IS-IS for IP Internets
Internet-Draft
Intended status: Standards Track
Expires: September 10, 2015

J. Farkas, Ed.
Ericsson
N. Bragg
Ciena
P. Unbehagen
Avaya
G. Parsons
Ericsson
P. Ashwood-Smith
Huawei Technologies
C. Bowers
Juniper Networks
March 9, 2015

IS-IS Path Computation and Reservation
draft-farkas-isis-pcr-02

Abstract

IEEE 802.1Qca Path Control and Reservation (PCR) specifies explicit path control via IS-IS in Layer 2 networks in order to move beyond the shortest path capabilities provided by IEEE 802.1aq Shortest Path Bridging (SPB). IS-IS PCR provides capabilities for the establishment and control of explicit forwarding trees in a Layer 2 network domain. This document specifies the sub-TLVs for IS-IS PCR.

Copyright Notice

Table of Contents

1. Introduction

IEEE 802.1Qca Path Control and Reservation (PCR) [IEEE8021Qca] specifies extensions to IS-IS for the control of Explicit Trees (ETs). The PCR extensions are compatible with the Shortest Path Bridging (SPB) extensions to IS-IS specified by [RFC6329] and [IEEE8021aq] (already rolled into [IEEE8021Q]). Furthermore, IS-IS with PCR extensions relies on the SPB architecture and terminology; and some of the IS-IS SPB sub-TLVs are also leveraged. IS-IS PCR builds upon IS-IS and uses IS-IS in a similar way to SPB. IS-IS PCR only addresses point-to-point physical links, although IS-IS also supports shared media LANs.

This document specifies five IS-IS sub-TLVs for the control of explicit trees by IS-IS PCR in a Layer 2 network as specified by IEEE 802.1Qca. In addition to the sub-TLVs specified here, IS-IS PCR relies on the following IS-IS SPB sub-TLVs specified by [RFC6329]:

- SPB Link Metric sub-TLV
- SPB Base VLAN-Identifiers sub-TLV
- SPB Instance sub-TLV
- SPBV MAC address sub-TLV
- SPBM Service Identifier and Unicast Address sub-TLV

These sub-TLVs are used to provide the link metric and the associations among bridges, MAC addresses, VIDs and I-SIDs within an IS-IS domain. The use of these SPB sub-TLVs for PCR is specified by IEEE 802.1Qca. Note that IS-IS PCR does not require the implementation of the full IS-IS SPB protocol but only the support of these SPB sub-TLVs. A bridge can support both IS-IS SPB and IS-IS PCR at the same time but when it supports both they are implemented by the same IS-IS entity on a per instance basis.

The sub-TLVs specified here can be also applied for Fast ReRoute using Maximally Redundant Trees (MRT-FRR) [I-D.ietf-rtgwg-mrt-frr-architecture] in a Layer 2 network. MRTs are computed as specified in [I-D.ietf-rtgwg-mrt-frr-algorithm]. If MRT computation is split such that the Generalized Almost Directed Acyclic Graph (GADAG) is computed centrally, then these sub-TLVs can be used to distribute the GADAG, which is identical for each network node throughout a network domain.

PCR uses IS-IS, the SPB sub-TLVs listed above, and the new sub-TLVs defined here. IS-IS PCR has no impact to IETF protocols.

2. 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].

The lowercase forms with an initial capital "Must", "Must Not", "Shall", "Shall Not", "Should", "Should Not", "May", and "Optional" in this document are to be interpreted in the sense defined in

[RFC2119], but are used where the normative behavior is defined in documents published by SDOs other than the IETF.

3. Terminology and Definitions

ADAG: Almost Directed Acyclic Graph - a digraph that can be transformed into a DAG by removing all arcs incoming to the root. [I-D.ietf-rtgwg-mrt-frr-architecture]

B-VID: Backbone VID. [IEEE8021Q]

Base VID: The VID used to identify a VLAN in management operations. [IEEE8021aq]

BLCE: Bridge Local Computation Engine - A computation engine in a bridge that performs path and routing computations. The BLCE implements e.g. SPF, CSPF, or the Maximally Redundant Trees Algorithm. [IEEE8021Qca]

Constrained tree: A tree meeting a certain constraint, e.g. providing a minimal available bandwidth. [IEEE8021Qca]

Cut-node: A node is a cut-node if removing it partitions the network. [I-D.ietf-rtgwg-mrt-frr-architecture]

Cut-link: A link is a cut-link if removing it partitions the network. [I-D.ietf-rtgwg-mrt-frr-architecture]

DAG: Directed Acyclic Graph - a digraph containing no directed cycle. [I-D.ietf-rtgwg-mrt-frr-architecture]

DEI: Drop Eligible Indicator. [IEEE8021Q]

ECT Algorithm: Equal Cost Tree Algorithm - The algorithm and mechanism that is used for the control of the active topology, i.e. forwarding trees. It can be one of the shortest path algorithms specified by IEEE 802.1aq. It can be also one of the explicit path control algorithms specified by IEEE 802.1Qca. Each ECT Algorithm has a 32-bit unique ID. [IEEE8021aq]

ET: Explicit Tree - An explicitly defined tree, which is specified by its end points and the paths among the end points. If only the end points are specified but the paths are not, then it is a loose explicit tree. If the paths are also specified, then it is a strict explicit tree. [IEEE8021Qca]

ETDB: Explicit Tree Database - A database storing explicit trees. [IEEE8021Qca]

FDB: Filtering Database. [IEEE8021Q]

GADAG: Generalized ADAG - a digraph, which has only ADAGs as all of its topology blocks. [I-D.ietf-rtgwg-mrt-frr-architecture]

Hop: A hop is specified by two nodes. A strict hop has no intermediate nodes, whereas a loose hop can have one or more intermediate nodes. IS-IS PCR specifies an explicit tree by an ordered list of hops starting at the root, each successive hop being defined by the next element of the list. [IEEE8021Qca]

I-SID: Backbone Service Instance Identifier - A 24-bit ID. [IEEE8021Q]

Maximally Redundant Trees (MRTs): A pair of trees with a common MRT Root where the path from any leaf node to the MRT Root along the first tree (MRT-Blue) and the path from the same leaf node along the second tree (MRT-Red) share the minimum number of nodes and the minimum number of links. Each such shared node is a cut-node. Any shared links are cut-links. [I-D.ietf-rtgwg-mrt-frr-architecture]

MRT-Blue: MRT-Blue is one of the two MRTs; specifically, MRT-Blue is the increasing MRT where links in the GADAG are taken in the direction from a lower topologically ordered node to a higher one. [I-D.ietf-rtgwg-mrt-frr-architecture]

MRT-Red: MRT-Red is one of the two MRTs; specifically, MRT-Red is the decreasing MRT where links in the GADAG are taken in the direction from a higher topologically ordered node to a lower one. [I-D.ietf-rtgwg-mrt-frr-architecture]

MRT Root: The common root of the two MRTs: MRT-Blue and MRT-Red. [I-D.ietf-rtgwg-mrt-frr-architecture]

MSRP: Multiple Stream Registration Protocol, standardized as IEEE 802.1Qat, already rolled into [IEEE8021Q].

PCA: Path Control Agent - The agent that is part of the IS-IS domain and thus can perform IS-IS operations on behalf of a PCE, e.g. maintain the LSDB and send LSPs. [IEEE8021Qca]

PCE: Path Computation Element - An entity that is capable of computing a path through a network based on a representation of the topology of the network (obtained by undefined means external to the PCE). [RFC4655]

PCP: Priority Code Point, which identifies a traffic class. [IEEE8021Q]

PTP: Precision Time Protocol specified by [IEEE1588].

Redundant trees: A pair of trees with a common Root where the paths from any leaf node to the Root along the first tree and the second tree are disjoint. [I-D.ietf-rtgwg-mrt-frr-architecture]

SPBM: SPB MAC - The SPB mode where a MAC or its shorthand (SPSourceID: Shortest Path Source ID) is used to identify an SPT. [IEEE8021aq]

SPBV: SPB VID - The SPB mode where a unique VID is assigned to each SPT Root bridge and is used to identify an SPT. [IEEE8021aq]

SPF: Shortest Path First.

SPT: Shortest Path Tree. [IEEE8021aq]

SRLG: Shared Risk Link Group - A set of links that share a resource whose failure affects each link. [RFC5307]

TAI: Temps Atomique International - International Atomic Time. [IEEE1588]

topology block: Either a maximally two-connected cluster, a cut-link with its endpoints, or an isolated node. [I-D.ietf-rtgwg-mrt-frr-architecture]

TED: Traffic Engineering Database - A database storing the traffic engineering information propagated by IS-IS. [RFC5305]

two-connected: A graph that has no cut-nodes. This is a graph that requires at least two nodes to be removed before gets partitioned. [I-D.ietf-rtgwg-mrt-frr-architecture]

VID: VLAN ID. [IEEE8021Q]

VLAN: Virtual Local Area Network. [IEEE8021Q]

4. Explicit Trees

An explicit tree is determined by a Path Computation Element (PCE) [RFC4655] and is not required to follow the shortest path. A PCE is an entity that is capable of computing a topology for forwarding based on a network topology, its corresponding attributes, and potential constraints. A PCE MUST explicitly describe a forwarding tree as described in Section 6.1. Either a single PCE or multiple PCEs determine explicit trees for a domain. Even if there are multiple PCEs in a domain, each explicit tree MUST be only determined by one PCE, which is referred to as the owner PCE of the tree. PCEs and IS-IS PCR can be used in combination with IS-IS SPB shortest path routing.

The PCE interacts with the active topology control protocol, i.e. with IS-IS. The collaboration with IS-IS can be provided by a Path Control Agent (PCA) on behalf of a PCE. Either the PCE or the corresponding PCA is part of the IS-IS domain. If the PCE is not part of the IS-IS domain, then the PCE MUST be associated with a PCA that is part of the IS-IS domain. The PCE or its PCA MUST establish IS-IS adjacency in order to receive all the LSPs transmitted by the bridges in the domain. The PCE, either on its own or via its PCA, can control the establishment of explicit trees in that domain by injecting an LSP conveying an explicit tree and thus instruct IS-IS to set up the explicit tree determined by the PCE. If instructed to do so by a PCE, IS-IS MAY also record and communicate bandwidth assignments, which MUST NOT be applied if reservation protocol (e.g. Multiple Stream Registration Protocol (MSRP)) is used in the domain. Both MSRP and IS-IS MUST NOT be used to make bandwidth assignments in the same domain.

The operation details of the PCE are not specified by this document or by IEEE 802.1Qca. If the PCE is part of the IS-IS domain, then the PCE uses IS-IS PDUs to communicate with the IS-IS domain and the PCE has a live IS-IS LSDB, (i.e. the PCE implements the PCA functions too). A PCE can instead communicate with the IS-IS domain via a PCA, e.g. to retrieve the LSDB or instruct the creation of an explicit tree. However, the means of communication between the PCE and the PCA is not specified by this document or by IEEE 802.1Qca.

An Explicit Tree (ET) is an undirected loop-free topology, whose use is under the control of the owner PCE by means of associating VIDs and MAC addresses with it. An ET MUST NOT contain Cycles. As it is undirected, an ET contains no assumptions about the direction of any flows that use it; it can be used in either direction as specified by the VIDs and MAC addresses associated with it. It is the responsibility of the PCE to ensure reverse path congruency and multicast-unicast congruency if that is required.

An explicit tree is either strict or loose. A strict explicit tree specifies all bridges and paths it comprises. A loose tree only specifies the bridges as a list of hops that have a special role in the tree, e.g. a traffic end point, and no path or path segment is specified between the bridges, which are therefore loose hops even if traffic end points are adjacent neighbors. The special role of a hop can be: traffic end point, root, leaf, a bridge to be avoided, or a transit hop in case of a tree with a single leaf. The path for a loose hop is determined by the Bridge Local Computation Engine (BLCE) of the bridges. The shortest path is used for a loose hop unless specified otherwise by the descriptor (Section 6.1) of the tree or by the corresponding ECT Algorithm (Section 5).

A loose explicit tree is constrained if the tree descriptor includes one or more constraints, e.g. the administrative group that the links of the tree have to belong to. The BLCE of the bridges then apply the Constrained Shortest Path First (CSPF) algorithm, which is Shortest Path First (SPF) on the topology that only contains the links meeting the constraint(s).

An explicit tree is specified by a Topology sub-TLV (Section 6.1). The Topology sub-TLV associates one or more VIDs with an explicit tree. The Topology sub-TLV includes two or more Hop sub-TLVs (Section 6.2), and a hop is specified by an IS-IS System ID. A Hop sub-TLV MAY include a delay constraint for a loose hop. A Topology sub-TLV MAY also include further sub-TLVs to constrain loose hops. The bridges involved in an explicit tree store the corresponding Topology sub-TLVs in their Explicit Tree Database (ETDB).

Explicit trees are propagated and set-up by IS-IS PCR in a domain. The PCE or its PCA assembles the Topology sub-TLVs (Section 6.1), and adds it into an LSP, which is flooded throughout the domain. The Topology sub-TLV is flooded by the same techniques used for the SPB LSPs. The bridges then MUST process the Topology sub-TLV upon reception. If the Topology sub-TLV specifies one or more loose trees, then the path for the loose hops is determined by the BLCE of the bridges. The bridges then install the appropriate FDB entries for frame forwarding along the tree described by the Topology sub-TLV, or the trees computed based on the Topology sub-TLV. Dynamic Filtering Entries are maintained by IS-IS for the VID, MAC address tuples associated with an ET.

Due to the LSP aging of IS-IS, the Topology sub-TLVs (Section 6.1) have to be refreshed similar to other IS-IS TLVs in order to keep the integrity of the LSDB. The corresponding Dynamic Filtering Entries are also refreshed in the FDB when a Topology sub-TLV is refreshed. Refreshing Topology sub-TLVs is the task of the entity being part of the IS-IS domain, i.e. either the PCE or the PCA.

There is no precedence order between Explicit Trees. Precedence order among bandwidth assignments recorded by IS-IS PCR is specified in Section 6.4.

If it is not possible to install an explicit tree, e.g. constraint(s) cannot be met or the Topology sub-TLV is ill-formed, then no tree is installed but a management report is generated.

The bridges MAY support the following IS-IS features for the computation of explicit trees. The Extended IS Reachability TLV (type 22) specified in [RFC5305] provides the following link attribute IS-IS sub-TLVs:

- Administrative Group (color, resource class) (sub-TLV type 3),
- Maximum Link Bandwidth (sub-TLV type 9),
- Maximum Reservable Link bandwidth (sub-TLV type 10),
- Unreserved Bandwidth (sub-TLV type 11),
- Traffic Engineering Default Metric (sub-TLV type 18).

When the Unreserved Bandwidth sub-TLV is used in a Layer 2 bridge network, the priority value encoded in the sub-TLV provides the PCP, i.e. identifies a traffic class (not a setup priority level).

Further attributes are provided by the IS-IS TE Metric Extension link attribute sub-TLVs specified in [I-D.ietf-isis-te-metric-extensions]:

- Unidirectional Link Delay,
- Min/Max Unidirectional Link Delay,
- Unidirectional Delay Variation,
- Unidirectional Link Loss,
- Unidirectional Residual Bandwidth,
- Unidirectional Available Bandwidth,
- Unidirectional Utilized Bandwidth.

The Shared Risk Link Group (SRLG) information provided by the SRLG TLV (type 138) [RFC5307] MAY be also used. In order to indicate that the interface is unnumbered in this case, the corresponding flag takes value 0. The Link Local Identifier is an Extended Local Circuit Identifier and the Link Remote Identifier is a Neighbor Extended Local Circuit ID.

5. Explicit ECT Algorithms

The exact IS-IS control mode of operation MUST be selected for a VLAN by associating its Base VID with the appropriate ECT Algorithm in the SPB Base VLAN-Identifiers sub-TLV [RFC6329], in addition to allocating the Base VID to IS-IS control. There are five distinct ECT Algorithms for the five explicit path control modes. The operation details of the explicit ECT Algorithms and their configuration is specified by IEEE 802.1Qca, a high level overview is given here. An ECT Algorithm value consists of the IEEE 802.1 OUI (Organizationally Unique Identifier) value 00-80-C2 concatenated with an index [RFC6329].

The Strict Tree (ST) ECT Algorithm MUST be used for a strict explicit tree. A strict ET is static as no other entity can update it but the owner PCE. In case of a topology change, it is the task of the owner PCE to detect the topology change, e.g. based on the changes in the LSDB, and to update the strict trees if needed. That is, the owner PCE computes the new tree, assembles its descriptor (Section 6.1), and then instructs IS-IS PCR to install it. The value for the ST ECT algorithm is 00-80-C2-17.

The Loose Tree (LT) ECT Algorithm MAY be also supported. It is used for a single loose explicit tree. The path for loose hops is determined by the BLCE of the bridges; therefore, the Topology sub-TLV (Section 6.1) specifying the tree MUST indicate which hop is the Root of the tree. The loose hops are maintained by IS-IS, i.e. restored upon a topology change if a loop-free path is available. If the tree computed by the BLCE visits the same bridge twice (implying that a loop or hairpin has been created), then that loop or hairpin MUST be pruned from the tree even if it contains a hop specified by the Topology sub-TLV. It is a constraint if a bridge is not to be included, which can be specified by the Exclude flag of a Hop sub-TLV (Section 6.2) conveyed by the Topology sub-TLV specifying the tree. The range of values for the LT ECT Algorithms is 00-80-C2-21...00-80-C2-30.

The Loose Tree Set (LTS) ECT Algorithm MAY be also supported. It is used if connectivity among the traffic end points specified by the Topology sub-TLV (Section 6.1) is to be provided by a set of loose trees such that one tree is rooted at each traffic end point. The BLCE of the bridges compute the loose trees, which are maintained by IS-IS, i.e. restored upon a topology change. One constraint can be to avoid some bridges in these trees, which can be specified by the Exclude flag (item c.6. in Section 6.2). Further constraints can be specified by the Topology sub-TLV. The range of values for the LT ECT Algorithms is 00-80-C2-31...00-80-C2-40.

The LT and LTS ECT Algorithms use the shortest paths after pruning the topology according to the constraint(s) if any. The shortest path tie-breaking specified by Section 12 of [RFC6329] is applied (see also subclauses 28.5 - 28.8 of [IEEE8021aq]), that's why range of values are associated with the LT and LTS ECT Algorithms. In case of the LT ECT Algorithm, the indexes are 0x21...0x30, and ECT-MASK{index-0x20} is applied to retrieve the ECT-MASK of Section 12 of [RFC6329]. In case of the LTS ECT Algorithm, the indexes are 0x31...0x40, and ECT-MASK{index-0x30} is applied to retrieve the ECT-MASK for shortest path tie-breaking.

The MRT ECT Algorithm MAY be also supported. It is used for the establishment and maintenance of MRTs in a distributed fashion. The MRT Lowpoint Algorithm specified by [I-D.ietf-rtgwg-mrt-frr-algorithm] MUST be used for the computation of MRTs. The MRT Lowpoint Algorithm first computes the GADAG then produces two MRTs for each MRT Root: MRT-Blue and MRT-Red. If the level of redundancy provided by each bridge being an MRT Root is not required, then the MRT Roots can be specified by a Topology sub-TLV (Section 6.1). Both the GADAG and the MRT computation steps are performed distributed, i.e. by each bridge. The value for the MRT ECT algorithm is 00-80-C2-18.

The MRT GADAG (MRTG) ECT Algorithm MAY be also supported. It splits the computation into two. As the GADAG is identical for each MRT within a domain, it is computed by a single entity, which is the GADAG Computer. The GADAG is then described in a Topology sub-TLV (Section 6.1), which is flooded in the domain. The bridges then compute the MRTs for the MRT Roots based on the GADAG received. Section 7 provides more details on the description of the GADAG. The value for the MRTG ECT algorithm is 00-80-C2-19.

MRTs are loose trees as bridges are involved in their computation and restoration. Thus both the MRT and the MRTG ECT Algorithms provide a set of loose trees: two MRTs for each MRT Root.

6. IS-IS PCR sub-TLVs

The following sub-TLVs are specified for IS-IS PCR. The Topology sub-TLV MUST be carried in an MT-Capability TLV, the rest of the sub-TLVs are conveyed by Topology sub-TLV.

6.1. Topology sub-TLV

The variable length Topology sub-TLV MUST be used to describe an explicit tree. The Topology sub-TLV MAY be also used for describing a Generalized Almost Directed Acyclic Graph (GADAG) as explained in Section 7 in detail. The Topology sub-TLV MUST be carried in an MT-

Capability TLV (type 144) [RFC6329] in a Link State PDU. A Topology sub-TLV specifying an explicit tree conveys one or more Base VIDs, two or more Hop sub-TLVs (Section 6.2). A Topology sub-TLV describing a loose tree MAY also convey further sub-TLVs to specify constraints. Figure 1 shows the format of the Topology sub-TLV.

```
 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      |                  (1 byte)
+-+-+-+-+-+-+-+-+
|    Length     |                  (1 byte)
+-+-+-+-+-+-+-+-+
| Num Base VIDs |                  (1 byte)
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Res  |  Base VID 1 (12 bits) |   (0 or 2 bytes)
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       .................
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Res  |  Base VID n (12 bits) |   (0 or 2 bytes)
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        sub-TLV 1  (variable)                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
                         .................
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        sub-TLV m  (variable)                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: Topology sub-TLV

The parameters of explicit trees are encoded by the Topology sub-TLV as follows:

a. Type (8 bits): The type of the sub-TLV, its value is TBD.

b. Length (8 bits): The total number of bytes contained in the Value field.

c. Number of Base VIDs (8 bits): The number of Base VIDs carried in the Topology sub-TLV. Its minimum value is 1 if the Topology sub-TLV specifies one or more explicit trees. Its value can be 0 if the Topology sub-TLV specifies a GADAG.

d. Reserved (Res) (4 bits): The reserved bits take value 0.

e. Base VID (12 bits): The Base VID parameter provides the Base VID of the VLAN that is associated with the explicit tree. Multiple Base VIDs can be associated with the same explicit tree. In addition to the Base VID, some of the explicit ECT Algorithms (Section 5) require further VIDs which are associated with the VLAN via the SPB Instance sub-TLV [RFC6329]. A Topology sub-TLV specifying a GADAG can have zero Base VID parameters. In this case, the given GADAG MUST be applied for each VLAN associated with the MRTG ECT Algorithm (Section 5).

f. sub TLVs: The rest conveys further sub-TLVs that specify the hops of the topology and can also specify constraints as described in the following.

A topology is specified by a list of Hop sub-TLVs (Section 6.2), and a hop is specified by an IS-IS System ID. An ill-formed Topology sub-TLV, e.g. specifying an invalid or inconsistent tree is ignored, no tree is installed but a management report is generated.

The Topology sub-TLV specifies a strict tree by decomposing the tree to branches. Each branch is a point-to-point path specified by an ordered list of hops where the end of each branch is a leaf. Each element of a branch is the direct link between adjacent neighbor bridges whose Hop sub-TLV is next to each other in the Topology sub-TLV. The first hop of the Topology sub-TLV is the root, hence, the first branch originates from the root. The rest of the branches fork from another branch. The first hop of a branch is a bridge that is already part of a former branch and the last hop is a leaf bridge. Therefore, the hop after a leaf hop is the beginning of a new branch, if any. A hop of a branch is created if and only if the bridge specified for that hop is directly connected to the preceding bridge of the same branch. The first branch MUST begin with the root and after that the order of the branches does not matter within the Topology sub-TLV. Figure 2 shows an example strict tree and its description.

```
                                          +-----------+
                                          |     A     |
                                          +-----------+
                                          |     I     |
                                          +-----------+
                                          |     H     |
                   [B]---[A]---[I]        +-----------+
                    |           |         |     G     |
                    |           |         +-----------+
                    |           |         |     E     |
                   [C]---[F]   [H]        +-----------+
                    |           |         |     A     |
                    |           |         +-----------+
                    |           |         |     B     |
                   [D]   [E]---[G]        +-----------+
                                          |     C     |
                                          +-----------+
                                          |     D     |
                                          +-----------+
                                          |     C     |
                                          +-----------+
                                          |     F     |
                                          +-----------+
```

Figure 2: A strict tree and its description; root = Node A

The Topology sub-TLV of a loose tree does not provide any path or path segment, but the hops which are to participate. The root MUST be the first hop. The leaves of a single loose tree MUST be also specified. Hop sub-TLVs can be included in a Topology sub-TLV to specify bridges that have to be avoided. If the Topology sub-TLV only specifies a single leaf, then one or more transit hops can be specified by the Topology sub-TLV to direct the path along a sequence of bridges, specified by the order of hops. If bridges whose respective Hop sub-TLVs are adjacent to each other in the Topology sub-TLV but are not topology neighbors, then it is a loose hop. If a Topology sub-TLV conveys one or more loose hops, then that sub-TLV defines a loose explicit tree and each hop is considered as a loose hop. The path of a loose hop MUST be pruned from the tree if the path would create a loop or hairpin.

If the Base VIDs of the Topology sub-TLV are associated with the LTS ECT Algorithm or the MRT ECT Algorithm, then the Hop sub-TLVs conveyed by the Topology sub-TLV belong to traffic end points or bridges to be excluded. The BLCEs compute the loose trees, e.g. MRTs, such that they span the traffic end points and are rooted at a traffic end point.

The Topology sub-TLV specifies a GADAG if the Base VIDs conveyed by the Topology sub-TLV are associated with the MRTG ECT Algorithm. Section 7 provides the details on the description of a GADAG by a Topology sub-TLV.

Each traffic end point of an explicit tree MUST be always specified in the Topology sub-TLV by the inclusion of the Hop sub-TLVs corresponding to the traffic end points. The traffic end points of a tree are identified by setting the Traffic End Point flag (item c.3. in Section 6.2) in the appropriate Hop sub-TLVs.

If the explicit tree is loose, then the Topology sub-TLV MAY convey further sub-TLVs to specify constraints, e.g. an Administrative Group sub-TLV [RFC5305] or a Bandwidth Constraint (Section 6.3). If it is not possible to meet the constraint(s) specified by the Topology sub-TLV, then no tree is installed but a management report is generated.

If IS-IS PCR is used for recording bandwidth assignment, then the Topology sub-TLV conveys Bandwidth Assignment sub-TLV (Section 6.4) and it can also convey Timestamp sub-TLV (Section 6.5). If the bandwidth assignment specified by the Topology sub-TLV is not possible, e.g. due to overbooking, then bandwidth assignment MUST NOT be performed and a management report is generated. If the Topology sub-TLV specifies a new valid explicit tree, then the tree is installed without bandwidth assignment.

6.2. Hop sub-TLV

The Hop sub-TLV MUST be used to specify a hop of a topology. Each Hop sub-TLV conveys an IS-IS System ID, which specifies a hop. A Hop sub-TLV is conveyed by a Topology sub-TLV (Section 6.1). A strict explicit tree is decomposed to branches where each branch is a point-to-point path specified by an ordered list of Hop sub-TLVs as specified in Section 6.1. A hop of a branch is created if and only if the bridge specified for that hop is directly connected to the preceding bridge in the path. That is, a point-to-point LAN is identified by the two bridges it interconnects; and the LAN is part of the strict tree if and only if the Hop sub-TLVs of the two bridges are next to each other in the Topology sub-TLV. A Hop sub-TLV can convey a Circuit ID in order to distinguish multiple links between adjacent neighbor bridges. A Hop sub-TLV also specifies the role of a bridge, e.g. if it is the root or a traffic end point. The Topology sub-TLV of a loose tree only comprises the Hop sub-TLV of the bridges that have special role in the tree. The Hop sub-TLV MAY also specify a delay budget for a loose hop.

By default, the traffic end points both transmit and receive with respect to each VID associated with an explicit tree, except for an

LTS (Section 5) associated with a learning VLAN, which uses a unidirectional VID per bridge. The Hop sub-TLV allows different configuration by means of the Transmit (T) and Receive (R) flags conveyed in the sub-TLV. The VID and its T/R flags are only present in the Hop sub-TLV if the behavior of the traffic end points differs from the default.

Figure 3 shows the format of the variable length Hop sub-TLV, which MUST be conveyed by a Topology sub-TLV (Section 6.1).

