BESS   Z. Zhang
Internet-Draft   Juniper Networks
Intended status: Standards Track   R. Raszuk
Expires: 2 July 2024   Arrcus
                                                            D. Pacella
                                                               Verizon
                                                              A. Gulko
                                          Edward Jones Wealth Management
                                                      30 December 2023

                    Controller Based BGP Multicast Signaling
                  draft-ietf-bess-bgp-multicast-controller-12

Abstract

   This document specifies a way that one or more centralized
   controllers can use BGP to set up multicast distribution trees
   (identified by either IP source/destination address pair, or mLDP
   FEC) in a network. Since the controllers calculate the trees, they
   can use sophisticated algorithms and constraints to achieve traffic
   engineering. The controllers directly signal dynamic replication
   state to tree nodes, leading to very simple multicast control plane
   on the tree nodes, as if they were using static routes. This can be
   used for both underlay and overlay multicast trees, including
   replacing BGP-MVPN signaling.

Requirements Language

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

Status of This Memo

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

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

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

   This Internet-Draft will expire on 2 July 2024.

Copyright Notice

Table of Contents

1. Overview

1.1. Terminology

   Some terminologies are originally introduced in [RFC6514]. They are
   reused in [I-D.ietf-bess-bgp-multicast] and in this document.

      PMSI [RFC6514]: Provider Multicast Service Interface - a
      conceptual interface for a PE to send customer multicast traffic
      to all or some PEs in the same VPN.

      I-PMSI: Inclusive PMSI - to all PEs in the same VPN.

      S-PMSI: Selective PMSI - to some of the PEs in the same VPN.

      I-PMSI A-D Route: Inclusive PMSI Auto-Discovery route used to
      advertise the tunnels that instantiate an I-PMSI.

      S-PMSI A-D route: Selective PMSI Auto-Discovery route used to
      advertise that particular C-flows are bound to (i.e., are
      traveling through) particular P-tunnels.

      Leaf A-D route: Leaf Auto-Discovery route used to advertise leaf/
      receiver information.

      PMSI Tunnel attribute (PTA): A BGP attribute used to identify a
      particular P-tunnel.

1.2. Introduction

   [I-D.ietf-bess-bgp-multicast] describes a way to use BGP as a
   replacement signaling for Protocol Independent Multicast (PIM)
   [RFC7761] or Label Distribution Protocol Extensions for P2MP and
   MP2MP Label Switched Paths (mLDP) [RFC6388]. The BGP-based multicast
   signaling described there provides a mechanism for setting up both
   (s,g)/(*,g) multicast trees (as PIM does, but optionally with labels)
   and labeled (MPLS) multicast tunnels (as mLDP does). Each router on
   a tree performs essentially the same procedures as it would perform
   if using PIM or mLDP, but all the inter-router signaling is done
   using BGP.

   These procedures allow the routers to set up a separate tree for each
   individual multicast (x,g) flow where the 'x' could be either 's' or
   '*', but they also allow the routers to set up trees that are used
   for more than one flow. In the latter case, the trees are often
   referred to as "multicast tunnels" or "multipoint tunnels", and
   specifically in this document they are mLDP tunnels (except that they
   are set up with BGP signaling). While it actually does not have to
   be restricted to mLDP tunnels, mLDP Forwarding Equivalent Class (FEC)
   [RFC6388] is conveniently borrowed to identify the tunnel. In the
   rest of the document, the term tree and tunnel are used
   interchangeably.

   The trees/tunnels are set up using the "receiver-initiated join"
   technique of PIM/mLDP, hop by hop from downstream routers towards the
   root but using BGP NLRIs of MCAST-TREE SAFI, either sent hop by hop
   between downstream routers and their upstream neighbors, or reflected
   by Route Reflectors (RRs).

   As an alternative to each hop independently determining its upstream
   router and signaling upstream towards the root (following PIM/mLDP
   model), the entire tree can be calculated by a centralized
   controller, and the signaling can be entirely done from the
   controller using the same MCAST-TREE SAFI. For that, some additional
   procedures and optimizations are specified in this document.

   [I-D.ietf-bess-bgp-multicast] uses some terminologies introduced in
   BGP-MVPN [RFC6514] because the main procedures and concepts are
   borrowed from there. While the same Leaf A-D routes in
   [I-D.ietf-bess-bgp-multicast] can be used to signal replication state
   to tree nodes from controllers, this document introduces a new route
   type "Replication State" for the same functionality so that
   familiarity with the BGP-MVPN concepts is not required.

While it is outside the scope of this document, signaling from the controllers could be done via other means as well, like Netconf or any other SDN methods.

1.3. Resilience

Each router could establish direct BGP sessions with one or more controllers, or it could establish BGP sessions with RRs who in turn peer with controllers. For the same tree/tunnel, each controller may independently calculate the tree/tunnel and signal the routers on the tree/tunnel using MCAST-TREE Replication State routes. How the calculations are done is outside the scope of this document.

On each router, BGP route selection rules will lead to one controller's route for the tree/tunnel being selected as the active route and used for setting up forwarding state. As long as all the routers on a tree/tunnel consistently pick the same controller's routes for the tree/tunnel, the setup should be consistent. If the tree/tunnel is labeled, different labels will be used from different controllers so there is no traffic loop issue even if the routers do not consistently select the same controller's routes. In the unlabeled case, to ensure the consistency the selection SHOULD be solely based on the identifier of the controller.

