IS-IS for IP Internets   J. Farkas, Ed.
Internet-Draft   Ericsson
Intended status: Standards Track   N. Bragg
Expires: June 19, 2016   Ciena
                                                            P. Unbehagen
                                                                   Avaya
                                                             G. Parsons
                                                                Ericsson
                                                        P. Ashwood-Smith
                                                     Huawei Technologies
                                                               C. Bowers
                                                         Juniper Networks
                                                       December 17, 2015

                    IS-IS Path Computation and Reservation
                           draft-ietf-isis-pcr-04

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.

Status of This Memo

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]:

   o SPB Link Metric sub-TLV

   o SPB Base VLAN-Identifiers sub-TLV

   o SPB Instance sub-TLV

   o SPBV MAC address sub-TLV

   o 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 directed graph (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 Edge Bridges and the paths among the Edge Bridges. If only
      the Edge Bridges 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 the two nodes at either end, 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

   Explicit trees may be determined in some fashion. For example, an
   explicit tree may be determined by a Path Computation Element (PCE)
   [RFC4655]. 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. If a PCE is

used, it 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
remainder of this section, and subsequent sections, are written
assuming PCE use.

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. an Edge Bridge, and no path or path segment is

specified between the bridges, which are therefore loose hops even if
Edge Bridges are adjacent neighbors. The special role of a hop can
be: Edge Bridge, 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.

The owner PCE can withdraw an explicit tree by sending an updated LSP
that does not include the Topology sub-TLV (Section 6.1). If a
Topology sub-TLV is removed from an LSP (or has been changed), so

that (previous) Topology sub-TLV is no longer present (or has been
changed) in the LSDB, then that (previous) Topology sub-TLV is
implicitly withdrawn. ISIS-PCR then removes (or updates) the
explicit tree.

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:

o Administrative Group (color, resource class) (sub-TLV type 3),

o Maximum Link Bandwidth (sub-TLV type 9),

o Maximum Reservable Link bandwidth (sub-TLV type 10),

o Unreserved Bandwidth (sub-TLV type 11),

o 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]:

o Unidirectional Link Delay (sub-TLV type 33),

o Min/Max Unidirectional Link Delay (sub-TLV type 34),

o Unidirectional Delay Variation (sub-TLV type 35),

o Unidirectional Link Loss (sub-TLV type 36),

o Unidirectional Residual Bandwidth (sub-TLV type 37),

o Unidirectional Available Bandwidth (sub-TLV type 38),

o Unidirectional Utilized Bandwidth (sub-TLV type 39).

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 Edge Bridges 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 Edge Bridge. 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.

The SPB Link Metric sub-TLV [RFC6329] specifies the metric of each link for IS-IS PCR if the LT, the LTS, the MRT, or the MRTG ECT Algorithm is used. If the SPB Link Metric values advertised by different ends of an adjacency are different, then the maximum value MUST be used.