```
 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      |                       (1 byte)
+-+-+-+-+-+-+-+-+
|    Length     |                       (1 byte)
+-+-+-+-+-+-+-+-+
|C|V|T|R|L|E|Res|                       (1 byte)
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           System ID                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          System ID            |       (6 bytes)
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Extended Local Circuit ID  (0 or 4 bytes)          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Num of VIDs  |                       (0 or 1 byte)
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|T|R|Res|     VID 1   (12 bits) |       (0 or 2 bytes)
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       .................
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|T|R|Res|     VID n   (12 bits) |       (0 or 2 bytes)
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Delay Constraint                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Delay Constraint       |       (0 or 6 bytes)
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: Hop sub-TLV

The parameters of a hop are encoded as follows:

a. Type (8 bits): The type of the sub-TLV, its value is TBD.

b. Length (8 bits): The total number of bytes contained in the Value field.

c. Hop Flags (8 bits): The Hop sub-TLV conveys six one-bit flags. The Circuit and the VID flags influence the length of the Hop sub-TLV. Two bits are reserved for future use, transmitted as 0 and ignored on receipt.

1. Circuit (C) flag (1 bit): The Circuit flag is a one-bit flag to indicate whether or not the Extended Local Circuit ID parameter is present. If the flag is set, then an Extended Local Circuit ID is also included in the Hop sub-TLV.

2. VID (V) flag (1 bit): The VID flag is a one-bit flag to indicate whether or not one or more VIDs are conveyed by the Hop sub-TLV. If the flag is set, then the Number of VIDs parameter is present and indicates how many VIDs are conveyed by the Hop sub-TLV. If the VID flag is reset, then neither the Number of VIDs parameter nor VIDs are present in the Hop sub-TLV.

3. Traffic End Point (T) flag (1 bit): The Traffic End Point flag is a one-bit flag to indicate whether or not the given System is a traffic end point, i.e. transmitter and/or receiver. If the System is a traffic end point, then the Traffic End Point flag MUSt be set. (The Traffic End Point flag indicates whether FDB entries are to be installed for the given hop.)

4. Root (R) flag (1 bit): The Root flag is a one-bit flag to indicate whether or not the given System is a Root of the explicit tree specified by the Topology sub-TLV. If the System is a root of a tree, then the Root flag MUST be set. If the Topology sub-TLV specifies a single tree, i.e. the Base VIDs conveyed by the Topology sub-TLV are associated with either the ST ECT Algorithm or the LT ECT Algorithm (Section 5), then the Root flag is only set for one of the Systems conveyed by the Topology sub-TLV. Furthermore, the first Hop sub-TLV of the Topology sub-TLV conveys the System that is the root of the tree. If the Topology sub-TLV specifies a Loose Tree Set, i.e. the Base VIDs conveyed by the Topology sub-TLV are associated with the LTS ECT Algorithm (Section 5), then the Root flag is set for each traffic end point as each of them roots a tree. If the Topology sub-TLV is used for MRT operations, i.e. the Base VIDs conveyed by the Topology sub-TLV are associated with either the MRT ECT Algorithm or the MRTG ECT Algorithm (Section 5), then the Root flag is set for each MRT Root. If no MRT Root is specified by a Topology sub-TLV specifying a GADAG, then each SPT Root is an MRT Root as well. If the Base VIDs conveyed by the Topology sub-TLV are associated

with the MRTG ECT Algorithm (Section 5), then the Topology sub-TLV specifies a GADAG and the very first Hop sub-TLV specifies the GADAG Root. There is no flag for indicating the GADAG Root.

5. Leaf (L) flag (1 bit): The Leaf flag is a one-bit flag to indicate whether or not the given System is a Leaf of the explicit tree specified by the Topology sub-TLV. If the System is a Leaf, then the Leaf flag MUST be set. The Leaf flag is only used to mark a leaf of a tree if the Topology sub-TLV specifies a single tree. The Leaf flag MUST be used to indicate the end of a topology block if the Topology sub-TLV specifies a GADAG, see Section 7.

6. Exclude (E) flag (1 bit): The Exclude flag is a one-bit flag to indicate if the given System MUST be excluded from the topology. The Exclude flag and the Root flag cannot be set for a given hop at the same time.

7. Reserved (Res) (2 bits): The reserved bits take value 0.

d. System ID (48 bits): The 6-byte IS-IS System Identifier of the bridge that the Hop sub-TLV refers to.

e. Extended Local Circuit ID (32 bits): The Extended Local Circuit ID [RFC5303] parameter is not necessarily present in the Hop sub-TLV. Its presence is indicated by the Circuit flag. Parallel links corresponding to different IS-IS adjacencies between a pair of neighbor bridges can be distinguished by means of the Extended Local Circuit ID. The Extended Local Circuit ID is conveyed by the Hop sub-TLV specifying the bridge nearer to the root of the tree, and identifies a circuit that attaches the given bridge to its neighbor cited by the next Hop sub-TLV of the Topology sub-TLV. The Extended Local Circuit ID can only be used in strict trees.

f. Number of VIDs (8 bits): The Number of VIDs parameter is not present if the Hop sub-TLV does not convey VIDs, which is indicated by the VID flag.

g. VID and its T/R flags (14 bits): The VID and its T/R flags are only present in the Hop sub-TLV if the given bridge is a traffic end point and it behaves differently from the default with respect to that particular VID.

1. T flag (1 bit): This is the Transmit allowed flag for the VID following the flag.

2. R flag (1 bit): This is the Receive allowed flag for the VID following the flag.

3. Reserved (Res) (2 bits): The reserved bits take value 0.

4. VID (12 bits): A VID.

h. Delay Constraint (48 bits): The last six bytes specify a delay constraint if they convey a Unidirectional Link Delay sub-TLV [I-D.ietf-isis-te-metric-extensions]. The delay constraint MAY be used in a Topology sub-TLV that specifies a single loose tree, i.e. the Base VIDs are associated with the LT ECT Algorithm (Section 5). If delay constraint is applied, then the loose hop MUST fit in the delay budget specified by the Delay parameter of the Unidirectional Link Delay sub-TLV conveyed by the Hop sub-TLV. If the Topology sub-TLV specifies a single leaf, then the path between the preceding Hop sub-TLV and the current Hop sub-TLV MUST meet the delay budget. If the Topology sub-TLV specifies multiple leaves, then the path between the root and the current Hop sub-TLV MUST to meet the delay budget. If the tree is used as a reverse congruent tree, then the delay constraint applies in both directions. If the tree is used as a directed tree, then the delay constraint applies in the direction of the tree. If it is not possible to meet the delay constraint specified by the Topology sub-TLV, then no tree is installed but a management report is generated.

6.3. Bandwidth Constraint sub-TLV

The Bandwidth Constraint sub-TLV MAY be included in a Topology sub-TLV (Section 6.1) in order to specify how much available bandwidth is to be provided by the constrained tree. Each loose hop MUST meet the bandwidth constraint. The bandwidth value of the constraint is a total value or it only refers to a single PCP as specified by the sub-TLV. Figure 4 shows the format of the Bandwidth Constraint sub-TLV.

```
 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      |                 (1 byte)
+-+-+-+-+-+-+-+-+
|    Length     |                 (1 byte)
+-+-+-+-+-+-+-+-+
| PCP |D|P| Res |                 (1 byte)
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Available Bandwidth  (4 bytes)                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4: Bandwidth Constraint sub-TLV

The parameters of the bandwidth constraint are encoded as follows:

a. Type (8 bits): The type of the sub-TLV, its value is TBD.

b. Length (8 bits): The total number of bytes contained in the Value field. The value of the Length field is 5 bytes.

c. PCP (4 bits): The Priority Code Point (PCP) parameter identifies the traffic class the Available Bandwidth parameter refers to, if any.

d. DEI (D) (1 bit): This is the Drop Eligible Indicator (DEI) parameter. If the DEI parameter is clear, then the bandwidth constraint refers to committed information rate. If the DEI parameter is set, then the bandwidth constraint refers to peak information rate.

e. PCP (P) flag (1 bit): If this flag is set, then the PCP parameter is taken into account.

f. Reserved (Res) (3 bits): The reserved bits take value 0.

g. Available Bandwidth (32 bits): The Available Bandwidth is specific to the traffic class identified by the PCP parameter if the PCP flag is set, otherwise, it is total bandwidth. In-line with the bandwidth parameters specified in [RFC5305], the Available Bandwidth is encoded as a 32-bit IEEE floating point number, and the units are bytes (not bits!) per second. When the Unreserved Bandwidth sub-TLV (sub-TLV type 11 specified by [RFC5305]) is used in a Layer 2 bridge network, the priority value encoded in the Unreserved Bandwidth sub-TLV provides the PCP, i.e. identifies a traffic class (not a setup priority level). Thus, the Available Bandwidth of a traffic class is easily comparable with the Unreserved Bandwidth stored in the TED for the given traffic class. The bandwidth constraint applies for both directions in case of symmetric explicit trees. Nevertheless, a VID associated with an explicit tree can be made unidirectional by means of the T/R flags belonging to the VID in the Hop sub-TLV (item g. in Section 6.2) of the traffic end points. If all the VIDs of the Topology sub-TLV (Section 6.1) are unidirectional and all belong to the traffic class identified by the PCP parameter of the Bandwidth Constraint sub-TLV, then it is enough to meet the bandwidth constraint in the direction applied for those VIDs.

6.4. Bandwidth Assignment sub-TLV

IS-IS PCR MAY be used for recording bandwidth assignment for explicitly placed data traffic in a domain if MSRP is not used within the domain. If MSRP is used in a domain, then only MSRP performs reservations. Both MSRP and IS-IS MUST NOT be used to make bandwidth assignments in the same domain.

The Bandwidth Assignment sub-TLV can be used to define the amount of bandwidth whose assignment is to be recorded by IS-IS PCR at each hop of the explicit tree described by the corresponding Topology sub-TLV (Section 6.1). The Bandwidth Assignment sub-TLV is used by IS-IS PCR for the recording of bandwidth assignment for a traffic class identified by the PCP parameter of a VLAN tag. If precedence order has to be determined among bandwidth assignments in a domain with multiple PCEs, then IS-IS PCR does it as described below. If the bandwidth assignment specified by the Topology sub-TLV is not possible, e.g. due to overbooking, then bandwidth recording MUST NOT be performed and a management report is generated. If the Topology sub-TLV specifies a new valid explicit tree, then the tree is installed without bandwidth assignment. The Bandwidth Assignment sub-TLV is conveyed by a Topology sub-TLV (Section 6.1). Figure 5 shows the format of the Bandwidth Assignment sub-TLV.

```
 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      |                 (1 byte)
+-+-+-+-+-+-+-+-+
|    Length     |                 (1 byte)
+-+-+-+-+-+-+-+-+
| PCP |D| Imp |R|                 (1 byte)
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Bandwidth  (4 bytes)                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 5: Bandwidth Assignment sub-TLV

The parameters of the bandwidth constraint are encoded as follows:

a. Type (8 bits): The type of the sub-TLV, its value is TBD.

b. Length (8 bits): The total number of bytes contained in the Value field. The value of the Length field is 5 bytes.

c. PCP (3 bits): The PCP parameter identifies the traffic class the bandwidth to be assigned for.

d. DEI (D) (1 bit): This is the Drop Eligible Indicator (DEI) parameter. If the DEI parameter is clear, then the bandwidth assignment is performed for providing committed information rate. If the DEI parameter is set, then the bandwidth assignment is performed for providing peak information rate.

e. Importance (Imp) (3 bits): This is the Importance parameter for determining precedence order among bandwidth assignments within a PCP as described below. Lower numerical value indicates more important bandwidth assignment within a PCP. The default value of the Importance parameter is 7.

f. Reserved (R) (1 bit): The reserved bit takes value 0.

g. Bandwidth (32 bits): This is the amount of bandwidth to be assigned for the traffic class identified by the PCP parameter. In-line with the bandwidth values specified in [RFC5305], the Bandwidth parameter is encoded as a 32-bit IEEE floating point number, and the units are bytes (not bits!) per second. The bandwidth assignment applies for both directions in case of symmetric explicit trees.

The PCEs are collectively responsible for making a consistent set of bandwidth assignments when IS-IS PCR is used for recording bandwidth allocations. If despite of that, precedence ordering is required among bandwidth assignments, then ordering based on the following parameters MUST be applied:

1. PCP parameter of Bandwidth Assignment sub-TLV,

2. Importance parameter of Bandwidth Assignment sub-TLV,

3. Timestamp sub-TLV (if present in the Topology sub-TLV).

A bandwidth assignment takes precedence if it has higher PCP, or higher Importance within a PCP, or earlier timestamp in case of equal Importance within a PCP. A bandwidth assignment associated with a timestamp takes precedence over a bandwidth assignment without timestamp. If resolution is not possible based on the above parameters or they are not available, e.g. each bandwidth assignment lacks timestamp or the same VID is called for, then the item is granted to the PCE whose LSP has the numerically least LSP ID.

6.5. Timestamp sub-TLV

The Timestamp sub-TLV MAY be included in a Topology sub-TLV (Section 6.1) in order to provide precedence order among equally important bandwidth assignments within a PCP as described in Section 6.4. Figure 6 shows the format of the Timestamp sub-TLV.

```
 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      |                   (1 byte)
+-+-+-+-+-+-+-+-+
|    Length     |                   (1 byte)
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Time       (4 bytes)                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 6: Timestamp sub-TLV

The timestamp represents a positive time with respect to the Precision Time Protocol (PTP) epoch and it is encoded as follows:

a. Type (8 bits): The type of the sub-TLV, its value is TBD.

b. Length (8 bits): The total number of bytes contained in the Value field. The value of the Length field is 4 bytes.

c. Time (32 bits): This is the time in units of seconds with respect to the PTP epoch.

The Timestamp sub-TLV carries the seconds portion of PTP as specified by [IEEE1588]. The epoch is 1970-01-01 00:00:00 TAI (i.e., the PTP time does not include leap seconds).

7. MRT-FRR Application

The application of MRT by [IEEE8021Qca] is discussed in detail in [I-D.bowers-rtgwg-mrt-applicability-to-8021qca]. This section describes some special considerations for the use of the MRT Lowpoint Algorithm [I-D.ietf-rtgwg-mrt-frr-algorithm], which are applicable both to the MRT ECT Algorithm and the MRTG ECT Algorithm. This section also explains details related to the MRTG ECT Algorithm and the application of the Topology sub-TLV in particular.

The SPB Link Metric sub-TLV [RFC6329] specifies the metric of each link for IS-IS PCR including the MRT Algorithms. If the SPB Link Metric values advertised by different ends of an adjacency are different, then the maximum value MUST be used. If equal cost (sub)paths are found during the MRT computation, then the default tie-breaking specified by Section 11 of [RFC6329] MUST be used, which is based on the lower Bridge ID. (The BridgeID is an 8-byte quantity whose upper 2 bytes are the node’s BridgePriority and lower 6 bytes are the node’s SYSID.) Note also that if MRTs are used for source specific multicast (see [IEEE8021Qca] for details), then the bridges have to compute the MRTs of the other bridges in addition to their own one in order to be able to install the appropriated FDB entries. (This is similar to the need for all pairs shortest path computation instead of Dijkstra for source specific shortest path multicast trees.)

The GADAG is identical for all the MRTs within a network domain, as a consequence of the use of the MRT Lowpoint Algorithm [I-D.ietf-rtgwg-mrt-frr-algorithm]. Therefore, it is beneficial to compute the GADAG by a single entity, which is referred to as the GADAG Computer and is either a PCE or the GADAG Root. If the MRTG ECT Algorithm is applied, then the GADAG MUST be only computed by the GADAG Computer, which then MUST flood the descriptor Topology sub-TLV of the GADAG. The bridges then compute the MRTs based on the received GADAG.

The GADAG computation requires the selection of the GADAG Root. The bridge with the best Bridge Identifier MUST be selected as the GADAG Root, where the numerically lower value indicates the better identifier. The Bridge Priority component of the Bridge Identifier allows the configuration of the GADAG Root by management action. The Bridge Priority is conveyed by the SPB Instance sub-TLV [RFC6329].

The GADAG Computer MUST perform the GADAG computation as specified by the MRT Lowpoint Algorithm [I-D.ietf-rtgwg-mrt-frr-algorithm]. The GADAG Computer then MUST encode the GADAG in a Topology sub-TLV (Section 6.1), which is then flooded throughout the domain. A GADAG is encoded in a Topology sub-TLV by means of directed ear decomposition as follows. A directed ear is a directed point-to-point path whose end points can coincide but no other element of the path is repeated in the ear. Each ear is specified by an ordered list of hops such that the order of hops is according to the direction of the arcs in the GADAG. There are no leaves in a GADAG, hence, the Leaf flag (item c.5. in Section 6.2) is used to mark the end of a topology block. (A GADAG with multiple blocks is illustrated in Figure 8.) The sequence of ears in the Topology sub-TLV is such that the end points of an ear belong to preceding ears. The GADAG Root is not marked by any flag but the GADAG Root is the first hop in the Topology sub-TLV, correspondingly the first ear starts and ends with the GADAG Root. MRT Roots MUST be marked by the Root flag (item c.4. in Section 6.2) and all other traffic end points are leaves of the given MRTs. If no MRT Root is specified, then each SPT Root is also an MRT Root.

Figure 7 shows an example GADAG. The figure also illustrates the description of the GADAG, it shows the System ID parameter of the Hop sub-TLV (Section 6.2) and the order of hops in the Topology sub-TLV (Section 6.1).

```
                                                       Leaf
                                               Hop     flag
                                          +-----------+---+
                                          |     A     |   |
                                          +-----------+---+
                                          |     B     |   |
                                          +-----------+---+
                                          |     C     |   |
                                          +-----------+---+
                                          |     F     |   |
               [B]<---[A]<---[I]          +-----------+---+
                |      ^      ^           |     A     |   |
                |      |      |           +-----------+---+
                V      |      |           |     C     |   |
               [C]--->[F]--->[H]          +-----------+---+
                |             ^           |     D     |   |
                |             |           +-----------+---+
                V             |           |     E     |   |
               [D]--->[E]--->[G]          +-----------+---+
                                          |     G     |   |
                                          +-----------+---+
                                          |     H     |   |
                                          +-----------+---+
                                          |     I     |   |
                                          +-----------+---+
                                          |     A     |   |
                                          +-----------+---+
                                          |     F     |   |
                                          +-----------+---+
                                          |     H     | X |
                                          +-----------+---+
```

Figure 7: A GADAG and its description; GADAG root = Node A

A topology can be comprised of multiple blocks, like the one illustrated in Figure 8(a). This example topology is comprised of four blocks as each cut-link is a block. A-B-C-D-E-F is a block, D-G is another block, G-H, and H-J-K are further blocks. The GADAG for this topology is shown in Figure 8(b). Note that the GADAG includes two arcs for each cut-link and the direction of each arc is different, e.g. D->G and G->D. The encoding starts with the Block (ADAG) involving the GADAG Root as illustrated in Figure 8. The first hop in the Topology sub-TLV is the GADAG Root (node A in this example.) The ADAG of the first block is then described using the ear decomposition, as described above. In this example, the first block has been completely traversed at the second occurrence of node A in the GADAG descriptor. The end of a block is indicated by setting the Leaf flag for the last hop of the block, e.g. for the second occurrence of node A in the example GADAG descriptor. The next node that appears in the GADAG descriptor (D in this case) is the localroot for the nodes in the next block. Continuing this process, the Leaf flag is set for the third occurrence of D, the third occurrence of G, and the third occurrence of H, each indicating the end of a block. The first hop of the first block is the GADAG Root, the fist hop in the rest of the blocks is the localroot. The position of the set Leaf flags helps to determine the localroot, which is the next hop. In the example GADAG descriptor, one can determine that A is the localroot for B,C,D,E,F (and A is the GADAG Root). D is the localroot for G. G is the localroot for H. And H is the localroot for J and K. The GADAG Root is assigned a localroot of None.

Block IDs are reconstructed while parsing a Topology sub-TLV specifying a GADAG. The current Block ID starts at 0 and is assigned to the GADAG Root. A node appearing in the GADAG descriptor without a previously-assigned Block ID value is assigned the current Block ID. And the current Block ID is incremented by 1 after processing the localroot of a block. Note that the localroot of a block will keep the Block ID of the first block in which it is assigned a Block ID. In the example in Figure 8, A has Block ID=0. B, C, D, E, and F have Block ID=1. G has Block ID=2. H has Block ID=3. J and K have Block ID=4.

```
                                                       Leaf
                                               Hop     flag
               [F]--[E]        |--[K]     +-----------+---+
                |    |         |   |      |     A     |   |
                |    |         |   |      +-----------+---+
               [A]  [D]--[G]--[H]  |      |     B     |   |
                |    |         |   |      +-----------+---+
                |    |         |   |      |     C     |   |
               [B]--[C]        |--[J]     +-----------+---+
                                          |     D     |   |
                    (a) topology          +-----------+---+
                                          |     E     |   |
                                          +-----------+---+
                                          |     F     |   |
                                          +-----------+---+
                                          |     A     | X |
                                          +-----------+---+
               [F]<-[E]        |--[K]     |     D     |   |
                |    ^         |   ^      +-----------+---+
                V    |         V   |      |     G     |   |
               [A]  [D]->[G]->[H]  |      +-----------+---+
                |   [D]<-[G]<-[H]  |      |     D     | X |
                |    ^         |   |      +-----------+---+
                V    |         |   |      |     G     |   |
               [B]->[C]        |->[J]     +-----------+---+
                                          |     H     |   |
                     (b) GADAG            +-----------+---+
                                          |     G     | X |
                                          +-----------+---+
                                          |     H     |   |
                                          +-----------+---+
                                          |     J     |   |
                                          +-----------+---+
                                          |     K     |   |
                                          +-----------+---+
                                          |     H     | X |
                                          +-----------+---+

                                       (c) GADAG descriptor
```

Figure 8: A GADAG with cut-links and its description; GADAG root = Node A

8. Summary

This document specifies IS-IS sub-TLVs for the control of explicit trees in Layer 2 networks. These sub-TLVs can be also used for the distribution of a centrally computed GADAG or MRTs if MFT-FRR is used.

9. IANA Considerations

Five new code points are required within MT-Capability [RFC6329] for the five new sub-TLVs:

- Topology sub-TLV
- Hop sub-TLV
- Bandwidth Constraint sub-TLV
- Bandwidth Assignment sub-TLV
- Timestamp sub-TLV

10. Security Considerations

This document adds no additional security risks to IS-IS, nor does it provide any additional security for IS-IS when used in a configured environment or a single-operator domain such as a data center. IS-IS PCR is not for zero configuration environments.

However, if IS-IS PCR is used to record bandwidth assignments in a network with multiple PCEs, then race conditions can appear and the precedence can be resolved by Importance parameter of the Bandwidth Assignment sub-TLV and the Time parameter of the Timestamp sub-TLV, especially if the different PCEs are administered by different entities.

11. Acknowledgements

The authors would like to thank Don Fedyk and Eric Gray for their comments and suggestions.

12. References

12.1. Normative References

[I-D.ietf-isis-te-metric-extensions]
Previdi, S., Giacalone, S., Ward, D., Drake, J., Atlas, A., Filsfils, C., and W. Wu, "IS-IS Traffic Engineering (TE) Metric Extensions", draft-ietf-isis-te-metric-extensions-04 (work in progress), October 2014.

[I-D.ietf-rtgwg-mrt-frr-algorithm]
Envedi, G., Csaszar, A., Atlas, A., Bowers, C., and A. Gopalan, "Algorithms for computing Maximally Redundant Trees for IP/LDP Fast- Reroute", draft-ietf-rtgwg-mrt-frr-algorithm-02 (work in progress), January 2015.

[IEEE8021Qca]
IEEE 802.1, "IEEE 802.1Qca Bridges and Bridged Networks - Amendment: Path Control and Reservation - Draft 1.4", (work in progress), February 12, 2015, <http://www.ieee802.org/1/pages/802.1ca.html>.

[IEEE8021aq]
IEEE 802.1, "IEEE 802.1aq: IEEE Standard for Local and metropolitan area networks - Media Access Control (MAC) Bridges and Virtual Bridged Local Area Networks - Amendment 20: Shortest Path Bridging", 2012, <http://standards.ieee.org/getieee802/download/802.1aq-2012.pdf>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC5303] Katz, D., Saluja, R., and D. Eastlake, "Three-Way Handshake for IS-IS Point-to-Point Adjacencies", RFC 5303, October 2008.

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

[RFC5307] Kompella, K. and Y. Rekhter, "IS-IS Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)", RFC 5307, October 2008.

[RFC6329] Fedyk, D., Ashwood-Smith, P., Allan, D., Bragg, A., and P. Unbehagen, "IS-IS Extensions Supporting IEEE 802.1aq Shortest Path Bridging", RFC 6329, April 2012.

12.2. Informative References

[I-D.bowers-rtgwg-mrt-applicability-to-8021qca]
Bowers, C. and J. Farkas, "Applicability of Maximally Redundant Trees to IEEE 802.1Qca Path Control and Reservation", draft-bowers-rtgwg-mrt-applicability-to-8021qca-00 (work in progress), February 2015.

[I-D.ietf-rtgwg-mrt-frr-architecture]
Atlas, A., Kebler, R., Bowers, C., Envedi, G., Csaszar, A., Tantsura, J., and R. White, "An Architecture for IP/LDP Fast-Reroute Using Maximally Redundant Trees", draft-ietf-rtgwg-mrt-frr-architecture-05 (work in progress), January 2015.

[IEEE1588]
IEEE 1588, "IEEE 1588: IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems", 2008, <http://standards.ieee.org/findstds/standard/1588-2008.html>.

[IEEE8021Q]
IEEE 802.1, "IEEE 802.1Q-2014: IEEE Standard for Local and metropolitan area networks - Bridges and Bridged Networks", 2014, <http://standards.ieee.org/getieee802/download/802.1aq-2012.pdf>.

[RFC4655] Farrel, A., Vasseur, J., and J. Ash, "A Path Computation Element (PCE)-Based Architecture", RFC 4655, August 2006.

Authors' Addresses

Janos Farkas (editor)
Ericsson
Konyves Kalman krt. 11/B
Budapest 1097
Hungary

Email: janos.farkas@ericsson.com

Nigel Bragg
Ciena
43-51 Worship Street
London EC2A 2DX
UK

Email: nbragg@ciena.com

Paul Unbehagen Jr
Avaya
1300 W. 120th Avenue
Westminster CO 80234
USA

Email: unbehagen@avaya.com

Glenn Parsons
Ericsson
349 Terry Fox Drive
Ottawa ON, K2K 2V6
Canada

Email: glenn.parsons@ericsson.com

Peter Ashwood-Smith
Huawei Technologies
303 Terry Fox Drive, Suite 400
Ottawa ON, K2K 3J1
Canada

Email: Peter.AshwoodSmith@huawei.com

Chris Bowers
Juniper Networks
1194 N. Mathilda Ave.
Sunnyvale, CA 94089
US

Email: cbowers@juniper.net

Networking Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 3, 2015

L. Ginsberg
Cisco Systems
B. Decraene
Orange
C. Filsfils
Cisco Systems
S. Litkowski
Orange Business Service
S. Previdi
Cisco Systems
March 2, 2015

IS-IS Prefix Attributes for Extended IP and IPv6 Reachability
draft-ginsberg-isis-prefix-attributes-01.txt

Abstract

This document introduces new sub-TLVs to support advertisement of prefix attribute flags and the source router id of the router which originated a prefix advertisement.

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

There are existing use cases in which knowing additional attributes of a prefix is useful. For example, it is useful to know whether an advertised prefix is directly connected to the advertising router or not. In the case of [SR] knowing whether a prefix is directly connected or not determines what action should be taken as regards processing of labels associated with an incoming packet. Current formats of the Extended Reachability TLVs for both IP and IPv6 are fixed and do not allow the introduction of additional flags without backwards compatibility issues. Therefore a new sub-TLV is introduced which allows for the advertisement of attribute flags associated with prefix advertisements.

It is also useful to know the source of a prefix advertisement when the advertisement has been leaked to another level. Therefore a new sub-TLV is introduced to advertise the router-id of the originator of a prefix advertisement.

2. New sub-TLVs for Extended Reachability TLVs

The following new sub-TLVs are introduced:

- IPv4/IPv6 Extended Reachability Attributes

- IPv4 Source Router ID

- IPv6 Source Router ID

All sub-TLVs are applicable to TLVs 135, 235, 236, and/or 237.

2.1. IPv4/IPv6 Extended Reachability Attribute Flags

This sub-TLV supports the advertisement of additional flags associated with a given prefix advertisement. The behavior of each flag when a prefix advertisement is leaked from one level to another (upwards or downwards) is explicitly defined below.

All flags are applicable to TLVs 135, 235, 236, 237 unless otherwise stated.

Prefix Attribute Flags
Type: 4 (suggested - to be assigned by IANA)
Length: Number of octets to follow
Value

```
     (Length * 8) bits.