Another consistency issue is when a bidirectional tree/tunnel needs to be re-routed. Because this is no longer triggered hop-by-hop from downstream to upstream, it is possible that the upstream change happens before the downstream, causing a traffic loop. In the unlabeled case, there is no good solution (other than that the controller issues upstream change only after it gets acknowledgement from downstream). In the labeled case, as long as a new label is used there should be no problem.

Besides the traffic loop issue, there could be transient traffic loss before both the upstream and downstream's forwarding state are updated. This could be mitigated if the upstream keep sending traffic on the old path (in addition to the new path) and the downstream keep accepting traffic on the old path (but not on the new path) for some time. When the downstream switches to the new path is a local matter - it could be data driven (e.g., after traffic arrives on the new path) or timer driven.

For each tree, multiple disjoint instances could be calculated and
signaled for live-live protection. Different labels are used for
different instances, so that the leaves can differentiate incoming
traffic on different instances. As far as transit routers are
concerned, the instances are just independent. Note that the two
instances are not expected to share common transit routers (it is
otherwise outside the scope of this document/revision).

1.4. Signaling

When a router receives a Replication State route, the re-
advertisement is blocked if a configured import RT matches the RT of
the route, which indicates that this router is the target and
consumer of the route hence it should not be re-advertised further.
The routes includes the forwarding information in the form of Tunnel
Encapsulation Attributes (TEA) [RFC9012], with enhancements specified
in this document.

Suppose that for a particular tree, there are two downstream routers
D1 and D2 for a particular upstream router U. A controller C sends
one Replication State route to U, with the Tree Node's IP Address
field (see Section 3.4) set to U's IP address and the TEA specifying
both the two downstreams and its upstream (see Section 3.1.5). In
this case, the Originating Router's Address field of the Replication
State route is set to the controller's address. Note that for a TEA
attached to a unicast NLRI, only one of the tunnels in a TEA is used
for forwarding a particular packet, while all the tunnels in a TEA
are used to reach multiple endpoints when it is attached to a
multicast NLRI.

It could be that U may need to replicate to many downstream routers,
say D1 through D1000. In that case, it may not be possible to encode
all those branches in a single TEA, or may not be optimal to update a
large TEA when a branch is added/removed. In that case, C may send
multiple Replication State routes, each with a different RD and a
different TEA that encodes a subset of the branches. This provides a
flexible way to optimize the encoding of large number of branches and
incremental updates of branches.

Notice that, in the case of labeled trees, the (x,g) or mLDP FEC
signaling is actually not needed to transit routers but only needed
to tunnel root/leaves. However, for consistency among the root/leaf/
transit nodes, and for consistency with the hop-by-hop signaling, the
same signaling (with tree identification encoded in the NLRI) is used
to all routers.

Nonetheless, a new NLRI route type of the MCAST-TREE SAFI is defined
to encode label/SID instead of tree identification in the NLRI, for
scenarios where there is really no need to signal tree
identification, e.g. as described in Section 2. On a tunnel root,
the tree's binding SID can be encoded in the NLRI.

For a tree node to acknowledge to the controller that it has received
the signaling and installed corresponding forwarding state, it
advertises a corresponding Replication State route, with the
Originating Router's IP Address set to itself and with a Route Target
to match the controller. For comparison, the tree signaling
Replication State route from the controller has the Originating
Router's IP Address set to the controller and the Route Target
matching the tree node. The two Replication State routes (for
controller to signal to a tree node and for a tree node to
acknowledge back) differ only in those two aspects.

With the acknowledgement Replication State routes, the controller
knows if tree setup is complete. The information can be used for
many purposes, e.g. the controller may instruct the ingress to start
forwarding traffic onto a tree only after it knows that the tree
setup has completed.

1.5. Label Allocation

In the case of labeled multicast signaled hop by hop towards the
root, whether it's (x,g) multicast or "mLDP" tunnel, labels are
assigned by a downstream router and advertised to its upstream router
(from traffic direction point of view). In the case of controller
based signaling, routers do not originate tree join routes anymore,
so the controllers have to assign labels on behalf of routers, and
there are three options for label assignment:

* From each router's SRLB that the controller learns

* From the common SRGB that the controller learns

* From the controller's local label space

Assignment from each router's SRLB is no different from each router
assigning labels from its own local label space in the hop-by-hop
signaling case. The assignments for one router is independent of
assignments for another router, even for the same tree.

Assignment from the controller's local label space is upstream-
assigned [RFC5331]. It is used if the controller does not learn the
common SRGB or each router's SRLB. Assignment from the SRGB
[RFC8402] is only meaningful if all SRGBs are the same and a single

common label is used for all the routers on a tree in the case of
unidirectional tree/tunnel (Section 1.5.1). Otherwise, assignment
from SRLB is preferred.

The choice of which of the options to use depends on many factors.
An operator may want to use a single common label per tree for ease
of monitoring and debugging, but that requires explicit RPF checking
and either common SRGB or upstream assigned labels, which may not be
supported due to either the software or hardware limitations (e.g.
label imposition/disposition limits). In an SR network, the
assignment from the common SRGB is used if it's required to use a
single common label per unidirectional tree; otherwise, the
assignment from SRLB is a good choice because it does not require
support for context label spaces.