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) |   (2 bytes if present)
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
          ................
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | Res  | Base VID n (12 bits) |   (2 bytes if present)
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                     sub-TLV 1  (variable)                     |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
                     ...............
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                     sub-TLV m  (variable)                     |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
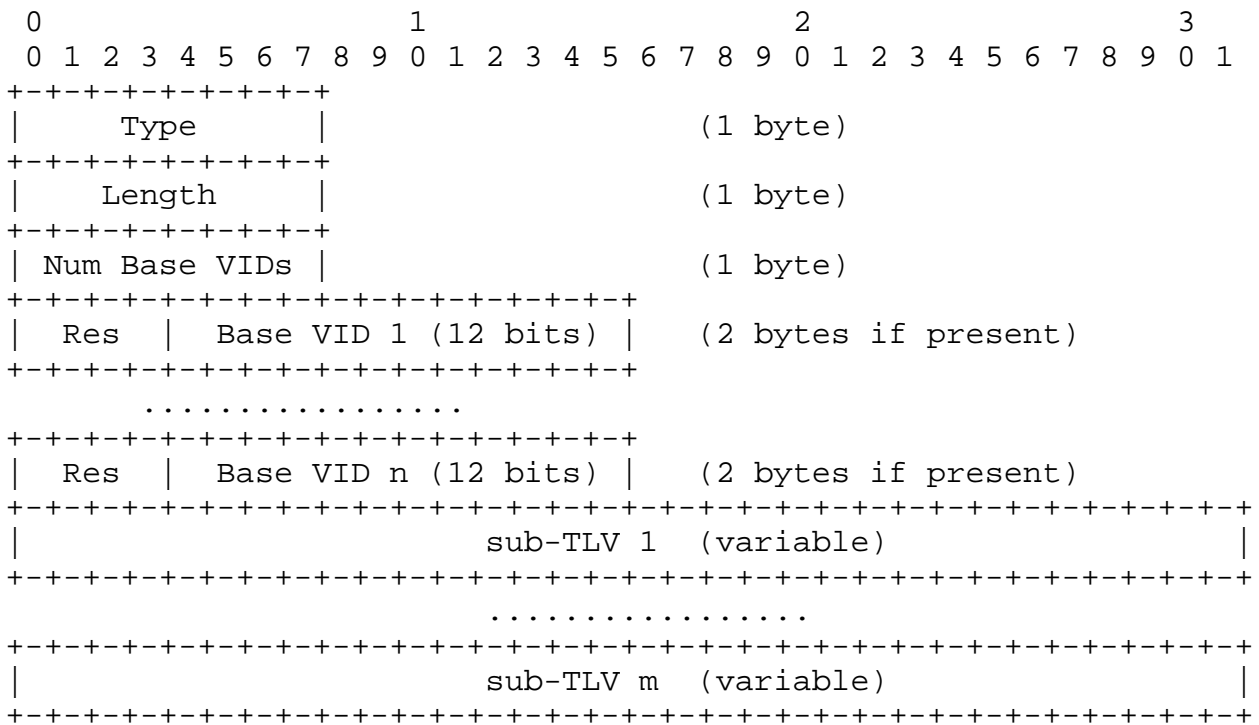
                      Figure 1: Topology sub-TLV

   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 21.

   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 MUST be set to 0 on
        sending and the value MUST be ignored on reception.