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

Bits are defined/sent starting with Bit #0 defined below. Additional bit definitions which may be defined in the future SHOULD be assigned in ascending bit order so as to minimize the number of bits which will need to be transmitted.

Undefined bits SHOULD be transmitted as 0 and MUST be ignored on receipt.

Bits which are NOT transmitted MUST be treated as if they are set to 0 on receipt.

X-Flag: External Prefix Flag (Bit 0)
Set if the prefix has been redistributed from another protocol. This includes the case where multiple virtual routers are supported and the source of the redistributed prefix is another IS-IS instance.
The flag is preserved when leaked between levels.
In TLVs 236 and 237 this flag SHOULD always be sent as 0 and MUST be ignored on receipt. This is because there is an existing X flag defined in the fixed format of these TLVs as specified in [RFC5308] and [RFC5120].

R-Flag: Re-advertisement Flag (Bit 1)
Set when the prefix has been leaked from one level to another (upwards or downwards).

N-flag: Node Flag (Bit 2)
Set when the prefix identifies the advertising router i.e., the prefix is a host prefix advertising a globally reachable address typically associated with a loopback address.
The advertising router MAY choose to NOT set this flag even when the above conditions are met.
If the flag is set and the prefix length is NOT a host prefix (/32 for IPV4, /128 for IPv6) then the flag MUST be ignored.
The flag is preserved when leaked between levels.

2.2. IPv4/IPv6 Source Router ID

When a reachability advertisement is leaked from one level to another, the source of the original advertisement is unknown. In cases where the advertisement is an identifier for the advertising router (e.g., N-flag set in the Extended Reachability Attribute sub-TLV as described in the previous section) it may be useful for other routers to know the source of the advertisement. The sub-TLVs defined below provide this information.

IPv4 Source Router ID
Type: 11 (suggested - to be assigned by IANA)
Length: 4
Value: IPv4 Router ID of the source of the advertisement

Inclusion of this TLV is optional and MAY occur in TLVs 135, 235, 236, or 237.

If present the sub-TLV MUST be included when the prefix advertisement is leaked to another level.

IPv6 Source Router ID
Type: 12 (suggested - to be assigned by IANA)
Length: 16
Value: IPv6 Router ID of the source of the advertisement

Inclusion of this TLV is optional and MAY occur in TLVs 135, 235, 236, or 237.

If present the sub-TLV MUST be included when the prefix advertisement is leaked to another level.

3. IANA Considerations

This document adds the following new sub-TLVs to the registry of sub-TLVs for TLVs 135, 235, 236, 237.

Value: 4 (suggested - to be assigned by IANA)

Name: Prefix Attribute Flags

Value: 11 (suggested - to be assigned by IANA)

Name: IPv4 Source Router ID

Value: 12 (suggested - to be assigned by IANA)

Name: IPv6 Source Router ID

This document also introduces a new registry for bit values in the Prefix Attribute Flags sub-TLV. Registration policy is Expert Review as defined in [RFC5226]. Defined values are:

Bit #	Name
0	External Prefix Flag
1	Re-advertisement Flag
2	Node Flag

4. Security Considerations

None.

5. Acknowledgements

TBD

6. References

6.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC5120] Przygienda, T., Shen, N., and N. Sheth, "M-ISIS: Multi Topology (MT) Routing in Intermediate System to Intermediate Systems (IS-ISs)", RFC 5120, February 2008.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[RFC5308] Hopps, C., "Routing IPv6 with IS-IS", RFC 5308, October 2008.

6.2. Informational References

[SR] "IS-IS Extensions for Segment Routing, draft-ietf-isis-segment-routing-extensions-03(work in progress)", October 2014.

Authors' Addresses

Les Ginsberg
Cisco Systems
510 McCarthy Blvd.
Milpitas, CA 95035
USA

Email: ginsberg@cisco.com

Bruno Decraene
Orange
38 rue du General Leclerc
MIssy Moulineaux cedex 9 92794
France

Email: bruno.decraene@orange.com

Clarence Filsfils
Cisco Systems

Email: cfilsfils@cisco.com

Stephane Litkowski
Orange Business Service

Email: stephane.litkowski@orange.com

Stefano Previdi
Cisco Systems
Via Del Serafico 200
Rome 0144
Italy

Email: sprevidi@cisco.com

Networking Working Group L. Ginsberg
Internet-Draft Cisco Systems
Updates: 5308 (if approved) S. Litkowski
Intended status: Standards Track Orange Business Service
Expires: April 18, 2016 S. Previdi
Cisco Systems
October 16, 2015

IS-IS Route Preference for Extended IP and IPv6 Reachability
draft-ietf-isis-route-preference-02.txt

Abstract

Existing specifications as regards route preference are not explicit when applied to IPv4/IPv6 Extended Reachability Type/Length/Value (TLVs). There are also inconsistencies in the definition of how the up/down bit applies to route preference when the prefix advertisement appears in Level 2 Link State Protocol Data Units (LSPs). This document addresses these issues.

This document, if approved, updates RFC 5308.

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

[RFC5302] defines the route preferences rules as they apply to TLVs 128 and 130. [RFC5305] introduced the IP Extended Reachability TLV 135 but did not explicitly adapt the route preference rules defined in [RFC5302] for the new TLV. [RFC5308] defines the IPv6 Reachability TLV 236 and does include an explicit statement as regards route preference - but the statement introduces use of the up/down bit in advertisements which appear in Level 2 LSPs which is inconsistent with statements made in [RFC5302] and [RFC5305]. This document defines explicit route preference rules for TLV 135, revises the route preferences rules for TLV 236, and clarifies the usage of the up/down bit when it appears in TLVs in Level 2 LSPs. This document is viewed as a clarification (NOT correction) of [RFC5302] and [RFC5305] and a correction of the route preference rules defined in [RFC5308] to be consistent with the rules for IPv4. It also makes explicit that the same rules apply for the Multi-Topology(MT) equivalent TLVs 235 and 237.

2. Use of the up/down Bit in Level 2 LSPs

The up/down bit was introduced in support of leaking prefixes downwards in the IS-IS level hierarchy. Routes which are leaked downwards have the bit set to 1. Such prefixes MUST NOT be leaked upwards in the hierarchy. So long as we confine ourselves to a single IS-IS instance and the current number of supported levels (two) it is impossible to have a prefix advertised in a Level 2 LSP and have the up/down bit set to 1. However, because [RFC5302] anticipated a future extension to IS-IS which might support additional levels it allowed for the possibility that the up/down bit might be set in a Level-2 LSP and in support of easier migration in the event such an extension was introduced Section 3.3 stated:

"...it is RECOMMENDED that implementations ignore the up/down bit in L2 LSPs, and accept the prefixes in L2 LSPs regardless of whether the up/down bit is set."

[RFC5305] addressed an additional case wherein an implementation included support for multiple virtual routers running IS-IS in different areas. In such a case it is possible to redistribute prefixes between two IS-IS instances in the same manner that prefixes are redistributed from other protocols into IS-IS. This introduced the possibility that a prefix could be redistributed from Level 1 to Level 1 (as well as between Level 2 and Level 2) and in the event the redistributed route was leaked from Level 1 to Level 2 two different routers in different areas would be advertising the same prefix into the Level 2 sub-domain. To prevent this [RFC5305] specified in Section 4.1:

"If a prefix is advertised from one area to another at the same level, then the up/down bit SHALL be set to 1."

However, the statement in [RFC5302] that the up/down bit is ignored in Level 2 LSPs is not altered by [RFC5305].

The conclusion then is that there is no "L2 inter-area route" - and indeed no such route type is defined by [RFC5302]. However, [RFC5308] ignored this fact and introduced such a route type in Section 5 when it specified a preference for " Level 2 down prefix". This is an error which this document corrects. As changing the use of the up/down bit in TLVs 236 and 237 may introduce interoperability issues implementors may wish to support transition mechanisms from the [RFC5308] behavior to the behavior specified in this document.

3. Types of Routes in IS-IS Supported by Extended Reachability TLVs

[RFC5302] is the authoritative reference for the types of routes supported by TLVs 128 and 130. However, a number of attributes supported by those TLVs are NOT supported by TLVs 135, 235, 236, 237. Distinction between internal/external metrics is not supported. In the case of IPv4 TLVs (135 and 235) the distinction between internal and external route types is not supported. However the Prefix Attribute Flags sub-TLV defined in [PFXATTR] reintroduces the distinction between internal and external route types. The definitions below include references to the relevant attribute bits from [PFXATTR].

3.1. Types of Routes Supported by TLVs 135 and 235

This section defines the types of route supported for IPv4 when using TLV 135 [RFC5305] and/or TLV 235 [RFC5120]. The text follows as closely as possible the original text from [RFC5302].

L1 intra-area routes: These are advertised in L1 LSPs, in TLV 135 or TLV 235. The up/down bit is set to 0. These IP prefixes are directly connected to the advertising router. If the Prefix Attribute Flags sub-TLV is included both the X-Flag and the R-Flag are set to 0.

L1 external routes: These are advertised in L1 LSPs, in TLV 135 or TLV 235. The up/down bit is set to 0. These IP prefixes are learned from other protocols and are usually not directly connected to the advertising router. If the Prefix Attribute Flags sub-TLV is included the X-Flag is set to 1 and the R-Flag is set to 0.

L2 intra-area routes: These are advertised in L2 LSPs, in TLV 135 or TLV 235. The up/down bit is set to 0. These IP prefixes are directly connected to the advertising router. If the Prefix Attribute Flags sub-TLV is included both the X-Flag and the R-Flag are set to 0.

L1->L2 inter-area routes: These are advertised in L2 LSPs, in TLV 135 or TLV 235. The up/down bit is set to 0. These IP prefixes are learned via L1 routing and were derived during the L1 Shortest Path First (SPF) computation from prefixes advertised in L1 LSPs in TLV 135 or TLV 235. If the Prefix Attribute Flags sub-TLV is included the R-Flag is set to 1.

L2->L2 inter-area routes: These are advertised in L2 LSPs, in TLV 135 or TLV 235. The up/down bit is set to 1 but is ignored and treated as if it were set to 0. These IP prefixes are learned from another IS-IS instance usually operating in another area. If the Prefix Attribute Flags sub-TLV is included the X-Flag is set to 1 and the R-Flag is set to 0.

L2 external routes: These are advertised in L2 LSPs, in TLV 135 or TLV 235. The up/down bit is set to 0. These IP prefixes are learned from other protocols and are usually not directly connected to the advertising router. If the Prefix Attribute Flags sub-TLV is included the X-Flag is set to 1 and the R-Flag is set to 0.

L2->L1 inter-area routes: These are advertised in L1 LSPs, in TLV 135 or TLV 235. The up/down bit is set to 1. These IP prefixes are learned via L2 routing and were derived during the L2 SPF computation from prefixes advertised in TLV 135 or TLV 235. If the Prefix Attribute Flags sub-TLV is included the R-Flag is set to 1.

L1->L1 inter-area routes: These are advertised in L1 LSPs, in TLV 135 or TLV 235. The up/down bit is set to 1. These IP prefixes are learned from another IS-IS instance usually operating in another area. If the Prefix Attribute Flags sub-TLV is included the X-Flag is set to 1 and the R-Flag is set to 0.

3.2. Types of Routes Supported by TLVs 236 and 237

This section defines the types of route supported for IPv6 when using TLV 236 [RFC5308] and/or TLV 237 [RFC5120].

L1 intra-area routes: These are advertised in L1 LSPs, in TLV 236 or TLV 237. The up/down bit is set to 0. The external bit is set to 0. These IPv6 prefixes are directly connected to the advertising router. If the Prefix Attribute Flags sub-TLV is included the R-Flag is set to 0.

L1 external routes: These are advertised in L1 LSPs, in TLV 236 or TLV 237. The up/down bit is set to 0. The external bit is set to 1. These IPv6 prefixes are learned from other protocols and are usually not directly connected to the advertising router. If the Prefix Attribute Flags sub-TLV is included the R-Flag is set to 0.

L2 intra-area routes: These are advertised in L2 LSPs, in TLV 236 or TLV 237. The up/down bit is set to 0. The external bit is set to 0. These IPv6 prefixes are directly connected to the advertising router. If the Prefix Attribute Flags sub-TLV is included the R-Flag is set to 0.

L1->L2 inter-area routes: These are advertised in L2 LSPs, in TLV 236 or TLV 237. The up/down bit is set to 0. The external bit is set to 0. These IPv6 prefixes are learned via L1 routing and were derived during the L1 Shortest Path First (SPF) computation from prefixes advertised in L1 LSPs in TLV 236 or TLV 237. If the Prefix Attribute Flags sub-TLV is included the R-Flag is set to 1.

L2 external routes: These are advertised in L2 LSPs, in TLV 236 or TLV 237. The up/down bit is set to 0. the external bit is set to 1. These IPv6 prefixes are learned from other protocols and are usually not directly connected to the advertising router. If the Prefix Attribute Flags sub-TLV is included the R-Flag is set to 0.

L1->L2 external routes: These are advertised in L2 LSPs, in TLV 236 or TLV 237. The up/down bit is set to 0. The external bit is set to 1. These IPv6 prefixes are learned via L1 routing and were derived during the L1 Shortest Path First (SPF) computation from L1 external routes advertised in L1 LSPs in TLV 236 or TLV 237. If the Prefix Attribute Flags sub-TLV is included the R-Flag is set to 1.

L2->L2 inter-area routes. These are advertised in L2 LSPs, in TLV 236 or TLV 237. The up/down bit is set to 1 but is ignored and treated as if it were set to 0. The external bit is set to 1. These IP prefixes are learned from another IS-IS instance usually operating in another area. If the Prefix Attribute Flags sub-TLV is included the R-Flag is set to 0.

L2->L1 inter-area routes: These are advertised in L1 LSPs, in TLV 236 or TLV 237. The up/down bit is set to 1. The external bit is set to 0. These IPv6 prefixes are learned via L2 routing and were derived during the L2 SPF computation from prefixes advertised in TLV 236 or TLV 237. If the Prefix Attribute Flags sub-TLV is included the R-Flag is set to 1.

L2->L1 external routes: These are advertised in L1 LSPs, in TLV 236 or TLV 237. The up/down bit is set to 1. The external bit is set to

1. These IPv6 prefixes are learned via L2 routing and were derived during the L2 SPF computation from prefixes advertised in TLV 236 or TLV 237. If the Prefix Attribute Flags sub-TLV is included the R-Flag is set to 1.

L1->L1 inter-area routes. These are advertised in L1 LSPs, in TLV 236 or TLV 237. The up/down bit is set to 1. The external bit is set to 1. These IP prefixes are learned from another IS-IS instance usually operating in another area. If the Prefix Attribute Flags sub-TLV is included the R-Flag is set to 0.

3.3. Order of Preference for all types of routes supported by TLVs 135 and 235

This document defines the following route preferences for IPv4 routes advertised in TLVs 135 or 235. Note that all types of routes listed for a given preference are treated equally.

1. L1 intra-area routes; L1 external routes

2. L2 intra-area routes; L2 external routes; L1->L2 inter-area routes; L2-L2 inter-area routes

3. L2->L1 inter-area routes; L1->L1 inter-area routes

3.4. Order of Preference for all types of routes supported by TLVs 236 and 237

This document defines the following route preferences for IPv6 routes advertised in TLVs 236 or 237. Note that all types of routes listed for a given preference are treated equally.

1. L1 intra-area routes; L1 external routes

2. L2 intra-area routes; L2 external routes; L1->L2 inter-area routes; L1-L2 external routes;L2-L2 inter-area routes

3. L2->L1 inter-area routes; L2->L1 external routes;L1->L1 inter-area routes

4. IANA Considerations

No IANA actions required.

5. Security Considerations

None.

6. Acknowledgements

The authors wish to thank Ahmed Bashandy for his insightful review.

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, <http://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, <http://www.rfc-editor.org/info/rfc5120>.

[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, <http://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, <http://www.rfc-editor.org/info/rfc5305>.

[RFC5308] Hopps, C., "Routing IPv6 with IS-IS", RFC 5308, DOI 10.17487/RFC5308, October 2008, <http://www.rfc-editor.org/info/rfc5308>.

7.2. Informational References

[PFXATTR] "IS-IS Prefix Attributes, draft-ietf-isis-prefix-attributes-01(work in progress)", June 2015.

Appendix A. Example Interoperability Issue

This documents a real world interoperability issue which occurs because implementations from different vendors have interpreted the use of the up/down bit in Level 2 LSPs inconsistently.

```
             L2       L2       L2     L2|L2      L2
    10/8 - R0 ----- R1 ----- R2 ----- R3 ----- R4 ---- 10/8
                                       |
```

Figure 1

Considering Figure 1, both R0 and R4 are advertising the prefix 10/8. Two ISIS Level 2 instances are running on R3 to separate the network into two areas. R3 is performing route-leaking and advertises prefixes from R4 to the other Level 2 process. The network is using extended metrics (TLV135 defined in [RFC5305]). R0 is advertising 10/8 with metric 2000 and R3 advertises 10/8 with metric 100. All links have a metric of 1. When advertising 10/8 in its Level 2 LSP, R3 sets the down bit as specified in [RFC5305].

R1, R2 and R3 are from three different vendors (R1->Vendor1, R2->Vendor2, R3->Vendor3). During interoperability testing, routing loops are observed in this scenario.

- R2 has two possible paths to reach 10/8, Level 2 route with metric 2002, up/down bit is 0 (from R0) and Level 2 route with metric 101, up/down bit is 1 (from R3). R2 selects R1 as nexthop to 10/8 because it prefers the route which does NOT have up/down bit set.

- R3 has two possible paths to reach 10/8, Level 2 route with metric 2003, up/down bit is 0 (from R0) and Level 2 route with metric 101, up/down bit is 0 (from R4). R3 selects R4 as nexthop due to lowest metric.

- R1 has two possible paths to reach 10/8, Level 2 route with metric 2001, up/down bit is 0 (from R0) and Level 2 metric 102, up/down bit is 1 (from R3). R1 selects R2 as nexthop due to lowest metric.

When R1 or R2 try to send traffic to 10/8, packets are looping due to inconsistent routing decision between R1 and R2.

Authors' Addresses

Les Ginsberg
Cisco Systems
510 McCarthy Blvd.
Milpitas, CA 95035
USA

Email: ginsberg@cisco.com

Stephane Litkowski
Orange Business Service

Email: stephane.litkowski@orange.com

Stefano Previdi
Cisco Systems
Via Del Serafico 200
Rome 0144
Italy

Email: sprevidi@cisco.com

Networking Working Group L. Ginsberg
Internet-Draft N. Akiya
Intended status: Standards Track Cisco Systems
Expires: September 3, 2015 M. Chen
Huawei
March 2, 2015

Advertising S-BFD Discriminators in IS-IS
draft-ietf-isis-sbfd-discriminator-02.txt

Abstract

This document defines a means of advertising one or more S-BFD Discriminators using the IS-IS Router Capability TLV.

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

[S-BFD] defines a simplified mechanism to use Bidirectional Forwarding Detection (BFD)[RFC5880]. This mechanism depends on network nodes knowing the BFD discriminators which each node in the network has reserved for this purpose. Use of the Intermediate System to Intermediate System (IS-IS)[IS-IS] protocol is one possible means of advertising these discriminators.

2. Encoding Format

The IS-IS Router CAPABILITY TLV as defined in [RFC4971] will be used to advertise S-BFD discriminators. A new sub-TLV is defined as described below. S-BFD Discriminators sub-TLVs are formatted as specified in [RFC5305].

```
                                           No. of octets
                 +-----------------------------+
                 | Type (to be assigned by     |     1
                 | IANA - suggested value 20)  |
                 +-----------------------------+
                 | Length (multiple of 4)      |     1
                 +-----------------------------+
                 | Discriminator Value(s)      |     4/Discriminator
                 :                             :
                 +-----------------------------+
```

Inclusion of the S-BFD Discriminators sub-TLV in a Router Capability TLV is optional. Multiple S-BFD Discriminators sub-TLVs MAY be advertised by an IS. When multiple S-BFD discriminators are advertised how a given discriminator is mapped to a specific use case is out of scope for this document.

S-BFD discriminator advertisements MAY be flooded within an area or throughout the domain using the procedures specified in [RFC4971]. The appropriate flooding scope depends on the intended use of S-BFD. If S-BFD use will be exclusively within a Level-1 area then area scope is appropriate. If S-BFD usage will span different L1 areas then domain scope is appropriate.

3. IANA Considerations

This document requires the definition of a new sub-TLV in the Sub-TLVs for TLV 242 registry. The value written below is a suggested value subject to assignment by IANA.

Value	Description
20	S-BFD Discriminators

4. Security Considerations

Security concerns for IS-IS are addressed in [IS-IS], [RFC5304], and [RFC5310]. Introduction of the S-BFD Discriminators sub-TLV introduces no new security risks for IS-IS.

Advertisement of the S-BFD discriminators does make it possible for attackers to initiate S-BFD sessions using the advertised information. The vulnerabilities this poses and how to mitigate them are discussed in the Security Considerations section of [S-BFD].

5. Acknowledgements

The authors wish to thank Sam Aldrin, Manav Bhatia, and Carlos Pignataro for input essential to defining the needed functionality.

6. Normative References

[IS-IS] "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, March 1997.

[RFC4971] Vasseur, JP., Shen, N., and R. Aggarwal, "Intermediate System to Intermediate System (IS-IS) Extensions for Advertising Router Information", RFC 4971, July 2007.

[RFC5304] Li, T. and R. Atkinson, "IS-IS Cryptographic Authentication", RFC 5304, October 2008.

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

[RFC5310] Bhatia, M., Manral, V., Li, T., Atkinson, R., White, R., and M. Fanto, "IS-IS Generic Cryptographic Authentication", RFC 5310, February 2009.

[RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, June 2010.

[S-BFD] "Seamless Bidirectional Forwarding Detection (S-BFD), draft-ietf-bfd-seamless-base-04(work in progress)", January 2015.

Authors' Addresses

Les Ginsberg
Cisco Systems
510 McCarthy Blvd.
Milpitas, CA 95035
USA

Email: ginsberg@cisco.com

Nobo Akiya
Cisco Systems

Email: nobo@cisco.com

Mach(Guoyi) Chen
Huawei

Email: mach.chen@huawei.com

IS-IS for IP Internets
Internet-Draft
Intended status: Standards Track
Expires: November 20, 2019

S. Previdi, Ed.
Huawei
L. Ginsberg, Ed.
C. Filsfils
Cisco Systems, Inc.
A. Bashandy
Arrcus
H. Gredler
RtBrick Inc.
B. Decraene
Orange
May 19, 2019

IS-IS Extensions for Segment Routing
draft-ietf-isis-segment-routing-extensions-25

Abstract

Segment Routing (SR) allows for 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 necessary IS-IS extensions that need to be introduced for Segment Routing operating on an 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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

Status of This Memo

time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on November 20, 2019.

Copyright Notice

Table of Contents

1. Introduction

Segment Routing (SR) allows for 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 of the 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 signaling (other than the regular IGP). For example, when used in MPLS networks, SR paths do not require any LDP or RSVP-TE signaling. Still, SR can interoperate in the presence of LSPs established with RSVP or LDP.

There are additional segment types, e.g., Binding SID defined in [RFC8402]. This document also defines an advertisement for one type of Binding SID: the Mirror Context segment.

This draft describes the necessary IS-IS extensions that need to be introduced for Segment Routing operating on an MPLS data-plane.