1.5.1. Using a Common per-tree Label for All Routers

MPLS labels only have local significance. For an LSP that goes
through a series of routers, each router allocates a label
independently and it swaps the incoming label (that it advertised to
its upstream) to an outgoing label (that it received from its
downstream) when it forwards a labeled packet. Even if the incoming
and outgoing labels happen to be the same on a particular router,
that is just incidental.

With Segment Routing, it is becoming a common practice that all
routers use the same SRGB so that a SID maps to the same label on all
routers. This makes it easier for operators to monitor and debug
their network. The same concept applies to multicast trees as well –
a common per-tree label can be used for a router to receive traffic
from its upstream neighbor and replicate traffic to all its
downstream neighbor.

However, a common per-tree label can only be used for unidirectional
trees. In the case of bidirectional trees, the common label needs to
be per– <tree, direction>. Additionally, unless the entire tree is
updated for every tree node to use a new common per-tree or
per–<tree, direction> label with any change in the tree (no matter
how small and local the change is), it requires each router to do
explicit RPF check, so that only packets from its expected upstream
neighbor are accepted. Otherwise, traffic loop may form during
topology changes, because the forwarding state update is no longer
ordered.

Traditionally, p2mp mpls forwarding does not require explicit RPF
check as a downstream router advertises a label only to its upstream
router and all traffic with that incoming label is presumed to be
from the upstream router and accepted. When a downstream router

switches to a different upstream router a different label will be
advertised, so it can determine if traffic is from its expected
upstream neighbor purely based on the label. Now with a single
common label used for all routers on a tree to send and receive
traffic with, a router can no longer determine if the traffic is from
its expected neighbor just based on that common tree label.
Therefore, explicit RPF check is needed. Instead of interface based
RPF checking as in PIM case, neighbor based RPF checking is used - a
label identifying the upstream neighbor precedes the common tree
label and the receiving router checks if that preceding neighbor
label matches its expected upstream neighbor. Notice that this is
similar to what's described in Section "9.1.1 Discarding Packets from
Wrong PE" of [RFC6513] (an egress PE discards traffic sent from a
wrong ingress PE). The only difference is one is used for label
based forwarding and the other is used for (s,g) based forwarding..

Both the common per-tree label and the neighbor label are allocated
either from the common SRGB or from the controller's local label
space. In the latter case, an additional label identifying the
controller's label space is needed, as described in the following
section.

1.5.2. Upstream-assignment from Controller's Local Label Space

In this case, in the multicast packet's label stack, the tree label
and upstream neighbor label (if used in the case of single common-
label per tree) are preceded by a downstream-assigned "context
label". The context label identifies a context-specific label space
(the controller's local label space), and the upstream-assigned label
that follows it is looked up in that space.

This specification requires that, in the case of upstream-assignment
from a controller's local label space, each router D to assign,
corresponding to each controller C, a context label that identifies
the upstream-assigned label space used by that controller. This
label, call it Lc-D, is communicated by D to C via BGP-LS [RFC7752].

Suppose a controller is setting up unidirectional tree T. It assigns
that tree the label Lt, and assigns label Lu to identify router U
which is the upstream of router D on tree T. C needs to tell U: "to
send a packet on the given tree/tunnel, one of the things you have to
do is push Lt onto the packet's label stack, then push Lu, then push
Lc-D onto the packet's label stack, then unicast the packet to D".
Controller C also needs to inform router D of the correspondence
between <Lc-D, Lu, Lt> and tree T.

To achieve that, when C sends a Replication State route, for each
tunnel in the TEA, it may include a label stack Sub-TLV [RFC9012],
with the outer label being the context label Lc-D (received by the
controller from the corresponding downstream), the next label being
the upstream neighbor label Lu, and the inner label being the label
Lt assigned by the controller for the tree. The router receiving the
route will use the label stacks to send traffic to its downstreams.

For C to signal the expected label stack for D to receive traffic
with, we overload a tunnel TLV in the TEA of the Replication State
route sent to D - if the tunnel TLV has a RPF sub-TLV
(Section 3.1.5), then it indicates that this is actually for
receiving traffic from the upstream.

1.6. Determining Root/Leaves

   For the controller to calculate a tree, it needs to determine the
   root and leaves of the tree. This may be based on provisioning
   (static or dynamically programmed), or based on BGP signaling as
   described in the following two sections.

   In both of the following cases, the BGP updates are targeted at the
   controller, via an address specific Route Target with Global
   Administration Field set to the controller's address and the Local
   Administration Field set to 0.

1.6.1. PIM-SSM/Bidir or mLDP

   In this case, the PIM Last Hop Routers (LHRs) with interested
   receivers or mLDP tunnel leaves encode a Leaf A-D route
   ([I-D.ietf-bess-bgp-multicast]) with the Upstream Router's IP Address
   field set to the controller's address and the Originating Router's IP
   Address set to the address of the LHR or the P2MP tunnel leaf. The
   encoded PIM SSM source or mLDP FEC provides root information and the
   Originating Router's IP Address provides leaf information.