   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     |
                                        +----------+
```
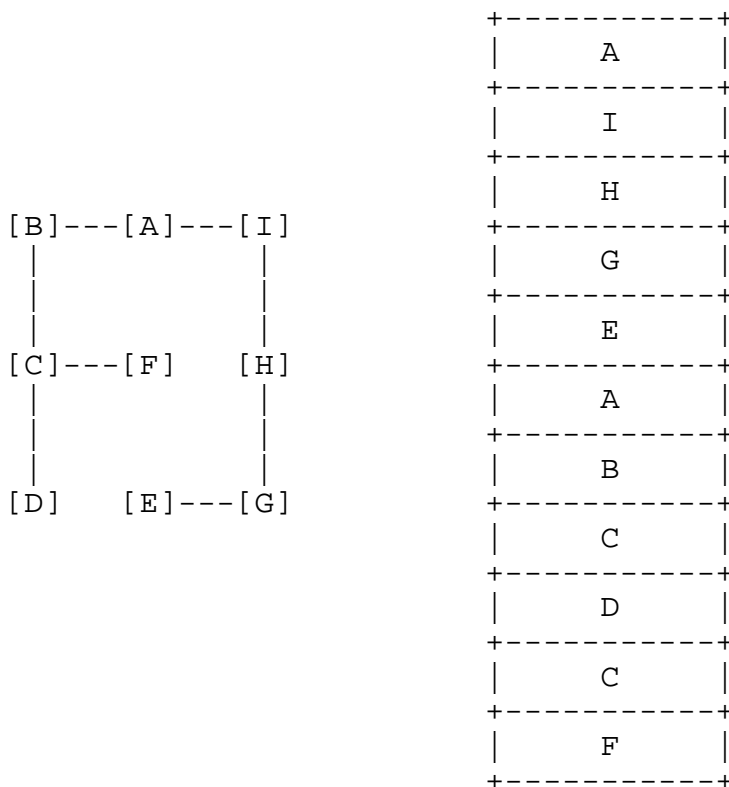
        Figure 2: A strict tree and its description; root = Node A

    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 Edge Bridges or bridges to
    be excluded. The BLCEs compute the loose trees, e.g. MRTs, such
    that they span the Edge Bridges and are rooted at an Edge Bridge.

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 Edge Bridge of an explicit tree MUST be always specified in the
Topology sub-TLV by the inclusion of the Hop sub-TLVs corresponding
to the Edge Bridges. The Edge Bridges of a tree are identified by
setting the Edge Bridge 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.

IS-IS PCR MAY be used for recording bandwidth assignment. In that
case, the Topology sub-TLV conveys Bandwidth Assignment sub-TLV
(Section 6.4) and it MAY also convey Timestamp sub-TLV (Section 6.5).
If assignment of the bandwidth indicated by the Bandwidth Assignment
sub-TLV of the Topology sub-TLV would result in overbooking any link
of the explicit tree, 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 an Edge Bridge. 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 Edge Bridges 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 Edge Bridges 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|B|R|L|E|Res|                        (1 byte)
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                           System ID                          |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |           System ID           |        (6 bytes)
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |              Extended Local Circuit ID  (4 bytes if present)  |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |  Num of VIDs  |                        (1 byte if present)
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |T|R|Res|     VID 1    (12 bits) |        (2 bytes if present)
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
            .................
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |T|R|Res|     VID n    (12 bits) |        (2 bytes if present)
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                        Delay Constraint                      |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |         Delay Constraint      |        (6 bytes if present)
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

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 22.

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. Edge Bridge (B) flag (1 bit): The Edge Bridge flag is a one-
          bit flag to indicate whether or not the given System is an
          Edge Bridge, i.e. transmitter and/or receiver. If the System
          is an Edge Bridge, then the Edge Bridge flag MUST be set.
          The Edge Bridge flag indicates that FDB entries have to be
          installed for the given hop as specified by the SPBV MAC
          address sub-TLV or SPBM Service Identifier and Unicast
          Address sub-TLV of the 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
          Edge Bridge 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 MUST be set to 0
    on sending and the value MUST be ignored on reception.

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 an Edge
    Bridge 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 MUST be set to 0
    on sending and the value MUST be ignored on reception.

4. VID (12 bits): A VID.

h. Delay Constraint (48 bits): A Hop sub-TLV MAY specify a delay
    constraint. The Length of the Hop sub-TLV indicates whether or
    not a delay constraint is present because the Length of a Hop
    sub-TLV conveying a delay constraint is six bytes greater than it
    would be without the delay constraint. The last six bytes then
    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-
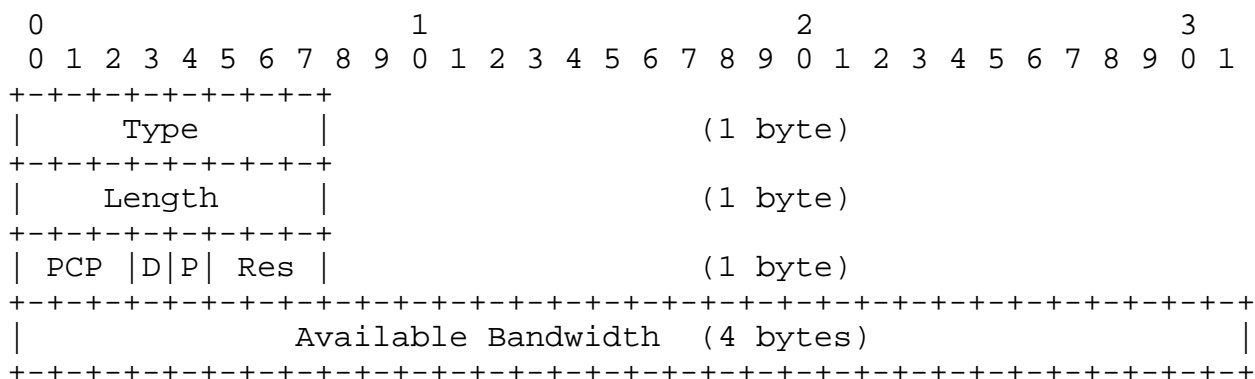   TLV.

```
      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 23.

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 MUST be set to 0 on
    sending and the value MUST be ignored on reception.

