Graceful Restart
                      RFC 5187: OSPFv3 Graceful Restart";
           leaf enable {
             type boolean;
             description
               "Enable/Disable graceful restart as defined in RFC 3623
                for OSPFv2 and RFC 5187 for OSPFv3.";
           }
           leaf helper-enable {
             type boolean;
             description
               "Enable graceful restart helper support for restarting
                routers (RFC 3623 Section 3).";
           }
           leaf restart-interval {
             type uint16 {
                range "1..1800";
             }
```

```
            units seconds;
            default "120";
            description
              "Interval to attempt graceful restart prior
               to failing (RFC 3623 Section B.1) (seconds)";
          }
          leaf helper-strict-lsa-checking {
            type boolean;
            description
              "Terminate graceful restart when an LSA topology change
               is detected (RFC 3623 Section B.2).";
          }
        }

        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.";
          }
        }

        container spf-control {
          leaf paths {
            if-feature max-ecmp;
            type uint16 {
              range "1..65535";
            }
            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.";
       }

       container database-control {
         leaf max-lsa {
           if-feature max-lsa;
           type uint32 {
             range "1..4294967294";
           }
           description
             "Maximum number of LSAs OSPF the router will accept.";
         }
         description "Database maintenance control.";
       }

       container stub-router {
         if-feature stub-router;
         description "Set maximum metric configuration";

         choice trigger {
           description
             "Specific triggers which will enable stub
              router state.";
           container always {
             presence
               "Enables unconditional stub router support";
             description
               "Unconditional stub router state (advertise
                transit links with MaxLinkMetric";
             reference "RFC 6987: OSPF Stub Router
                        Advertisement";
           }
         }
       }

       container mpls {
         description
           "OSPF MPLS config state.";
         container te-rid {
           if-feature te-rid;
           description
             "Stable OSPF Router IP Address used for Traffic
```

```
              Engineering (TE)";
           leaf ipv4-router-id {
             type inet:ipv4-address;
             description
               "Explicitly configure the TE IPv4 Router ID.";
           }
           leaf ipv6-router-id {
             type inet:ipv6-address;
             description
               "Explicitly configure the TE IPv6 Router ID.";
           }
         }
         container ldp {
           description
             "OSPF MPLS LDP config state.";
           leaf igp-sync {
             if-feature ldp-igp-sync;
             type boolean;
             description
               "Enable LDP IGP synchronization.";
           }
         }
       }
       uses instance-fast-reroute-config;
       uses node-tag-config;
     }

     grouping instance-state {
       description
         "OSPF instance operational state.";

       leaf router-id {
         type rt-types:router-id;
         config false;
         description
           "Defined in RFC 2328. A 32-bit number
            that uniquely identifies the router.";
       }

       uses local-rib;

       container statistics {
         config false;
         description "Per-instance statistics";
         uses instance-stat;
       }

       container database {
         config false;
         description "AS-scope Link State Database.";
         list as-scope-lsa-type {
           key "lsa-type";
           description "List OSPF AS-scope LSAs.";
           leaf lsa-type {
             type uint16;
             description "OSPF AS scope LSA type.";
           }
           container as-scope-lsas {
             description "All AS-scope of LSA of this LSA type.";
             list as-scope-lsa {
               key "lsa-id adv-router";
               description "List of OSPF AS-scope LSAs";
               uses lsa-key;
               uses lsa {
                 refine "version/ospfv2/ospfv2" {
                   must "derived-from-or-self( "
                      + "../../../../../../../"
                      + "rt:type, 'ospfv2')" {
                     description "OSPFv2 LSA.";
                   }
                 }
                 refine "version/ospfv3/ospfv3" {
                   must "derived-from-or-self( "
                      + "../../../../../../../"
                      + "rt:type, 'ospfv3')" {
                     description "OSPFv3 LSA.";
                   }
                 }
               }
             }
           }
         }
       }
       uses spf-log;
       uses lsa-log;
     }

     grouping multi-topology-area-common-config {
       description
         "OSPF multi-topology area common configuration state.";
       leaf summary {
         when "derived-from(../../../area-type, 'stub-nssa-area')" {
           description
             "Summary advertisement into the stub/NSSA area.";
         }
         type boolean;
         description
           "Enable/Disable summary advertisement into the
            topology in the stub or NSSA area.";
       }
       leaf default-cost {
         when "derived-from(../../../area-type, 'stub-nssa-area')" {
           description
             "Cost for LSA default route advertised into the
              topology into the stub or NSSA area.";
         }
         type ospf-metric;
         description
           "Set the summary default route cost for a
            stub or NSSA area.";
       }
     }

     grouping multi-topology-area-config {
       description
         "OSPF multi-topology area configuration state.";

       uses multi-topology-area-common-config;
       uses address-family-area-config;
     }

     grouping multi-topology-state {
       description
         "OSPF multi-topology operational state.";

       uses local-rib;
     }

     grouping multi-topology-interface-config {
       description
         "OSPF multi-topology configuration state.";

       leaf cost {
        type ospf-link-metric;
        description
          "Interface cost for this topology.";
       }
     }

     grouping ospfv3-interface-config {
       description
         "OSPFv3 interface specific configuration state.";

       leaf instance-id {
         type uint8 {
           range "0 .. 31";
         }
         description
           "OSPFv3 instance ID.";
       }
     }

     grouping ospfv3-interface-state {
       description
         "OSPFv3 interface specific operational state.";

       leaf interface-id {
         type uint16;
         config false;
         description
           "OSPFv3 interface ID.";
       }
     }

     grouping lsa-identifiers {
       description
         "The parameters that uniquely identify an LSA.";
       leaf area-id {
         type area-id-type;
         description
           "Area ID";
       }
       leaf type {
         type uint16;
         description
           "LSA type.";
       }
       leaf lsa-id {
         type union {
           type inet:ipv4-address;
           type yang:dotted-quad;
         }
         description "Link-State ID.";
       }
       leaf adv-router {
         type rt-types:router-id;
         description
           "LSA advertising router.";
       }
       leaf seq-num {
         type uint32;
         description
```

```
            "LSA sequence number.";
        }
      }

      grouping spf-log {
        description
          "Grouping for SPF log.";
        container spf-log {
          config false;
          description
            "This container lists the SPF log.";
          list event {
            key id;
            description
              "List of SPF log entries represented
               as a wrapping buffer in chronological
               order with the oldest entry returned
               first.";
            leaf id {
              type uint32;
              description
                "Event identifier - Purely internal value.";
            }
            leaf spf-type {
              type enumeration {
                enum full {
                  description
                    "SPF computation was a Full SPF.";
                }
                enum intra {
                  description
                    "SPF computation was only for intra-area routes.";
                }
                enum inter {
                  description
                    "SPF computation was only for inter-area
                     summary routes.";
                }
                enum external {
                  description
                    "SPF computation was only for AS external routes.";
                }
              }
              description
                "The SPF computation type for the SPF log entry.";
            }
            leaf schedule-timestamp {
              type yang:timestamp;
```

```
             description
               "This is the timestamp when the computation was
                scheduled.";
           }
           leaf start-timestamp {
             type yang:timestamp;
             description
               "This is the timestamp when the computation was
                started.";
           }
           leaf end-timestamp {
             type yang:timestamp;
             description
               "This the timestamp when the computation was
                completed.";
           }
           list trigger-lsa {
             description
               "The list of LSAs that triggered the computation.";
             uses lsa-identifiers;
           }
         }
       }
     }

     grouping lsa-log {
       description
         "Grouping for the LSA log.";
       container lsa-log {
         config false;
         description
           "This container lists the LSA log.
            Local LSA modifications are also included
            in the list.";
         list event {
           key id;
           description
             "List of LSA log entries represented
              as a wrapping buffer in chronological order
              with the oldest entries returned first.";
           leaf id {
             type uint32;
             description
               "Event identifier - purely internal value.";
           }
           container lsa {
             description
               "This container describes the logged LSA.";
             uses lsa-identifiers;
           }
           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 lsa-log-reason;
             }
             description
               "This reason for the LSA log entry.";
           }
         }
       }
     }

     augment "/rt:routing/rt:control-plane-protocols/"
           + "rt:control-plane-protocol" {
        when "derived-from(rt:type, 'ospf')" {
         description
           "This augmentation is only valid for a routing protocol
            instance of OSPF (type 'ospfv2' or 'ospfv3').";
       }
       description "OSPF protocol ietf-routing module
                    control-plane-protocol augmentation.";

       container ospf {
         description
           "OSPF protocol Instance";

         leaf address-family {
           type iana-rt-types:address-family;
           description
             "Address-family of the instance.";
         }

         uses instance-config;
         uses instance-state;

         container areas {
           description "All areas.";
           list area {
             key "area-id";
             description
```

```
               "List of OSPF areas";
             leaf area-id {
               type area-id-type;
               description
                 "Area ID";
             }

             uses area-config;
             uses area-state;

             container virtual-links {
               when "derived-from-or-self(../area-type, ’normal-area’) "
                  + "and ../area-id = ’0.0.0.0’" {
                 description
                   "Virtual links must be in backbone area.";
                 }
               description "All virtual links.";
               list virtual-link {
                 key "transit-area-id router-id";
                 description
                   "OSPF virtual link";
                 leaf transit-area-id {
                   type leafref {
                     path "../../../../area/area-id";
                   }
                   must "derived-from-or-self("
                     + "../../../../area[area-id=current()]/area-type, "
                     + "’normal-area’) and "
                     + "../../../../area[area-id=current()]/area-id != "
                     + "’0.0.0.0’" {
                     error-message "Virtual link transit area must "
                                 + "be non-zero.";
                     description
                       "Virtual-link transit area must be
                        non-zero area.";
                   }
                   description
                     "Virtual link transit area ID.";
                 }
                 leaf router-id {
                   type rt-types:router-id;
                   description
                     "Virtual Link remote endpoint Router ID.";
                 }

                 uses virtual-link-config;
                 uses virtual-link-state;
               }
             }
             container sham-links {
               if-feature pe-ce-protocol;
               description "All sham links.";
               list sham-link {
                 key "local-id remote-id";
                 description
                   "OSPF sham link";
                 leaf local-id {
                   type inet:ip-address;
                   description
                     "Address of the local sham Link endpoint.";
                 }
                 leaf remote-id {
                   type inet:ip-address;
                   description
                     "Address of the remote sham Link endpoint.";
                 }
                 uses sham-link-config;
                 uses sham-link-state;
               }
             }
             container interfaces {
               description "All interfaces.";
               list interface {
                 key "name";
                 description
                   "List of OSPF interfaces.";
                 leaf name {
                   type if:interface-ref;
                   description
                     "Interface name reference.";
                 }
                 uses interface-config;
                 uses interface-state;
               }
             }
           }
         }
       }
     }

     augment "/rt:routing/rt:control-plane-protocols/"
           + "rt:control-plane-protocol/ospf" {
       when "derived-from(../rt:type, 'ospf')" {
         description
           "This augmentation is only valid for OSPF
            (type 'ospfv2' or 'ospfv3').";
       }
       if-feature multi-topology;
       description
         "OSPF multi-topology instance configuration
          state augmentation.";
       container topologies {
         description "All topologies.";
         list topology {
           key "name";
           description
             "OSPF topology - The OSPF topology address-family
              must coincide with the routing-instance
              address-family.";
           leaf name {
             type leafref {
               path "../../../../../../rt:ribs/rt:rib/rt:name";
             }
             description "RIB name corresponding to the OSPF
                          topology.";
           }