1.6.2. PIM ASM

   In this case, the First Hop Routers (FHRs) originate Source Active
   routes which provides root information, and the LHRs originate Leaf
   A-D routes, encoded as in the PIM-SSM case except that it is (*,G)
   instead of (S,G). The Leaf A-D routes provide leaf information.

1.7. Multiple Domains

   An end to end multicast tree may span multiple routing domains, and
   the setup of the tree in each domain may be done differently as
   specified in [I-D.ietf-bess-bgp-multicast]. This section discusses a
   few aspects specific to controller signaling.

   Consider two adjacent domains each with its own controller in the
   following configuration where router B is an upstream node of C for a
   multicast tree:

```
                          |
             domain 1     |   domain 2
                          |
              ctrlr1      |     ctrlr2
               /\         |      /\
              /  \        |     /  \
             /    \       |    /    \
           A--...-B--|--C--...-D
                          |
```
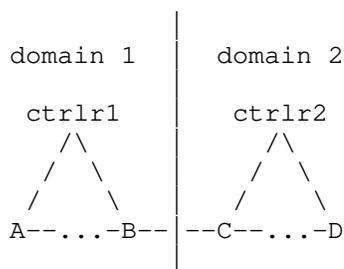
   In the case of native (un-labeled) IP multicast, nothing special is
   needed. Controller 1 signals B to send traffic out of B-C link while
   Controller 2 signals C to accept traffic on the B-C link.

   In the case of labeled IP multicast or mLDP tunnel, the controllers
   may be able to coordinate their actions such that Controller 1
   signals B to send traffic out of B-C link with label X while
   Controller 2 signals C to accept traffic with the same label X on the
   B-C link. If the coordination is not possible, then C needs to use
   hop-by-hop BGP signaling to signal towards B, as specified in
   [I-D.ietf-bess-bgp-multicast].

   The configuration could also be as follows, where router B borders
   both domain 1 and domain 2 and is controlled by both controllers:

```
                        |
            domain 1    | domain 2
                        |
             ctrlr1     | ctrlr2
              /\        |   /\
             /  \       |  /  \
            /    \      | /    \
           /      \|/      \
         A--...---B--...---C
                   |
```
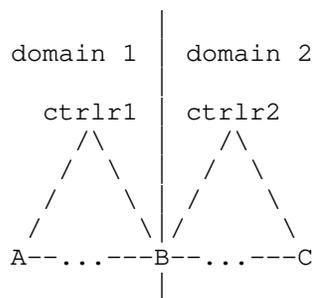
As discussed in Section 1.3, when B receives signaling from both
Controller 1 and Controller 2, only one of the routes would be
selected as the best route and used for programming the forwarding
state of the corresponding segment. For B to stitch the two segments
together, it is expected for B to know by provisioning that it is a
border router so that B will look for the other segment (represented
by the signaling from the other controller) and stitch the two
together.

2. Alternative to BGP-MVPN

Multicast with BGP signaling from controllers can be an alternative
to BGP-MVPN [RFC6514]. It is an attractive option especially when
the controller can easily determine the source and leaf information.

With BGP-MVPN, distributed signaling is used for the following:

* Egress PEs advertise C-multicast (Type-6/7) Auto-Discovery (A-D)
   routes to join C-multicast trees at the overlay (PE-PE).

* In the case of ASM, ingress PEs advertise Source Active (Type-5)
   A-D routes to advertise sources so that egress PEs can establish
   Shortest Path Trees (SPT).

* PEs advertise I/S-PMSI (Type-1/2/3) A-D routes to advertise the
   binding of overlay/customer traffic to underlay/provider tunnels.
   For some types of tunnels, Leaf (Type-4) A-D routes are advertised
   by egress PEs in response to I/S-PMSI A-D routes so that the
   ingress PE can discover the leaves.

Based on the above signaled information, an ingress PE builds
forwarding state to forward traffic arriving on the PE-CE interface
to the provider tunnel (and local interfaces if there are local
downstream receivers), and an egress PE builds forwarding state to
forward traffic arriving on a provider tunnel to local interfaces
with downstream receivers.

Notice that multicast with BGP signaling from controllers essentially
programs "static" forwarding state onto multicast tree nodes. As
long as a controller can determine how a C-multicast flow should be
forwarded on ingress/egress PEs, it can signal to the ingress/egress
PEs using the procedures in this document to set up forwarding state,
removing the need of the above-mentioned distributed signaling and
processing.

For the controller to learn the egress PEs for a C-multicast tree (so
that it can set up or find a corresponding provider tunnel), the
egress PEs advertise MCAST-TREE Leaf A-D routes (Section 1.6.1)

towards the controller to signal its desire to join C-multicast
trees, each with an appropriate RD and an extended community derived
from the Route Target for the VPN
([I-D.ietf-idr-rt-derived-community]) so that the controller knows
which VPN it is for. The controller then advertises corresponding
MCAST-TREE Replication State routes to set up C-multicast forwarding
state on ingress and egress PEs. To encode the provider tunnel
information in the MCAST-TREE Replication State route for an ingress
PE, the TEA can explicitly list all replication branches of the
tunnel, or just the binding SID for the provider tunnel in the form
of Segment List tunnel type, if the tunnel has a binding SID.