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 Edge Bridges. 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 and IS-IS does not. 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)                 |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
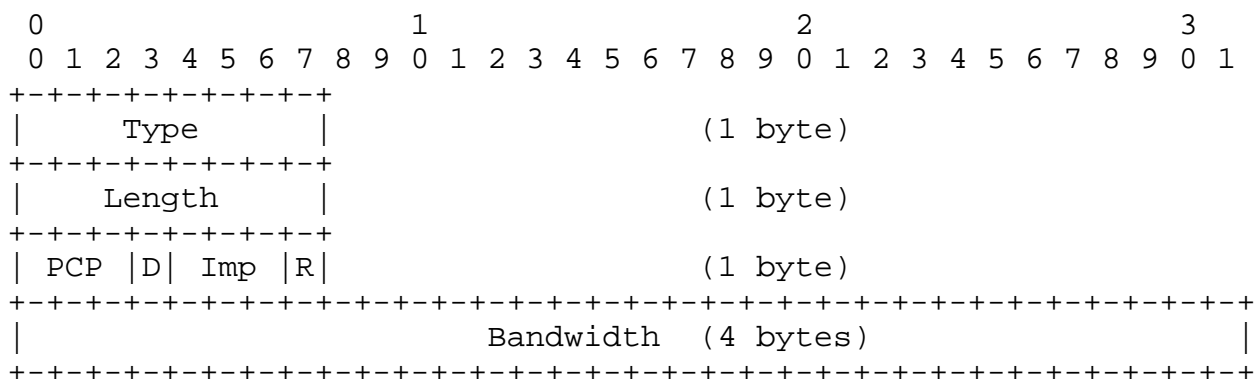
                  Figure 5: Bandwidth Assignment sub-TLV

   The parameters of the bandwidth assignment are encoded as follows:

   a. Type (8 bits): The type of the sub-TLV, its value is 24.

   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 MUST be set to 0 on
       sending and the value MUST be ignored on reception.

   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 when PCP and Importance of different bandwidth assignments are both equal. If resolution is not possible based on the above parameters or they are not available, e.g. each bandwidth assignment lacks timestamp or the precedence order has to be determined for the use of a VID, 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

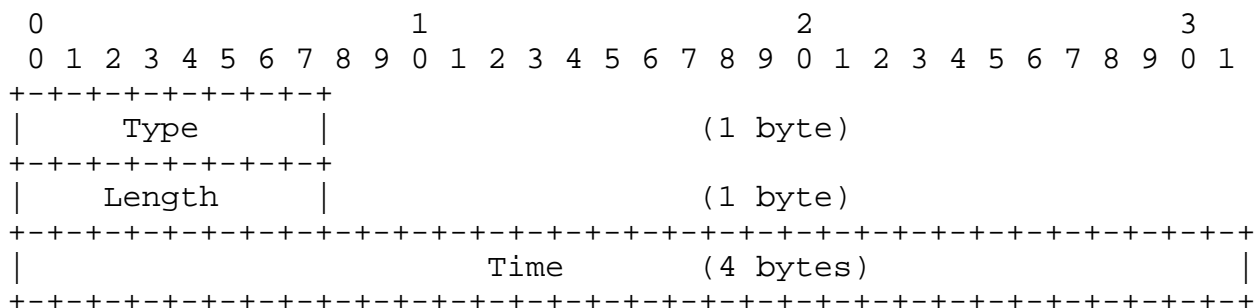The timestamp represents a positive time with respect to the Precision Time Protocol (PTP) epoch and it is encoded as follows:

a. Type (8 bits): The type of the sub-TLV, its value is 25.