           uses multi-topology-state;
         }
       }
     }

     augment "/rt:routing/rt:control-plane-protocols/"
           + "rt:control-plane-protocol/ospf/"
           + "areas/area" {
       when "derived-from-or-self(../../../rt:type, "
          + "'ospfv2')" {
         description
           "This augmentation is only valid for OSPFv2.";
       }
       if-feature multi-topology;
       description
         "OSPF multi-topology area configuration state
          augmentation.";
       container topologies {
         description "All topologies for the area.";
         list topology {
           key "name";
           description "OSPF area topology.";
           leaf name {
             type leafref {
               path "../../../../../../../../"
                  + "rt:ribs/rt:rib/rt:name";
             }
             description
               "Single topology enabled for this area.";
           }

           uses multi-topology-area-config;
         }
       }
     }

     augment "/rt:routing/rt:control-plane-protocols/"
           + "rt:control-plane-protocol/ospf/"
           + "areas/area/interfaces/interface" {
       when "derived-from-or-self(../../../../../rt:type, "
          + "'ospfv2')" {
         description
           "This augmentation is only valid for OSPFv2.";
       }
       if-feature multi-topology;
       description
         "OSPF multi-topology interface configuration state
          augmentation.";
       container topologies {
         description "All topologies for the interface.";
         list topology {
           key "name";
           description "OSPF interface topology.";
           leaf name {
             type leafref {
               path "../../../../../../../../../../../"
                  + "rt:ribs/rt:rib/rt:name";
             }
             description
               "Single topology enabled on this interface.";
           }

           uses multi-topology-interface-config;
         }
       }
     }

     augment "/rt:routing/rt:control-plane-protocols/"
           + "rt:control-plane-protocol/ospf/"
           + "areas/area/interfaces/interface" {
       when "derived-from-or-self(../../../../../rt:type, "
          + "'ospfv3')" {
         description
           "This augmentation is only valid for OSPFv3.";
       }
       description
         "OSPFv3 interface specific configuration state
          augmentation.";
       uses ospfv3-interface-config;
       uses ospfv3-interface-state;
     }

     grouping route-content {
       description
         "This grouping defines OSPF-specific route attributes.";
       leaf metric {
         type uint32;
         description "OSPF route metric.";
       }
       leaf tag {
         type uint32;
         default "0";
         description "OSPF route tag.";
       }
       leaf route-type {
         type route-type;
         description "OSPF route type";
       }
     }

     augment "/rt:routing/rt:ribs/rt:rib/rt:routes/rt:route" {
       when "derived-from(rt:source-protocol, 'ospf')" {
         description
           "This augmentation is only valid for routes whose
            source protocol is OSPF.";
       }
       description
         "OSPF-specific route attributes.";
       uses route-content;
     }

     /*
      * RPCs
      */

     rpc clear-neighbor {
       description
         "This RPC request clears a particular set of OSPF neighbors.
          If the operation fails for OSPF internal reason, then
          error-tag and error-app-tag should be set to a meaningful
          value.";
       input {
         leaf routing-protocol-name {
           type leafref {
             path "/rt:routing/rt:control-plane-protocols/"
                + "rt:control-plane-protocol/rt:name";
           }
           mandatory "true";
           description
             "OSPF protocol instance which information for neighbors
              are to be cleared.

              If the referenced OSPF instance doesn't exist, then
              this operation SHALL fail with error-tag 'data-missing'
              and error-app-tag
              'routing-protocol-instance-not-found'.";
         }

         leaf interface {
           type if:interface-ref;
             description
               "Name of the OSPF interface for which neighbors are to
                be cleared.

                If the referenced OSPF interface doesn't exist, then
                this operation SHALL fail with error-tag
                'data-missing'  and error-app-tag
                'ospf-interface-not-found'.";
         }
       }
     }

     rpc clear-database {
       description
         "This RPC request clears a particular OSPF Link State
          Database. If the operation fails for OSPF internal reason,
          then error-tag and error-app-tag should be set to a
          meaningful value.";
       input {
         leaf routing-protocol-name {
           type leafref {
             path "/rt:routing/rt:control-plane-protocols/"
                + "rt:control-plane-protocol/rt:name";
           }
           mandatory "true";
           description
             "OSPF protocol instance whose Link State Database is to
              be cleared.

              If the referenced OSPF instance doesn't exist, then
              this operation SHALL fail with error-tag 'data-missing'
              and error-app-tag
              'routing-protocol-instance-not-found'.";
         }
       }
     }

     /*
      * Notifications
      */

     grouping notification-instance-hdr {
       description
        "This grouping describes common instance specific
         data for OSPF notifications.";

       leaf routing-protocol-name {
         type leafref {
           path "/rt:routing/rt:control-plane-protocols/"
              + "rt:control-plane-protocol/rt:name";
         }
         must "derived-from( "
           + "/rt:routing/rt:control-plane-protocols/"
           + "rt:control-plane-protocol[rt:name=current()]/"
           + "rt:type, 'ospf')";
         description
          "OSPF routing protocol instance name.";
       }

       leaf address-family {
         type leafref {
           path "/rt:routing/"
              + "rt:control-plane-protocols/rt:control-plane-protocol"
              + "[rt:name=current()/../routing-protocol-name]/"
              + "ospf/address-family";
         }
         description
           "Address family of the OSPF instance.";
       }
     }

     grouping notification-interface {
       description
         "This grouping provides interface information
          for the OSPF interface specific notification.";

       choice if-link-type-selection {
         description
           "Options for link type.";
         container interface {
           description "Normal interface.";
           leaf interface {
             type if:interface-ref;
             description "Interface.";
           }
         }
         container virtual-link {
           description "virtual-link.";
           leaf transit-area-id {
             type area-id-type;
             description "Area ID.";
           }
           leaf neighbor-router-id {
             type rt-types:router-id;
             description "Neighbor Router ID.";
           }
         }
         container sham-link {
           description "sham link.";
           leaf area-id {
             type area-id-type;
             description "Area ID.";
           }
           leaf local-ip-addr {
             type inet:ip-address;
             description "Sham link local address.";
           }
           leaf remote-ip-addr {
             type inet:ip-address;
             description "Sham link remote address.";
           }
         }
       }
     }

     grouping notification-neighbor {
       description
         "This grouping provides the neighbor information
          for neighbor specific notifications.";

       leaf neighbor-router-id {
         type rt-types:router-id;
         description "Neighbor Router ID.";
       }

       leaf neighbor-ip-addr {
         type inet:ip-address;
         description "Neighbor address.";
       }
     }

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

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

     notification if-config-error {
       uses notification-instance-hdr;
       uses notification-interface;

       leaf packet-source {
         type inet:ip-address;
         description "Source address.";
       }

       leaf packet-type {
         type packet-type;
         description "OSPF packet type.";
       }

       leaf error {
         type enumeration {
           enum "bad-version" {
             description "Bad version.";
           }
           enum "area-mismatch" {
             description "Area mismatch.";
           }
           enum "unknown-nbma-nbr" {
             description "Unknown NBMA neighbor.";
           }
           enum "unknown-virtual-nbr" {
             description "Unknown virtual link neighbor.";
           }
           enum "auth-type-mismatch" {
             description "Auth type mismatch.";
           }
           enum "auth-failure" {
             description "Auth failure.";
           }
           enum "net-mask-mismatch" {
             description "Network mask mismatch.";
           }
           enum "hello-interval-mismatch" {
             description "Hello interval mismatch.";
           }
           enum "dead-interval-mismatch" {
             description "Dead interval mismatch.";
           }
           enum "option-mismatch" {
             description "Option mismatch.";
           }
           enum "mtu-mismatch" {
             description "MTU mismatch.";
           }
           enum "duplicate-router-id" {
             description "Duplicate Router ID.";
           }
           enum "no-error" {
             description "No error.";
           }
         }
         description "Error code.";
       }
       description
         "This notification is sent when an interface
          config error is detected.";
     }

     notification nbr-state-change {
       uses notification-instance-hdr;
       uses notification-interface;
       uses notification-neighbor;

       leaf state {
         type nbr-state-type;
         description "Neighbor state.";
       }

       description
         "This notification is sent when a neighbor
          state change is detected.";
     }

     notification nbr-restart-helper-status-change {
       uses notification-instance-hdr;
       uses notification-interface;
       uses notification-neighbor;

       leaf status {
         type restart-helper-status-type;
         description "Restart helper status.";
       }

       leaf age {
         type rt-types:timer-value-seconds16;
         description
           "Remaining time in current OSPF graceful restart
            interval when the router is acting as a restart
            helper for the neighbor.";
       }

       leaf exit-reason {
         type restart-exit-reason-type;
         description
           "Restart helper exit reason.";
       }
       description
         "This notification is sent when a neighbor restart
          helper status change is detected.";
     }

     notification if-rx-bad-packet {
       uses notification-instance-hdr;
       uses notification-interface;

       leaf packet-source {
         type inet:ip-address;
         description "Source address.";
       }

       leaf packet-type {
         type packet-type;
         description "OSPF packet type.";
       }

       description
         "This notification is sent when an OSPF packet that
          cannot be parsed is received on an OSPF interface.";
     }

     notification lsdb-approaching-overflow {
       uses notification-instance-hdr;
       leaf ext-lsdb-limit {
         type uint32;
         description
           "The maximum number of non-default AS-external LSAs
            entries that can be stored in the Link State Database.";
       }

       description
         "This notification is sent when the number of LSAs
          in the router's Link State Database has exceeded
          ninety percent of the AS-external limit (ext-lsdb-limit).";
     }

     notification lsdb-overflow {
       uses notification-instance-hdr;

       leaf ext-lsdb-limit {
         type uint32;
         description
           "The maximum number of non-default AS-external LSAs
            entries that can be stored in the Link State Database.";
       }

       description
         "This notification is sent when the number of LSAs
          in the router's Link State Database has exceeded the
          AS-external limit (ext-lsdb-limit).";
     }

     notification nssa-translator-status-change {
       uses notification-instance-hdr;

       leaf area-id {
         type area-id-type;
         description "Area ID.";
       }

       leaf status {
         type nssa-translator-state-type;
         description
           "NSSA translator status.";
       }

       description
         "This notification is sent when there is a change
          in the router's role in translating OSPF NSSA LSAs
          to OSPF AS-External LSAs.";
     }
     notification restart-status-change {
       uses notification-instance-hdr;

       leaf status {
         type restart-status-type;
         description
           "Restart status.";
       }

       leaf restart-interval {
         type uint16 {
           range 1..1800;
         }
         units seconds;
         default "120";
         description
           "Restart interval.";
       }

       leaf exit-reason {
         type restart-exit-reason-type;
         description
           "Restart exit reason.";
       }

       description
         "This notification is sent when the graceful restart
          state for the router has changed.";
     }
   }
   <CODE ENDS>
```

4. Security Considerations

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

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

There are a number of data nodes defined in ietf-ospf.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. Writable data node represent configuration of each instance, area, virtual link, sham-link, and interface. These correspond to the following schema nodes:

```
/ospf

/ospf/areas/

/ospf/areas/area[area-id]

/ospf/virtual-links/

/ospf/virtual-links/virtual-link[transit-area-id router-id]

/ospf/areas/area[area-id]/interfaces

/ospf/areas/area[area-id]/interfaces/interface[name]