The Replication State route may also include a PMSI Tunnel Attribute
(PTA) attached to specify the provider tunnel, while the TEA
specifies the local PE-CE interfaces where traffic need to be
directed. This not only allows provider tunnel without a binding SID
(e.g., in a non-SR network) to be specified without explicitly
listing its replication branches, but also allows the service
controller for MVPN overlay state to be independent of provider
tunnel setup (which could be from a different transport controller or
even without a controller).

However, notice that if the service controller and transport
controller are different, then the service controller needs to signal
the transport controller the tree information: identification, set of
leaves, and applicable constraints. While this can be achieved (see
Section 1.6.1), it is easier for the service and transport controller
to be the same.

Depending on local policy, a PE may add PE-CE interfaces to its
replication state based on local signaling (e.g., IGMP/PIM) instead
of completely relying on signaling from controllers.

If there is a need for dynamic switching between inclusive and
selective tunnels based on data rate, the ingress PE can advertise or
withdraw S-PMSI routes targeted only at the controllers, without
attaching a PMSI Tunnel Attribute. The controller then updates
relevant MCAST-TREE Replication State routes to update C-multicast
forwarding states on PEs to switch to a new tunnel.

3. Specification

3.1. Enhancements to TEA

   A TEA can encode a list of tunnels. A TEA attached to an MCAST-TREE
   NLRI encodes replication information for a <tree, node > that is
   identified by the NLRI. Each tunnel in the TEA identifies a branch –
   either an upstream branch towards the tree root (Section 3.1.5) or a
   downstream branch towards some leaves. A tunnel in the TEA could
   have an outer encapsulation (e.g. MPLS label stack) or it could just
   be a one-hop direct connection for native IP multicast forwarding
   without any outer encapsulation.

   This document specifies three new Tunnel Types and four new sub-TLVs.
   The type codes will be assigned by IANA from the "BGP Tunnel
   Encapsulation Attribute Tunnel Types".

3.1.1. Any-Encapsulation Tunnel

   When a multicast packet needs to be sent from an upstream node to a
   downstream node, it may not matter how it is sent – natively when the
   two nodes are directly connected or tunneled otherwise. In the case
   of tunneling, it may not matter what kind of tunnel is used – MPLS,
   GRE, IPinIP, or whatever.

   To support this, an "Any-Encapsulation" tunnel type of value 20 is
   defined. This tunnel MAY have a Tunnel Egress Endpoint and other
   Sub-TLVs. The Tunnel Egress Endpoint Sub-TLV specifies an IP
   address, which could be any of the following:

   * An interface's local address – when a packet needs to be sent out
      of the corresponding interface natively. On a LAN multicast MAC
      address MUST be used.

   * A directly connected neighbor's interface address – when a packet
      needs to unicast to the address natively.

   * An address that is not directly connected – when a packet needs to
      be tunneled to the address (any tunnel type/instance can be used).

3.1.2. Load-balancing Tunnel

   Consider that a multicast packet needs to be sent to a downstream
   node, which could be reached via four paths P1~P4. If it does not
   matter which of path is taken, an "Any-Encapsulation" tunnel with the
   Tunnel Egress Endpoint Sub-TLV specifying the downstream node's
   loopback address works well. If the controller wants to specify that
   only P1~P2 should be used, then a "Load-balancing" tunnel needs to be
   used, listing P1 and P2 as member tunnels of the "Load-balancing"
   tunnel.

A load-balancing tunnel is of type TBD1. It has one "Member Tunnels"
Sub-TLV of type TBD3 defined in this document. The Sub-TLV is a list
of tunnels, each specifying a way to reach the downstream. A packet
will be sent out of one of the tunnels listed in the Member Tunnels
Sub-TLV of the load-balancing tunnel.

3.1.3. Segment List Tunnel

A Segment List tunnel is of type TBD2. It has a Segment List sub-
TLV. The encoding of the sub-TLV is as specified in Section 2.4.4 of
[I-D.ietf-idr-segment-routing-te-policy].

3.1.4. Receiving MPLS Label Stack

While [I-D.ietf-bess-bgp-multicast] uses S-PMSI A-D routes to signal
forwarding information for MP2MP upstream traffic, when controller
signaling is used, a single Replication State route is used for both
upstream and downstream traffic. Since different upstream and
downstream labels need to be used, a new "Receiving MPLS Label Stack"
of type TBD5 is added as a tunnel sub-TLV in addition to the existing
MPLS Label Stack sub-TLV. Other than type difference, the two are
encoded the same way.

The Receiving MPLS Label Stack sub-TLV is added to each downstream
tunnel in the TEA of Replication State route for an MP2MP tunnel to
specify the forwarding information for upstream traffic from the
corresponding downstream node. A label stack instead of a single
label is used because of the need for neighbor based RPF check, as
further explained in the following section.

The Receiving MPLS Label Stack sub-TLV is also used for downstream
traffic from the upstream for both P2MP and MP2MP, as specified
below.

3.1.5. RPF Sub-TLV

The RPF sub-TLV is of type 124 and has a one-octet length. The
length is 0 currently, but if necessary in the future, sub-sub-TLVs
could be placed in its value part. If the RPF sub-TLV appears in a
tunnel, it indicates that the "tunnel" is for the upstream node
instead of a downstream node.