   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 Edge Bridges 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 |
                                     +-----------+---+
```
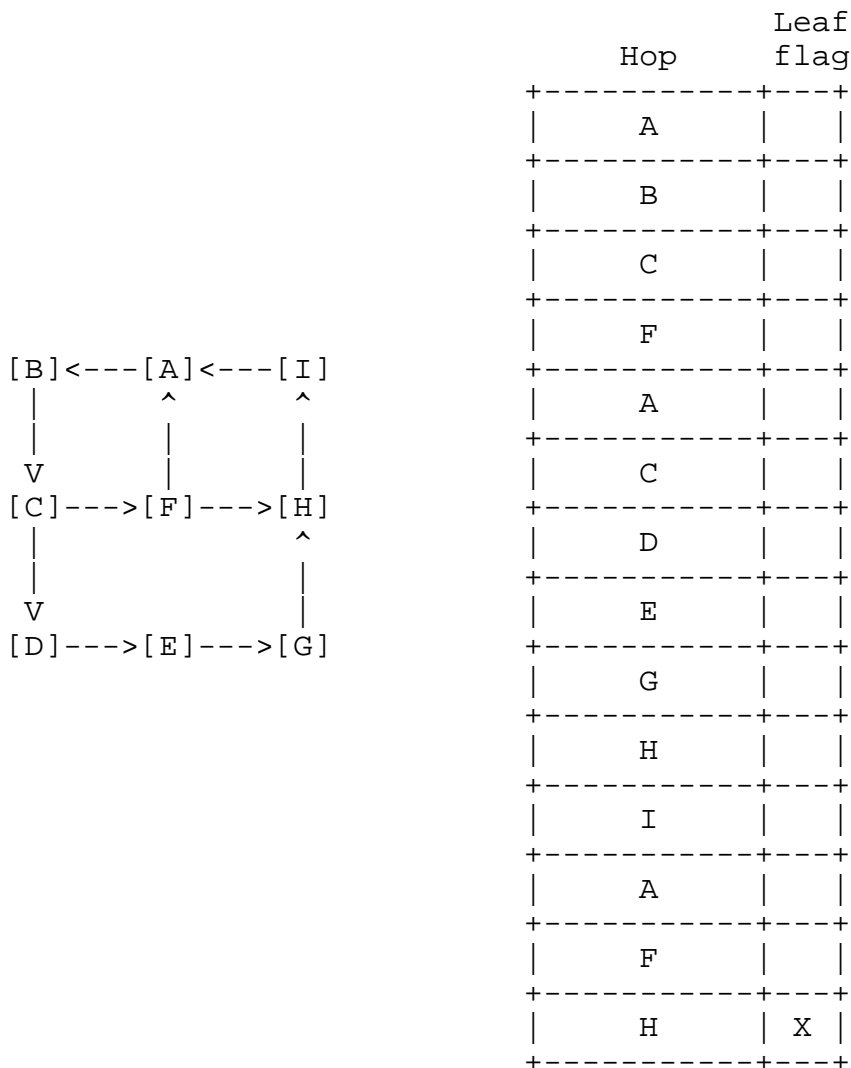
     Figure 7: A GADAG and its description; GADAG Root = Node A

   A topology can comprise multiple blocks, like the one illustrated in
   Figure 8(a). This example topology comprises 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. A GADAG for this topology is
   shown in Figure 8(b). Note that two arcs with opposite direction
   represent a cut-link in a GADAG, see e.g. the cut-link between D and
   G. 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 |
                                 +------------+---+
     +-+   +-+           +-+      |     D      |   |
     |F|<-|E|         +--|K|      +------------+---+
     +-+   +-+         |  +-+     |     G      |   |
      |     ^          |   ^      +------------+---+
      |     |          V   |      |     D      | X |
      V    +-+  +-+  +-+    |      +------------+---+
     +-+   | |->| |->| |    |      |     G      |   |
     |A|   |D|  |G|  |H|    |      +------------+---+
     +-+   | |<-| |<-| |    |      |     H      |   |
      |    +-+  +-+  +-+    |      +------------+---+
      |     ^          |    |      |     G      | X |
      V     |          |    |      +------------+---+
     +-+   +-+         |   +-+     |     H      |   |
     |B|->|C|         +->|J|       +------------+---+
     +-+   +-+           +-+       |     J      |   |
                                  +------------+---+
           (b) GADAG              |     K      |   |
                                  +------------+---+
                                  |     H      | X |
                                  +------------+---+

                                   (c) GADAG descriptor
```
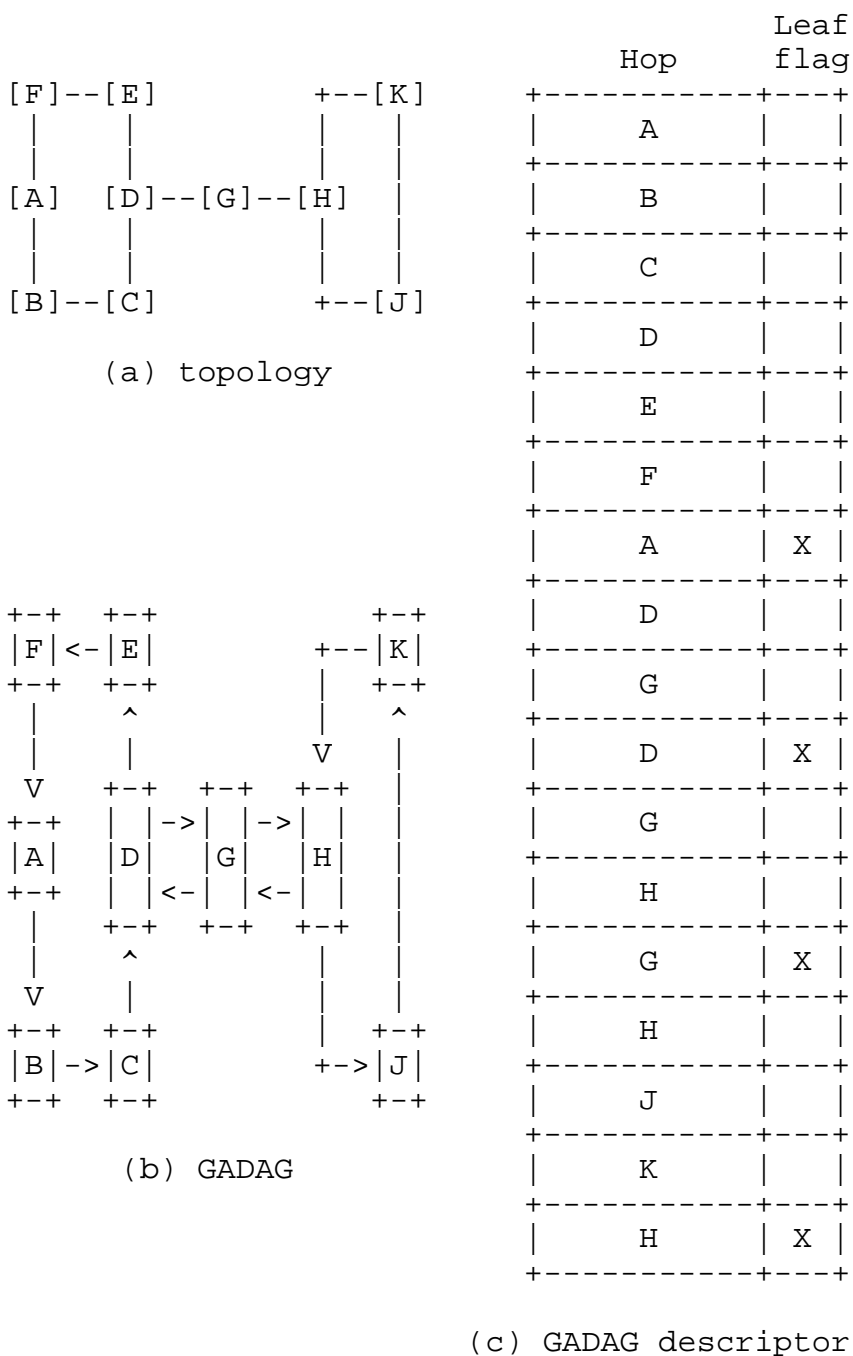
      Figure 8: A GADAG with cut-links and its description; GADAG Root =
                                  Node A