/ospf/area/area[area-id]/sham-links

/ospf/area/area[area-id]/sham-links/sham-link[local-id remote-id]
```

For OSPF, the ability to modify OSPF configuration will allow the entire OSPF domain to be compromised including peering with unauthorized routers to misroute traffic or mount a massive Denial-of-Service (DoS) attack. For example, adding OSPF on any unprotected interface could allow an OSPF adjacency to be formed with an unauthorized and malicious neighbor. Once an adjacency is formed, traffic could be hijacked. As a simpler example, a Denial-of-Service attack could be mounted by changing the cost of an OSPF interface to be asymmetric such that a hard routing loop ensues. In general, unauthorized modification of most OSPF features will pose there own set of security risks and the "Security Considerations" in the respective reference RFCs should be consulted.

Some of the readable data nodes in the ietf-ospf.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. There is a separate Link State Database for each instance, area, virtual link, sham-link, and interface. These correspond to the following schema nodes:

/ospf/database

/ospf/areas/area[area-id]/database

/ospf/virtual-links/virtual-link[transit-area-id router-id]/database

/ospf/areas/area[area-id]/interfaces/interface[name]/database

/ospf/area/area[area-id]/sham-links/sham-link[local-id remote-id]/database

Exposure of the Link State Database includes information beyond the scope of the OSPF router and this may be undesirable since exposure may facilitate other attacks. Additionally, in the case of an area LSDB, the complete IP network topology and, if deployed, the traffic engineering topology of the OSPF area can be reconstucted. Network operators may consider their topologies to be sensitive confidential data.

For OSPF authentication, configuration is supported via the specification of key-chains [RFC8177] or the direct 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.

Additionally, local specification of OSPF authentication keys and the associated authentication algorithm is supported for legacy implementations that do not support key-chains [RFC8177] It is RECOMMENDED that implementations migrate to key-chains due the seamless support of key and algorithm rollover, as well as, the hexadecimal key specification affording more key entropy, and encryption of keys using the Advanced Encryption Standard (AES) Key Wrap Padding Algorithm [RFC5649].

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 supports the "clear-neighbor" 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.

The actual authentication key data (whether locally specified or part of a key-chain) is sensitive and needs to be kept secret from unauthorized parties; compromise of the key data would allow an attacker to forge OSPF traffic that would be accepted as authentic, potentially compromising the entirety OSPF domain.

5. IANA Considerations

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

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

This document registers a YANG module in the YANG Module Names registry [RFC6020].

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

6. Acknowledgements

The authors wish to thank Yi Yang, Alexander Clemm, Gaurav Gupta, Ladislav Lhotka, Stephane Litkowski, Greg Hankins, Manish Gupta, Michael Darwish, and Alan Davey for their thorough reviews and helpful comments.

Thanks to Tom Petch for last call review and improvement of the document organization.

Thanks to Alvaro Retana for AD comments.

Thanks to Benjamin Kaduk, Suresh Krishnan, and Roman Dannyliw for IESG review comments.

This document was produced using Marshall Rose's xml2rfc tool.

Author affiliation with The MITRE Corporation is provided for identification purposes only, and is not intended to convey or imply MITRE's concurrence with, or support for, the positions, opinions or viewpoints expressed. MITRE has approved this document for Public Release, Distribution Unlimited, with Public Release Case Number 18-3194.

7. References

7.1. 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.

[RFC1765] Moy, J., "OSPF Database Overflow", RFC 1765, DOI 10.17487/RFC1765, March 1995, <https://www.rfc-editor.org/info/rfc1765>.

[RFC1793] Moy, J., "Extending OSPF to Support Demand Circuits", RFC 1793, DOI 10.17487/RFC1793, April 1995, <https://www.rfc-editor.org/info/rfc1793>.

[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>.

[RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, DOI 10.17487/RFC2328, April 1998, <https://www.rfc-editor.org/info/rfc2328>.

[RFC3101] Murphy, P., "The OSPF Not-So-Stubby Area (NSSA) Option", RFC 3101, DOI 10.17487/RFC3101, January 2003, <https://www.rfc-editor.org/info/rfc3101>.

[RFC3623] Moy, J., Pillay-Esnault, P., and A. Lindem, "Graceful OSPF Restart", RFC 3623, DOI 10.17487/RFC3623, November 2003, <https://www.rfc-editor.org/info/rfc3623>.

[RFC3630] Katz, D., Kompella, K., and D. Yeung, "Traffic Engineering (TE) Extensions to OSPF Version 2", RFC 3630, DOI 10.17487/RFC3630, September 2003, <https://www.rfc-editor.org/info/rfc3630>.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <https://www.rfc-editor.org/info/rfc3688>.

[RFC4552] Gupta, M. and N. Melam, "Authentication/Confidentiality for OSPFv3", RFC 4552, DOI 10.17487/RFC4552, June 2006, <https://www.rfc-editor.org/info/rfc4552>.

[RFC4576] Rosen, E., Psenak, P., and P. Pillay-Esnault, "Using a Link State Advertisement (LSA) Options Bit to Prevent Looping in BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4576, DOI 10.17487/RFC4576, June 2006, <https://www.rfc-editor.org/info/rfc4576>.

[RFC4577] Rosen, E., Psenak, P., and P. Pillay-Esnault, "OSPF as the Provider/Customer Edge Protocol for BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4577, DOI 10.17487/RFC4577, June 2006, <https://www.rfc-editor.org/info/rfc4577>.

[RFC4915] Psenak, P., Mirtorabi, S., Roy, A., Nguyen, L., and P. Pillay-Esnault, "Multi-Topology (MT) Routing in OSPF", RFC 4915, DOI 10.17487/RFC4915, June 2007, <https://www.rfc-editor.org/info/rfc4915>.

[RFC4973] Srisuresh, P. and P. Joseph, "OSPF-xTE: Experimental Extension to OSPF for Traffic Engineering", RFC 4973, DOI 10.17487/RFC4973, July 2007, <https://www.rfc-editor.org/info/rfc4973>.

[RFC5082] Gill, V., Heasley, J., Meyer, D., Savola, P., Ed., and C. Pignataro, "The Generalized TTL Security Mechanism (GTSM)", RFC 5082, DOI 10.17487/RFC5082, October 2007, <https://www.rfc-editor.org/info/rfc5082>.

[RFC5185] Mirtorabi, S., Psenak, P., Lindem, A., Ed., and A. Oswal, "OSPF Multi-Area Adjacency", RFC 5185, DOI 10.17487/RFC5185, May 2008, <https://www.rfc-editor.org/info/rfc5185>.

[RFC5187] Pillay-Esnault, P. and A. Lindem, "OSPFv3 Graceful Restart", RFC 5187, DOI 10.17487/RFC5187, June 2008, <https://www.rfc-editor.org/info/rfc5187>.

[RFC5250] Berger, L., Bryskin, I., Zinin, A., and R. Coltun, "The OSPF Opaque LSA Option", RFC 5250, DOI 10.17487/RFC5250, July 2008, <https://www.rfc-editor.org/info/rfc5250>.

[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>.

[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>.

[RFC5329] Ishiguro, K., Manral, V., Davey, A., and A. Lindem, Ed., "Traffic Engineering Extensions to OSPF Version 3", RFC 5329, DOI 10.17487/RFC5329, September 2008, <https://www.rfc-editor.org/info/rfc5329>.

[RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, DOI 10.17487/RFC5340, July 2008, <https://www.rfc-editor.org/info/rfc5340>.

[RFC5613] Zinin, A., Roy, A., Nguyen, L., Friedman, B., and D. Yeung, "OSPF Link-Local Signaling", RFC 5613, DOI 10.17487/RFC5613, August 2009, <https://www.rfc-editor.org/info/rfc5613>.

[RFC5642] Venkata, S., Harwani, S., Pignataro, C., and D. McPherson, "Dynamic Hostname Exchange Mechanism for OSPF", RFC 5642, DOI 10.17487/RFC5642, August 2009, <https://www.rfc-editor.org/info/rfc5642>.

[RFC5709] Bhatia, M., Manral, V., Fanto, M., White, R., Barnes, M., Li, T., and R. Atkinson, "OSPFv2 HMAC-SHA Cryptographic Authentication", RFC 5709, DOI 10.17487/RFC5709, October 2009, <https://www.rfc-editor.org/info/rfc5709>.

[RFC5714] Shand, M. and S. Bryant, "IP Fast Reroute Framework", RFC 5714, DOI 10.17487/RFC5714, January 2010, <https://www.rfc-editor.org/info/rfc5714>.

[RFC5838] Lindem, A., Ed., Mirtorabi, S., Roy, A., Barnes, M., and R. Aggarwal, "Support of Address Families in OSPFv3", RFC 5838, DOI 10.17487/RFC5838, April 2010, <https://www.rfc-editor.org/info/rfc5838>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/info/rfc6020>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <https://www.rfc-editor.org/info/rfc6242>.

[RFC6565] Pillay-Esnault, P., Moyer, P., Doyle, J., Ertekin, E., and M. Lundberg, "OSPFv3 as a Provider Edge to Customer Edge (PE-CE) Routing Protocol", RFC 6565, DOI 10.17487/RFC6565, June 2012, <https://www.rfc-editor.org/info/rfc6565>.

[RFC6845] Sheth, N., Wang, L., and J. Zhang, "OSPF Hybrid Broadcast and Point-to-Multipoint Interface Type", RFC 6845, DOI 10.17487/RFC6845, January 2013, <https://www.rfc-editor.org/info/rfc6845>.

[RFC6860] Yang, Y., Retana, A., and A. Roy, "Hiding Transit-Only Networks in OSPF", RFC 6860, DOI 10.17487/RFC6860, January 2013, <https://www.rfc-editor.org/info/rfc6860>.

[RFC6987] Retana, A., Nguyen, L., Zinin, A., White, R., and D. McPherson, "OSPF Stub Router Advertisement", RFC 6987, DOI 10.17487/RFC6987, September 2013, <https://www.rfc-editor.org/info/rfc6987>.

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

[RFC7166] Bhatia, M., Manral, V., and A. Lindem, "Supporting Authentication Trailer for OSPFv3", RFC 7166, DOI 10.17487/RFC7166, March 2014, <https://www.rfc-editor.org/info/rfc7166>.

[RFC7474] Bhatia, M., Hartman, S., Zhang, D., and A. Lindem, Ed., "Security Extension for OSPFv2 When Using Manual Key Management", RFC 7474, DOI 10.17487/RFC7474, April 2015, <https://www.rfc-editor.org/info/rfc7474>.

[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>.

[RFC7684] Psenak, P., Gredler, H., Shakir, R., Henderickx, W., Tantsura, J., and A. Lindem, "OSPFv2 Prefix/Link Attribute Advertisement", RFC 7684, DOI 10.17487/RFC7684, November 2015, <https://www.rfc-editor.org/info/rfc7684>.

[RFC7770] Lindem, A., Ed., Shen, N., Vasseur, JP., Aggarwal, R., and S. Shaffer, "Extensions to OSPF for Advertising Optional Router Capabilities", RFC 7770, DOI 10.17487/RFC7770, February 2016, <https://www.rfc-editor.org/info/rfc7770>.

[RFC7777] Hegde, S., Shakir, R., Smirnov, A., Li, Z., and B. Decraene, "Advertising Node Administrative Tags in OSPF", RFC 7777, DOI 10.17487/RFC7777, March 2016, <https://www.rfc-editor.org/info/rfc7777>.

[RFC7884] Pignataro, C., Bhatia, M., Aldrin, S., and T. Ranganath, "OSPF Extensions to Advertise Seamless Bidirectional Forwarding Detection (S-BFD) Target Discriminators", RFC 7884, DOI 10.17487/RFC7884, July 2016, <https://www.rfc-editor.org/info/rfc7884>.

[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>.

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

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

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

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

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

[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>.

[RFC8476] Tantsura, J., Chunduri, U., Aldrin, S., and P. Psenak, "Signaling Maximum SID Depth (MSD) Using OSPF", RFC 8476, DOI 10.17487/RFC8476, December 2018, <https://www.rfc-editor.org/info/rfc8476>.

7.2. Informative References

[RFC0905] "ISO Transport Protocol specification ISO DP 8073", RFC 905, DOI 10.17487/RFC0905, April 1984, <https://www.rfc-editor.org/info/rfc905>.

[RFC4750] Joyal, D., Ed., Galecki, P., Ed., Giacalone, S., Ed., Coltun, R., and F. Baker, "OSPF Version 2 Management Information Base", RFC 4750, DOI 10.17487/RFC4750, December 2006, <https://www.rfc-editor.org/info/rfc4750>.

[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>.

[RFC5643] Joyal, D., Ed. and V. Manral, Ed., "Management Information Base for OSPFv3", RFC 5643, DOI 10.17487/RFC5643, August 2009, <https://www.rfc-editor.org/info/rfc5643>.

[RFC5649] Housley, R. and M. Dworkin, "Advanced Encryption Standard (AES) Key Wrap with Padding Algorithm", RFC 5649, DOI 10.17487/RFC5649, September 2009, <https://www.rfc-editor.org/info/rfc5649>.

[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>.

Appendix A. Contributors' Addresses

Dean Bogdanovic
Volta Networks, Inc.

EMail: dean@voltanet.io

Kiran Koushik Agrahara Sreenivasa
Verizon
500 W Dove Rd
Southlake, TX 76092
USA

EMail: kk@employees.org

Authors' Addresses

Derek Yeung
Arrcus

EMail: derek@arrcus.com

Yingzhen Qu
Futurewei
2330 Central Expressway
Santa Clara, CA 95050
USA

EMail: yingzhen.qu@futurewei.com

Jeffrey Zhang
Juniper Networks
10 Technology Park Drive
Westford, MA 01886
USA

EMail: zzhang@juniper.net

Ing-Wher Chen
The MITRE Corporation

EMail: ingwherchen@mitre.org

Acee Lindem
Cisco Systems
301 Midenhall Way
Cary, NC 27513

EMail: acee@cisco.com

Link State Routing
Internet-Draft
Intended status: Standards Track
Expires: September 2, 2018

K. Talaulikar
P. Psenak
Cisco Systems, Inc.
H. Johnston
AT&T Labs

March 1, 2018

OSPF Reverse Metric
draft-ketant-ospf-reverse-metric-00

Abstract

This document specifies the extensions to OSPF that enables a router to signal to its neighbor the metric that the neighbor should use towards itself using link-local advertisement between them. The signalling of this reverse metric, to be used on link(s) towards itself, allows a router to influence the amount of traffic flowing towards itself and in certain use-cases enables routers to maintain symmetric metric on both sides of a link between them.

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

Copyright Notice

Table of Contents

1. Introduction

Routers running the Open Shortest Path First (OSPFv2) [RFC2328] and OSPFv3 [RFC5340] routing protocols originate a Router-LSA (Link State Advertisement) that describes all its links to its neighbors and includes a metric which indicates its "cost" of reaching the neighbor over that link. Consider two routers R1 and R2 that are connected via a link. The metric for this link in direction R1->R2 is configured on R1 and in the direction R2->R1 is configured on R2. Thus the configuration on R1 influences the traffic that it forwards towards R2 but does not influence the traffic that it may receive from R2 on that same link.

This document describes certain use-cases where it is desirable for R1 to be able to signal what we call as the reverse metric (RM) that R2 should use on the link towards R1. Once R1 signals its reverse metric on its link to R2, then R2 advertises this value as its metric to R1 in its Router-LSA instead of its locally configured value. Once this information is part of the topology then all other routers do their computation using this value which results in the desired change in traffic distribution that R1 wanted to achieve towards itself over the link from R2.

This document proposes an extension to OSPF link-local signaling (LLS) [RFC5613] for signalling the OSPF Reverse Metric using the LLS Reverse Metric TLV in Section 4 and describes the related procedures in section Section 5.

2. Use Cases

This section describes certain use-cases that OSPF reverse metric helps to address. The usage of OSPF reverse metric need not be limited to these cases and is intended to be a generic mechanism.

2.1. Symmetrical Metric Based on Reference Bandwidth

Certain OSPF implementations and deployments deduce the metric of links based on their bandwidth using a reference bandwidth. The OSPF MIB [RFC4750] has ospfReferenceBandwidth that is used by entries in the ospfIfMetricTable. This mechanism is leveraged in deployments where the link metrics get lowered or increased as bandwidth capacity is removed or added e.g. consider layer-2 links bundled as a layer-3 interface on which OSPF is enabled. In the situations where these layer-2 links are directly connected to the two routers, the link and bandwidth availability is detected and updated on both sides. This allows for schemes where the metric is maintained to be symmetric in both directions based on the bandwidth.

Now consider variation of the same deployment where the links between routers are not directly connected and instead are provisioned over a layer-2 network consisting of switches or other mechanisms for a layer-2 emulation. In such scenarios, as show in Figure 1, the router on one side of the link would not detect when the neighboring router has lost one of its layer-2 link and has reduced capacity to its layer-2 switch. Note that the number of links and their capacities on the router R0 may not be the same as those on R1, R2 and R3. The left hand side diagram shows the actual physical topology in terms of the layer-2 links while the right hand side diagram shows the logical layer-3 link topology between the routers.