In the case of MPLS, the tunnel contains an Receiving MPLS Label
Stack sub-TLV for downstream traffic from the upstream node, and in
the case of MP2MP it also contains a regular MPLS Label Stack sub-TLV
for upstream traffic to the upstream node.

The inner most label in the Receiving MPLS Label Stack is the
incoming label identifying the tree (for comparison the inner most
label for a regular MPLS Label Stack is the outgoing label). If the
Receiving MPLS Label Stack sub-TLVe has more than one labels, the
second inner most label in the stack identifies the expected upstream
neighbor and explicit RPF checking needs to be set up for the tree
label accordingly.

3.1.6. Tree Label Stack sub-TLV

The MPLS Label Stack sub-TLV can be utilized to specify the complete
label stack used to transmit traffic, with the stack including both a
transport label (stack) and label(s) that identify the (tree,
neighbor) to the downstream node. There are cases where the
controller only wants to specify the tree-identifying labels but
leave the transport details to the router itself. For example, the
router could locally determine a transport label (stack) and combine
with the tree-identifying labels signaled from the controller to get
the complete outgoing label stack.

For that purpose, a new Tree Label Stack sub-TLV of type 125 is
defined, with a one-octet length field. It MAY appear in an Any-
Encapsulation tunnel. The value field contains a label stack with
the same encoding as the value field of the MPLS Label Stack sub-TLV,
but with a different type. A stack is specified because it may take
up to three labels (see Section 1.5):

* If different nodes use different labels (allocated from the common
   SRGB or the node's SRLB) for a (tree, neighbor) tuple, only a
   single label is in the stack. This is similar to current mLDP hop
   by hop signaling case.

* If different nodes use the same tree label, then an additional
   neighbor-identifying label is needed in front of the tree label.

* For the previous bullet, if the neighbor-identifying label is
   allocated from the controller's local label space, then an
   additional context label is needed in front of the neighbor label.

3.1.7. Backup Tunnel sub-TLV

The Backup Tunnel sub-TLV is used to specify the backup paths for an
Any-Encapsulation or Segment List tunnel. The type is TBD4. The
length is two-octet. The value field encodes a one-octet flags field
and a variable length Tunnel Encapsulation Attribute. If the tunnel
goes down, traffic that is normally sent out of the tunnel is fast
rerouted to the tunnels listed in the encoded TEA.

```
              +- - - - - - - - - - - - - - - - +
              │ Sub-TLV Type (1 Octet, TBD4)   │
              +- - - - - - - - - - - - - - - - +
              │ Sub-TLV Length (2 Octets)      │
              +- - - - - - - - - - - - - - - - +
              │ P │ rest of 1 Octet Flags      │
              +- - - - - - - - - - - - - - - - +
              │ Backup TEA (variable length)   │
              +- - - - - - - - - - - - - - - - +
```
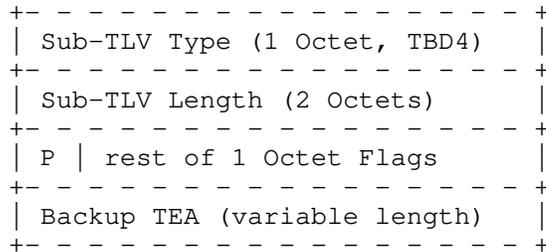
The backup tunnels can lead to the same or different nodes reached by
the original tunnel.

If the tunnel carries a RPF sub-TLV and a Backup Tunnel sub-TLV, then
both traffic arriving on the original tunnel and on the tunnels
encoded in the Backup Tunnel sub-TLV's TEA can be accepted, provided
that the Parallel (P-)bit in the flags field is set. If the P-bit is
not set, then traffic arriving on the backup tunnel is accepted only
if router has switched to receiving on the backup tunnel (this is the
equivalent of PIM/mLDP MoFRR).

3.2. Context Label TLV in BGP-LS Node Attribute

For a router to signal the context label that it assigns for a
controller (or any label allocator that assigns labels – from its
local label space – that will be received by this router), a new BGP-
LS Node Attribute TLV of type TBD6 is defined:

```
 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 TBD6          |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Context Label                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             IPv4/v6 Address of Label Space Owner             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
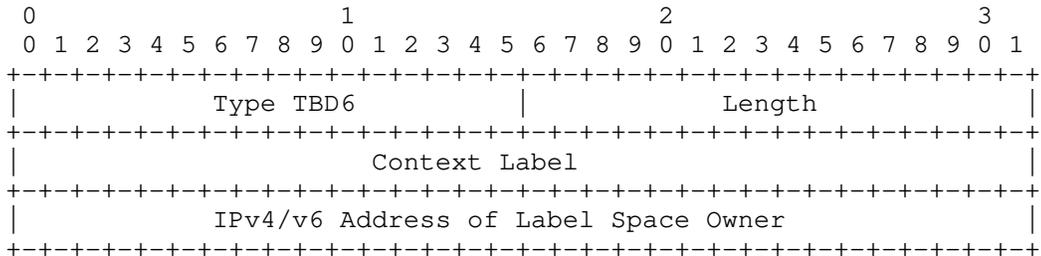
The Length field implies the type of the address. Multiple Context
Label TLVs may be included in a Node Attribute, one for each label
space owner.

An as example, a controller with address 198.51.100.1 allocates label
200 from its own label space, and router A assigns label 100 to
identify this controller's label space. The router includes the
Context Label TLV (100, 198.51.100.1) in its BGP-LS Node Attribute
and the controller instructs router B to send traffic to router A
with a label stack (100, 200), and router A uses label 100 to
determine the Label FIB in which to look up label 200.

3.3. MCAST Extended Community

A tree node needs to acknowledge to the controller the success/
failure in installing forwarding state for a tree. In case of
failure, an MCAST NACK Extended Community is attached. The value
field is set to 0. In the future, flag bits may be defined to signal
specific failure.

The MCAST NACK Extended Community is an MCAST Extended Community with
a sub-type TBD9 to be assigned by IANA. The MCAST Extended Community
is a new Extended Community with a type TBD8 to be assigned by IANA.