The Segment Routing architecture is described in [RFC8402].

Segment Routing use cases are described in [RFC7855].

2. Segment Routing Identifiers

The Segment Routing architecture [RFC8402] defines different types of Segment Identifiers (SID). This document defines the IS-IS encodings for the IGP-Prefix Segment, the IGP-Adjacency Segment, the IGP-LAN-Adjacency Segment and the Binding Segment.

2.1. Prefix Segment Identifier (Prefix-SID Sub-TLV)

A new IS-IS sub-TLV is defined: the Prefix Segment Identifier sub-TLV (Prefix-SID sub-TLV).

The Prefix-SID sub-TLV carries the Segment Routing IGP-Prefix-SID as defined in [RFC8402]. The 'Prefix SID' MUST be unique within a given IGP domain (when the L-flag is not set).

A Prefix-SID sub-TLV is associated to a prefix advertised by a node and MAY be present in any of the following TLVs:

TLV-135 (Extended IPv4 reachability) defined in [RFC5305].

TLV-235 (Multitopology IPv4 Reachability) defined in [RFC5120].

TLV-236 (IPv6 IP Reachability) defined in [RFC5308].

TLV-237 (Multitopology IPv6 IP Reachability) defined in [RFC5120].

Binding-TLV and Multi-Topology Binding-TLV defined in Section 2.4 and Section 2.5 respectively.

The Prefix-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   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     SID/Index/Label (variable)                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

Type: 3

Length: 5 or 6 depending on the size of the SID (described below)

Flags: 1 octet field of following flags:

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|R|N|P|E|V|L|   |
+-+-+-+-+-+-+-+-+
```

where:

R-Flag: Re-advertisement flag. If set, then the prefix to which this Prefix-SID is attached, has been propagated by the router either from another level (i.e., from level-1 to level-2 or the opposite) or from redistribution (e.g.: from another protocol).

N-Flag: Node-SID flag. If set, then the Prefix-SID refers to the router identified by the prefix. Typically, the N-Flag is set on Prefix-SIDs attached to a router loopback address. The N-Flag is set when the Prefix-SID is a Node-SID as described in [RFC8402].

P-Flag: no-PHP flag. If set, then the penultimate hop MUST NOT pop the Prefix-SID before delivering the packet to the node that advertised the Prefix-SID.

E-Flag: Explicit-Null Flag. If set, any upstream neighbor of the Prefix-SID originator MUST replace the Prefix-SID with a Prefix-SID having an Explicit-NULL value (0 for IPv4 and 2 for IPv6) before forwarding the packet.

V-Flag: Value flag. If set, then the Prefix-SID carries a value (instead of an index). By default the flag is UNSET.

L-Flag: Local Flag. If set, then the value/index carried by the Prefix-SID has local significance. By default the flag is UNSET.

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

Algorithm: the router may use various algorithms when calculating reachability to other nodes or to prefixes attached to these nodes. Algorithm identifiers are defined in Section 3.2. Examples of these algorithms are metric based Shortest Path First (SPF), various sorts of Constrained SPF, etc. The algorithm field of the Prefix-SID contains the identifier of the algorithm the router uses to compute the reachability of the prefix to which the Prefix-SID is associated.

At origination, the Prefix-SID algorithm field MUST be set to 0 or to any value advertised in the SR-Algorithm sub-TLV (Section 3.2).

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 (Section 3.2) MUST ignore the Prefix-SID sub-TLV.

SID/Index/Label as defined in Section 2.1.1.1.

When the Prefix SID is an index (the V-flag is not set) the value is used to determine the actual label value inside the set of all advertised label ranges of a given router. This allows a receiving router to construct forwarding state to a particular destination router.

In many use-cases a 'stable transport' address is overloaded as an identifier of a given node. Because Prefixes may be re-advertised into other levels there may be some ambiguity (e.g. Originating router vs. L1L2 router) for which node a particular IP prefix serves as identifier. The Prefix-SID sub-TLV contains the necessary flags to disambiguate Prefix to node mappings. Furthermore if a given node has several 'stable transport' addresses there are flags to differentiate those among other Prefixes advertised from a given node.

2.1.1. Flags

2.1.1.1. V and L Flags

The V-flag indicates whether the SID/Index/Label field is a value or an index.

The L-Flag indicates whether the value/index in the SID/Index/Label field has local or global significance.

The following settings for V and L flags are valid:

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 using the encodings defined in Section 3.1.

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.

2.1.1.2. R and N Flags

The R-Flag MUST be set for prefixes that are not local to the router and either:

advertised because of propagation (Level-1 into Level-2);

advertised because of leaking (Level-2 into Level-1);

advertised because of redistribution (e.g.: from another protocol).

In the case where a Level-1-2 router has local interface addresses configured in one level, it may also propagate these addresses into the other level. In such case, the Level-1-2 router MUST NOT set the R bit.

The N-Flag is used in order to define a Node-SID. A router MAY set the N-Flag only if all of the following conditions are met:

The prefix to which the Prefix-SID is attached is local to the router (i.e., the prefix is configured on one of the local interfaces, e.g., a 'stable transport' loopback).

The prefix to which the Prefix-SID is attached has a Prefix length of either /32 (IPv4) or /128 (IPv6).

The router MUST ignore the N-Flag on a received Prefix-SID if the prefix has a Prefix length different than /32 (IPv4) or /128 (IPv6).

The Prefix Attributes Flags sub-TLV [RFC7794] also defines the N and R flags and with the same semantics of the equivalent flags defined in this document. Whenever the Prefix Attributes Flags sub-TLV is present for a given prefix the values of the N and R flags advertised in that sub-TLV MUST be used and the values in a corresponding Prefix SID sub-TLV (if present) MUST be ignored.

2.1.1.3. E and P Flags

The following behavior is associated with the settings of the E and P flags:

o If the P-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 which improves performance of the ultimate hop. MPLS EXP bits of the Prefix-SID are not preserved to the ultimate hop (the Prefix-SID being removed). If the P-flag is unset the received E-flag is ignored.

o If the P-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, e.g., the originator of the

Prefix-SID must stitch the incoming packet into a continuing MPLS LSP to the final destination. This could occur at an inter-area border router (prefix propagation from one area to another) or at an inter-domain border router (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 PrefixSID with a Prefix-SID having an Explicit-NULL value. 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 EXP bits.

When propagating (either from Level-1 to Level-2 or vice versa) a reachability advertisement originated by another IS-IS speaker, the router MUST set the P-flag and MUST clear the E-flag of the related Prefix-SIDs.

2.1.2. Prefix-SID Propagation

The Prefix-SID sub-TLV MUST be included when the associated Prefix Reachability TLV is propagated across level boundaries.

The level-1-2 router that propagates the Prefix-SID sub-TLV between levels maintains the content (flags and SID) except as noted in Section 2.1.1.2 and Section 2.1.1.3.

2.2. Adjacency Segment Identifier

A new IS-IS sub-TLV is defined: the Adjacency Segment Identifier sub-TLV (Adj-SID sub-TLV).

The Adj-SID sub-TLV is an optional sub-TLV carrying the Segment Routing IGP-Adjacency-SID as defined in [RFC8402] with flags and fields that may be used, in future extensions of Segment Routing, for carrying other types of SIDs.

IS-IS adjacencies are advertised using one of the IS-Neighbor TLVs below:

TLV-22 (Extended IS reachability)[RFC5305]

TLV-222 (Multitopology IS)[RFC5120]

TLV-23 (IS Neighbor Attribute)[RFC5311]

TLV-223 (Multitopology IS Neighbor Attribute)[RFC5311]

TLV-141 (inter-AS reachability information)[RFC5316]

Multiple Adj-SID sub-TLVs MAY be associated with a single IS-neighbor.

2.2.1. Adjacency Segment Identifier (Adj-SID) Sub-TLV

The following format is defined for the Adj-SID sub-TLV:

```
 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    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    SID/Label/Index (variable)                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

Type: 31

Length: 5 or 6 depending on size of the SID

Flags: 1 octet field of following flags:

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|F|B|V|L|S|P|   |
+-+-+-+-+-+-+-+-+
```

where:

F-Flag: Address-Family flag. If unset, then the Adj-SID is used when forwarding IPv4 encapsulated traffic to the neighbor. If set then the Adj-SID is used when forwarding IPv6 encapsulated traffic to the neighbor.

B-Flag: Backup flag. If set, the Adj-SID is eligible for protection (e.g.: using IPFRR or MPLS-FRR) as described in [RFC8402].

V-Flag: Value flag. If set, then the Adj-SID carries a value. By default the flag is SET.

L-Flag: Local Flag. If set, then the value/index carried by the Adj-SID has local significance. By default the flag is SET.

S-Flag. Set flag. When set, the S-Flag indicates that the Adj-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 Adj-SID is persistently allocated, i.e., the Adj-SID value remains consistent across router restart and/or interface flap.

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

Weight: 1 octet. The value represents the weight of the Adj-SID for the purpose of load balancing. The use of the weight is defined in [RFC8402].

SID/Index/Label as defined in Section 2.1.1.1.

An SR capable router MAY allocate an Adj-SID for each of its adjacencies

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.

Examples of use of the Adj-SID sub-TLV are described in [RFC8402].

The F-flag is used in order for the router to advertise the outgoing encapsulation of the adjacency the Adj-SID is attached to.

2.2.2. Adjacency Segment Identifiers in LANs

In LAN subnetworks, the Designated Intermediate System (DIS) is elected and originates the Pseudonode-LSP (PN-LSP) including all neighbors of the DIS.

When Segment Routing is used, each router in the LAN MAY advertise the Adj-SID of each of its neighbors. Since, on LANs, each router only advertises one adjacency to the DIS (and doesn't advertise any other adjacency), each router advertises the set of Adj-SIDs (for each of its neighbors) inside a newly defined sub-TLV part of the TLV advertising the adjacency to the DIS (e.g.: TLV-22).

The following new sub-TLV is defined: LAN-Adj-SID containing the set of Adj-SIDs the router assigned to each of its LAN neighbors.

The format of the LAN-Adj-SID sub-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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Type        |     Length    |     Flags     |    Weight     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Neighbor System-ID (ID length octets)        |
+                               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   SID/Label/Index (variable)                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

Type: 32

Length: variable.

Flags: 1 octet field of following flags:

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|F|B|V|L|S|P|   |
+-+-+-+-+-+-+-+-+
```

where F, B, V, L, S and P flags are defined in Section 2.2.1. Other bits: MUST be zero when originated and ignored when received.

Weight: 1 octet. The value represents the weight of the Adj-SID for the purpose of load balancing. The use of the weight is defined in [RFC8402].

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

SID/Index/Label as defined in Section 2.1.1.1.

Multiple LAN-Adj-SID sub-TLVs MAY be encoded.

Note that this sub-TLV MUST NOT appear in TLV 141.

In case one TLV-22/23/222/223 (reporting the adjacency to the DIS) can't contain the whole set of LAN-Adj-SID sub-TLVs, multiple advertisements of the adjacency to the DIS MUST be used and all advertisements MUST have the same metric.

Each router within the level, by receiving the DIS PN LSP as well as the non-PN LSP of each router in the LAN, is capable of reconstructing the LAN topology as well as the set of Adj-SIDs each router uses for each of its neighbors.

2.3. SID/Label Sub-TLV

The SID/Label sub-TLV may be present in the following TLVs/sub-TLVs defined in this document:

SR-Capabilities Sub-TLV (Section 3.1)

SR Local Block Sub-TLV (Section 3.3)

SID/Label Binding TLV (Section 2.4)

Multi-Topology SID/Label Binding TLV (Section 2.5)

Note that the code point used in all of the above cases is the SID/Label Sub-TLV code point specified in the new "sub-TLVs for TLV 149 and 150" registry created by this document.

The SID/Label sub-TLV contains a SID or a MPLS Label. The SID/Label 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    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          SID/Label (variable)                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

Type: 1

Length: 3 or 4

SID/Label: if length is set to 3 then the 20 rightmost bits represent a MPLS label. If length is set to 4 then the value is a 32 bit index

2.4. SID/Label Binding TLV

The SID/Label Binding TLV MAY be originated by any router in an IS-IS domain. There are multiple uses of the SID/Label Binding TLV.

The SID/Label Binding TLV may be used to advertise prefixes to SID/Label mappings. This functionality is called the Segment Routing Mapping Server (SRMS). The behavior of the SRMS is defined in [I-D.ietf-spring-segment-routing-ldp-interop].

The SID/Label Binding TLV may also be used to advertise a Mirror SID to advertise the ability to process traffic originally destined to another IGP node. This behavior is defined in [RFC8402].

The SID/Label Binding 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     |   RESERVED    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             Range             | Prefix Length |    Prefix     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
//               Prefix (continued, variable)                  //
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Sub-TLVs (variable)                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: SID/Label Binding TLV format

- Type: 149
- Length: variable.
- 1 octet of flags
- 1 octet of RESERVED (SHOULD be transmitted as 0 and MUST be ignored on receipt)
- 2 octets of Range
- 1 octet of Prefix Length
- 0-16 octets of Prefix

o sub-TLVs, where each sub-TLV consists of a sequence of:

  * 1 octet of sub-TLV type

  * 1 octet of length of the value field of the sub-TLV

  * 0-243 octets of value

2.4.1. Flags

Flags: 1 octet field of following flags:

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

where:

F-Flag: Address Family flag. If unset, then the Prefix carries an IPv4 Prefix. If set then the Prefix carries an IPv6 Prefix.

M-Flag: Mirror Context flag. Set if the advertised SID corresponds to a mirrored context. The use of a mirrored context is described in [RFC8402].

S-Flag: If set, the SID/Label Binding TLV SHOULD be flooded across the entire routing domain. If the S flag is not set, the SID/ Label Binding TLV MUST NOT be leaked between levels. This bit MUST NOT be altered during the TLV leaking.

D-Flag: when the SID/Label Binding TLV is leaked from level-2 to level-1, the D-Flag MUST be set. Otherwise, this flag MUST be clear. SID/Label Binding TLVs with the D-Flag set MUST NOT be leaked from level-1 to level-2. This is to prevent TLV looping across levels.

A-Flag: Attached flag. The originator of the SID/Label Binding TLV MAY set the A bit in order to signal that the prefixes and SIDs advertised in the SID/Label Binding TLV are directly connected to their originators. The mechanisms through which the originator of the SID/Label Binding TLV can figure out if a prefix is attached or not are outside the scope of this document (e.g.: through explicit configuration). If the Binding TLV is leaked to other areas/levels the A-flag MUST be cleared.

An implementation may decide not to honor the S-flag in order not to leak Binding TLV’s between levels (for policy reasons).

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

2.4.2. Range

The 'Range' field provides the ability to specify a range of addresses and their associated Prefix SIDs. This advertisement supports the SRMS functionality. It is essentially a compression scheme to distribute a continuous Prefix and their continuous, corresponding SID/Label Block. If a single SID is advertised then the range field MUST be set to one. For range advertisements > 1, the range field MUST be set to the number of addresses that need to be mapped into a Prefix-SID. In either case the prefix is the first address to which a SID is to be assigned.

2.4.3. Prefix Length, Prefix

The 'Prefix' represents the Forwarding equivalence class at the tail-end of the advertised path. The 'Prefix' does not need to correspond to a routable prefix of the originating node.

The 'Prefix Length' field contains the length of the prefix in bits. Only the most significant octets of the Prefix are encoded (i.e., 1 octet for prefix length 1 up to 8, 2 octets for prefix length 9 to 16, 3 octets for prefix length 17 up to 24 and 4 octets for prefix length 25 up to 32, ..., 16 octets for prefix length 113 up to 128).

2.4.4. Mapping Server Prefix-SID

The Prefix-SID sub-TLV is defined in Section 2.1 and contains the SID/index/label value associated with the prefix and range. The Prefix-SID Sub-TLV MUST be present in the SID/Label Binding TLV when the M-flag is clear. The Prefix-SID Sub-TLV MUST NOT be present when the M-flag is set.

2.4.4.1. Prefix-SID Flags

The Prefix-SID flags are defined in Section 2.1. The Mapping Server MAY advertise a mapping with the N flag set when the prefix being mapped is known in the link-state topology with a mask length of 32 (IPv4) or 128 (IPv6) and when the prefix represents a node. The mechanisms through which the operator defines that a prefix represents a node are outside the scope of this document (typically it will be through configuration).

The other flags defined in Section 2.1 are not used by the Mapping Server and MUST be ignored at reception.

2.4.4.2. PHP Behavior when using Mapping Server Advertisements

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, if additional information is available PHP behavior may safely be done. The required information consists of:

- o A prefix reachability advertisement for the prefix has been received which includes the Prefix Attribute Flags sub-TLV [RFC7794].

- o X and R flags are both set to 0 in the Prefix Attribute Flags sub-TLV.

In the absence of an Prefix Attribute Flags sub-TLV [RFC7794] the A flag in the binding TLV indicates that the originator of a prefix reachability advertisement is directly connected to the prefix and thus PHP MUST be done by the neighbors of the router originating the prefix reachability advertisement. Note that A-flag is only valid in the original area in which the Binding TLV is advertised.

2.4.4.3. Prefix-SID Algorithm

The algorithm field contains the identifier of the algorithm associated with the SIDs for the prefix(es) in the range. Use of the algorithm field is described in Section 2.1.

2.4.5. SID/Label Sub-TLV

The SID/Label sub-TLV (Type: 1) contains the SID/Label value as defined in Section 2.3. It MUST be present in the SID/Label Binding TLV when the M-flag is set in the Flags field of the parent TLV.

2.4.6. Example Encodings

Example 1: if the following IPv4 router addresses (loopback addresses) need to be mapped into the corresponding Prefix SID indexes.

```
Router-A: 192.0.2.1/32, Prefix-SID: Index 1
Router-B: 192.0.2.2/32, Prefix-SID: Index 2
Router-C: 192.0.2.3/32, Prefix-SID: Index 3
Router-D: 192.0.2.4/32, Prefix-SID: Index 4
```

```
     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    |0|0|0|0|0|     |    RESERVED   |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |           Range = 4           |      32       |      192      |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |       0       |       2       |       1       |Prefix-SID Type|
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    | sub-TLV Length|     Flags     |   Algorithm   |               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                             1 |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Example-2: If the following IPv4 prefixes need to be mapped into the corresponding Prefix-SID indexes:

10.1.1/24, Prefix-SID: Index 51
10.1.2/24, Prefix-SID: Index 52
10.1.3/24, Prefix-SID: Index 53
10.1.4/24, Prefix-SID: Index 54
10.1.5/24, Prefix-SID: Index 55
10.1.6/24, Prefix-SID: Index 56
10.1.7/24, Prefix-SID: Index 57

```
     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    |0|0|0|0|0|     |    RESERVED   |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |           Range = 7           |      24       |      10       |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |       1       |       1       |Prefix-SID Type| sub-TLV Length|
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |     Flags     | Algorithm     |                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                            51 |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Example-3: If the following IPv6 prefixes need to be mapped into the corresponding Prefix-SID indexes:

2001:db8:1/48, Prefix-SID: Index 151
2001:db8:2/48, Prefix-SID: Index 152
2001:db8:3/48, Prefix-SID: Index 153
2001:db8:4/48, Prefix-SID: Index 154

```
 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     |1|0|0|0|0|     |   RESERVED    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Range = 4            |      48       |     0x20      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     0x01      |     0x0d      |     0xb8      |     0x00      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   0x01        |Prefix-SID Type| sub-TLV Length|  Flags        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Algorithm     |                 0                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     151       |
+-+-+-+-+-+-+-+-+
```

It is not expected that a network operator will be able to keep fully continuous Prefix / SID/Index mappings. In order to support noncontinuous mapping ranges an implementation MAY generate several instances of Binding TLVs.

For example if a router wants to advertise the following ranges:

Range 16: { 192.0.2.1-15, Index 1-15 }

Range 6: { 192.0.2.22-27, Index 22-27 }

Range 41: { 192.0.2.44-84, Index 80-120 }

A router would need to advertise three instances of the Binding TLV.

2.5. Multi-Topology SID/Label Binding TLV

The Multi-Topology SID/Label Binding TLV allows the support of M-ISIS as defined in [RFC5120]. The Multi-Topology SID/Label Binding TLV has the same format as the SID/Label Binding TLV defined in Section 2.4 with the difference consisting of a Multitopology Identifier (MTID) as defined here 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     |    Length     |              MTID             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Flags     |    RESERVED   |             Range             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Prefix Length |            Prefix (variable)                 //
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Sub-TLVs (variable)                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: Multi-Topology SID/Label Binding TLV format

where:

Type: 150

Length: variable

MTID is the multitopology identifier defined as:

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

RESVD: reserved bits. MUST be reset on transmission and ignored on receive.

MTID: a 12-bit field containing the non-zero ID of the topology being announced. The TLV MUST be ignored if the ID is zero. This is to ensure the consistent view of the standard unicast topology.

The other fields and Sub-TLVs are defined in Section 2.4.

3. Router Capabilities

This section defines sub-TLVs which are inserted into the IS-IS Router Capability TLV-242 that is defined in [RFC7981].

3.1. SR-Capabilities Sub-TLV

Segment Routing requires each router to advertise its SR data-plane capability and the range of MPLS label values it uses for Segment Routing in the case where global SIDs are allocated (i.e., global indexes). Data-plane capabilities and label ranges are advertised using the newly defined SR-Capabilities sub-TLV.

The Router Capability TLV specifies flags that control its advertisement. The SR Capabilities sub-TLV MUST be propagated throughout the level and MUST NOT be advertised across level boundaries. Therefore Router Capability TLV distribution flags are set accordingly, i.e., the S flag in the Router Capability TLV [RFC7981] MUST be unset.

The SR Capabilities 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    |    Flags      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Range                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
//                SID/Label Sub-TLV (variable)                 //
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type: 2

Length: variable.

Flags: 1 octet of flags. The following are defined:

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

where:

I-Flag: MPLS IPv4 flag. If set, then the router is capable of processing SR MPLS encapsulated IPv4 packets on all interfaces.

V-Flag: MPLS IPv6 flag. If set, then the router is capable of processing SR MPLS encapsulated IPv6 packets on all interfaces.

One or more SRGB Descriptor entries, each of which have the following format:

Range: 3 octets.

SID/Label sub-TLV (as defined in Section 2.3).

SID/Label sub-TLV contains the first value of the SRGB while the range contains the number of SRGB elements. The range value MUST be higher than 0.

The SR-Capabilities sub-TLV MAY be advertised in an LSP of any number but a router MUST NOT advertise more than one SR-Capabilities sub-TLV. A router receiving multiple SR-Capabilities sub-TLVs from the same originator SHOULD select the first advertisement in the lowest numbered LSP.

When multiple SRGB Descriptors are advertised the entries define an ordered set of ranges on which a SID index is to be applied. For this reason changing the order in which the descriptors are advertised will have a disruptive effect on forwarding.

When a router adds a new SRGB Descriptor to an existing SR-Capabilities sub-TLV the new Descriptor SHOULD add the newly configured block at the end of the sub-TLV and SHOULD NOT change the order of previously advertised blocks. Changing the order of the advertised descriptors will create label churn in the FIB and blackhole / misdirect some traffic during the IGP convergence. In particular, if a range which is not the last is extended it’s preferable to add a new range rather than extending the previously advertised range.

The originating router MUST ensure the order is unchanged after a graceful restart (using checkpointing, non-volatile storage or any other mechanism).

The originating router MUST NOT advertise overlapping ranges.

When a router receives multiple overlapping ranges, it MUST conform to the procedures defined in [I-D.ietf-spring-segment-routing-mpls].

Here follows an example of advertisement of multiple ranges:

The originating router advertises following ranges:

```
SR-Cap: range: 100, SID value: 100
SR-Cap: range: 100, SID value: 1000
SR-Cap: range: 100, SID value: 500
```

The receiving routers concatenate the ranges in the received order and build the SRGB as follows:

```
SRGB = [100, 199]
       [1000, 1099]
       [500, 599]
```

The indexes span multiple ranges:

```
index=0   means label 100
...
index 99  means label 199
index 100 means label 1000
index 199 means label 1099
...
index 200 means label 500
...
```

3.2. SR-Algorithm Sub-TLV

The router may use various algorithms when calculating reachability to other nodes or to prefixes attached to these nodes. Examples of these algorithms are metric based Shortest Path First (SPF), various sorts of Constrained SPF, etc. The SR-Algorithm sub-TLV allows the router to advertise the algorithms that the router is currently using. Algorithm values are defined in the "IGP Algorithm Type" registry defined in [I-D.ietf-ospf-segment-routing-extensions]. The following values have been defined:

0: Shortest Path First (SPF) algorithm based on link metric. This is the well-known shortest path algorithm as computed by the IS-IS Decision process. Consistent with the deployed practice for link-state protocols, algorithm 0 permits any node to overwrite the SPF path with a different path based on local policy.

1: Strict Shortest Path First (SPF) algorithm based on link metric. The algorithm is identical to algorithm 0 but algorithm 1 requires that all nodes along the path will honor the SPF routing decision. Local policy MUST NOT alter the forwarding decision computed by algorithm 1 at the node claiming to support algorithm 1.

The Router Capability TLV specifies flags that control its advertisement. The SR-Algorithm MUST be propagated throughout the level and MUST NOT be advertised across level boundaries. Therefore Router Capability TLV distribution flags are set accordingly, i.e., the S flag MUST be unset.

The SR-Algorithm sub-TLV is optional. It MUST NOT be advertsied more than once at a given level. A router receiving multiple SR-Algorithm sub-TLVs from the same originator SHOULD select the first advertisement in the lowest numbered LSP.

When the originating router does not advertise the SR-Algorithm sub-TLV, this implies that the only algorithm supported by routers supporting the extensions defined in this document is Algorithm 0.

When the originating router does advertise the SR-Algorithm sub-TLV, then algorithm 0 MUST be present while non-zero algorithms MAY be present.

The SR-Algorithm 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 1   | Algorithm 2   | Algorithm ... |  Algorithm n  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

Type: 19

Length: variable.

Algorithm: 1 octet of algorithm

3.3. SR Local Block Sub-TLV

The SR Local Block (SRLB) Sub-TLV contains the range of labels the node has reserved for local SIDs. Local SIDs are used, e.g., for Adjacency-SIDs, and may also be allocated by components other than the IS-IS protocol. As an example, an application or a controller may instruct the router to allocate a specific local SID. Therefore, in order for such applications or controllers to know what are the local SIDs available in the router, it is required that the router advertises its SRLB.

The SRLB Sub-TLV is used for this purpose and 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      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Range                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
//                SID/Label Sub-TLV (variable)                 //
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type: 22

Length: variable.

Flags: 1 octet of flags. None are defined at this stage.

One or more SRLB Descriptor entries, each of which have the following format:

Range: 3 octets.

SID/Label sub-TLV (as defined in Section 2.3).

SID/Label sub-TLV contains the first value of the SRLB while the range contains the number of SRLB elements. The range value MUST be higher than 0.