```
                +--------+
                |   R0   |
                | Router |
                +--------+                       +--------+
    (a) Physical   ^ ^ ^           (b) Layer-3   |   R0   |
        Topology   | | |              Topology   +--------+
                   v v v                           ^ ^ ^
             +----------------+                    | | |
             | Layer 2 Switch |                    | | |
             |  (Aggregation) |                +---+ | +---+
             +----------------+                |     |     |
              ^^  ^ ^ ^ ^   ^                  v     |     v
              ||  | | | |   |              +------+  |  +------+
         +----+|  | | | |   |              |  R1  |  |  |  R3  |
         | +---+  | | | |   +----+         +------+  |  +------+
         v v      v v v v        v                   v
    +--------+  +--------+  +--------+           +--------+
    |   R1   |  |   R2   |  |   R3   |           |   R2   |
    | Router |  | Router |  | Router |           +--------+
    +-- -----+  +--------+  +--------+
```

Figure 1: Routers Interconnected over Layer-2 Network

In such a scenario, the amount of traffic that can be forwarded in bidirectional manner between say R0 and R1 is dictated by the lower of the link capacity of R0 and R1 to the layer-2 transport network. In this scenario, when one of the link from R1 to the switch goes down, it would increase its link metric to R0 from say 20 to 40. However, similarly R0 also needs to increase its link metric to R1 as well from 20 to 40 as otherwise, the traffic will hit congestion and get dropped.

When R1 has the ability to signal the OSPF reverse metric of 40 towards itself to R0, then R0 can also update its metric without any manual intervention to ensure the correct traffic distribution. Consider if some destinations were reachable from R0 via R1 previously and this automatic metric adjustment now makes some of those destinations reachable from R0 via R3. This allows some traffic load on the link R0 to R1 to now flow via R3 to these destinations.

2.2. Adaptive Metric Signaling

Now consider another deployment scenario where, as show in Figure 2, two routers AGGR1 and AGGR2 are connected to a bunch of routers R1 thru RN that are dual homed to them and aggregating the traffic from them towards a core network. At some point T, AGGR1 loses some of its capacity towards the core or is facing some congestion issue towards the core and it needs to reduce the traffic going through it and perhaps redirect some of that load via AGGR2 which is not facing a similar issue. Altering its own metric towards Rx routers would influence the traffic flowing through it in the direction from core to the Rx but not the other way around as desired.