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

   This document defines the following IS-IS sub-TLVs within the MT-Capability TLV (type 144). They are listed in the IS-IS TLV codepoint registry.

Type	Description	Length
21	Topology sub-TLV	variable
22	Hop sub-TLV	variable
23	Bandwidth Constraint sub-TLV	5
24	Bandwidth Assignment sub-TLV	5
25	Timestamp sub-TLV	4

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.

   Any mechanism that chooses forwarding paths, and allocates resources to those paths, is potentially vulnerable to attack. The security considerations section of [RFC4655] describes the risks associated with the use of PCE for this purpose and should be referred to. Use of any other means to determine paths should only be used after considering similar concerns.

   Because the mechanism assumed for distributing tree information relies on IS-IS routing, IS-IS routing security considerations (Section 6, [RFC1195]) and mechanisms (e.g. [RFC5310]) used to authenticate peer advertisements apply.

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-07 (work in progress), June 2015.

[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-06 (work in progress), October 2015.

[IEEE8021Qca]
          IEEE 802.1, "IEEE 802.1Qca Bridges and Bridged Networks -
          Amendment: Path Control and Reservation - Draft 2.1",
          (work in progress), June 24, 2015,
          <http://www.ieee802.org/1/pages/802.1ca.html>.

[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>.

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

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

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

[RFC6329] Fedyk, D., Ed., Ashwood-Smith, P., Ed., Allan, D., Bragg,
          A., and P. Unbehagen, "IS-IS Extensions Supporting IEEE
          802.1aq Shortest Path Bridging", RFC 6329,
          DOI 10.17487/RFC6329, April 2012,
          <http://www.rfc-editor.org/info/rfc6329>.

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-01 (work in progress), July 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-07 (work in progress),
             October 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>.

   [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>.

   [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>.

   [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>.

   [RFC4655] Farrel, A., Vasseur, J., and J. Ash, "A Path Computation
             Element (PCE)-Based Architecture", RFC 4655,
             DOI 10.17487/RFC4655, August 2006,
             <http://www.rfc-editor.org/info/rfc4655>.

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

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