The SRLB sub-TLV MAY be advertised in an LSP of any number but a router MUST NOT advertise more than one SRLB sub-TLV. A router receiving multiple SRLB sub-TLVs, from the same originator, SHOULD select the first advertisement in the lowest numbered LSP.

The originating router MUST NOT advertise overlapping ranges.

When a router receives multiple overlapping ranges, it MUST conform to the procedures defined in [I-D.ietf-spring-segment-routing-mpls].

It is important to note that each time a SID from the SRLB is allocated, it should also be reported to all components (e.g.: controller or applications) in order for these components to have an up-to-date view of the current SRLB allocation and in order to avoid collision between allocation instructions.

Within the context of IS-IS, the reporting of local SIDs is done through IS-IS Sub-TLVs such as the Adjacency-SID. However, the reporting of allocated local SIDs may also be done through other means and protocols which are outside the scope of this document.

A router advertising the SRLB sub-TLV may also have other label ranges, outside the SRLB, for its local allocation purposes which are NOT advertised in the SRLB. For example, it is possible that an Adjacency-SID is allocated using a local label not part of the SRLB.

3.4. SRMS Preference Sub-TLV

The Segment Routing Mapping Server (SRMS) Preference sub-TLV is used in order to associate a preference with SRMS advertisements from a particular source.

The SRMS Preference 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    | Preference    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type: 24

Length: 1.

Preference: 1 octet. Unsigned 8 bit SRMS preference.

The SRMS Preference sub-TLV MAY be advertised in an LSP of any number but a router MUST NOT advertise more than one SRMS Preference sub-TLV. A router receiving multiple SRMS Preference sub-TLVs, from the same originator, SHOULD select the first advertisement in the lowest numbered LSP.

The use of the SRMS Preference during the SID selection process is described in [I-D.ietf-spring-segment-routing-ldp-interop]

4. IANA Considerations

This document requests allocation for the following TLVs and Sub-TLVs.

4.1. Sub TLVs for Type 22,23,25,141,222, and 223

This document makes the following registrations in the "sub-TLVs for TLV 22, 23, 25, 141, 222 and 223" registry.

Type	Description	22	23	25	141	222	223
31	Adjacency Segment Identifier	y	y	n	y	y	y
32	LAN Adjacency Segment Identifier	y	y	n	y	y	y

4.2. Sub TLVs for Type 135,235,236 and 237

This document makes the following registrations in the "sub-TLVs for TLV 135,235,236 and 237" registry.

Type	Description	135	235	236	237
3	Prefix Segment Identifier	y	y	y	y

4.3. Sub TLVs for Type 242

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

Type	Description
2	Segment Routing Capability
19	Segment Routing Algorithm
22	Segment Routing Local Block (SRLB)
24	Segment Routing Mapping Server Preference (SRMS Preference)

4.4. New TLV Codepoint and Sub-TLV registry

This document registers the following TLV:

Value	Name	IIH	LSP	SNP	Purge
149	Segment Identifier/Label Binding	n	y	n	n
150	Multi-Topology Segment Identifier /Label Binding	n	y	n	n

This document creates the following sub-TLV Registry:

Name: sub-TLVs for TLVs 149 and 150
Registration Procedure: Expert Review

Type	Description
0	Reserved
1	SID/Label
2	Unassigned
3	Prefix SID
4-255	Unassigned

5. Security Considerations

With the use of the extensions defined in this document, IS-IS carries information which will be used to program the MPLS data plane [RFC3031]. In general, the same types of attacks that can be carried out on the IP/IPv6 control plane can be carried out on the MPLS control plane resulting in traffic being misrouted in the respective data planes. However, the latter may be more difficult to detect and isolate.

Existing security extensions as described in [RFC5304] and [RFC5310] apply to these segment routing extensions.

6. Acknowledgements

We would like to thank Dave Ward, Dan Frost, Stewart Bryant, Pierre Francois and Jesper Skrivers for their contribution to the content of this document.

7. Contributors

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

Stephane Litkowski
Orange
FR

Email: stephane.litkowski@orange.com

Jeff Tantsura
Apstra, Inc.

Email: jefftant@gmail.com

Peter Psenak

Cisco Systems Inc.
US

Email: ppsenak@cisco.com

Martin Horneffer
Deutsche Telekom
DE

Email: Martin.Horneffer@telekom.de

Wim Henderickx
Nokia
BE

Email: wim.henderickx@nokia.com

Edward Crabbe
Oracle
US

Email: edward.crabbe@oracle.com

Rob Shakir
Google
UK

Email: robjs@google.com

Igor Milojevic
Individual
RS

Email: milojevicigor@gmail.com

Saku Ytti
TDC
FI

Email: saku@ytti.fi

Steven Luong
Cisco Systems Inc.

US

Email: sluong@cisco.com

8. References

8.1. Normative References

[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-22 (work in progress), May 2019.

[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>.

[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>.

[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>.

[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>.

[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>.

[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>.

8.2. Informative References

[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>.

[RFC5308] Hopps, C., "Routing IPv6 with IS-IS", RFC 5308, DOI 10.17487/RFC5308, October 2008, <https://www.rfc-editor.org/info/rfc5308>.

[RFC5311] McPherson, D., Ed., Ginsberg, L., Previdi, S., and M. Shand, "Simplified Extension of Link State PDU (LSP) Space for IS-IS", RFC 5311, DOI 10.17487/RFC5311, February 2009, <https://www.rfc-editor.org/info/rfc5311>.

[RFC5316] Chen, M., Zhang, R., and X. Duan, "ISIS Extensions in Support of Inter-Autonomous System (AS) MPLS and GMPLS Traffic Engineering", RFC 5316, DOI 10.17487/RFC5316, December 2008, <https://www.rfc-editor.org/info/rfc5316>.

[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

Stefano Previdi (editor)
Huawei
IT

Email: stefano@previdi.net

Les Ginsberg (editor)
Cisco Systems, Inc.
USA

Email: ginsberg@cisco.com

Clarence Filsfils
Cisco Systems, Inc.
Brussels
BE

Email: cfilsfil@cisco.com

Ahmed Bashandy
Arrcus

Email: abashandy.ietf@gmail.com

Hannes Gredler
RtBrick Inc.

Email: hannes@rtbrick.com

Bruno Decraene
Orange
FR

Email: bruno.decraene@orange.com

IS-IS Working Group S. Litkowski
Internet-Draft Orange
Intended status: Standards Track D. Yeung
Expires: March 12, 2020 Arrcus, Inc
A. Lindem
Cisco Systems
J. Zhang
Juniper Networks
L. Lhotka
CZ.NIC
September 9, 2019

YANG Data Model for IS-IS Protocol
draft-ietf-isis-yang-isis-cfg-37

Abstract

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

Requirements Language

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

Status of This Memo

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 operational states.

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

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

2. Design of the Data Model

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

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

```
module: ietf-isis
  augment /rt:routing/rt:ribs/rt:rib/rt:routes/rt:route:
    +--ro metric?       uint32
    +--ro tag*          uint64
    +--ro route-type?   enumeration
  augment /if:interfaces/if:interface:
    +--rw clns-mtu?   uint16 {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 address-family-stats* [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 in:

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

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

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

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

2.2. Multitopology Parameters

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

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

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

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

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

2.3. Per-Level Parameters

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

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

- o a level-1 container: corresponds to level-1 specific parameters.

- o a level-2 container: corresponds to level-2 specific parameters.

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

Example:

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

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

Some parameters like "overload bit" and "route preference" are not modeled to support a per level configuration. If an implementation supports per level configuration for such parameter, this implementation 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 previous section. An interface-specific parameter MUST be preferred over an IS-IS global parameter.

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

```
       +--rw interfaces
          +--rw interface* [name]
             +--rw name                        if:interface-ref
             +--rw 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 a global authentication is configured, an implementation SHOULD authenticate PSNPs (Partial Sequence Number Packets), CSNPs (Complete Sequence Number Packets) and LSPs (Link State Packets) with the authentication parameters supplied. The authentication of HELLO PDUs (Protocol Data Units) can be activated on a per interface basis.

2.6. IGP/LDP synchronization

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

2.7. ISO parameters

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

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

The clns-mtu can be defined under the interface.

2.8. IP FRR

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

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

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

The "candidate-enable" allows to mark an interface to be used as a backup.

2.9. Operational States

Operational states are provided in the module in various places:

- system-counters: provides statistical informations about the global system.

- interface : provides configuration state informations for each interface.

- adjacencies: provides state informations about current IS-IS adjacencies.

- spf-log: provides informations about SPF events on the node. This SHOULD be implemented as a wrapping buffer.

- lsp-log: provides informations about LSP events on the node (reception of an LSP or modification of local LSP). This SHOULD be implemented as a wrapping buffer and an implementation MAY decide to log refresh LSPs or not.

- local-rib: provides the IS-IS internal routing table view.

- database: provides details on the current LSDB.

- hostnames: provides informations about system-id to hostname mappings [RFC5301].

- fast-reroute: provides informations about IP FRR.

3. RPC Operations

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

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

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

4. Notifications

The "ietf-isis" module introduces some notifications :

database-overload: raised when overload condition is changed.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

5. Interaction with Other YANG Modules

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

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

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

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

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

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

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

6. IS-IS YANG Module

The following RFCs, drafts and external standards are not referenced in the document text but are referenced in the ietf-isis.yang module: [ISO-10589], [RFC1195], [RFC4090],[RFC5029], [RFC5130], [RFC5302], [RFC5305], [RFC5306], [RFC5307], [RFC5308], [RFC5880], [RFC5881], [RFC6119], [RFC6232], [RFC7794], [RFC7981], [RFC8570], [RFC7917], [RFC8405].

```
<CODE BEGINS> file "ietf-isis@2019-09-09.yang"
module ietf-isis {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-isis";

  prefix isis;

  import ietf-routing {
    prefix "rt";
```

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

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

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

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

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

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

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

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

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

  }

  organization
```

```
       "IETF LSR Working Group";

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

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

           Derek Yeung
             <mailto:derek@arrcus.com>
           Acee Lindem
             <mailto:acee@cisco.com>
           Jeffrey Zhang
             <mailto:zzhang@juniper.net>
           Ladislav Lhotka
             <mailto:llhotka@nic.cz>

       ";

     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 2019-09-09 {
       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 on the path.";
     }
     identity frr-protection-available-other-type {
       base frr-protection-available-type;
       description "The level of protection is unknown.";
     }


     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 referred metric.";
     }
     identity lsp-attached-delay-metric-flag {
         base lsp-flag;
         description "Set when originator is attached to
              another area using the referred metric.";
     }
     identity lsp-attached-expense-metric-flag {
         base lsp-flag;
         description "Set when originator is attached to
              another area using the referred metric.";
```

```
     }
     identity lsp-attached-default-metric-flag {
         base lsp-flag;
         description "Set when originator is attached to
              another area using the referred 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
         "LDP IGP synchronization.";
       reference "RFC5443 - LDP IGP Synchronization.";
     }
     feature fast-reroute {
       description
         "Support for IP Fast Reroute (IP-FRR).";
     }
     feature nsr {
       description
         "Non-Stop-Routing (NSR) support. 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
         "Calculate IS-IS interface metric according to
          reference bandwidth.";
     }

     feature te-rid {
       description
         "Traffic-Engineering Router-ID.";
       reference "RFC5305 - IS-IS Extensions for Traffic Engineering
                  RFC6119 - IPv6 Traffic Engineering in IS-IS";
     }
     feature max-ecmp {
       description
         "Setting maximum number of ECMP paths.";
     }
     feature multi-topology {
       description
         "Support for Multiple-Topology Routing (MTR).";
       reference "RFC5120 - M-IS-IS: Multi Topology Routing in IS-IS";
     }
     feature nlpid-control {
       description
         "This feature controls the advertisement
          of support NLPID within IS-IS configuration.";
     }
```

```
     feature graceful-restart {
       description
         "IS-IS Graceful restart support.";
       reference "RFC5306 - Restart Signaling in IS-IS";
     }

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

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

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

     /* Type definitions */

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

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

     typedef interface-type {
       type enumeration {
         enum broadcast {
           description
             "Broadcast interface type.";
         }
         enum point-to-point {
           description
             "Point-to-point interface type.";
         }
       }
```

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

      }

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

      }

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

```
    description
      "This type defines states of an adjacency";
  }

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

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

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

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

  typedef area-address {
    type string {
      pattern '[0-9A-Fa-f]{2}(\.[0-9A-Fa-f]{4}){0,6}';
    }
    description
      "This type defines the area address format.";
  }

  typedef snpa {
    type string {
      length "0 .. 20";
```

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

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

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

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

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

     /* Grouping for notifications */

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

     grouping notification-interface-hdr {
       description
         "Interface specific IS-IS notification data grouping";
       leaf interface-name {
         type if:interface-ref;
         description "IS-IS interface name";
       }
       leaf interface-level {
         type level;
         description "IS-IS level of the interface.";
       }
```

```
       leaf extended-circuit-id {
         type extended-circuit-id;
         description "Extended circuit-id of the interface.";
       }
     }


    /* Groupings for IP Fast Reroute */

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

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

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

          leaf address-family {
            type iana-rt-types:address-family;
            description
              "Address-family";
          }
          leaf prefix {
            type inet:ip-prefix;
            description
              "Protected prefix.";
```

```
           }
           leaf alternate {
             type inet:ip-address;
             description
               "Alternate next hop for the prefix.";
           }
           leaf alternate-type {
             type enumeration {
               enum equal-cost {
                 description
                   "ECMP alternate.";
               }
               enum lfa {
                 description
                   "LFA alternate.";
               }
               enum remote-lfa {
                 description
                   "Remote LFA alternate.";
               }
               enum tunnel {
                 description
                   "Tunnel based alternate
                    (like RSVP-TE or GRE).";
               }
               enum ti-lfa {
                 description
                   "TI-LFA alternate.";
               }
               enum mrt {
                 description
                   "MRT alternate.";
               }
               enum other {
                 description
                   "Unknown alternate type.";
               }
             }
             description
               "Type of alternate.";
           }
           leaf best {
             type boolean;
             description
               "Is set when the alternate is the preferred one,
                is unset otherwise.";
           }
           leaf non-best-reason {
```

```
            type string {
              length "1..255";
            }
            description
              "Information field to describe why the alternate
               is not best. The length should be limited to 255
               unicode characters. The expected format is a single
               line text.";
          }
          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 address-family-stats {
           key "address-family prefix";

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

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

       list protection-statistics {
         key frr-protection-method;
         config false;
         leaf frr-protection-method {
           type 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 uint32;
             description "Total prefixes.";
           }
           leaf unprotected-routes {
             type uint32;
             description
               "Total prefixes that are not protected.";
           }
           leaf protected-routes {
```

```
            type uint32;
            description
              "Total prefixes that are protected.";
          }
          leaf linkprotected-routes {
            type uint32;
            description
              "Total prefixes that are link protected.";
          }
          leaf nodeprotected-routes {
            type uint32;
            description
              "Total prefixes that are node protected.";
          }
          description
            "Per AF protected prefix statistics.";
        }

        description "Global protection statistics.";
      }
    }

          /* Route table and local RIB groupings */

    grouping local-rib {
      description "Local-rib - RIB for Routes computed by the local
                   IS-IS routing instance.";
      container local-rib {
        config false;
        description "Local-rib.";
        list route {
          key "prefix";
          description "Routes";
          leaf prefix {
            type inet:ip-prefix;
            description "Destination prefix.";
          }
          container next-hops {
            description "Next hops for the route.";
            list next-hop {
              key "next-hop";
              description "List of next hops for the route";
              leaf outgoing-interface {
                type if:interface-ref;
                description
                  "Name of the outgoing interface.";
              }
              leaf next-hop {
```

```
                type inet:ip-address;
                description "Next hop address.";
              }
            }
          }
          leaf metric {
            type uint32;
            description "Metric for this route.";
          }
          leaf level {
            type level-number;
            description "Level number for this route.";
          }
          leaf route-tag {
            type uint32;
            description "Route tag for this route.";
          }
        }
      }
    }

    grouping route-content {
      description
        "IS-IS protocol-specific route properties grouping.";
      leaf metric {
        type uint32;
        description "IS-IS metric of a route.";
      }
      leaf-list tag {
        type uint64;
        description
          "List of tags associated with the route. The leaf
           describes both 32-bit and 64-bit tags.";
      }
      leaf route-type {
        type enumeration {
          enum l2-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 on the adjacency.
                On a p2p link this might be level 1 and 2,
                but on a LAN, the usage will be level 1
                between 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";
           }
         }
         default wide-only;
         description
           "Type of metric to be generated:
            - wide-only means only new metric style
               is generated,
            - old-only means that only old-style metric
              is generated,
            - both means that both are advertised.
            This leaf is only affecting IPv4 metrics.";
       }
       description
         "Grouping for global metric style config.";
     }

     grouping default-metric-global-cfg {
       leaf value {
         type wide-metric;
         default "10";
         description  "Value of the metric";
```

```
       }
       description
         "Global default metric config grouping.";
     }


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

     grouping overload-max-metric-global-cfg {
       leaf timeout {
         type rt-types:timer-value-seconds16;
         units "seconds";
         description
           "Timeout (in seconds) of the overload condition.";
       }
       description
         "Overload maximum metric configuration grouping";
     }

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

```
          description
            "Choice for implementation of route preference.";
        }
        description
          "Global route preference grouping";
      }

      grouping hello-authentication-cfg {
        choice authentication-type {
          case key-chain {
            if-feature key-chain;
            leaf key-chain {
              type key-chain:key-chain-ref;
              description "Reference to a key-chain.";
            }
          }
          case password {
            leaf key {
              type string;
              description "Authentication key specification - The
                           length of the key may be dependent on the
                           cryptographic algorithm.";
            }
            leaf crypto-algorithm {
              type identityref {
                base key-chain:crypto-algorithm;
              }
              description
                "Cryptographic algorithm associated with key.";
            }
          }
          description "Choice of authentication.";
        }
        description "Grouping for hello authentication.";
      }

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

        description "Interval between hello messages.";
      }
```

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

     grouping priority-cfg {
       leaf value {
         type uint8 {
           range "0 .. 127";
         }
         default 64;
         description
           "Priority of interface for DIS election.";
       }

       description "Interface DIS election priority grouping";
     }

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


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

```
       }

       container default-metric {
         uses default-metric-global-cfg;
         container level-1 {
           uses default-metric-global-cfg;
           description "level-1 specific configuration";
         }
         container level-2 {
           uses default-metric-global-cfg;
           description "level-2 specific configuration";
         }
         description "Default metric global configuration";
       }
       container auto-cost {
         if-feature auto-cost;
         description
           "Interface Auto-cost configuration state.";
         leaf enable {
           type boolean;
           description
             "Enable/Disable interface auto-cost.";
         }
         leaf reference-bandwidth {
           when "../enable = 'true'" {
             description "Only when auto cost is enabled";
           }
           type uint32 {
             range "1..4294967";
           }
           units Mbits;
           description
             "Configure reference bandwidth used to automatically
              determine interface cost (Mbits). The cost is the
              reference bandwidth divided by the interface speed
              with 1 being the minimum cost.";
         }
       }

       description "Grouping for global metric parameters.";
     }

     grouping high-availability-parameters {
       container graceful-restart {
         if-feature graceful-restart;
         leaf enable {
           type boolean;
           default false;
```

```
          description "Enable graceful restart.";
        }
        leaf restart-interval {
          type rt-types:timer-value-seconds16;
          units "seconds";
          description
            "Interval (in seconds) to attempt graceful restart prior
             to failure.";
        }
        leaf helper-enable {
          type boolean;
          default "true";
          description
            "Enable local IS-IS router as graceful restart helper.";
        }
        description "Graceful-Restart Configuration.";
      }
      container nsr {
        if-feature nsr;
        description "Non-Stop Routing (NSR) configuration.";
        leaf enable {
          type boolean;
          default false;
          description "Enable/Disable Non-Stop Routing (NSR).";
        }
      }
      description "Grouping for High Availability parameters.";
    }

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

        container level-1 {
          uses authentication-global-cfg;
          description "level-1 specific configuration";
        }
        container level-2 {
          uses authentication-global-cfg;
          description "level-2 specific configuration";
        }
        description "Authentication global configuration for
                     both LSPs and SNPs.";
      }
      description "Grouping for authentication parameters";
    }
    grouping address-family-parameters {
      container address-families {
```

```
         if-feature nlpid-control;
         list address-family-list {
           key address-family;
           leaf address-family {
             type iana-rt-types:address-family;
             description "Address-family";
           }
           leaf enable {
             type boolean;
             description "Activate the address family.";
           }
           description
             "List of address families and whether or not they
              are activated.";
         }
         description "Address Family configuration";
       }
       description "Grouping for address family parameters.";
     }

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

```
       }
       description "Grouping for MPLS global parameters.";
     }

     grouping lsp-parameters {
       leaf lsp-mtu {
         type uint16;
         units "bytes";
         default 1492;
         description
           "Maximum size of an LSP PDU in bytes.";
       }
       leaf lsp-lifetime {
         type uint16 {
           range "1..65535";
         }
         units "seconds";
         description
           "Lifetime of the router's LSPs in seconds.";
       }
       leaf lsp-refresh {
         if-feature lsp-refresh;
         type rt-types:timer-value-seconds16;
         units "seconds";
         description
           "Refresh interval of the router's LSPs in seconds.";
       }
       leaf poi-tlv {
         if-feature poi-tlv;
         type boolean;
         default false;
         description
           "Enable advertisement of IS-IS purge TLV.";
       }
       description "Grouping for LSP global parameters.";
     }
     grouping spf-parameters {
       container spf-control {
           leaf paths {
             if-feature max-ecmp;
             type uint16 {
               range "1..32";
             }
             description
               "Maximum number of Equal-Cost Multi-Path (ECMP) paths.";
           }
           container ietf-spf-delay {
             if-feature ietf-spf-delay;
```

```
             uses ietf-spf-delay;
             description "IETF SPF delay algorithm configuration.";
           }
           description
             "SPF calculation control.";
       }
       description "Grouping for SPF global parameters.";
     }
     grouping instance-config {
       description "IS-IS global configuration grouping";

       uses admin-control;

       leaf level-type {
         type level;
         default "level-all";
         description
           "Level of an IS-IS node - can be level-1,
            level-2 or level-all.";
       }

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

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

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

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

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

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

       container overload-max-metric {
         if-feature overload-max-metric;
         uses overload-max-metric-global-cfg;
         description
           "Router protocol instance overload maximum
            metric advertisement configuration.";
       }
     }

     grouping instance-state {
       description
         "IS-IS instance operational state.";
       uses spf-log;
       uses lsp-log;
       uses hostname-db;
       uses lsdb;
       uses local-rib;
       uses system-counters;
       uses instance-fast-reroute-state;
       leaf discontinuity-time {
         type yang:date-and-time;
         description
           "The time on 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 on 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;
          container level-1 {
            uses hello-interval-cfg;
            description "level-1 specific configuration";
          }
          container level-2 {
            uses hello-interval-cfg;
            description "level-2 specific configuration";
          }
          description "Interval between hello messages.";
        }
        container hello-multiplier {
          uses hello-multiplier-cfg;
          container level-1 {
            uses hello-multiplier-cfg;
            description "level-1 specific configuration";
          }
          container level-2 {
            uses hello-multiplier-cfg;
            description "level-2 specific configuration";
          }
          description "Hello multiplier configuration.";
        }
        container priority {
          must '../interface-type = "broadcast"' {
            error-message
              "Priority only applies to broadcast interfaces.";
            description "Check for broadcast interface.";
          }
          uses priority-cfg;
```

```
         container level-1 {
           uses priority-cfg;
           description "level-1 specific configuration";
         }
         container level-2 {
           uses priority-cfg;
           description "level-2 specific configuration";
         }
         description "Priority for DIS election.";
       }
       container metric {
         uses metric-cfg;
         container level-1 {
           uses metric-cfg;
           description "level-1 specific configuration";
         }
         container level-2 {
           uses metric-cfg;
           description "level-2 specific configuration";
         }
         description "Metric configuration.";
       }
       container bfd {
         if-feature bfd;
         description "BFD Client Configuration.";
         uses bfd-types:client-cfg-parms;

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

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

       }
       container address-families {
         if-feature nlpid-control;
         list address-family-list {
           key address-family;
           leaf address-family {
             type iana-rt-types:address-family;
             description  "Address-family";
           }
           description "List of AFs.";
         }
         description "Interface address-families";
       }
       container mpls {
         container ldp {
```

```
          leaf igp-sync {
            if-feature ldp-igp-sync;
            type boolean;
            default false;
            description "Enables IGP/LDP synchronization";
          }
          description "LDP protocol related configuration.";
        }
        description "MPLS configuration for IS-IS interfaces";
      }
      uses interface-fast-reroute-config;
    }

    grouping multi-topology-interface-config {
      description "IS-IS interface topology configuration.";
      container metric {
        uses metric-cfg;
        container level-1 {
          uses metric-cfg;
          description "level-1 specific configuration";
        }
        container level-2 {
          uses metric-cfg;
          description "level-2 specific configuration";
        }
        description "Metric IS-IS interface configuration.";
      }
    }
    grouping interface-state {
      description
        "IS-IS interface operational state.";
      uses adjacency-state;
      uses event-counters;
      uses packet-counters;
      leaf discontinuity-time {
        type yang:date-and-time;
        description
          "The time on the most recent occasion at which any one
           or more of this IS-IS interfaces'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 uint32;
```

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

```
             }
             description
               "This list includes the LSPs that triggered the
                SPF computation.";
           }
           description
             "List of computation events - implemented as a
              wrapping buffer.";
         }

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

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

           leaf id {
             type uint32;
             description
               "Event identifier -  purely internal value.";
           }
           leaf level {
             type level-number;
             description
               "IS-IS level number for LSP";
           }
           container lsp {
             leaf lsp {
               type lsp-id;
               description
                 "LSP ID of the LSP.";
             }
             leaf sequence {
               type uint32;
               description
                 "Sequence number of the LSP.";
             }
             description
               "LSP identification container - either the received
                LSP or the locally generated LSP.";
           }
```