```
    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=TBD8     | Sub-Type=TBD9 |           Reserved=0          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                          Reserved=0                           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
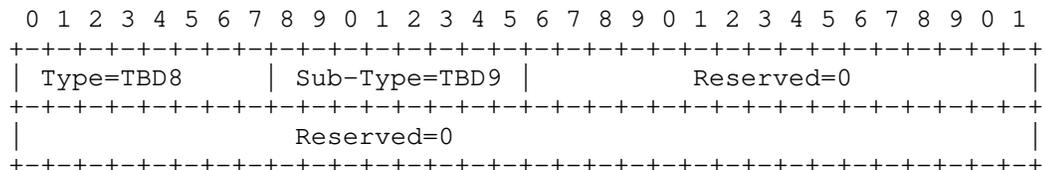
3.4. Replication State Route Type

The NLRI route type (TBD7) for signaling from controllers to tree
nodes is "Replication State". The NLRI has the following format:

```
                +-------------------------------+
                |Route Type - Replication State TBD7|
                +-------------------------------+
                |      Length (1 octet)         |
                +-------------------------------+
                |     Tree Type (1 octet)       |
                +-------------------------------+
                |Tree Type Specific Length (1 octet)|
                +-------------------------------+
                |        RD (8 octets)          |
                +-------------------------------+
                ~  Tree Identification (variable)  ~
                +-------------------------------+
                |     Tree Node's IP Address    |
                +-------------------------------+
                |    Originator's IP Address    |
                +-------------------------------+
```

                      Replication State NLRI

   Notice that Replication State is just a new route type with the same
   format of Leaf A-D route except some fields are renamed:

   * Tree Type in Replication State route matches the PMSI route type
      in Leaf A-D route

   * Tree Node's IP Address matches the Upstream Router's IP Address of
      the PMSI route key in Leaf A-D route

   With this arrangement, IP multicast tree and mLDP tunnel can be
   signaled via Replication State routes from controllers, or via Leaf
   A-D routes either hop by hop or from controllers with maximum code
   reuse, while newer types of trees can be signaled via Replication
   State routes with maximum code reuse as well.

3.5. Replication State Route with Label Stack for Tree Identification

   As described in Section 1.4, the tree label, instead of tree
   identification could be encoded in the NLRI to identify the tree in
   the control plane as well as in the forwarding plane. For that a new
   Tree Type of 2 is used and the Replication State route has the
   following format:

```
+-----------------------------------+
|     Route Type - Replication State |
+-----------------------------------+
|     Length (1 octet)              |
+-----------------------------------+
|     Tree Type 2 (Label as Tree ID) |
+-----------------------------------+
|Tree Type specific Length (1 octet) |
+-----------------------------------+
|     RD    (8 octets)              |
+-----------------------------------+
~     Label Stack (variable)        ~
+-----------------------------------+
|   Tree Node's IP Address          |
+-----------------------------------+
|   Originating Router's IP Address |
+-----------------------------------+
```
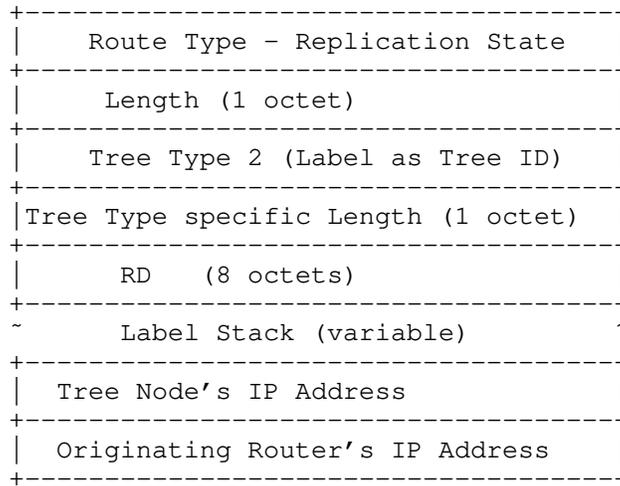
        Replication State route for tree identification by label stack

   As discussed in Section 1.5.2, a label stack may have to be used to
   identify a tree in the data plane so a label stack is encoded here.
   The number of labels is derived from the Tree Type Specific Length
   field. Each label stack entry is encoded 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
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                   Label                   |0 0 0 0 0 0 0 0 0 0 0 0|
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
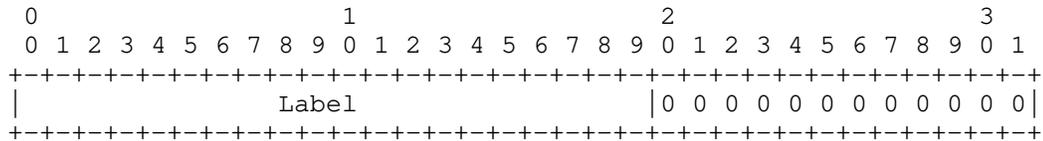
4. Procedures

   This section applies to MPLS data plane. While the concept of BGP
   signaling applies to SRv6 data plane as well, SRv6 related
   specification is outside the scope of this document. Note that, this
   document does not assume Segment Routing is used, even though the
   SRGB/SRLB terminologies are used to describe label blocks, and some
   scenarios of Segment routing are considered.