```
              Core Network
          ^                ^
          |                |
          V                v
     +----------+    +----------+
     |  AGGR1   |    |  AGGR2   |
     +----------+    +----------+
       ^      ^        ^      ^
       |      |        |      |
       |      +-----------+   |
       |               |  |   |
       |      +--------+  |   |
       v      v           v   v
    +-----------+      +-----------+
    |    R1     |      |    RN     |
    |  Router   | ...  |  Router   |
    +-----------+      +-----------+
```

Figure 2: Adaptive Metric for Dual Gateways

In such a scenario, the AGGR1 router could signal an incremental value of OSPF reverse metric towards some or all of the Rx routers. When the Rx routers apply this signaled reverse metric offset value to the original metric on their links towards AGGR1 then the path via AGGR2 becomes a better path causing traffic towards the core getting diverted away from it. Note that the reverse metric mechanism allows such adaptive metric changes to be applied on the AGGR1 as opposed to being provisioning statically on the possibly large number of Rx routers.

3. Solution

To address the use-cases described earlier and to allow an OSPF router to indicate its reverse metric for a specific point-to-point or point-to-multipoint link to its neighbor, this document proposes to extend OSPF link-local signaling to advertise the Reverse Metric TLV in OSPF Hello packets. This ensures that the RM signaling is scoped ONLY to each specific link individually. The router continues to include the Reverse Metric TLV in its Hello packets on the link as long as it needs its neighbor to use that metric value towards itself. Further details of the procedures involve are specified in Section 5.

The RM signaling specified in this document is not required for broadcast or non-broadcast-multi-access (NBMA) links since the same objective is achieved there using the OSPF Two-Part Metric mechanism [RFC8042].

4. LLS Reverse Metric TLV

The Reverse Metric TLV is a new LLS TLV. It has following format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Type             |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Flags        |O|H|        Reverse Metric         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

Type: TBD, suggested value 19

Length: 4 octet

Flags: Following are defined currently and the rest MUST be set to 0 and ignored on reception.

- * H (0x1) : Indicates that neighbor should use value only if higher than its current metric value in use
- * O (0x2) : Indicates that the reverse metric value provided is an offset that is to be added to the original metric

Reverse Metric: The value or offset of reverse metric to be used

5. Procedures

When a router needs to signal a RM value that its neigbhor(s) should use towards itself, it includes the Reverse Metric TLV in the LLS block of its hello messages sent on the link and continues to include this TLV for as long as it needs it’s neighbor to use this value. The mechanisms used to determine the value to be used for the RM is specific to the implementation and use-case and is outside the scope of this document. e.g. in the use-case related to symmetric metric described in Section 2.1, the RM value may be derived based on the router’s link’s bandwidth with respect to the reference bandwidth.

A router receiving a hello packet from its neighbor that contains the Reverse Metric TLV on its link SHOULD use the RM value to derive the metric for the link in its Router-LSA to the advertising router. When the O flag is set, the value in the TLV needs to be added to the existing original metric provisioned on the link to derive the new metric value to be used. When the O flag is clear, the value in the TLV should be directly used as the metric to be used. When H flag is set and O flag is clear, this is done only when the RM value signaled is higher than the provisioned metric value being used already. This mechanism applies only for point-to-point, point-to-multipoint and hybrid broadcast point-to-multipoint ([RFC6845]) links. For broadcast and NBMA links the OSPF Two-Part Metric mechanism [RFC8042] should be used in similar use-cases.

Implementations SHOULD provide a configuration option to enable the signaling of RM from a router to its neighbors and MAY provide a configuration option to disable the acceptance of the RM from its neighbors.

A router stops including the Reverse Metric TLV in its hello messages when it needs its neighbors to go back to using their own provisioned metric values. When that happens, a router which had modified its metric in response to receiving a Reverse Metric TLV from its neighbor should revert back to using its original provisioned metric value.

In certain scenarios, it is possible that two or more routers start the RM signaling on the same link. This could create collision scenarios. The following rules MUST be adopted by routers to ensure that there is no instability in the network due to churn in their metric due to signaling of RM:

- The RM value that is signaled by a router to its neighbor MUST NOT be derived from the reverse metric being signaled by any of its neighbor on any of its links.

- The RM value that is signaled by a router MUST NOT be derived from its own metric which has been modified on account of a RM signaled from any of its neighbors on any of its links. RM signaling from other routers can affect the router's own metric advertised in its Router-LSA. When deriving the RM values that a router signals to its neighbors, it should use its "original" local metric values not influenced by any RM signaling.

Based on these rules, a router MUST never start or stop or change its RM metric signaling based on the RM metric signaling initiated by some other router. Based on the local configuration policy, each router would end up accepting the RM value signaled by its neighbor and there would be no churn of metrics on the link or the network on account of RM signaling.

In certain use-case as described in Section 2.1 when symmetrical metrics are desired, the RM signaling can be enabled on routers on either ends of a link. In other use-cases as described in Section 2.2 RM signaling may need to be enabled on only router at one end of a link.

6. Backward Compatibility

The signaling specified in this document happens at a link-local level between routers on that link. A router which does not support this specification would ignore the Reverse Metric LLS TLV and take no actions to updates its metric in the other LSAs. As a result, the behavior would be the same as before this specification. Therefore, there are no backward compatibility related issues or considerations that need to be taken care of when implementing this specification.

7. IANA Considerations

This specification updates Link Local Signalling TLV Identifiers registry.

Following values are requested for allocation:

o TBD (Suggested value 19) - Reverse Metric TLV

8. Security Considerations

Implementations must assure that malformed LLS TLV and Sub-TLV permutations do not result in errors which cause hard OSPF failures.

9. Contributors

Thanks to Jay Karthik for his contributions on the use-cases related to symmetric metric and the review of the solution.

10. Acknowledgements

11. References

11.1. Normative References

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

[RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, DOI 10.17487/RFC2328, April 1998, <https://www.rfc-editor.org/info/rfc2328>.

[RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, DOI 10.17487/RFC5340, July 2008, <https://www.rfc-editor.org/info/rfc5340>.

[RFC5613] Zinin, A., Roy, A., Nguyen, L., Friedman, B., and D. Yeung, "OSPF Link-Local Signaling", RFC 5613, DOI 10.17487/RFC5613, August 2009, <https://www.rfc-editor.org/info/rfc5613>.

11.2. Informative References

[RFC4750] Joyal, D., Ed., Galecki, P., Ed., Giacalone, S., Ed., Coltun, R., and F. Baker, "OSPF Version 2 Management Information Base", RFC 4750, DOI 10.17487/RFC4750, December 2006, <https://www.rfc-editor.org/info/rfc4750>.

[RFC6845] Sheth, N., Wang, L., and J. Zhang, "OSPF Hybrid Broadcast and Point-to-Multipoint Interface Type", RFC 6845, DOI 10.17487/RFC6845, January 2013, <https://www.rfc-editor.org/info/rfc6845>.

[RFC8042] Zhang, Z., Wang, L., and A. Lindem, "OSPF Two-Part Metric", RFC 8042, DOI 10.17487/RFC8042, December 2016, <https://www.rfc-editor.org/info/rfc8042>.

Authors' Addresses

Ketan Talaulikar
Cisco Systems, Inc.
S.No. 154/6, Phase I, Hinjawadi
PUNE, MAHARASHTRA 411 057
India

Email: ketant@cisco.com

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

Email: ppsenak@cisco.com

Hugh Johnston
AT&T Labs
USA

Email: hugh_johnston@labs.att.com

Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: December 30, 2018

T. Li
Arista Networks
P. Psenak
Cisco Systems, Inc.

June 28, 2018

Dynamic Flooding on Dense Graphs
draft-li-dynamic-flooding-05

Abstract

Routing with link state protocols in dense network topologies can result in sub-optimal convergence times due to the overhead associated with flooding. This can be addressed by decreasing the flooding topology so that it is less dense.

This document discusses the problem in some depth and an architectural solution. Specific protocol changes for IS-IS, OSPFv2, and OSPFv3 are described in this document.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

In recent years, there has been increased focused on how to address the dynamic routing of networks that have a bipartite (a.k.a. spine-leaf or leaf-spine), Clos [Clos], or Fat Tree [Leiserson] topology. Conventional Interior Gateway Protocols (IGPs, i.e., IS-IS [ISO10589], OSPFv2 [RFC2328], and OSPFv3 [RFC5340]) under-perform, redundantly flooding information throughout the dense topology, leading to overloaded control plane inputs and thereby creating operational issues. For practical considerations, network architects have resorted to applying unconventional techniques to address the problem, applying BGP in the data center [RFC7938]. However it is very clear that using an Exterior Gateway Protocol as an IGP is sub-optimal, if only due to the configuration overhead.

The primary issue that is demonstrated when conventional mechanisms are applied is the poor reaction of the network to topology changes. Normal link state routing protocols rely on a flooding algorithm for state distribution. In a dense topology, this flooding algorithm is highly redundant, resulting in unnecessary overhead. Each node in the topology receives each link state update multiple times. Ultimately, all of the redundant copies will be discarded, but only after they have reached the control plane and been processed. This creates issues because significant link state database updates can become queued behind many redundant copies of another update. This delays convergence as the link state database does not stabilize promptly.

In a real world implementation, the packet queues leading to the control plane are necessarily of finite size, so if the flooding rate exceeds the update processing rate for long enough, the control plane will be obligated to drop incoming updates. If these lost updates are of significance, this will further delay stabilization of the link state database and the convergence of the network.

This is not a new problem. Historically, when routing protocols have been deployed in networks where the underlying topology is a complete graph, there have been similar issues. This was more common when the underlying link layer fabric presented the network layer with a full mesh of virtual connections. This was addressed by reducing the flooding topology through IS-IS Mesh Groups [RFC2973], but this approach requires careful configuration of the flooding topology.

Thus, the root problem is not limited to massively scalable data centers. It exists with any dense topology at scale.

This problem is not entirely surprising. Link state routing protocols were conceived when links were very expensive and topologies were sparse. The fact that those same designs are sub-optimal in a dense topology should not come as a huge surprise. The fundamental premise that was addressed by the original designs was an environment of extreme cost and scarcity. Technology has progressed to the point where links are cheap and common. This represents a complete reversal in the economic fundamentals of network engineering. The original designs are to be commended for continuing to provide correct operation to this point, and optimizations for operation in today's environment are to be expected.

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. Problem Statement

In a dense topology, the flooding algorithm that is the heart of conventional link state routing protocols causes a great deal of redundant messaging. This is exacerbated by scale. While the protocol can survive this combination, the redundant messaging is unnecessary overhead and delays convergence. Thus, the problem is to provide routing in dense, scalable topologies with rapid convergence.

3. Solution Requirements

A solution to this problem must then meet the following requirements:

Requirement 1 Provide a dynamic routing solution. Reachability must be restored after any topology change.

Requirement 2 Provide a significant improvement in convergence.

Requirement 3 The solution should address a variety of dense topologies. Just addressing a complete bipartite topology such as K5,8 is insufficient. Multi-stage Clos topologies must also be addressed, as well as topologies that are slight variants. Addressing complete graphs is a good demonstration of generality.

Requirement 4 There must be no single point of failure. The loss of any link or node should not unduly hinder convergence.

Requirement 5 Dense topologies are subgraphs of much larger topologies. Operational efficiency requires that the dense subgraph not operate in a radically different manner than the remainder of the topology. While some operational differences are permissible, they should be minimized. Changes to nodes outside of the dense subgraph are not acceptable. These situations occur when massively scaled data centers are part of an overall larger wide-area network. Having a second protocol operating just on this subgraph would add much more complexity at the edge of the subgraph where the two protocols would have to inter-operate.

4. Dynamic Flooding

We have observed that the combination of the dense topology and flooding on the physical topology in a scalable network is sub-optimal. However, if we decouple the flooding topology from the physical topology and only flood on a greatly reduced portion of that topology, we can have efficient flooding and retain all of the resilience of existing protocols.

In this idea, the flooding topology is computed either centrally on an elected node or in a distributed manner on all nodes that are supporting Dynamic Flooding. If the flooding topology is computed centrally, it is encoded into and distributed as part of the normal link state database. We call this the centralized mode of operation. If the flooding topology is computed in a distributed fashion, we call this the distributed mode of operation. Nodes within the dense topology would only flood on the flooding topology. On links outside of the normal flooding topology, normal database synchronization mechanisms (i.e., OSPF database exchange, IS-IS CSNPs) would apply, but flooding would not. New link state information that arrives from outside of the flooding topology suggests that the sender has a different or no flooding topology information and that the link state update should be flooded on the flooding topology as well.

Since the flooding topology is computed prior to topology changes, it does not factor into the convergence time and can be done when the topology is stable. The speed of the computation and its distribution, in the case of a centralized mode, is not a significant issue.

If a node does not have any flooding topology information when it receives new link state information, it should flood according to legacy flooding rules. This situation will occur when the dense topology is first established, but is unlikely to recur.

When centralized mode is used and if, during a transient, there are multiple flooding topologies being advertised, then nodes should flood link state updates on all of the flooding topologies. Each node should locally evaluate the election of the lead node for the dense subgraph and first flood on the topology of the lead node. The rationale behind this is straightforward: if there is a transient and there has been a recent change in the elected node, then propagating topology information promptly along the most likely flooding topology should be the priority.

During transients, it is possible that loops will form in the flooding topology. This is not problematic, as the legacy flooding rules would cause duplicate updates to be ignored. Similarly, during transients, it is possible that the forwarding topology may become disconnected. To address this, nodes can perform a database synchronization check anytime a link is added to or removed from the flooding topology.

4.1. Applicability

In a complete graph, this approach is appealing because it drastically decreases the flooding topology without the manual configuration of mesh groups. By controlling the diameter of the flooding topology, as well as the maximum degree node in the flooding topology, convergence time goals can be met and the stability of the control plane can be assured.

Similarly, in a massively scaled data center, where there are many opportunities for redundant flooding, this mechanism ensures that flooding is redundant, with each leaf and spine well connected, while ensuring that no update need make too many hops and that no node shares an undue portion of the flooding effort.

In a network where only a portion of the nodes support Dynamic Flooding, the remaining nodes will continue to perform universal flooding. This is not an issue for correctness, as no node can become isolated.

Flooding that is initiated within the flooding topology will remain within that flooding topology until it reaches a legacy node, which will resume legacy flooding. Legacy flooding will be bounded by the flooding topology, which can help limit the propagation of unnecessary flooding. Whether or not the network can remain stable in this condition is unknown and may be very dependent on the number and location of the nodes that support Dynamic Flooding.

4.2. Leader election

A single node within the dense topology is elected as an Area Leader.

A generalization of the mechanisms used in existing Designated Router (OSPF) or Designated Intermediate-System (IS-IS) elections suffices. The elected node is known as the Area Leader.

In the case of centralized mode, the Area Leader is responsible for computing and distributing the flooding topology. When a new node is elected and has distributed new flooding topology information, then the old node should withdraw its flooding topology information from the link state database. If the old node does not return to the topology in a timely manner, the new node may remove the old node’s information from the link state database.

In the case of distributed mode, the distributed algorithm advertised by the Area Leader MUST be used by all routers that participate in Dynamic Flooding.

Not every router needs to be a candidate to be Area Leader within an area, as a single candidate is sufficient for correct operation. For redundancy, however, it is strongly RECOMMENDED that there be multiple candidates.

4.3. Computing the Flooding Topology

There is a great deal of flexibility in how the flooding topology may be computed. For resilience, it needs to at least contain a cycle of all nodes in the dense subgraph. However, additional links could be added to decrease the convergence time. The trade-off between the density of the flooding topology and the convergence time is a matter for further study. The exact algorithm for computing the flooding topology in the case of the centralized computation need not be standardized, as it is not an interoperability issue. Only the encoding of the result needs to be documented. In the case of distributed mode, all nodes in the IGP area need to use the same algorithm to compute the flooding topology. It is possible to use private algorithms to compute flooding topology, so long as all nodes in the IGP area use the same algorithm.

While the flooding topology should be a covering cycle, it need not be a Hamiltonian cycle where each node appears only once. In fact, in many relevant topologies this will not be possible e.g., K5,8. This is fortunate, as computing a Hamiltonian cycle is known to be NP-complete.

A simple algorithm to compute the topology for a complete bipartite graph is to simply select unvisited nodes on each side of the graph until both sides are completely visited. If the number of nodes on each side of the graph are unequal, then revisiting nodes on the less populated side of the graph will be inevitable. This algorithm can run in O(N) time, so is quite efficient.

While a simple cycle is adequate for correctness and resiliency, it may not be optimal for convergence. At scale, a cycle may have a diameter that is half the number of nodes in the graph. This could cause an undue delay in link state update propagation. Therefore it may be useful to have a bound on the diameter of the flooding topology. Introducing more links into the flooding topology would reduce the diameter, but at the trade-off of possibly adding redundant messaging. The optimal trade-off between convergence time and graph diameter is for further study.

Similarly, if additional redundancy is added to the flooding topology, specific nodes in that topology may end up with a very high degree. This could result in overloading the control plane of those nodes, resulting in poor convergence. Thus, it may be optimal to have an upper bound on the degree of nodes in the flooding topology. Again, the optimal trade-off between graph diameter, node degree, and convergence time, and topology computation time is for further study.

If the leader chooses to include a multi-node broadcast LAN segment as part of the flooding topology, all of the connectivity to that LAN segment should be included as well. Once updates are flooded onto the LAN, they will be received by every attached node.

4.4. Topologies on Complete Bipartite Graphs

Complete bipartite graph topologies have become popular for data center applications and are commonly called leaf-spine or spine-leaf topologies. In this section, we discuss some flooding topologies that are of particular interest in these networks.

4.4.1. A Minimal Flooding Topology

We define a Minimal Flooding Topology on a complete bipartite graph as one in which the topology is connected and each node has at least degree two. This is of interest because it guarantees that the flooding topology has no single points of failure.

In practice, this implies that every leaf node in the flooding topology will have a degree of two. As there are usually more leaves than spines, the degree of the spines will be higher, but the load on the individual spines can be evenly distributed.

This type of flooding topology is also of interest because it scales well. As the number of leaves increases, we can construct flooding topologies that perform well. Specifically, for n spines and m leaves, if m >= n(n/2-1), then there is a flooding topology that has a diameter of four.

4.4.2. Xia Topologies

We define a Xia Topology on a complete bipartite graph as one in which all spine nodes are bi-connected through leaves with degree two, but the remaining leaves all have degree one and are evenly distributed across the spines.

Constructively, we can create a Xia topology by iterating through the spines. Each spine can be connected to the next spine by selecting any unused leaf. Since leaves are connected to all spines, all leaves will have a connection to both the first and second spine and we can therefore choose any leaf without loss of generality. Continuing this iteration across all of the spines, selecting a new leaf at each iteration, will result in a path that connects all spines. Adding one more leaf between the last and first spine will produce a cycle of n spines and n leaves.

At this point, m-n leaves remain unconnected. These can be distributed evenly across the remaining spines, connected by a single link.

Xia topologies represent a compromise that trades off increased risk and decreased performance for lower flooding amplification. Xia topologies will have a larger diameter. For m spines, the diameter will be m + 2.

In a Xia topology, some leaves are singly connected. This represents a risk in that in some failures, convergence may be delayed. However, there may be some alternate behaviors that can be employed to mitigate these risks. If a leaf node sees that its single link on the flooding topology has failed, it can compensate by performing a database synchronization check with a different spine. Similarly, if a leaf determines that its connected spine on the flooding topology has failed, it can compensate by performing a database synchronization check with a different spine. In both of these cases, the synchronization check is intended to ameliorate any delays in link state propagation due to the fragmentation of the flooding topology.

The benefit of this topology is that flooding load is easily understood. Each node in the spine cycle will never receive an update more than twice. For n leaves and m spines, a spine never transmits more than m/n updates.

4.4.3. Optimization

If two systems have multiple links between them, only one of the links should be part of the flooding topology. Moreover, symmetric selection of the link to use for flooding is not required.

4.5. Encoding the Flooding Topology

There are a variety of ways that the flooding topology could be encoded efficiently. If the topology was only a cycle, a simple list of the nodes in the topology would suffice. However, this is insufficiently flexible as it would require a slightly different encoding scheme as soon as a single additional link is added. Instead, we choose to encode the flooding topology as a set of intersecting paths, where each path is a set of connected edges.

Other encodings are certainly possible. We have attempted to make a useful trade off between simplicity, generality, and space.

4.6. Analysis of Topology Changes

In this section, we explicitly consider a variety of different topological failures in the network and how dynamic flooding should address them.

4.6.1. Link Addition

If a link is added to the topology, the protocol will form a normal adjacency on the link and update the appropriate link state advertisements for the routers on either end of the link. These link state updates will be flooded on the flooding topology.

In centralized mode, the Area Leader, upon receiving these updates, may choose to retain the existing flooding topology or may choose to modify the flooding topology. If it elects to change the flooding topology, it will update the flooding topology in the link state database and flood it using the new flooding topology.

In distributed mode, any change in the topology, including the link addition, should trigger the flooding topology recalculation. This is done to ensure that all nodes converge on the same flooding topology, regardless of the time of the calculation.

4.6.2. Node Addition

In centralized mode, if a node is added to the topology, then at least one link is also added to the topology. The paragraph above applies and the Area Leader will necessarily need to add the new node to the flooding topology.

In distributed mode, the addition of a node should trigger flooding topology recalculation.

Until the new node is incorporated into the flooding topology at least a single link towards the new node MUST be added to the flooding topology locally on all of its neighbors.

4.6.3. Link Failures Off the Flooding Topology

If a link that is not part of the flooding topology fails, then the adjoining routers will update their link state advertisements and flood them on the flooding topology. There is no need for changes to the flooding topology.

4.6.4. Failure of the Area Leader

The failure of the Area Leader can be detected by observing that it is disconnected from the area topology. In this case, the Area Leader election process is repeated and a new Area Leader is elected.

In the centralized mode, the new Area Leader will compute a new flooding topology and flood it using the new flooding topology.

As an optimization, applicable to centralized mode, the new Area Leader MAY compute a new flooding topology that has as much in common as possible with the old flooding topology. This will minimize the risk of over-flooding.

4.6.5. Failures on the Flooding Topology

If there is a failure on the flooding topology, the adjoining routers will update their link state advertisements and flood them. If the original flooding topology is bi-connected, the flooding topology should still be connected despite a single failure.

In centralized mode, the Area Leader will notice the change in the flooding topology, recompute the flooding topology, and flood it using the new flooding topology.

In distributed mode, all routers supporting dynamic flooding will notice the change in the flooding topology and recompute the new flooding topology.

4.6.6. Recovery from Multiple Failures

In the unlikely event of multiple failures on the flooding topology, it may become disconnected. The nodes that remain active on the edges of the flooding topology will recognize this, update their own link state advertisements and flood them on the remainder of the flooding topology. At this point, nodes will be able to compute that the flooding topology is partitioned.

Note that this is very different from partitioning the area itself. The area may remain connected and forwarding may still be effective.

When this condition is detected, the flooding topology can no longer be expected to deliver link state updates in a prompt manner. Nodes on the edges of the flooding topology should perform database synchronization on all links not on the flooding topology. Updates received from off of the flooding topology should be flooded on the remaining flooding topology. Any links that provide updates or require updates that are not part of the flooding topology should temporarily be added to the flooding topology. This should repair the current flooding topology, albeit in a sub-optimal manner.

In centrailzed mode, the Area Leader will also detect this condition, compute a new flooding topology, and flood it using the new flooding topology.

In distributed mode, all routers that actively participate in Dynamic Flooding will compute the new flooding topology.

5. Protocol Elements

5.1. IS-IS TLVs

The following TLVs are added to IS-IS:

1. A TLV that an IS may inject into its LSP to indicate its preference for becoming Area Leader.

2. A TLV to carry the list of system IDs that compromise the flooding topology for the area.

3. A TLV to carry the adjacency matrix for the flooding topology for the area.

5.1.1. IS-IS Area Leader Sub-TLV

The Area Leader Sub-TLV allows a system to:

1. Indicate its eligibility and priority for becoming Area Leader.

2. Indicate whether centralized or distributed mode is to be used to compute the flooding topology in the area.

3. Indicate the algorithm identifier for the algorithm that is used to compute the flooding topology in distributed mode.

Intermediate Systems (routers) that are not advertising this Sub-TLV are not eligible to become Area Leader.

The Area Leader is the router with the numerically highest Area Leader priority in the area. In the event of ties, the router with the numerically highest system ID is the Area Leader. Due to transients during database flooding, different routers may not agree on the Area Leader.

The Area Leader Sub-TLV is advertised as a Sub-TLV of the IS-IS Router Capability TLV-242 that is defined in [RFC7981] and has the following format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| TLV Type      | TLV Length    | Priority      |   Algorithm   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

TLV Type: TBD1

TLV Length: 2

Priority: 0-255, unsigned integer

Algorithm - a numeric identifier in the range 0-255 that identifies the algorithm used to calculate the flooding topology. The following values are defined:

0: Centralized computation by the Area Leader.

1-127: Standardized distributed algorithms. Individual values area assigned and managed by IANA. Before any assignments can be made, there MUST be an IETF specification that specifies IANA allocation for any value from this range (see Section 7.3).

128-254: Private distributed algorithms. Values from this range will not be registered with IANA and MUST NOT be mentioned by RFCs.

255: Reserved

5.1.2. IS-IS Area System IDs TLV

IS-IS Area System IDs TLV is only used in centralized mode.

The Area System IDs TLV is used by the Area Leader to enumerate the system IDs that it has used in computing the flooding topology. Conceptually, the Area Leader creates a list of system IDs for all routers in the area, assigning indices to each system, starting with index 0.

Because the space in a single TLV is small, more than one TLV may be required to encode all of the system IDs in the area. This TLV may be present in multiple LSPs.

The format of the Area System IDs TLV is:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| TLV Type      | TLV Length    | Starting Index                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|L| Reserved    | System IDs ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
System IDs continued ....
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

TLV Type: TBD2

TLV Length: 3 + (System ID length * (number of System IDs))

Starting index: The index of the first system ID that appears in this TLV.

L (Last): This bit is set if the index of the last system ID that appears in this TLV is equal to the last index in the full list of system IDs for the area.

System IDs: A concatenated list of system IDs for the area.

If there are multiple IS-IS Area System IDs TLVs with the L bit set advertised by the same router, the TLV which specifies the smaller maximum index is used and the other TLV(s) with L bit set are ignored. TLVs which specify system IDs with indices greater than that specified by the TLV with the L bit set are also ignored.

5.1.3. IS-IS Flooding Path TLV

IS-IS Flooding Path TLV is only used in centralized mode.

The Flooding Path TLV is used to denote a path in the flooding topology. The goal is an efficient encoding of the links of the topology. A single link is a simple case of a path that only covers two nodes. A connected path may be described as a sequence of indices: (I1, I2, I3, ...), denoting a link from the system with index 1 to the system with index 2, a link from the system with index 2 to the system with index 3, and so on.

If a path exceeds the size that can be stored in a single TLV, then the path may be distributed across multiple TLVs by the replication of a single system index.

Complex topologies that are not a single path can be described using multiple TLVs.

The Flooding Path TLV contains a list of system indices relative to the systems advertised through the Area System IDs TLV. At least 2 indices must be included in the TLV. Due to the length restriction of TLVs, this TLV can contain at most 126 system indices.

The Flooding Path TLV has the format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| TLV Type      | TLV Length    | Starting Index                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Index 2                       | Additional indices ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

TLV Type: TBD3

TLV Length: 2 * (number of indices in the path)

Starting index: The index of the first system in the path.

Index 2: The index of the next system in the path.

Additional indices (optional): A sequence of additional indices to systems along the path.

5.2. OSPF LSAs and TLVs

This section defines new LSAs and TLVs for both OSPFv2 and OSPFv3.

5.2.1. OSPF Area Leader Sub-TLV

The usage of the OSPF Area Leader Sub-TLV is identical to IS-IS and is described in Section 5.1.1.

The OSPF Area Leader Sub-TLV is used by both OSPFv2 and OSPFv3.

The OSPF Area Leader Sub-TLV is advertised as a top-level TLV of the RI LSA that is defined in [RFC7770] and has the following format:

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |              Type             |             Length            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |    Priority   |   Algorithm   |            Reserved           |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type: TBD4

Length: 4 octets

Priority: 0-255, unsigned integer

Algorithm: as defined in Section 5.1.1.

5.2.2. OSPFv2 Dynamic Flooding Opaque LSA

The OSPFv2 Dynamic Flooding Opaque LSA is only used in centralized mode.

The OSPFv2 Dynamic Flooding Opaque LSA is used to advertise additional data related to the dynamic flooding in OSPFv2. OSPFv2 Opaque LSAs are described in [RFC5250].

Multiple OSPFv2 Dynamic Flooding Opaque LSAs can be advertised by an OSPFv2 router. The flooding scope of the OSPFv2 Dynamic Flooding Opaque LSA is area-local.

The format of the OSPFv2 Dynamic Flooding Opaque LSA is as follows:

```
       0                   1                   2                   3
       0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |            LS age             |    Options    |    LS Type    |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |      TBD5     |                 Opaque ID                     |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                     Advertising Router                        |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                     LS sequence number                        |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |         LS checksum           |             Length            |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                                                               |
      +-                            TLVs                             -+
      |                             ...                               |
```

OSPFv2 Dynamic Flooding Opaque LSA

The opaque type used by OSPFv2 Dynamic Flooding Opaque LSA is TBD. The opaque type is used to differentiate the various type of OSPFv2 Opaque LSAs and is described in section 3 of [RFC5250]. The LS Type is 10. The LSA Length field [RFC2328] represents the total length (in octets) of the Opaque LSA including the LSA header and all TLVs (including padding).

The Opaque ID field is an arbitrary value used to maintain multiple Dynamic Flooding Opaque LSAs. For OSPFv2 Dynamic Flooding Opaque LSAs, the Opaque ID has no semantic significance other than to differentiate Dynamic Flooding Opaque LSAs originated by the same OSPFv2 router.

The format of the TLVs within the body of the OSPFv2 Dynamic Flooding Opaque LSA is the same as the format used by the Traffic Engineering Extensions to OSPF [RFC3630].

The Length field defines the length of the value portion in octets (thus a TLV with no value portion would have a length of 0). The TLV is padded to 4-octet alignment; padding is not included in the length field (so a 3-octet value would have a length of 3, but the total size of the TLV would be 8 octets). Nested TLVs are also 32-bit aligned. For example, a 1-octet value would have the length field set to 1, and 3 octets of padding would be added to the end of the value portion of the TLV. The padding is composed of zeros.

5.2.3. OSPFv3 Dynamic Flooding LSA

The OSPFv3 Dynamic Flooding Opaque LSA is only used in centralized mode.

The OSPFv3 Dynamic Flooding LSA is used to advertise additional data related to the dynamic flooding in OSPFv3.

The OSPFv3 Dynamic Flooding LSA has a function code of TBD. The flooding scope of the OSPFv3 Dynamic Flooding LSA is area-local. The U bit will be set indicating that the OSPFv3 Dynamic Flooding LSA should be flooded even if it is not understood. The Link State ID (LSID) value for this LSA is the Instance ID. OSPFv3 routers MAY advertise multiple Dynamic Flooding Opaque LSAs in each area.

The format of the OSPFv3 Dynamic Flooding LSA is as follows:

```
        0                   1                   2                   3
        0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |            LS age             |1|0|1|          TBD6           |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                    Link State ID                              |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                    Advertising Router                         |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                    LS sequence number                         |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |        LS checksum            |           Length              |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                                                               |
       +-                            TLVs                             -+
       |                             ...                               |
```

OSPFv3 Dynamic Flooding LSA

5.2.4. OSPF Area Router IDs TLV

The OSPF Area Router IDs TLV is a top level TLV of the OSPFv2 Dynamic Flooding Opaque LSA and OSPFv3 Dynamic Flooding LSA.

The OSPF Area Router IDs TLV is used by the Area Leader to enumerate the Router IDs that it has used in computing the flooding topology. Conceptually, the Area Leader creates a list of Router IDs for all routers in the area, assigning indices to each router, starting with index 0.

Because the space in a single OSPF Area Router IDs TLV is limited, more than one TLV may ve required to encode all of the Router IDs in the area. This TLV may also recur in multiple OSPFv2 Dynamic Flooding Opaque LSAs or OSPFv3 Dynamic Flooding LSA, so that all Router IDs can be advertised.

The format of the Area Router IDs TLV is:

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |              Type             |             Length            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |    Starting Index             |L| Flags       |   Reserved    |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    +-                        Router IDs                           -+
    |                           ...                                 |
```

OSPF Area Router IDs TLV

TLV Type: 1

TLV Length: 4 + (Router ID length * (number of Router IDs))

Starting index: The index of the first Router ID that appears in this TLV.

L (Last): This bit is set if the index of the last system ID that appears in this TLV is equal to the last index in the full list of Rourer IDs for the area.

Router IDs: A concatenated list of Router IDs for the area.

If there are multiple OSPF Area Router IDs TLVs with the L bit set advertised by the same router, the TLV which specifies the smaller maximum index is used and the other TLV(s) with L bit set are ignored. TLVs which specify Router IDs with indices greater than that specified by the TLV with the L bit set are also ignored.

5.2.5. OSPF Flooding Path TLV

The OSPF Flooding Path TLV is a top level TLV of the OSPFv2 Dynamic Flooding Opaque LSAs and OSPFv3 Dynamic Flooding LSA.

The usage of the OSPF Flooding Path TLV is identical to IS-IS and is described in Section 5.1.3.

The OSPF Flooding Path TLV contains a list of Router ID indices relative to the Router IDs advertised through the OSPF Area Router IDs TLV. At least 2 indices must be included in the TLV.

Multiple OSPF Flooding Path TLVs can be advertised in a single OSPFv2 Dynamic Flooding Opaque LSA or OSPFv3 Dynamic Flooding LSA. OSPF Flooding Path TLVs can also be advertised in multiple OSPFv2 Dynamic Flooding Opaque LSAs or OSPFv3 Dynamic Flooding LSA, if they all can not fit in a single LSA.

The Flooding Path TLV has the format:

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |              Type             |             Length            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |    Starting Index             |       Index 2                 |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    +-                       Additional Indices                    -+
    |                             ...                               |
```

OSPF Flooding Path TLV

TLV Type: 2

TLV Length: 2 * (number of indices in the path)

Starting index: The index of the first Router ID in the path.

Index 2: The index of the next Router ID in the path.

Additional indices (optional): A sequence of additional indices to Router IDs along the path.

6. Behavioral Specification

In this section, we specify the detailed behaviors of the nodes participating in the IGP.

6.1. Leader Election

Any node that is capable MAY advertise its eligibility to become Area Leader.

Nodes that are not reachable are not eligible as Area Leader. Nodes that do not advertise their eligibility to become Area Leader are not eligible. Amongst the eligible nodes, the node with the numerically highest priority is the Area Leader. If multiple nodes all have the highest priority, then the node with the numerically highest system identifier in the case of IS-IS, or Router-ID in the case of OSPFv2 and OSPFv3 is the Area Leader.

6.2. Area Leader Responsibilities

If the Area Leader operates in centralized mode, it MUST advertise algorithm 0 in its Area Leader Sub-TLV. It also MUST compute and advertise a flooding topology for the area. The Area Leader MAY update the flooding topology at any time, however, it should not destabilize the network with undue or overly frequent topology changes.

The flooding topology MUST include all reachable nodes in the area. If nodes become unreachable on the flooding topology, the flooding topology MUST be recalculated. In centralized mode, the Area Leader MUST advertise a new flooding topology.

The flooding topology MAY be bi-connected. This is strongly RECOMMENDED but not required.

6.3. Distributed Flooding Topology Calculation

If the Area Leader advertises a non-zero algorithm in its Area Leader Sub-TLV, all routers in the area that support Dynamic Flooding and the value of algorithm advertised by the Area Leader MUST compute the flooding topology based on the Area Leader's advertised algorithm. Routers that do not support the value of algorithm advertised by the Area Leader MUST continue to use legacy flooding mechanism as defined by the protocol.

If the value of the algorithm advertised by the Area Leader is from the range 128-254 (Private distributed algorithms), it is the responsibility of the network operator to guarantee that all nodes in the area have a common understanding of what the given algorithm value represents.

6.4. Flooding Behavior

Nodes that support Dynamic Flooding MUST use the flooding topology for flooding. The flooding topology is calculated locally in the case of distributed mode. In centralized mode the flooding topology is advertised in the area link state database. Link state updates received on one link in the flooding topology MUST be flooded on all other links in the flooding topology other than the link on which the update has been received. Link state updates received on a link not in the flooding topology MUST be flooded on all links in the flooding topology.

In centralized mode, if multiple flooding topologies are present in the area link state database, the node SHOULD flood on the union of the topologies.

When the flooding topology changes on a node, either as a result of the local computation in distributed mode or as a result of the advertisement from the Area Leader in centralized mode, the node MUST continue to flood on the union of the old and new flooding topology for a limited amount of time. This is required to provide all nodes sufficient time to migrate to the new flooding topology.

When failures occur, nodes will learn about them from link state updates and can compare those to the existing flooding topology. If the flooding topology becomes disconnected, then the nodes at the edges of the flooding topology should perform a database synchronization on all links. While the flooding topology is disconnected, if a new link state update is received on a link not in the flooding topology, then the node SHOULD temporarily consider the link as part of the flooding topology. When a new flooding topology is received or locally calculated, this MUST be discontinued.

7. IANA Considerations

7.1. IS-IS

This document requests the following code point from the "sub-TLVs for TLV 242" registry (IS-IS Router CAPABILITY TLV).

Type: TBD1

Description: IS-IS Area Leader Sub-TLV

Reference: This document (Section 5.1.1)

This document requests that IANA allocate and assign two code points from the "IS-IS TLV Codepoints" registry. One for each of the following TLVs:

Type: TBD2

Description: IS-IS Area System IDs TLV

Reference: This document (Section 5.1.2)

Type: TBD3

Description: IS-IS Flooding Path TLV

Reference: This document (Section 5.1.3)

7.2. OSPF

This document requests the following code point from the "OSPF Router Information (RI) TLVs" registry:

Type: TBD4

Description: OSPF Area Leader Sub-TLV

Reference: This document (Section 5.2.1)

This document requests the following code point from the "Opaque Link-State Advertisements (LSA) Option Types" registry:

Type: TBD5

Description: OSPFv2 Dynamic Flooding Opaque LSA

Reference: This document (Section 5.2.2)

This document requests the following code point from the "OSPFv3 LSA Function Codes" registry:

Type: TBD6

Description: OSPFv3 Dynamic Flooding LSA

Reference: This document (Section 5.2.3)

7.2.1. OSPF Dynamic Flooding LSA TLVs Registry

This specification also requests one new registry - "OSPF Dynamic Flooding LSA TLVs". New values can be allocated via IETF Review or IESG Approval

The "OSPF Dynamic Flooding LSA TLVs" registry will define top-level TLVs for the OSPFv2 Dynamic Flooding Opaque LSA and OSPFv3 Dynamic Flooding LSAs. It should be added to the "Open Shortest Path First (OSPF) Parameters" registries group.

The following initial values are allocated:

Type: 0

Description: Reserved

Reference: This document

Type: 1

Description: OSPF Area Router IDs TLV

Reference: This document (Section 5.2.4)

Type: 2

Description: OSPF Flooding Path TLV

Reference: This document (Section 5.2.5)

Types in the range 32768-33023 are for experimental use; these will not be registered with IANA, and MUST NOT be mentioned by RFCs.

Types in the range 33024-65535 are not to be assigned at this time. Before any assignments can be made in the 33024-65535 range, there MUST be an IETF specification that specifies IANA Considerations that covers the range being assigned.

7.3. IGP

IANA is requested to set up a registry called "IGP Algorithm Type For Computing Flooding Topology" under an existing "Interior Gateway Protocol (IGP) Parameters" IANA registries. The registration policy for this registry is "Standards Action" ([RFC8126] and [RFC7120]).

Values in this registry come from the range 0-255.

The initial values in the IGP Algorithm Type For Computing Flooding Topology registry are:

0: Reserved for centralized mode.

1-127: Available for standards action.

128-254: Reserved for private use.

255: Reserved.

8. Security Considerations

This document introduces no new security issues. Security of routing within a domain is already addressed as part of the routing protocols themselves. This document proposes no changes to those security architectures.

It is possible that an attacker could become Area Leader and introduce a flawed flooding algorithm into the network thus compromising the operation of the protocol. Authentication methods as describe in [RFC5304] and [RFC5310] for IS-IS, [RFC2328] and [RFC7474] for OSPFv2 and [RFC5340] and [RFC4552] for OSPFv3 SHOULD be used to prevent such attack.

9. Acknowledgements

The authors would like to thank Les Ginsberg, Zeqing (Fred) Xia, Naiming Shen, Adam Sweeney and Olufemi Komolafe for their helpful comments.

The authors would like to thank Tom Edsall for initially introducing them to the problem.

10. References

10.1. Normative References

[ISO10589]
International Organization for Standardization, "Intermediate System to Intermediate System Intra-Domain Routing Exchange Protocol for use in Conjunction with the Protocol for Providing the Connectionless-mode Network Service (ISO 8473)", ISO/IEC 10589:2002, 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>.

[RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, DOI 10.17487/RFC2328, April 1998, <https://www.rfc-editor.org/info/rfc2328>.

[RFC4552] Gupta, M. and N. Melam, "Authentication/Confidentiality for OSPFv3", RFC 4552, DOI 10.17487/RFC4552, June 2006, <https://www.rfc-editor.org/info/rfc4552>.

[RFC5250] Berger, L., Bryskin, I., Zinin, A., and R. Coltun, "The OSPF Opaque LSA Option", RFC 5250, DOI 10.17487/RFC5250, July 2008, <https://www.rfc-editor.org/info/rfc5250>.

[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>.

[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>.

[RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, DOI 10.17487/RFC5340, July 2008, <https://www.rfc-editor.org/info/rfc5340>.

[RFC7120] Cotton, M., "Early IANA Allocation of Standards Track Code Points", BCP 100, RFC 7120, DOI 10.17487/RFC7120, January 2014, <https://www.rfc-editor.org/info/rfc7120>.

[RFC7474] Bhatia, M., Hartman, S., Zhang, D., and A. Lindem, Ed., "Security Extension for OSPFv2 When Using Manual Key Management", RFC 7474, DOI 10.17487/RFC7474, April 2015, <https://www.rfc-editor.org/info/rfc7474>.

[RFC7770] Lindem, A., Ed., Shen, N., Vasseur, JP., Aggarwal, R., and S. Shaffer, "Extensions to OSPF for Advertising Optional Router Capabilities", RFC 7770, DOI 10.17487/RFC7770, February 2016, <https://www.rfc-editor.org/info/rfc7770>.

[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>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

10.2. Informative References

[Clos] Clos, C., "A Study of Non-Blocking Switching Networks", The Bell System Technical Journal Vol. 32(2), DOI 10.1002/j.1538-7305.1953.tb01433.x, March 1953, <http://dx.doi.org/10.1002/j.1538-7305.1953.tb01433.x>.

[Leiserson] Leiserson, C., "Fat-Trees: Universal Networks for Hardware-Efficient Supercomputing", IEEE Transactions on Computers 34(10):892-901, 1985.

[RFC2973] Balay, R., Katz, D., and J. Parker, "IS-IS Mesh Groups", RFC 2973, DOI 10.17487/RFC2973, October 2000, <https://www.rfc-editor.org/info/rfc2973>.

[RFC3630] Katz, D., Kompella, K., and D. Yeung, "Traffic Engineering (TE) Extensions to OSPF Version 2", RFC 3630, DOI 10.17487/RFC3630, September 2003, <https://www.rfc-editor.org/info/rfc3630>.

[RFC7938] Lapukhov, P., Premji, A., and J. Mitchell, Ed., "Use of BGP for Routing in Large-Scale Data Centers", RFC 7938, DOI 10.17487/RFC7938, August 2016, <https://www.rfc-editor.org/info/rfc7938>.

Authors’ Addresses

Tony Li
Arista Networks
5453 Great America Parkway
Santa Clara, California 95054
USA

Email: tony.li@tony.li

Peter Psenak
Cisco Systems, Inc.
Eurovea Centre, Central 3
Pribinova Street 10
Bratislava 81109
Slovakia

Email: ppsenak@cisco.com

Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: September 19, 2018

T. Li
Arista Networks

March 18, 2018

Dynamic Flooding for IS-IS
draft-li-dynamic-flooding-isis-01

Abstract

Routing with link state protocols in dense network topologies can result in sub-optimal convergence times due to the overhead associated with flooding. This can be addressed by decreasing the flooding topology so that it is less dense.

This document discusses extensions to the IS-IS routing protocol to support a solution to flooding in dense subgraphs.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

In recent years, there has been increased focused on how to address the dynamic routing of networks that have a bipartite (a.k.a. spine-leaf), Clos [Clos], or Fat Tree [Leiserson] topology. Conventional Interior Gateway Protocols (IGPs, i.e. IS-IS [ISO10589], OSPF [RFC5340]) under-perform, redundantly flooding information throughout the dense topology, leading to overloaded control plane inputs and thereby creating operational issues. For practical considerations, network architects have resorted to applying unconventional techniques to address the problem, applying BGP in the data center [RFC7938], however it is very clear that using an Exterior Gateway Protocol as an IGP is sub-optimal, if only due to the configuration overhead.

This problem is discussed in more detail in [Architecture], along with an architectural solution for the problem. The remainder of this document is focused on describing extensions to the IS-IS protocol to implement that architecture. Three additions appear to be necessary.

1. A TLV that an IS may inject into its LSP to indicate its preference for becoming Area Leader.

2. A TLV to carry the list of system IDs that compromise the flooding topology for the area.

3. A TLV to carry the adjacency matrix for the flooding topology for the area.

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. Area Leader TLV

The Area Leader TLV allows a system to indicate its eligibility and priority for becoming Area Leader. Intermediate Systems (routers) not advertising this TLV are not eligible to become Area Leader.

The Area Leader is the router with the numerically highest Area Leader priority in the area. In the event of ties, the router with the numerically highest system ID is the Area Leader. Due to transients during database flooding, different routers may not agree on the Area Leader.

The format of the Area Leader TLV is:

```
    0                   1                   2
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | TLV Type      | TLV Length    | Priority      |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

TLV Type: XXX

TLV Length: 1

Priority: 0-255, unsigned integer

3. Area System IDs TLV

The Area System IDs TLV is used by the Area Leader to enumerate the system IDs that it has used in computing the flooding topology. Conceptually, the Area Leader creates a list of system IDs for all routers in the area, assigning indices to each system, starting with index 0.

Because the space in a single TLV is small, it may require more than one TLV to encode all of the system IDs in the area. This TLV may recur in multiple LSP segments so that all system IDs can be advertised.

The format of the Area System IDs TLV is:

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    | TLV Type      | TLV Length    | Starting Index                |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    | Ending Index                  |L| Reserved    | System IDs ...
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     System IDs continued ....
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

TLV Type: YYY

TLV Length: 9 + (ID length * N)

Starting index: The index of the first system ID that appears in this TLV.

Ending index: The index of the last system ID that appears in this TLV.

L (Last): This bit is set if the ending index of this TLV is the last index in the full list of system IDs for the area.

System IDs: A concatenated list of system IDs for the area.

4. Flooding Path TLV

The Flooding Path TLV is used to denote a path in the flooding topology. The goal is an efficient encoding of the links of the topology. A single link is a simple case of a path that only covers two nodes. A connected path may be described as a sequence of indices: (I1, I2, I3, ...), denoting a link from the system with index 1 to the system with index 2, a link from the system with index 2 to the system with index 3, and so on.

If a path exceeds the size that can be stored in a single TLV, then the path may be distributed across multiple TLVs by the replication of a single system index.

Complex topologies that are not a single path can be described using multiple TLVs.

The Flooding Path TLV contains a list of system indices relative to the systems advertised through the Area System IDs TLV. At least 2 indices must be included in the TLV. Due to the lenth restriction of TLVs, this TLV can contain at most 126 system indices.

The Flooding Path TLV has the format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| TLV Type      | TLV Length    | Starting Index                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Index 2                       | Additional indices ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

TLV Type: ZZZ

TLV Length: 9 + Length of Matrix octet contents

Starting index: The index of the first system in the path.

Index 2: The index of the next system in the path.

Additional indices: A sequence of additional indices to systems along the path.

Matrix: The concatenated rows of the upper right triangular portion of the adjacency matrix for the flooding topology, padded with 0 bits to an octet boundary.

5. Acknowledgements

The author would like to thank Adam Sweeney for his diligent review.

6. IANA Considerations

This memo requests that IANA allocate and assign three code points from the IS-IS TLV Codepoints registry. One for each of the following TLVs:

1. Area Leader TLV

2. Area System IDs TLV

3. Flooding Path TLV

7. Security Considerations

This document introduces no new security issues. Security of routing within a domain is already addressed as part of the routing protocols themselves. This document proposes no changes to those security architectures.

8. References

8.1. Normative References

[ISO10589]
International Organization for Standardization, "Intermediate System to Intermediate System Intra-Domain Routing Exchange Protocol for use in Conjunction with the Protocol for Providing the Connectionless-mode Network Service (ISO 8473)", ISO/IEC 10589:2002, 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>.

8.2. Informative References

[Architecture]
Li, T., "An Architecture for Dynamic Flooding on Dense Graphs", Internet draft draft-li-dynamic-flooding, Jan. 2018.

[Clos] Clos, C., "A Study of Non-Blocking Switching Networks", The Bell System Technical Journal Vol. 32(2), DOI 10.1002/j.1538-7305.1953.tb01433.x, March 1953, <http://dx.doi.org/10.1002/j.1538-7305.1953.tb01433.x>.

[Leiserson]
Leiserson, C., "Fat-Trees: Universal Networks for Hardware-Efficient Supercomputing", IEEE Transactions on Computers 34(10):892-901, 1985.

[RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, DOI 10.17487/RFC5340, July 2008, <https://www.rfc-editor.org/info/rfc5340>.

[RFC7938] Lapukhov, P., Premji, A., and J. Mitchell, Ed., "Use of BGP for Routing in Large-Scale Data Centers", RFC 7938, DOI 10.17487/RFC7938, August 2016, <https://www.rfc-editor.org/info/rfc7938>.

Author’s Address

Tony Li
Arista Networks
5453 Great America Parkway
Santa Clara, California 95054
USA

Email: tony.li@tony.li