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

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

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

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

    } description "Grouping for LSP log.";
  }



  /* Groupings for the LSDB description */

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

```
     }

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

     /* TLVs and sub-TLVs for prefixes */

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

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

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

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

     grouping prefix-ipv6-source-router-id {
```

```
        description
          "Grouping for the IPv6 source router ID of a prefix
           advertisement.";

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

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

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

      grouping prefix-ipv4-std {
        description
          "Grouping for attributes of an IPv4 standard prefix
           as defined in RFC1195.";
        leaf ip-prefix {
          type inet:ipv4-address;
          description "IPv4 prefix address";
        }
        leaf prefix-len {
          type uint8;
          description "IPv4 prefix length (in bits)";
        }
        leaf i-e {
          type boolean;
          description "Internal or External (I/E) Metric bit value.";
        }
        container default-metric {
          leaf metric {
            type std-metric;
            description "Default IS-IS metric for IPv4 prefix";
          }
          description "IS-IS default metric container.";
        }
        container delay-metric {
          leaf metric {
            type std-metric;
            description "IS-IS delay metric for IPv4 prefix";
          }
          leaf supported {
```

```
          type boolean;
          default "false";
          description
            "Indicates whether IS-IS delay metric is supported.";
        }
        description "IS-IS delay metric container.";
      }
      container expense-metric {
        leaf metric {
          type std-metric;
          description "IS-IS expense metric for IPv4 prefix";
        }
        leaf supported {
          type boolean;
          default "false";
          description
            "Indicates whether IS-IS delay metric is supported.";
        }
        description "IS-IS expense metric container.";
      }
      container error-metric {
        leaf metric {
          type std-metric;
          description
            "This leaf describes the IS-IS error metric value";
        }
        leaf supported {
          type boolean;
          default "false";
          description "IS-IS error metric for IPv4 prefix";
        }
        description "IS-IS error metric container.";
      }
    }

    grouping prefix-ipv4-extended {
      description
        "Grouping for attributes of an IPv4 extended prefix
         as defined in RFC5305.";
      leaf up-down {
        type boolean;
        description  "Value of up/down bit.";
      }
      leaf ip-prefix {
        type inet:ipv4-address;
        description "IPv4 prefix address";
      }
      leaf prefix-len {
```

```
         type uint8;
         description "IPv4 prefix length (in bits)";
       }
       leaf metric {
         type wide-metric;
         description "IS-IS wide metric value";
       }
       leaf-list tag {
         type uint32;
         description
           "List of 32-bit tags associated with the IPv4 prefix.";
       }
       leaf-list tag64 {
         type uint64;
         description
           "List of 32-bit tags associated with the IPv4 prefix.";
       }
       uses prefix-attributes-extension;
     }

     grouping prefix-ipv6-extended {
       description "Grouping for attributes of an IPv6 prefix
                    as defined in RFC5308.";
       leaf up-down {
         type boolean;
         description "Value of up/down bit.";
       }
       leaf ip-prefix {
         type inet:ipv6-address;
         description "IPv6 prefix address";
       }
       leaf prefix-len {
         type uint8;
         description  "IPv4 prefix length (in bits)";
       }
       leaf metric {
         type wide-metric;
         description  "IS-IS wide metric value";
       }
       leaf-list tag {
         type uint32;
         description
           "List of 32-bit tags associated with the IPv4 prefix.";
       }
       leaf-list tag64 {
         type uint64;
         description
           "List of 32-bit tags associated with the IPv4 prefix.";
```

```
       }
       uses prefix-attributes-extension;
     }

     /* TLVs and sub-TLVs for neighbors */

     grouping neighbor-link-attributes {
       description
         "Grouping for link attributes as defined
         in RFC5029";
       leaf link-attributes-flags {
         type uint16;
         description
           "Flags for the link attributes";
       }
     }
     grouping neighbor-gmpls-extensions {
       description
         "Grouping for GMPLS attributes of a neighbor as defined
         in RFC5307";
       leaf link-local-id {
         type uint32;
         description
           "Local identifier of the link.";
       }
       leaf remote-local-id {
         type uint32;
         description
           "Remote identifier of the link.";
       }
       leaf protection-capability {
         type uint8;
         description
           "Describes the protection capabilities
           of the link. This is the value of the
           first octet of the sub-TLV type 20 value.";
       }
       container interface-switching-capability {
         description
           "Interface switching capabilities of the link.";
         leaf switching-capability {
           type uint8;
           description
             "Switching capability of the link.";
         }
         leaf encoding {
           type uint8;
           description
```

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

          leaf minimum-lsp-bandwidth {
            type rt-types:bandwidth-ieee-float32;
            description "minimum LSP bandwidth.";
          }
          leaf indication {
            type uint8;
            description
              "The indication whether the interface supports Standard
              or Arbitrary SONET/SDH.";
          }
        }
        container psc-specific {
          when "../switching-capability >= 1 and
                ../switching-capability <= 4";
          description
            "Switching Capability-specific information applicable
            when switching type is PSC1,PSC2,PSC3 or PSC4.";

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

```
           leaf mtu {
             type uint16;
             units bytes;
             description
               "Interface MTU";
           }
         }
       }
     }

     grouping neighbor-extended-te-extensions {
       description
         "Grouping for TE attributes of a neighbor as defined
         in 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";
            }
            container default-metric {
              leaf metric {
                type std-metric;
                description "IS-IS default metric value";
              }
             description "IS-IS default metric container";
            }
            container delay-metric {
              leaf metric {
                type std-metric;
                description "IS-IS delay metric value";
              }
              leaf supported {
                type boolean;
                default "false";
                description "IS-IS delay metric supported";
              }
              description "IS-IS delay metric container";
            }
            container expense-metric {
              leaf metric {
                type std-metric;
                description "IS-IS delay expense metric value";
              }
              leaf supported {
                type boolean;
                default "false";
                description "IS-IS delay expense metric supported";
              }
              description "IS-IS delay expense metric container";
            }
            container error-metric {
              leaf metric {
                type std-metric;
                description "IS-IS error metric value";
              }
              leaf supported {
                type boolean;
                default "false";
                description "IS-IS error metric supported";
```

```
            }
            description "IS-IS error metric container";
          }
          description "Instance of a particular adjacency
                       as defined in ISO10589.";
        }
      }
    }

    /* Top-level TLVs */

    grouping tlv132-ipv4-addresses {
      leaf-list ipv4-addresses {
        type inet:ipv4-address;
        description
          "List of IPv4 addresses of the IS-IS node - IS-IS
           reference is TLV 132.";
      }
      description "Grouping for TLV132.";
    }
    grouping tlv232-ipv6-addresses {
      leaf-list ipv6-addresses {
        type inet:ipv6-address;
        description
          "List of IPv6 addresses of the IS-IS node - IS-IS
           reference is TLV 232.";
      }
      description "Grouping for TLV232.";
    }
    grouping tlv134-ipv4-te-rid {
      leaf ipv4-te-routerid {
        type inet:ipv4-address;
        description
          "IPv4 Traffic Engineering router ID of the IS-IS node -
           IS-IS reference is TLV 134.";
      }
      description "Grouping for TLV134.";
    }
    grouping tlv140-ipv6-te-rid {
      leaf ipv6-te-routerid {
        type inet:ipv6-address;
        description
          "IPv6 Traffic Engineering router ID of the IS-IS node -
           IS-IS reference is TLV 140.";
      }
      description "Grouping for TLV140.";
    }
    grouping tlv129-protocols {
```

```
       leaf-list protocol-supported {
         type uint8;
         description
         "List of supported protocols of the IS-IS node -
          IS-IS reference is TLV 129.";
       }
       description "Grouping for TLV129.";
     }
     grouping tlv137-hostname {
       leaf dynamic-hostname {
         type string;
         description
           "Host Name of the IS-IS node - IS-IS reference
            is TLV 137.";
       }
       description "Grouping for TLV137.";
     }
     grouping tlv10-authentication {
       container authentication {
         leaf authentication-type {
           type identityref {
               base key-chain:crypto-algorithm;
           }
           description
             "Authentication type to be used with IS-IS node.";
         }
         leaf authentication-key {
           type string;
           description
             "Authentication key to be used. For security reasons,
              the authentication key MUST NOT be presented in
              a clear text format in response to any request
             (e.g., via get, get-config).";
         }
         description
           "IS-IS node authentication information container -
            IS-IS reference is TLV 10.";
       }
       description "Grouping for TLV10.";
     }
     grouping tlv229-mt {
       container mt-entries {
         list topology {
           description
             "List of topologies supported";

           leaf mt-id {
             type uint16 {
```

```
               range "0 .. 4095";
             }
             description
               "Multi-Topology identifier of topology.";
           }
           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 "1..255";
         }
         description
           "The system may provide a reason to reject the
            adjacency. If the reason is not available,
            an empty string will be returned.
            The expected format is a single line text.";
       }
       description
         "This notification is sent when the system receives a
          Hello PDU from an IS but does not establish an adjacency
          for some reason. The notification generation must be
          throttled with at least 5 seconds between successive
          notifications.";
     }


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

     notification lsp-error-detected {
```

```
       uses notification-instance-hdr;
       uses notification-interface-hdr;
       leaf lsp-id {
         type lsp-id;
         description "LSP ID.";
       }
       leaf raw-pdu {
         type binary;
         description "Received raw PDU.";
       }
       leaf error-offset {
         type uint32;
         description
           "If the problem is a malformed TLV, the error-offset
            points to the start of the TLV. If the problem is with
            the LSP header, the error-offset points to the errant
            byte";
       }
       leaf tlv-type {
         type uint8;
         description
           "If the problem is a malformed TLV, the tlv-type is set
            to the type value of the suspicious TLV. Otherwise,
            this leaf is not present.";
       }
       description
         "This notification is sent when the system receives an
          LSP with a parse error. The notification generation must
          be throttled with at least 5 seconds between successive
          notifications.";
     }

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

```
          type system-id;
          description "Neighbor system-id";
        }
        leaf state {
          type adj-state-type;

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

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

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

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

```
           least 5 seconds between successive notifications.";
       }

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

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

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

7. Security Considerations

The YANG 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 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-isi.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. The Link State Database (LSDB) is represented by the following schema node:

/isis/database

Exposure of the Link State Database includes 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 reconstucted. 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 direction specification of key and authentication algorithm. Hence, authentification 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 too 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.

8. Contributors

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

9. IANA Considerations

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

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

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

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

10. References

10.1. 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>.

10.2. Informative References

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

Appendix A. Example of IS-IS configuration in XML

This section gives an example of configuration of an IS-IS instance on a device. The example is written in XML.


```
<?xml version="1.0" encoding="utf-8"?>
<data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <routing xmlns="urn:ietf:params:xml:ns:yang:ietf-routing">
      <name>SLI</name>
      <router-id>1.1.1.1</router-id>
      <control-plane-protocols>
        <control-plane-protocol>
          <name>ISIS-example</name>
          <description/>
          <type>
            <type xmlns:isis="urn:ietf:params:xml:ns:yang:ietf-isis">
            isis:isis
            </type>
          </type>
          <isis xmlns="urn:ietf:params:xml:ns:yang:ietf-isis">
              <enable>true</enable>
              <level-type>level-2</level-type>
              <system-id>87FC.FCDF.4432</system-id>
              <area-address>49.0001</area-address>
              <mpls>
                <te-rid>
                  <ipv4-router-id>192.0.2.1</ipv4-router-id>
                </te-rid>
              </mpls>
              <lsp-lifetime>65535</lsp-lifetime>
              <lsp-refresh>65000</lsp-refresh>
              <metric-type>
                <value>wide-only</value>
              </metric-type>
              <default-metric>
                <value>111111</value>
              </default-metric>
              <address-families>
                <address-family-list>
                  <address-family>ipv4</address-family>
                  <enable>true</enable>
                </address-family-list>
```

```
                   <address-family-list>
                     <address-family>ipv6</address-family>
                     <enable>true</enable>
                   </address-family-list>
                 </address-families>
                 <interfaces>
                   <interface>
                     <name>Loopback0</name>
                     <tag>200</tag>
                     <metric>
                       <value>0</value>
                     </metric>
                     <passive>true</passive>
                   </interface>
                   <interface>
                     <name>Eth1</name>
                     <level-type>level-2</level-type>
                     <interface-type>point-to-point</interface-type>
                     <metric>
                       <value>167890</value>
                     </metric>
                   </interface>
                 </interfaces>
             </isis>
           </control-plane-protocol>
         </control-plane-protocols>
     </routing>
     <interfaces xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
       <interface>
         <name>Loopback0</name>
         <description/>
         <type xmlns:ianaift="urn:ietf:params:xml:ns:yang:iana-if-type">
         ianaift:softwareLoopback
         </type>
         <link-up-down-trap-enable>enabled</link-up-down-trap-enable>
         <ipv4 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
           <address>
             <ip>192.0.2.1</ip>
             <prefix-length>32</prefix-length>
           </address>
         </ipv4>
         <ipv6 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
           <address>
             <ip>2001:DB8::1</ip>
             <prefix-length>128</prefix-length>
           </address>
         </ipv6>
       </interface>
```

```
      <interface>
        <name>Eth1</name>
        <description/>
        <type xmlns:ianaift="urn:ietf:params:xml:ns:yang:iana-if-type">
        ianaift:ethernetCsmacd
        </type>
        <link-up-down-trap-enable>enabled</link-up-down-trap-enable>
        <ipv4 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
          <address>
            <ip>198.51.100.1</ip>
            <prefix-length>30</prefix-length>
          </address>
        </ipv4>
        <ipv6 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
          <address>
            <ip>2001:DB8:0:0:FF::1</ip>
            <prefix-length>64</prefix-length>
          </address>
        </ipv6>
      </interface>
    </interfaces>
  </data>
```


Authors' Addresses

Stephane Litkowski
Orange

Email: stephane.litkowski@orange.com

Derek Yeung
Arrcus, Inc

Email: derek@arrcus.com

Acee Lindem
Cisco Systems

Email: acee@cisco.com

Jeffrey Zhang
Juniper Networks

Email: zzhang@juniper.net

Ladislav Lhotka
CZ.NIC

Email: lhotka@nic.cz

isis D. Lamparter
Internet-Draft NetDEF
Intended status: Standards Track October 20, 2014
Expires: April 23, 2015

IS-IS Reachability with critical Sub-TLVs
draft-lamparter-isis-reachability-critical-subtlvs-00

Abstract

While previously existing TLVs for IP Reachability extensibly support Sub-TLVs, these cannot be marked as critical. This is required for extending router behaviour with additional qualifiers on routes, hence this document introduces new Reachability TLVs that support critical Sub-TLVs.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

IS-IS is very extensible by design; Newly defined Sub-TLVs can be added in many places. However, the behaviour for unknown Sub-TLVs is always assumed to be "ignore", there is currently no way to prescribe different behaviour. Therefore, a system that receives a Reachability TLV with a Sub-TLV it doesn't recognise will silently process the Reachability with a reduced set of specified information.

This is not desirable for situations where Sub-TLVs provide essential information for the reachability, in particular if that information restricts the usability of the reachability. At the time of writing, usage by extensions of the following types is envisioned:

o further qualifications for the route target, e.g. restricted source address or flowlabel. In this case the reachability information is incomplete (and the route does not match) without these critical fields.

o mandatory encapsulation specifications, e.g. routing headers or labels required for the egress router or systems outside the domain. Here, ignorance of that information would render these systems unable to apply correct forwarding decisions.

Other future developments may find even more use cases for this TLV. The functionality defined here could also have been used for M-ISIS

[RFC5120] reachabilities in order to hide them from non-M-ISIS routers without introducing a new TLV type.

Therefore, this document creates a new Reachability TLV with a critical Sub-TLV part, where the specified behavior on unrecognized Sub-TLVs is to ignore the entire Reachability TLV, not just the Sub-TLV.

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 [RFC2119].

2. Design considerations

This document specifies new Reachability TLVs for IPv4 and IPv6. These new TLVs have two Sub-TLV blocks: one critical and one optional. Sub-TLVs in the optional block behave exactly as Sub-TLVs in previous Reachability TLVs (135, 235, 236, and 237.) This includes application of the same TLV namespace, all TLVs defined for these four TLVs are also applicable in the optional part of the new two TLVs.

The critical Sub-TLV block constitutes a separate namespace. A system MUST keep these separated, and specifications MUST define to which part exactly they apply. Expected combinations are:

- o 135, 235, 236, 237, TBD1 and TBD2 optional
- o 135, 235, TBD1 optional (IPv4)
- o 236, 237, TBD2 optional (IPv6)
- o TBD1 and TBD2 critical
- o TBD1 critical (IPv4)
- o TBD2 critical (IPv6)

Though no such use is foreseen at this point, a specification MAY specify a TLV to be valid in either the optional or critical part. This TLV may end up with different codepoints in each of the namespaces.

A system MUST NOT originate these new TLVs with an empty critical part. Doing so would create an alternate encoding of the previous TLVs, breaking interoperability. Systems SHOULD process a new TLV with an empty critical block.

There is no need for non-MT variants of these TLVs. If a system does not implement M-ISIS, it MUST ignore all TLVs with a MT ID other than zero.

3. SPF Functional specification

This document assumes that all transit routers need to support processing of the feature associated with a respective critical Sub-TLV. Hence, on calculating a path for a reachability with critical Sub-TLV A, all intermediate systems that do not indicate support for Sub-TLV A must be excluded.

The logical result from this is essentially that separate SPF trees MUST be calculated for each set of critical Sub-TLVs.

Calculation of these extra trees can be optimized by sharing intermediate calculation results as far as critical Sub-TLV support is identical.

A system MUST NOT blindly use a "more Sub-TLVs supported" SPF calculation result for calculating paths that require only a subset of these Sub-TLVs. This would result in a disagreement on shortest path with other routers, which correctly used a SPF tree for the specific combination.

3.1. Simplified SPF

TBD: It is possible to construct a variant of this that doesn’t implicitly work with multiple topologies, instead marking routes as unreachable if they transit over routers that do not support the critical TLVs. This may be useful for simpler implementations.

4. TLV formats

4.1. IP/IPv6 Reachability TLV

The encoding for TLVs TBD1 and TBD2 is modified from TLVs 235 and 237 by inserting a second length field for the critical Sub-TLV part before the existing length field for the optional Sub-TLV part. The critical Sub-TLV part follows after the length field, then the optional part.

This results in the following TLV structure:

```
(2/4 bytes TLV header)
2 octets of MT ID (12 bits, top 4 bits reserved)
-- multiple (n >= 1) occurences of the following:
4 octets of metric information
1 octet of control information, consisting of
     1 bit of up/down information
     1 bit indicating the presence of optional sub-TLVs
     6 bits of prefix length
     0-4/0-16 octets of IPv4/IPv6 prefix
4-n optional octets of sub-TLVs, if present consisting of
     1/2 octets of length of critical sub-TLVs
     2-n octets of critical sub-TLVs,
     -- depending on presence of optional sub-TLVs indication:
     0-2 octets of length of optional sub-TLVs
     0-n octets of optional sub-TLVs,
          where each sub-TLV (critical or optional) is a sequence of
               1/2 octets of sub-type
               1/2 octets of length of the value field of the sub-TLV
               0-n octets of value
```

Unlike MT Reachability TLVs, this TLV MUST NOT be ignored if the MT ID is zero. Instead, the information applies to the "standard" topology.

The size of offset and length fields depends on the PDU in which the TLV is found, as per [RFC7356].

The critical sub-TLV part MUST NOT be empty. Reachability TLVs without a critical Sub-TLV field MUST be used instead in this case.

As in TLVs 235 and 237, the optional sub-TLV length and data fields are only present if the "presence of optional sub-TLVs" bit is one.

4.2. Reachability Critical Sub-TLVs Supported TLV

Supported Critical Information Sub-TLV format

```
 0                   1        LSB
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
+-+-+-+-+-+-+-+-+ + + + + + + + +
|   Type = TBD3 |               : (1 / 2 bytes)
+-+-+-+-+-+-+-+-+ + + + + + + + +
|     Length    |               : (1 / 2 bytes)
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Applicable TLV length      |0| (2 bytes)
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Applicable TLV numbers       | (n * 2 bytes)
:                               :

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ (multiple of this block allowed)
|  Sub-TLV combination length |C| (2 bytes)
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Sub-TLV numbers              | (n * 2 bytes)
:                               :
```

Applicable TLV length and numbers specify which (parent) TLVs this information applies to. Both fields are always in 2-octet units each, which means the length is even. Thus, the LSB of the length field MUST be set to 0 on TLV origination. Systems SHOULD ignore the entire TLV if the applicable TLV length field is not even. The same applies if the applicable TLV length is zero, systems SHOULD ignore the entire TLV.

Sub-TLV combination length and numbers specify supported Sub-TLVs for the TLVs with applicable TLV numbers listed before. As with Applicable TLVs, these are units of 2 octets each.

The LSB of the combination length is redefined to be the "Combinatorial" bit. Any mixture (present or not present) of Sub-TLVs listed with C=1, plus any Sub-TLVs present in at most one list with C=0 are understood to be supported by the router. A combination of Sub-TLVs present in two distinct lists with C=0 MUST NOT be assumed to be in a router’s supported set.

The block of Sub-TLV combination length and numbers MAY occur multiple times, as MAY the entire TLV. The information MUST be merged.

Systems MUST process known TLVs even if unknown TLVs are present. The latter MUST be ignored.

5. IANA Considerations

5.1. IS-IS TLV Codepoints

This document requests the allocation of two codepoints from the IS-IS TLV Codepoints registry. Suggested values are 238 for TBD1 and 239 for TBD2.

Top-level codepoints

Value	Name	IIH	LSP	SNP	Purge
TBD1	MT IPv4 Reach with Critical Sub-TLVs	n	y	n	n
TBD2	MT IPv6 Reach with Critical Sub-TLVs	n	y	n	n

A codepoint from the Sub-TLVs for TLV 144 registry is also requested:

TLV 144 Sub-TLV codepoints

Value	Name
TBD3	Reachability Critical Sub-TLVs Supported

5.2. TLVs 135, 235, 236, 237 Sub-TLV Registry

The registry for Sub-TLVs below TLVs 135, 235, 236, and 237 is requested to be renamed to "Sub-TLVs for TLVs 135, 235, 236, 237, TBD1 (optional) and TBD2 (optional)". Two new columns are added to the table: "TBD1 (optional)" and "TBD2 (optional)". The value for preexisting entries is copied from 235 to TBD1 and from 237 to TBD2. This document is added as reference.

5.3. TLVs TBD1, TBD2 critical Sub-TLV Registry

This document requests creation of a new registry named "Sub-TLVs for TLVs TBD1 (critical), and TBD2 (critical)". Procedures and experts are inherited from the registry in the previous paragraph. The registry’s table is initially empty and has a total of two applicability columns titled "TBD1 (critical)" and "TBD2 (critical)". The starting value for allocations is 1.

6. Security Considerations

The mechanism outlined in this document can be used to perform memory and processor resource exhaustion attacks against routers. By introducing reachabilities with different sets of critical Sub-TLVs present, participating routers are forced to calculate different SPF trees.

As a countermeasure, routers SHOULD:

- o only calculate SPF trees for critical TLV combinations they support

- o conflate SPF trees where logically correct, i.e. where routers' lists of critical TLV combinations overlap

7. Privacy Considerations

No privacy considerations apply to this document, as it only specifies routing control plane information.

8. Acknowledgements

This document is largely the result of discussions with Fred Baker.

9. Change Log

Initial Version: October 2014

10. Normative References

[IS-IS] ISO/IEC, "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, Second Edition, 2002.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC5120] Przygienda, T., Shen, N., and N. Sheth, "M-ISIS: Multi Topology (MT) Routing in Intermediate System to Intermediate Systems (IS-ISs)", RFC 5120, February 2008.

[RFC7356] Ginsberg, L., Previdi, S., and Y. Yang, "IS-IS Flooding Scope Link State PDUs (LSPs)", RFC 7356, September 2014.

Author's Address

David Lamparter
NetDEF
Leipzig 04103
Germany

Email: david@opensourcerouting.org

isis B. Liu, Ed.
Internet-Draft Huawei Technologies
Intended status: Standards Track B. Decraene
Expires: April 21, 2016 Orange
I. Farrer
Deutsche Telekom AG
M. Abrahamsson
T-Systems
L. Ginsberg
Cisco Systems
October 19, 2015

ISIS Auto-Configuration
draft-liu-isis-auto-conf-06

Abstract

This document specifies an IS-IS auto-configuration technology. The key mechanisms of this technology are IS-IS System ID self-generation, duplication detection and duplication resolution. This technology fits the environment where plug-and-play is expected.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This document describes mechanisms for IS-IS [RFC1195] [ISO_IEC10589][RFC5308] to be auto-configuring. Such mechanisms could reduce the management burden to configure a network. Home networks and small or medium size enterprise networks where plug-and-play is expected can benefit from these mechanisms.

This document also defines mechanisms which prevent unintentional interoperation of autoconfigured routers with non-autoconfigured routers. See Section 3.3.1 .

IS-IS auto-configuration contains the following aspects:

1. IS-IS default configurations

2. IS-IS System ID self-generation

3. System ID duplication detection and resolution

4. ISIS TLVs utilization such as Authentication TLV, Wide Metric TLV etc.

2. Scope

The auto-configuring mechanisms support both IPv4 and IPv6 deployments.

This auto-configuration mechanism aims at simple case. The following advanced features are out of scope:

- o Multiple IS-IS instances

- o Multi-area and level-2 routing

- o Interworking with other routing protocols

3. Protocol Specification

3.1. IS-IS Default Configuration

- o IS-IS interfaces MUST be auto-configured to an interface type corresponding to their layer-2 capability. For example, Ethernet interfaces will be auto-configured as broadcast networks and Point-to-Point Protocol (PPP) interfaces will be auto-configured as Point-to-Point interfaces.

- o IS-IS auto-configuration instance MUST be configured with level-1, so that the interfaces operate at level-1 only.

- o IS-IS auto-configuration SHOULD allow P2P mode on Ethernet interfaces.

3.2. IS-IS NET Generation

In IS-IS, a router (known as an Intermediate System) is identified by an NET which is the address of a Network Service Access Point (NSAP) and represented with an IS-IS specific address format. The NSAP is a logical entity which represents an instance of the IS-IS protocol running on an Intermediate System.

The autoconfiguration mechanism generates the IS-IS NET as the following:

o Area address

   This field is 1 to 13 octets in length. In IS-IS auto-configuration, this field MUST be 13 octets of all 0.

o System ID

   This field follows the area address field, and is 6 octets in length. There are two basic requirements for the System ID generation:

   - As specified in IS-IS protocol, this field must be unique among all routers in the same area.

   - In order to make the routing system stable, the System ID SHOULD remain the same after it is firstly generated. It SHOULD not be changed due to device status change (such as interface enable/disable, interface plug in/off, device reboot, firmware update etc.) or configuration change (such as changing system configurations or IS-IS configurations etc.); but it MUST allow be changed by collision resolution and SHOULD allow be cleared by user enforced system reset.

   More specific considerations for System ID generation are described in Section 3.3.3 .

3.3. IS-IS System ID Duplication Detection and Resolution

The System ID of each node MUST be unique. As described in Section 3.3.3, the System ID is generated based on entropies such as MAC address which are supposed to be unique, but in theory there is still possibility of duplication. This section defines how IS-IS detects and resolves System ID duplication.

3.3.1. Router-Fingerprint TLV

The Router-Fingerprint TLV basically re-uses the design of Router-Hardware-Fingerprint TLV defined in [RFC7503]. However, there is one difference that one flag is added to indicate the node is in "start-up mode" which is defined in Section 3.3.2 .

```
 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     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|S|A| Reserved  |                                               |
+-+-+-+-+-+-+-+-+        Router Fingerprint (Variable)
             .
.                                                               .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The length of the Router-Fingerprint is variable but must be 32 octets or greater; and the content is also supposed to be unique among all the routers.

- o Type: to be assigned by IANA.
- o Length: the length of the value field.
- o S flag: indicates the router is in "start-up" mode as described below.
- o A flag: indicates the router is operating in autoconfiguration mode. This flag is in case the TLV gets used outside of autoconfiguration. If A flag setting does not match in hellos then no adjacency should be formed.
- o Reserved: these bits MUST be set to zero and MUST be ignored when received.
- o Router Fingerprint: uniquely identifies a router, variable length.

More specific considerations for Router-Fingerprint is described in Section 3.3.3 .

3.3.2. System ID Duplication Detection and Resolution Procedures

This section describes the System ID duplication detection and resolution between two neighbors and two non-neighbors respectively. This is because the routing messages between neighbors and non-neighbors are a bit different.

3.3.2.1. Start-up Mode

While in startup-mode, an auto-configuration router forms adjacencies but generates only LSP #0 which contains only the Router-Fingerprint TLV. A router remains in startup-mode until it has successfully completed LSPDB synchronization with all neighbors or until 1 minute has elapsed - whichever is longer. If duplicate system-ID is detected while in startup-mode the router MUST clear all adjacencies, select a new system-id (subject to rules defined in Section 3.3.2.2), and reenter Startup-mode.

The start-up mode is to minimize the occurrence of System ID changes for a router once it has become fully operational. It has minimal impact on a running network because the startup node is not yet being used for forwarding traffic. Once duplicate System ID has been resolved the router begins normal operation. If two routers are both in startup mode (or both NOT in startup mode) and duplicate system-id is detected then they determine which one changes its system-id based on fingerprint.

When an IS-IS auto-configuration router boots up, it MUST operate in start-up mode until duplicate system-id detection has successfully completed.

3.3.2.2. Duplication Between Neighbors

In case of System ID duplication occurs between neighbors, an IS-IS auto-configuration router MUST include the Router-Fingerprint TLV in the Hello messages, so that the duplication could be detected before adjacency forming.

Procedures of the nodes in Start-up Mode:

1. Boot up, advertise the Router-Fingerprint TLV in Hello message

   The router sends Hellos which include the Router-Fingerprint TLV. Adjacencies are formed as normal but MUST NOT be advertised in LSPs until the router exits startup-mode.

2. Receive Hello message(s), and verifies System ID duplication

   Received hellos are inspected for possible duplicate System ID. If duplication is detected, the router MUST check the S flag of the Router-Fingerprint TLV.

   + If the S flag is NOT set (which means the Hello was NOT generated by a neighbor also in Start-up mode), then the router MUST re-generate the System ID and reenter Startup-mode.

   + If the S flag is set (which means the neighbor is also in Startup-mode),

- the router which has a numerically smaller Router-Fingerprint MUST re-generate the System ID and reenter Startup-mode. Fingerprint comparison is performed octet by octet until octets are different. Then the smaller fingerprint is the one with the smaller octet (unsigned integer). If the fingerprints have different lengths, then the shorter length fingerprint MUST be padding with zero for comparison.

- If Router Fingerprints are identical, both routers MUST re-generate the System ID and the Router Fingerprint, and reenter Startup-mode.

3. Run in normal operation

After the System ID duplication procedure is done, the router begins to run in normal operation. The router MUST re-advertise the Router-Fingerprint TLV with the S flag off.

Procedures of the nodes NOT in Start-up Mode:

1. Compare the System ID in received Hello messages

When receiving a Hello message, the router MUST check the System ID of the Hello. If the System ID is the same as its own, it indicates a System ID duplication occurs.

If there is no Router-Fingerprint TLV in the Hello message, it means a non-autoconfiguration router by accident connected to the auto-configuration domain or other unexpected bad behaviors. In this case, the auto-configuration router MUST NOT form adjacency with the non-autoconfiguration router.

2. Duplication resolution

When System ID duplication occurs, the non-startup mode router MUST check the S flag of the duplicated Router-Fingerprint TLV:

+ If the S flag is NOT set, then the router with the numerically smaller or equal Router-Fingerprint MUST generate a new System ID. Note that, the router MUST compare the two Router-Fingerprint in terms of two numeric numbers.

+ If the S flag is set, then router does nothing, because it MUST be the node which is in start-up mode re-generates the System ID.

3. Re-join the network with the new System ID (if required)

The router with the smaller Router-Fingerprint advertise new Hellos based on the newly generated NET to re-join the IS-IS auto-configuration network. The router with the highest Router-Fingerprint MUST re-advertise its own LSP (after increasing the sequence number).

The newly generated System ID SHOULD take a duplication detection as well.

3.3.2.3. Duplication Between Non-neighbors

System ID duplication may also occur between non-neighbors, so an IS-IS auto-configuration router MUST also include the Router-Fingerprint TLV in the LSP messages. Specific procedures are as the following.

Procedures of the nodes in Start-up Mode:

1. Boot up, form adjacency

2. Acquire LSPDB and verifies System ID duplication

   The router generates only LSP #0 which contains only the Fingerprint TLV; and that Fingerprint is only sent in LSP #0. A router remains in startup-mode until it has successfully completed LSPDB synchronization with all neighbors or until 1 minute has elapsed - whichever is longer. If duplicate system-ID is detected, the router MUST check the S flag of the Router-Fingerprint TLV of the LSP that contains the duplicated System ID.

   + If the S flag is not set, it means the LSP was not generated at the Start-up Mode, then the router itself MUST clear all adjacencies, re-generate a new system-id and reenter Startup-mode.

   + If the S flag is set, then the router which has a numerically smaller Router-Fingerprint MUST generate a new System ID and reenter Startup-mode.

3. Run in normal operation

   After the System ID duplication procedure is done, the router begins to run in normal operation. The router MUST re-advertise the Router-Fingerprint TLV with the S flag off.

Procedures of the nodes not in Start-up Mode:

1. Compare the received Router-Fingerprint TLVs

   When receiving a LSP containing its own System ID, the router MUST check the Router-Fingerprint TLV. If the Router-Fingerprint TLV is different from its own, it indicates a System ID duplication occurs.

2. Duplication resolution

   When System ID duplication occurs, the non-startup mode router MUST check the S flag of the duplicated Router-Fingerprint TLV:

   + If the S flag is NOT set, then the router with the numerically smaller Router-Fingerprint MUST generate a new System ID. Note that, the router MUST compare the two Router-Fingerprint in terms of two numeric numbers.

   + If the S flag is set, then router does nothing, because according to the start-up mode procedure, the start-up node MUST re-generate the System ID.

3. Re-join the network with the new System ID

   The router changing its system ID advertise new LSPs based on the newly generated System ID to re-join the IS-IS auto-configuration network. The router with the highest Router-Fingerprint MUST re-advertise its own LSP (after increasing the sequence number).

   The newly generated System SHOULD take a duplication detection as well.

3.3.3. System ID and Router-Fingerprint Generation Considerations

As specified in this document, there are two distinguisher need to be self-generated, which is System ID and Router-Fingerprint. In a network device, normally there are resources which provide an extremely high probability of uniqueness thus could be used as seeds to derive distinguisher (e.g. hashing or generating pseudo-random numbers), such as:

o MAC address(es)

o Configured IP address(es)

- o Hardware IDs (e.g. CPU ID)
- o Device serial number(s)
- o System clock at a certain specific time
- o Arbitrary received packet

This document recommends to use an IEEE 802 48-bit MAC address associated with the router as the initial System ID. This document does not specify a specific method to re-generate the System ID when duplication happens.

This document also does not specify a specific method to generate the Router-Fingerprint. However, the generation of System ID and Router-Fingerprint MUST be based on different seeds so that the two distinguisher would not collide.

There is an important concern that the seeds listed above (except MAC address) might not be available in some small devices such as home routers. This is because of the hardware/software limitation and the lack of sufficient communication packets at the initial stage in the home routers when doing ISIS-autoconfiguration. In this case, this document suggests to use MAC address as System ID and generate a pseudo-random number based on another seed (such as the memory address of a certain variable in the program) as Router-Fingerprint. The pseudo-random number might not have a very high quality in this solution, but should be sufficient in home networks scenarios.

Note that, the Router-Fingerprint SHOULD also remain the same after it is firstly generated. It SHOULD not be changed due to device status change (such as interface enable/disable, interface plug in/off, device reboot, firmware update etc.) or configuration change (such as changing system configurations or IS-IS configurations etc.); but it MUST allow be changed by double-duplication resolution Section 3.3.4 and SHOULD allow be cleared by user enforced system reset.

3.3.4. Double-Duplication of both System ID and Router-Fingerprint

As described above, the resources for generating the distinguisher might be very constrained at the initial stage. Hence, the double-duplication of both System ID and Router-Fingerprint needs to be considered.

ISIS-autoconfiguring routers SHOULD support detecting System ID duplication by LSP war. LSP war is a phenomenon that if a router receives a LSP originated with its System ID, but it doesn’t find it in the database, or it does not match the one the router has (e.g. It advertises IP prefixes that the router doesn't own, or IS neighbors that the router doesn't see), then per ISIS specification, the router must re-originate its LSP with an increased sequence number. If double-duplication happens, the duplicated two routers will both continuously have the above behavior. After multiples iterations, the program should be able to deduce that double-duplication happens.

At the point when double-duplication happens, routers should have much more entropies available. Thus, the router is to extend or re-generate its Router-Fingerprint (one simple way is just adding the LSP sequence number of the next LSP it will send to the Router-Fingerprint). (Optimized solution TBD.)

3.4. IS-IS TLVs Usage

This section describes several TLVs that are utilized by IS-IS auto-configuration.

3.4.1. Authentication TLV

It is RECOMMENDED that IS-IS routers supporting this specification minimally offer an option to explicitly configure a single password for HMAC-MD5 authentication, which is Type 54 authentication mode of [RFC5304]. In this case, the Authentication TLV (TLV 10) is needed.

3.4.2. Wide Metric TLV

IS-IS auto-configuration routers MUST support TLVs using wide metric as defined in [RFC5305]).

It is recommended that IS-IS auto-configuration routers use a high metric value (e.g. 1000000) as default in order to typically prefer the manually configured adjacencies rather than the auto-configuring ones.

3.4.3. Dynamic Host Name TLV

IS-IS auto-configuration routers MAY advertise their Dynamic Host Names TLV (TLV 137, [RFC5301]). The host names could be provisioned by an IT system, or just use the name of vendor, device type or serial number etc. Note that, the hostname needs to be unique so that it could be useful.

3.5. Routing Behavior Considerations

3.5.1. Adjacency Formation

Since ISIS does not require strict hold timers matching to form adjacency, this document does not specify specific hold timers. However, the timers should be within a reasonable range based on current practise in the industry. (For example, the defaults defined in [ISO_IEC10589] .)

4. Security Considerations

In general, auto-configuration is mutually incompatible with authentication. This is a common problem that IS-IS auto-configuration can not avoid.

For wired deployment, the wired line itself could be considered as an implicit authentication that normally unwanted routers are not able to connect to the wire line; for wireless deployment, the authentication could be achieve at the lower wireless link layer.

Malicious router could modify the System ID field to keep causing System ID duplication detection and resolution thus cause the routing system oscillate. However, this is not a new attack vector as without this document the consequences would be higher as other routers would not try to adapt.

5. IANA Considerations

The Router-Fingerprint TLV type code needs an assignment by IANA.

6. Acknowledgements

This document was heavily inspired by [RFC7503].

Martin Winter, Christian Franke and David Lamparter gave essential feedback to improve the technical design based on their implementation experience.

Many useful comments were made by Acee Lindem, Karsten Thomannby, Hannes Gredler, Peter Lothberg, Uma Chundury, Qin Wu, Sheng Jiang and Nan Wu, etc.

This document was produced using the xml2rfc tool [RFC2629]. (initially prepared using 2-Word-v2.0.template.dot.)

7. References

7.1. Normative References

[ISO_IEC10589]
""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", November 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, <http://www.rfc-editor.org/info/rfc1195>.

[RFC2629] Rose, M., "Writing I-Ds and RFCs using XML", RFC 2629, DOI 10.17487/RFC2629, June 1999, <http://www.rfc-editor.org/info/rfc2629>.

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

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

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

[RFC5308] Hopps, C., "Routing IPv6 with IS-IS", RFC 5308, DOI 10.17487/RFC5308, October 2008, <http://www.rfc-editor.org/info/rfc5308>.

[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, <http://www.rfc-editor.org/info/rfc6232>.

7.2. Informative References

[I-D.ietf-homenet-hncp]
Stenberg, M., Barth, S., and P. Pfister, "Home Networking Control Protocol", draft-ietf-homenet-hncp-09 (work in progress), August 2015.

[RFC7503] Lindem, A. and J. Arkko, "OSPFv3 Autoconfiguration", RFC 7503, DOI 10.17487/RFC7503, April 2015, <http://www.rfc-editor.org/info/rfc7503>.

Authors' Addresses

Bing Liu
Huawei Technologies
Q14, Huawei Campus, No.156 Beiqing Road
Hai-Dian District, Beijing, 100095
P.R. China

Email: leo.liubing@huawei.com

Bruno Decraene
Orange
38 rue du General Leclerc
Issy-les-Moulineaux FR
FR

Email: bruno.decraene@orange.com

Ian Farrer
Deutsche Telekom AG
Bonn
Germany

Email: ian.farrer@telekom.de

Mikael Abrahamsson
T-Systems
Stockholm
Sweden

Email: mikael.abrahamsson@t-systems.se

Les Ginsberg
Cisco Systems
510 McCarthy Blvd.
Milpitas CA 95035
USA

Email: ginsberg@cisco.com

Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: August 3, 2015

A. Przygienda
Ericsson
L. Ginsberg
Cisco Systems
S. Aldrin
Huawei
J. Zhang
Juniper Networks, Inc.
January 30, 2015

BIER support via ISIS
draft-przygienda-bier-isis-ranges-02

Abstract

Specification of an ISIS extension to support BIER domains and sub-domains.

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

Bit Index Explicit Replication (BIER) [I-D.draft-wijnands-bier-architecture-02] defines an architecture where all intended multicast receivers are encoded as bitmask in the Multicast packet header within different encapsulations such as [I-D.draft-wijnands-mpls-bier-encapsulation-02]. A router that receives such a packet will forward the packet based on the Bit Position in the packet header towards the receiver(s), following a precomputed tree for each of the bits in the packet. Each receiver is represented by a unique bit in the bitmask.

This document presents necessary extensions to the currently deployed ISIS for IP [RFC1195] protocol to support distribution of information necessary for operation of BIER domains and sub-domains. This document defines a new TLV to be advertised by every router participating in BIER signaling.

2. Terminology

Some of the terminology specified in [I-D.draft-wijnands-bier-architecture-02] is replicated here and extended by necessary definitions:

BIER: Bit Index Explicit Replication (The overall architecture of forwarding multicast using a Bit Position).

BIER-OL: BIER Overlay Signaling. (The method for the BFIR to learn about BFER's).

BFR: Bit Forwarding Router (A router that participates in Bit Index Multipoint Forwarding). A BFR is identified by a unique BFR-prefix in a BIER domain.

BFIR: Bit Forwarding Ingress Router (The ingress border router that inserts the BM into the packet).

BFER: Bit Forwarding Egress Router. A router that participates in Bit Index Forwarding as leaf. Each BFER must be a BFR. Each BFER must have a valid BFR-id assigned.

BFT: Bit Forwarding Tree used to reach all BFERs in a domain.

BIFT: Bit Index Forwarding Table.

BMS: Bit Mask Set. Set containing bit positions of all BFER participating in a set.

BMP: Bit Mask Position, a given bit in a BMS.

Invalid BMP: Unassigned Bit Mask Position, consisting of all 0s.

IGP signalled BIER domain: A BIER underlay where the BIER synchronization information is carried in IGP. Observe that a multi-topology is NOT a separate BIER domain in IGP.

BIER sub-domain: A further distinction within a BIER domain identified by its unique sub-domain identifier. A BIER sub-domain can support multiple BitString Lengths.

BFR-id: An optional, unique identifier for a BFR within a BIER sub-domain.

Invalid BFR-id: Unassigned BFR-id, consisting of all 0s.

3. IANA Considerations

This document adds the following new sub-TLVs to the registry of sub-TLVs for TLVs 235, 237 [RFC5120] and TLVs 135,236 [RFC5305],[RFC5308].

Value: 32 (suggested - to be assigned by IANA)

Name: BIER Info

4. Concepts

4.1. BIER Domains and Sub-Domains

An ISIS signalled BIER domain is aligned with the scope of distribution of BFR-prefixes that identify the BFRs within ISIS. ISIS acts in such a case as the according BIER underlay.

Within such a domain, ISIS extensions are capable of carrying BIER information for multiple BIER sub-domains. Each sub-domain is uniquely identified by its subdomain-id and each subdomain can reside in any of the ISIS topologies [RFC5120]. The mapping of sub-domains to topologies is a local decision of each BFR currently but is advertised throughout the domain to ensure routing consistency.

Each BIER sub-domain has as its unique attributes the encapsulation used and the type of tree it is using to forward BIER frames (currently always SPF). Additionally, per supported bitstring length in the sub-domain, each router will advertise the necessary label ranges to support it.

This RFC introduces a sub-TLV in the extended reachability TLVs to distribute such information about BIER sub-domains. To satisfy the requirements for BIER prefixes per [I-D.draft-wijnands-bier-architecture-02] additional information will be carried in [I-D.draft-ginsberg-isis-prefix-attributes].

5. Procedures

5.1. Enabling a BIER Sub-Domain

A given sub-domain with identifier BS with supported bitstring lengths MLs in a multi-topology MT [RFC5120] is denoted further as <MT,SD,MLs> and is normally not advertised to preserve the scaling of the protocol (i.e. ISIS carries no TLVs containing any of the elements related to <MT,SD>) and is enabled by a first BIER sub-TLV (Section 6.1) containing <MT,SD> being advertised into the area. The trigger itself is outside the scope of this RFC but can be for example a VPN desiring to initiate a BIER sub-domain as MI-PMSI [RFC6513] tree. It is outside the scope of this document to describe what trigger for a router capable of participating in <MT,SD> is used to start the origination of the necessary information to join into it.

5.2. Multi Topology and Sub-Domain

All routers in the flooding scope of the BIER TLVs MUST advertise a sub-domain within the same multi-topology. A router discovering a sub-domain advertised within a topology that is different from its own MUST report a misconfiguration of a specific sub-domain. Each router MUST compute BFTs for a sub-domain using only routers advertising it in the same topology.

5.3. Encapsulation

All routers in the flooding scope of the BIER TLVs MUST advertise the same encapsulation for a given <MT,SD>. A router discovering encapsulation advertised that is different from its own MUST report a misconfiguration of a specific <MT,SD>. Each router MUST compute BFTs for <MT,SD> using only routers having the same encapsulation as its own advertised encapsulation in BIER sub-TLV for <MT,SD>.

5.4. Tree Type

All routers in the flooding scope of the BIER TLVs MUST advertise the same tree type for a given <MT,SD>. In case of mismatch the behavior is analogous to Section 5.3.

5.5. Label Advertisements for MPLS encapsulated BIER sub-domains

Each router MAY advertise within the BIER MPLS Encapsulation sub-sub-TLV (Section 6.2) of a BIER Info sub-TLV (Section 6.1, denoted as TLV<MT,SD>) for <MT,SD> for every supported bitstring length a valid starting label value and a non-zero range length. It MUST advertise at least one valid label value and a non-zero range length for the required bitstring lengths per [I-D.draft-wijnands-bier-architecture-02] in case it has computed itself as being on the BFT rooted at any of the BFRs with valid BFR-ids (except itself if it does NOT have a valid BFR-id) participating in <MT,SD>.

A router MAY decide to not advertise the BIER Info sub-TLV (Section 6.1) for <MT,SD> if it does not want to participate in the sub-domain due to resource constraints, label space optimization, administrative configuration or any other reasons.

5.5.1. Special Consideration

A router MUST advertise for each bitstring length it supports in <MT,SD> a label range size that guarantees to cover the maximum BFR-id injected into <MT,SD> (which implies a certain maximum set id per bitstring length as described in [I-D.draft-wijnands-bier-architecture-02]). Any router that violates this condition MUST be excluded from BIER BFTs for <MT,SD>.

5.6. BFR-id Advertisements

Each BFER MAY advertise with its TLV<MT,SD> the BFR-id that it has administratively chosen.

If a router discovers that two BFRs it can reach advertise the same value for BFR-id for <MT,SD>, it MUST report a misconfiguration and disregard those routers for all BIER calculations and procedures for <MT,SD> to align with [I-D.draft-wijnands-bier-architecture-02]. It is worth observing that based on this procedure routers with colliding BFR-id assignments in <MT,SD> MAY still act as BFIRs in <MT,SD> but will be never able to receive traffic from other BFRs in <MT,SD>.

5.7. Flooding

BIER domain information SHOULD change and force flooding infrequently. Especially, the router SHOULD make every possible attempt to bundle all the changes necessary to sub-domains and ranges advertised with those into least possible updates.

5.8. Version

This RFC specifies Version 0 of the BIER extension encodings. Packet encoding supports introduction of future, higher versions with e.g. new sub-sub-TLVs or redefining reserved bits that can maintain the compatiblity to Version 0 or choose to indicate that the compatibility cannot be maintained anymore (changes that cannot work with the provided encoding would necessitate obviously introduction of completely new sub-TLV for BIER).

This kind of 'versioning' allows to introduce e.g. backwards-compatible automatic assignment of unique BFR-ids within sub-domains or addition of optional sub-sub-TLVs that can be ignored by version 0 BIER routers without the danger of incompatiblity.

This is a quite common technique in software development today to maintain and extend backwards compatible APIs.

6. Packet Formats

All ISIS BIER information is carried within the TLVs 235, 237 [RFC5120] and TLVs 135,236 [RFC5305], [RFC5308].

6.1. BIER Info sub-TLV

This sub-TLV carries the information for the BIER sub-domains that the router participates in as BFR. It can repeat multiple times for different sub-domain <MT,SD> combinations.

The sub-TLV carries a single <MT,SD> combination followed by optional sub-sub-TLVs specified within its context such as e.g. BIER MPLS Encapsulation per Section 6.2.

On violation of any of the following conditions, the receiving router SHOULD signal a misconfiguration condition. Further results are unspecified unless described in the according section of this RFC:

- o The subdomain-id MUST be included only within a single 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Type       |   Length      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Ver|C| Reserved| subdomain-id  |   BFR-id                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type: as indicated in IANA section.

Length: 1 octet.

Version: Version of the BIER TLV advertised, must be 0 on transmission by router implementing this RFC. Behavior on reception depends on the 'C' bit. 2 bits

C-BIT: Compatibility bit indicating that the TLV can be interpreted by routers implementing lower than the advertised version. Router implementing this version of the RFC MUST set it to 1. On reception, IF the version of the protocol is higher than 0 AND the bit is set (i.e. its value is 1), the TLV MUST be processed normally, IF the bit is clear (i.e. its value is 0), the TLV MUST be ignored for further processing completely independent of the advertised version. When processing this sub-TLV with compatibility bit set, all sub-sub-TLV of unknown type MUST and CAN be safely ignored. 1 bit

Reserved: reserved, must be 0 on transmission, ignored on reception. May be used in future versions. 5 bits

subdomain-id: Unique value identifying the BIER sub-domain. 1 octet

BFR-id: A 2 octet field encoding the BFR-id, as documented in [I-D.draft-wijnands-bier-architecture-02]. If set to the invalid BFR-id advertising router is not owning a BFR-id in the sub-domain.

6.2. BIER MPLS Encapsulation sub-sub-TLV

This sub-sub-TLV carries the information for the BIER MPLS encapsulation and the necessary label ranges per bitstring length for a certain <MT,SD> and is carried within the BIER Info sub-TLV (Section 6.1) that the router participates in as BFR.

On violation of any of the following conditions, the receiving router SHOULD signal a misconfiguration condition. Further results are unspecified:

- The sub-sub-TLV MUST be included once AND ONLY once within the sub-TLV.

- Label ranges within the sub-sub-TLV MUST NOT overlap. A receiving BFR MAY additionally check whether any of the ranges in all the sub-sub-TLVs advertised by another BFR overlap and apply the same treatement on violations.

- Bitstring lengths within the sub-sub-TLV MUST NOT repeat.

- The sub-sub-TLV MUST include the required bitstring lengths per [I-D.draft-wijnands-bier-architecture-02].

o All label range sizes MUST be greater than 0.

o All labels MUST represent valid label values.

```
 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      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ <-+
| Lbl Range Size|BS Len |                   Label               |   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+   |
        ~~ (number repetitions derived from TLV length) ~~          ~~~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+   |
| Lbl Range Size|BS Len |                   Label               |   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ <-+
```

Type: value of 0 indicating MPLS encapsulation.

Length: 1 octet.

Local BitString Length (BS Len): Bitstring length for the label range that this router is advertising per [I-D.draft-wijnands-mpls-bier-encapsulation-02]. 4 bits.

Label Range Size: Number of labels in the range used on encapsulation for this BIER sub-domain for this bitstring length, 1 octet. This MUST never be advertised as 0 (zero) and otherwise, this sub-sub-TLV must be treated as if not present for BFT calculations and a misconfiguration SHOULD be reported by the receiving router.

Label: First label of the range used on encapsulation for this BIER sub-domain for this bitstring length, 20 bits. The label is used for example by [I-D.draft-wijnands-mpls-bier-encapsulation-02] to forward traffic to sets of BFERs.

6.3. Optional BIER sub-domain Tree Type sub-sub-TLV

This sub-sub-TLV carries the information of the BIER tree type for a certain <MT,SD>. It is carried within the BIER Info sub-TLV (Section 6.1) that the router participates in as BFR. This sub-sub-TLV is optional and its absence indicates the same as its presence with Tree Type value 0 (SPF). BIER implementation following this version of the RFC SHOULD NOT advertise this TLV.

On violation of any of the following conditions, the receiving router implementing this RFC SHOULD signal a misconfiguration condition. Further results are unspecified unless described further:

o The sub-sub-TLV MUST be included once AND ONLY once.

o The advertised BIER TLV version is 0 and the value of Tree Type MUST be 0 (SPF).

```
 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      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Tree Type     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Tree Type specific opaque data|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   ~~ up to TLV Length ~~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Tree Type specific opaque data|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type: value of 1 indicating BIER Tree Type.

Length: 1 octet.

Tree Type: The only supported value today is 0 and indicates that BIER uses normal SPF computed reachability to construct BIFT. BIER implementation following this RFC MUST ignore the node for purposes of the sub-domain <MT,SD> if this field has any value except 0.

Tree type specific opaque data: Opaque data up to the length of the TLV carrying tree type specific parameters. For Tree Type 0 (SPF) no such data is included and therefore TLV Length is 1.

7. Security Considerations

Implementations must assure that malformed TLV and Sub-TLV permutations do not result in errors which cause hard protocol failures.

8. Acknowledgements

The RFC is aligned with the [I-D.draft-psenak-ospf-bier-extension-01] draft as far as the protocol mechanisms overlap.

Many thanks for comments from (in no particular order) Hannes Gredler, Ijsbrand Wijnands and Peter Psenak.

9. Normative References

[I-D.draft-ginsberg-isis-prefix-attributes]
Ginsberg et al., U., "IS-IS Prefix Attributes for Extended IP and IPv6 Reachability", internet-draft draft-ginsberg-isis-prefix-attributes-00.txt, October 2014.

[I-D.draft-psenak-ospf-bier-extension-01]
Psenak, P. and IJ. Wijnands, "OSPF Extension for Bit Index Explicit Replication", internet-draft draft-ietf-ospf-prefix-link-attr-01.txt, October 2014.

[I-D.draft-wijnands-bier-architecture-02]
Wijnands, IJ., "Stateless Multicast using Bit Index Explicit Replication Architecture", internet-draft draft-wijnands-bier-architecture-02.txt, February 2014.

[I-D.draft-wijnands-mpls-bier-encapsulation-02]
Wijnands et al., IJ., "Bit Index Explicit Replication using MPLS encapsulation", internet-draft draft-wijnands-mpls-bier-encapsulation-02.txt, February 2014.

[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, March 1997.

[RFC4971] Vasseur, JP., Shen, N., and R. Aggarwal, "Intermediate System to Intermediate System (IS-IS) Extensions for Advertising Router Information", RFC 4971, July 2007.

[RFC5120] Przygienda, T., Shen, N., and N. Sheth, "M-ISIS: Multi Topology (MT) Routing in Intermediate System to Intermediate Systems (IS-ISs)", RFC 5120, February 2008.

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

[RFC5308] Hopps, C., "Routing IPv6 with IS-IS", RFC 5308, October 2008.

[RFC6513] Rosen, E. and R. Aggarwal, "Multicast in MPLS/BGP IP VPNs", RFC 6513, February 2012.

Authors' Addresses

Tony Przygienda
Ericsson
300 Holger Way
San Jose, CA 95134
USA

Email: antoni.przygienda@ericsson.com

Les Ginsberg
Cisco Systems
510 McCarthy Blvd.
Milpitas, CA 95035
USA

Email: ginsberg@cisco.com

Sam Aldrin
Huawei
2330 Central Expressway
Santa Clara, CA 95051
USA

Email: aldrin.ietf@gmail.com

Jeffrey (Zhaohui) Zhang
Juniper Networks, Inc.
10 Technology Park Drive
Westford, MA 01886
USA

Email: zzhang@juniper.net

IS-IS for IP Internets
Internet-Draft
Intended status: Standards Track
Expires: April 27, 2015

P. Sarkar, Ed.
H. Gredler
S. Hegde
Juniper Networks, Inc.
S. Litkowski
B. Decraene
Orange
Z. Li
Huawei Technologies
H. Raghuveer

October 24, 2014

Advertising Per-node Admin Tags in IS-IS
draft-psarkar-isis-node-admin-tag-03

Abstract

This document describes an extension to IS-IS protocol [ISO10589], [RFC1195] to add an optional operational capability, that allows tagging and grouping ofthe nodes in an IS-IS domain. This allows simple management and easy control over route and path selection, based on local configured policies.

This document describes the protocol extensions to disseminate per-node administrative tags in IS-IS protocols.

Requirements Language

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

Status of This Memo

This Internet-Draft will expire on April 27, 2015.

Copyright Notice

Table of Contents

1. Introduction

This document provides mechanisms to advertise per-node administrative tags in the IS-IS Link State PDU [RFC1195]. In certain path-selection applications like for example in traffic-engineering or LFA [RFC5286] selection there is a need to tag the nodes based on their roles in the network and have policies to prefer or prune a certain group of nodes.

2. Administrative Tag

For the purpose of advertising per-node administrative tags within IS-IS, a new sub-TLV to the IS-IS Router Capability TLV-242 that is defined in [RFC4971] is proposed. Because path selection is a functional set which applies both to TE and non-TE applications the same has not been added as a new sub-TLV in the Traffic Engineering TLVs [RFC5305].

An administrative Tag is a 32-bit integer value that can be used to identify a group of nodes in the IS-IS domain. The new sub-TLV specifies one or more administrative tag values. An IS-IS router advertises the set of groups it is part of in the specific IS-IS level. As an example, all PE-nodes may be configured with certain tag value, whereas all P-nodes are configured with a different tag value in.

The new sub-TLV defined will be carried inside the IS-IS Router Capability TLV-242 (defined in [RFC4971]) in the Link State PDUs originated by the router. Link State PDUs [ISO10589] that has either level-wise (i.e. L1 or L2) or domain-wide flooding scope. Choosing the flooding scope to flood the group tags are defined by the needs of the operator's usage and is a matter of local policy or configuration.

Operator may choose to advertise a set of per-node administrative tags across levels and another set of per-node administrative tags within the specific level. But evidently the same set of per-node administrative tags cannot be advertised both across levels and within a specific level. A receiving IS-IS router will not be able to distinguish between the significance of a per-node administrative tag advertised globally from that of a administrative tag advertised locally if they have the same value associated but different significance across different scopes.

Implementations SHOULD allow configuring one or more 'global' as well as 'level-wide' administrative tags. A operator may only need to advertise and flood a specific per-node administrative tag, either across all levels, or only within a specific level. Hence implementations MUST NOT allow configuring the same per-node administrative tag values in both 'global' and 'level-wide' scopes. However the same administrative tag value MAY be allowed to be configured and advertised for multiple levels with 'level-wide' flooding scope.

The 'global' per-node administrative tags shall have significance across the entire administrative domain and hence MUST be advertised in a Router-Capability TLV with 'global' scope (i.e. S-bit set to 1), and inserted in the LSP PDUs generated for all levels applicable. The 'level-wide' administrative tags should be copied in to a Router-Capability with 'level-wide' scope only (i.e S-bit reset to 0) and copied into the LSP PDU for the specific level.

In deployments using multi-topology routing [RFC5120], since multiple topologies within same IS-IS level share the same flooding scope configuring the same per-node administrative tag across different topologies, SHOULD NOT be allowed. Advertising the same tag value across multiple topologies will lead to same inconsistencies as with the case of advertising same tag value across 'global' and 'level-wide' flooding scope. If there is need to distinguish between the per-node administrative tags used for one topology to another, operators are advised to use disjoint sets of per-node administrative tags across such topologies.

3. TLV format

3.1. Per-node Admin Tag sub-TLV

The new Per-node Administrative Tag sub-TLV, like other ISIS Capability sub-TLVs, is formatted as Type/Length/Value (TLV)triplets. Figure 1 below shows the format of the new sub-TLV.

```
 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     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Administrative Tag #1                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Administrative Tag #2                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
//                                                             //
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Administrative Tag #N                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

Type :  TBA

Length: A 8-bit field that indicates the length of the value
        portion in octets and will be a multiple of 4 octets
        dependent on the number of tags advertised.

Value:  A sequence of multiple 4 octets defining the
        administrative tags.
```

Figure 1: IS-IS Per-node Administrative Tag sub-TLV

The 'Per-node Admin Tag' sub-TLV may be generated more than once by an originating router. This MAY happen if a node carries more than 63 per-node administrative groups and a single sub-TLV does not provide sufficient space. As such occurence of the 'Per-node Admin Tag' sub-TLV does not cancel previous announcements, but rather is cumulative.

4. Elements of Procedure

Meaning of the Per-node administrative tags is generally opaque to IS-IS. Router advertising the per-node administrative tag (or tags) may be configured to do so without knowing (or even explicitly supporting) functionality implied by the tag.

Interpretation of tag values is specific to the administrative domain of a particular network operator. The meaning of a per-node administrative tag is defined by the network local policy and is controlled via the configuration. If a receiving node does not understand the tag value, it ignores the specific tag and floods the Router Capability TLV without any change as defined in [RFC4971].

The semantics of the tag order has no meaning. There is no implied meaning to the ordering of the tags that indicates a certain operation or set of operations that need to be performed based on the ordering.

Each tag SHOULD be treated as an independent identifier that MAY be used in policy to perform a policy action. Tags carried by the administrative tag TLV SHOULD be used to indicate independent characteristics of a node. The TLV SHOULD be considered as an unordered list. Whilst policies may be implemented based on the presence of multiple tags (e.g., if tag A AND tag B are present), they MUST NOT be reliant upon the order of the tags (i.e., all policies should be considered commutative operations, such that tag A preceding or following tag B does not change their outcome).

As mentioned earlier, to avoid incomplete or inconsistent interpretations of the per-node administrative tags the same tag value MUST NOT be advertised by a router in Router Capabilities of different scopes. Implementations MUST NOT allow configuring the same tag value across domain-wide and 'level-wide' scopes. The same tag value MAY be allowed to be configured and advertised under 'level-wide' scope for all levels. A IS-IS Area Border Routers (ABR) participating in both levels 1 and 2 MAY advertise the same tag value in the level-specific Router Capability TLVs with 'level-wide' scope (S-bit reset to 0) generated by it. But the same tag value MUST not be advertised in any of level 1 or level 2 Router-Capability TLV with 'global' scope (S-bit set to 1).

The per-node administrative tags are not meant to be extended by the future IS-IS standards. The new IS-IS extensions MUST NOT require use of per-node administrative tags or define well-known tag values. Per-node administrative tags are for generic use and do not require IANA registry. The future IS-IS extensions requiring well known values MAY use new Capability sub-TLVs tailored to the needs of the feature, as defined in [RFC4971].

Being part of the Router Capability TLV, the per-node administrative tag sub-TLV MUST be reasonably small and stable. In particular, but not limited to, implementations supporting the per-node administrative tags MUST NOT tie advertised tags to changes in the network topology (both within and outside the IS-IS domain) or reachability of routes.

5. Applications

This section lists several examples of how implementations might use the Per-node administrative tags. These examples are given only to demonstrate generic usefulness of the router tagging mechanism. Implementation supporting this specification is not required to implement any of the use cases. It is also worth noting that in some described use cases routers configured to advertise tags help other routers in their calculations but do not themselves implement the same functionality.

1. Auto-discovery of Services

   Router tagging may be used to automatically discover group of routers sharing a particular service.

   For example, service provider might desire to establish full mesh of MPLS TE tunnels between all PE routers in the area of MPLS VPN network. Marking all PE routers with a tag and configuring devices with a policy to create MPLS TE tunnels to all other devices advertising this tag will automate maintenance of the full mesh. When new PE router is added to the area, all other PE devices will open TE tunnels to it without the need of reconfiguring them.

2. Policy-based Fast-Reroute

   Increased deployment of Loop Free Alternates (LFA) as defined in [RFC5286] poses operation and management challenges. [I-D.ietf-rtgwg-lfa-manageability] proposes policies which, when implemented, will ease LFA operation concerns.

   One of the proposed refinements is to be able to group the nodes in IGP domain with administrative tags and engineer the LFA based on configured policies.

(a) Administrative limitation of LFA scope

Service provider access infrastructure is frequently designed in layered approach with each layer of devices serving different purposes and thus having different hardware capabilities and configured software features. When LFA repair paths are being computed, it may be desirable to exclude devices from being considered as LFA candidates based on their layer.

For example, if the access infrastructure is divided into the Access, Distribution and Core layers it may be desirable for a Distribution device to compute LFA only via Distribution or Core devices but not via Access devices. This may be due to features enabled on Access routers; due to capacity limitations or due to the security requirements. Managing such a policy via configuration of the router computing LFA is cumbersome and error prone.

With the Per-node administrative tags it is possible to assign a tag to each layer and implement LFA policy of computing LFA repair paths only via neighbors which advertise the Core or Distribution tag. This requires minimal per-node configuration and network automatically adapts when new links or routers are added.

(b) Optimizing LFA calculations

Calculation of LFA paths may require significant resources of the router. One execution of Dijkstra algorithm is required for each neighbor eligible to become next hop of repair paths. Thus a router with a few hundreds of neighbors may need to execute the algorithm hundreds of times before the best (or even valid) repair path is found. Manually excluding from the calculation neighbors which are known to provide no valid LFA (such as single-connected routers) may significantly reduce number of Dijkstra algorithm runs.

LFA calculation policy may be configured so that routers advertising certain tag value are excluded from LFA calculation even if they are otherwise suitable.

3. Controlling Remote LFA tunnel termination

[I-D.ietf-rtgwg-remote-lfa] proposed method of tunneling traffic after connected link failure to extend the basic LFA coverage and algorithm to find tunnel tail-end routers fitting LFA requirement. In most cases proposed algorithm finds more than one candidate tail-end router. In real life network it may be desirable to exclude some nodes from the list of candidates based on the local policy. This may be either due to known limitations of the per-node (the router does accept targeted LDP sessions required to implement Remote LFA tunneling) or due to administrative requirements (for example, it may be desirable to choose tail-end router among co-located devices).

The Per-node administrative tag delivers simple and scalable solution. Remote LFA can be configured with a policy to accept during the tail-end router calculation as candidates only routers advertising certain tag. Tagging routers allows to both exclude nodes not capable of serving as Remote LFA tunnel tail-ends and to define a region from which tail-end router must be selected.

4. Mobile backhaul network service deployment

The topology of mobile backhaul network usually adopts ring topology to save fiber resource and it is divided into the aggregate network and the access network. Cell Site Gateways(CSGs) connects the eNodeBs and RNC(Radio Network Controller) Site Gateways(RSGs)connects the RNCs. The mobile traffic is transported from CSGs to RSGs. The network takes a typical aggregate traffic model that more than one access rings will attach to one pair of aggregate site gateways(ASGs) and more than one aggregate rings will attach to one pair of RSGs.

```
                    ----------------
                   /                \
                  /                  \
                 /                    \
    +------+   +----+    Access     +----+
    |eNodeB|---|CSG1|    Ring 1     |ASG1|-------------
    +------+   +----+               +----+             \
                 \                    /                 \
                  \                  /                   +----+    +---+
                   \             +----+                  |RSG1|----|RNC|
                    -------------|    |    Aggregate     +----+    +---+
                                 |ASG2|      Ring          |
                    -------------|    |                  +----+    +---+
                   /             +----+                  |RSG2|----|RNC|
                  /                  \                   +----+    +---+
                 /                    \                 /
    +------+   +----+     Access     +----+            /
    |eNodeB|---|CSG2|     Ring 2     |ASG3|------------
    +------+   +----+                +----+
                \                     /
                 \                   /
                  \                 /
                   -----------------
```

Figure 2: Mobile Backhaul Network

A typical mobile backhaul network with access rings and aggregate links is shown in figure above. The mobile backhaul networks deploy traffic engineering due to the strict Service Level Agreements(SLA). The TE paths may have additional constraints to avoid passing via different access rings or to get completely disjoint backup TE paths. The mobile backhaul networks towards the access side change frequently due to the growing mobile traffic and addition of new eNodeBs. It's complex to satisfy the requirements using cost, link color or explicit path configurations. The per-node administrative tag defined in this document can be effectively used to solve the problem for mobile backhaul networks. The nodes in different rings can be assigned with specific tags. TE path computation can be enhanced to consider additional constraints based on per-node administrative tags.

5. Policy-based Explicit Routing

Partially meshed network provides multiple paths between any two nodes in the network. In a data center environment, the topology is usually highly symmetric with many/all paths having equal cost. In a long distance network, this is usually less the case for a variety of reasons (e.g. historic, fiber availability constraints, different distances between transit nodes, different roles ...). Hence between a given source and destination, a path is typically preferred over the others, while between the same source and another destination, a different path may be preferred.

```
              +--------------------+
              |                    |
              |    +----------+    |
              |    |          |    |
              T-10-T          |    |
             /|   /|          |    |
            / |  / |          |    |
         --+  | |  |          |    |
        /  +--+-+ 100         |    |
       /  /   |    |          |    |
      /  /    R-18-R          |    |
     /  /    /\   /\          |    |
    /  |    /  \ /  \         |    |
   /   |   /    x    \        |    |
  A-25-A  10  10 \    \       |    |
         /    /   10   10     |    |
        /    /     \    \     |    |
       A-25-A       A-25-A    |    |
        \    \     /    /     |    |
        201  201  201 201     |    |
          \    \ /    /       |    |
           \    x    /        |    |
            \  / \  /         |    |
             \/   \/          |    |
             I-24-I          100  100
             |    |           |    |
             |    +-----------+    |
             |                     |
             +---------------------+
```

Figure 3: Explicit Routing topology

In the above topology, operator may want to enforce the following high level explicitly routed policies: - Traffic from A nodes to A nodes must not go through I nodes - Traffic from A nodes to I nodes must not go through R and T nodes with per-node administrative tag, tag A can be configured on all A nodes, (similarly I, R, T), and then configure this single CSPF policy on all A nodes to avoid I nodes for path calculation.

6. Security Considerations

This document does not introduce any further security issues other than those discussed in [ISO10589] and [RFC1195].

7. IANA Considerations

IANA maintains the registry for the Router Capability sub-TLVs. IS-IS Administrative Tags will require new type code for the following new sub-TLV defined in this document.

i) Per-Node-Admin-Tag Sub-TLV, Type: TBD

8. Acknowledgments

Many thanks to Les Ginsberg, Dhruv Dhody, Uma Chunduri for useful inputs. Thanks to Chris Bowers for providing useful inputs to remove ambiguity related to tag-ordering.

9. References

9.1. Normative References

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

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

9.2. Informative References

[I-D.ietf-rtgwg-lfa-manageability] Litkowski, S., Decraene, B., Filsfils, C., Raza, K., Horneffer, M., and p. psarkar@juniper.net, "Operational management of Loop Free Alternates", draft-ietf-rtgwg-lfa-manageability-04 (work in progress), August 2014.

[I-D.ietf-rtgwg-remote-lfa]
Bryant, S., Filsfils, C., Previdi, S., Shand, M., and N. So, "Remote LFA FRR", draft-ietf-rtgwg-remote-lfa-08 (work in progress), September 2014.

[RFC1195] Callon, R., "Use of OSI IS-IS for routing in TCP/IP and dual environments", RFC 1195, December 1990.

[RFC4971] Vasseur, JP., Shen, N., and R. Aggarwal, "Intermediate System to Intermediate System (IS-IS) Extensions for Advertising Router Information", RFC 4971, July 2007.

[RFC5120] Przygienda, T., Shen, N., and N. Sheth, "M-ISIS: Multi Topology (MT) Routing in Intermediate System to Intermediate Systems (IS-ISs)", RFC 5120, February 2008.

[RFC5286] Atlas, A. and A. Zinin, "Basic Specification for IP Fast Reroute: Loop-Free Alternates", RFC 5286, September 2008.

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

Authors' Addresses

Pushpasis Sarkar (editor)
Juniper Networks, Inc.
Electra, Exora Business Park
Bangalore, KA 560103
India

Email: psarkar@juniper.net

Hannes Gredler
Juniper Networks, Inc.
1194 N. Mathilda Ave.
Sunnyvale, CA 94089
US

Email: hannes@juniper.net

Shraddha Hegde
Juniper Networks, Inc.
Electra, Exora Business Park
Bangalore, KA 560103
India

Email: shraddha@juniper.net

Stephane Litkowski
Orange

Email: stephane.litkowski@orange.com

Bruno Decraene
Orange

Email: bruno.decraene@orange.com

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

Email: lizhenbin@huawei.com

Harish Raghuveer

Email: harish.r.prabhu@gmail.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: March 29, 2015

X. Xu
Huawei
U. Chunduri
Ericsson

September 25, 2014

Carrying Routable IP Addresses in IS-IS Router Capability TLV
draft-xu-isis-routable-ip-address-01

Abstract

This document proposes two new sub-TLVs of the IS-IS Router CAPABILITY TLV, called routable IPv4 address sub-TLV and routable IPv6 address sub-TLV respectively.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

There are several situations where it is useful for IS-IS routers within a given area to identify the routable IP address of IS-IS routers within the same area. In deployments, multiple IP addresses can be hosted by an IS-IS router for a particular purpose and all these addresses cannot be represented through the currently defined TE Router ID TLVs [RFC5305] [RFC6119]. For example, a private address range used for PQ nodes, as specified in Section 12 of [I-D.ietf-rtgwg-remote-lfa], can be easily represented through the sub-TLVs defined in this document without having to parse through all TE reachability TLVs [RFC5305] [RFC5120] where a receiving node can not identify if the prefix is an external prefix, an inter-area leaked prefix or a locally attached prefix. Similarly, in the case where a topology database is learned by an orchestrator or controller [I-D.ietf-i2rs-problem-statement], an application can easily determine the router reachability from a private address range (specifically for this purpose), advertised through the sub-TLVs defined in this document.

Meanwhile, there are also several situations where it is required for IS-IS routers in one area to find correlations between routable IP addresses and capabilities of IS-IS routers in another area. One example is the Entropy Label Capability (ELC) advertisement [I-D.xu-isis-mpls-elc] across the IS-IS domain. In this example, assume the ELC TLV originated by a router in one area is propagated to another area, those routers in the latter area need to find routable IP addresses of the router originating that ELC TLV before inserting the Entropy Label (EL) for packets going to the Label Switch Path (LSP) tunnel towards one of the above routable IP addresses. Another example is the S-BFD discriminator distribution [I-D.ginsberg-isis-sbfd-discriminator] across the IS-IS domain. In this example, assume the S-BFD Discriminator sub-TLV originated by a router in one area is propagated to another area, those routers in the latter area need to find routable IP addresses of the router originating that S-BFD Discriminator sub-TLV so as to set up S-BDF sessions with that originating router.

However, in the IS-IS Router CAPABILITY TLV as defined in [RFC4971], which is applicable for both IPv4 IS-IS [RFC1195] [RFC5305] and IPv6 MT IS-IS [RFC5120], there is no such field for containing the routable IP address. Although TE Router ID sub-TLVs defined in [RFC5316] can be used to carry routable IP addresses, TE Router ID sub-TLVs are specifically designed for TE purpose. Therefore, this document propose two new sub-TLVs of this CAPABILITY TLV to carry a routable IPv4 and IPv6 address of the router originating the CAPABILITY TLV respectively. These two sub-TLVs could be used for non-TE purpose. A router MUST NOT advertise the sub-TLVs defined in this document if the same has been advertised through TE Router ID sub-TLVs.

1.1. Requirements Language

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

2. Terminology

This memo makes use of the terms defined in [RFC4971].

3. Routable IPv4 Address Sub-TLV

A new sub-TLV of the IS-IS Router Capability TLV, called Routable IPv4 Address sub-TLV is defined to carry one or more routable /32 IPv4 address of the router originating the CAPABILITY TLV. The Type of this sub-TLV is TBD, the Length is variable (multiple of 4), and the Value field contains one or more routable IPv4 address of the router originating the CAPABILITY TLV. For every IPv4 address of this type, a corresponding IP reachability TLV [RFC5305] or MT IP reachability TLV [RFC5120] MUST be included in its LSP. An implementation receiving a Routable IPv4 Address sub-TLV defined in this document MUST NOT consider these /32 reachable prefixes in the standard SPF calculation because this can lead to forwarding loops when interacting with systems that do not support this TLV.

4. Routable IPv6 Address Sub-TLV

A new sub-TLV of the IS-IS Router Capability TLV, called Routable IPv6 Address sub-TLV is defined to carry one or more routable /128 IPv6 global address of the router originating the CAPABILITY TLV.

The Type of this sub-TLV is TBD, the Length is variable (multiple of 16), and the Value field contains one or more routable IPv6 global address of the router originating the CAPABILITY TLV. For every IPv6 address of this type,a corresponding IPv6 reachability TLV [RFC5308] orMT IPv6 reachability TLV [RFC5120] MUST be included in its LSP. An implementation receiving a Routable IPv6 Address sub-TLV defined in this document MUST NOT consider these /128 reachable prefixes in the standard SPF calculation because this can lead to forwarding loops when interacting with systems that do not support this TLV.

5. Acknowledgements

Thanks Karsten Thomann, Anton Smirnov, Joel Jaeggli, Joel M. Halpern, Wes George and Acee Lindem for their valuable comments on the initial idea of this draft.

6. IANA Considerations

This memo includes a request to IANA to allocate two sub-TLV type codes within the IS-IS Router Capability TLV for the Routable IPv4 Address Sub-TLV and the Routable IPv6 Address Sub-TLV respectively.

7. Security Considerations

This document does not introduce any new security risk.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC4971] Vasseur, JP., Shen, N., and R. Aggarwal, "Intermediate System to Intermediate System (IS-IS) Extensions for Advertising Router Information", RFC 4971, July 2007.

8.2. Informative References

[I-D.ginsberg-isis-sbfd-discriminator] Ginsberg, L., Akiya, N., and M. Chen, "Advertising S-BFD Discriminators in IS-IS", draft-ginsberg-isis-sbfd-discriminator-00 (work in progress), May 2014.

[I-D.ietf-i2rs-problem-statement] Atlas, A., Nadeau, T., and D. Ward, "Interface to the Routing System Problem Statement", draft-ietf-i2rs-problem-statement-04 (work in progress), June 2014.

[I-D.ietf-rtgwg-remote-lfa]
Bryant, S., Filsfils, C., Previdi, S., Shand, M., and S. Ning, "Remote LFA FRR", draft-ietf-rtgwg-remote-lfa-06 (work in progress), May 2014.

[I-D.xu-isis-mpls-elc]
Xu, X., Kini, S., Sivabalan, S., and C. Filsfils, "Signaling Entropy Label Capability Using IS-IS", draft-xu-isis-mpls-elc-00 (work in progress), December 2013.

[RFC1195] Callon, R., "Use of OSI IS-IS for routing in TCP/IP and dual environments", RFC 1195, December 1990.

[RFC5120] Przygienda, T., Shen, N., and N. Sheth, "M-ISIS: Multi Topology (MT) Routing in Intermediate System to Intermediate Systems (IS-ISs)", RFC 5120, February 2008.

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

[RFC5308] Hopps, C., "Routing IPv6 with IS-IS", RFC 5308, October 2008.

[RFC5316] Chen, M., Zhang, R., and X. Duan, "ISIS Extensions in Support of Inter-Autonomous System (AS) MPLS and GMPLS Traffic Engineering", RFC 5316, December 2008.

[RFC6119] Harrison, J., Berger, J., and M. Bartlett, "IPv6 Traffic Engineering in IS-IS", RFC 6119, February 2011.

Authors' Addresses

Xiaohu Xu
Huawei

Email: xuxiaohu@huawei.com

Uma Chunduri
Ericsson

Email: uma.chunduri@ericsson.com