4.1. Label Space and Tree Label Allocation

   In the case of labeled trees for either (x, g) IP multicast or mLDP
   tunnels, an operator first determines which of the following methods
   is used to allocate tree-identifying labels, as explained in
   Section 1.5:

1. A common per-tree label on all nodes of a P2MP tree, or a common
    per-<tree, direction> label on all nodes of a MP2MP tree,
    allocated from the controller's own label space.

2. A common per-tree label on all nodes of a P2MP tree, or a common
    per-<tree, direction> label on all nodes of a MP2MP tree,
    allocated from a common SRGB.

3. Uncorrelated labels independently allocated from each node's
    SRLB.

For option 2 and 3, the process through which the controller learns
the common SRGB or each node's SRLB is outside the scope of this
document.

For option 1, each tree node MUST advertise a label from its default
label space to identify the controller's label space, via the Context
Label TLV in BGP-LS Node Attribute (Section 3.2). The tree-
identifying label in TEA and packets MUST be preceded by the label-
space-identifying label.

For option 1 and 2, the operator also determines if the controller
allocates a new label for each tree or <tree, direction> and resignal
to all tree nodes even when only some tree nodes need to be changed.
If not, then another neighbor-identifying label needs to precede the
tree-identifying label (and follow the label-space-identifying label
in the case of option 1). The neighbor-identifying label MUST be
allocated from the same label space or SRGB from which the tree-
identifying label is allocated.

To generalize, a label stack can contain one label (for option 3),
two labels (for option 2 and 1 if neighbor-identifying label is not
needed), or three labels (for option 2 and 1 if neighbor-identifying
label is needed). In the rest of the document, tree-identifying
label(-stack) term is used generically.

4.2. Advertising Replication State Routes

   After the controller calculates a tree, it constructs one or more
   Replication State Routes for each tree node as follows:

   * If the tree is for the default routing instance and only one route
      is needed, the RD MAY be set to 0:0. Otherwise, the RD is set to
      a value to distinguish the routes for trees in different routing
      instances but with the same tree identifier (e.g., (x, g) or mLDP
      FEC for a VPN), or to distinguish the multiple routes needed for
      the same <tree, node>.

* The Route Type, Length, Tree Type, Tree Type Specific Length, and
  Tree Identification are set accordingly.

* The Tree Node's IP Address is set to an address of the tree node,
  typically the loopback address.

* The Originator's IP Address is set to the controller's address.

* An IP Address Specific Route Target is attached, with the Global
  Administration Field set to match the Tree Node's IP Address in
  the NLRI, and the Local Admin Field set to 0.

* In the case of VPN, an Extended Community derived from the Route
  Target for the VPN ([I-D.ietf-idr-rt-derived-community]) is
  attached.

* A Tunnel Encapsulation Attribute (TEA) is attached to encode the
  replication information, as detailed below.

The TEA encompasses one or more tunnels. If the route is for the
root node, a single tunnel of type Any-Encapsulation MAY be included
with a RPF sub-TLV and a Receiving MPLS Label Stack sub-TLV to encode
a binding SID, if it is assigned for the tree. If the route is for a
leaf or bud node (which is both a leaf node and transit node
simultaneously), a single tunnel of type Any-Encapsulation MUST be
included with a Tunnel Egress Endpoint sub-TLV. The address of this
sub-TLV MUST be set to a loopback address on the node.

Additionally, for any node, a tunnel is included for each upstream/
downstream node. Each tunnel MUST include a Tunnel Egress Endpoint
sub-TLV if it is needed to derive forwarding information. Otherwise,
the sub-TLV MAY be included for informational purposes.

* For any neighbor from which labeled traffic may be received for
  the tree (notice that on a bidirectional tree traffic may be
  received on multiple branches), the tunnel MUST include a
  Receiving MPLS Label Stack sub-TLV to encode the incoming tree-
  identifying label(-stack).

* For any neighbor from which unlabeled IP multicast traffic may be
  received for the IP multicast tree (notice that on a bidirectional
  tree traffic may be received on multiple branches), the tunnel
  MUST be of type Any-Encapsulation with a Tunnel Egress Endpoint
  sub-TLV with the address set to the local address of incoming
  interface.

* A tunnel MAY be one of the following types:

- Any-Encapsulation: any encapsulation can be used to send or receive traffic. it MUST include a Tunnel Egress Endpoint Sub-TLV, with the address set to either a local address of incoming/outgoing interface, or the address of a neighbor to which outgoing traffic is sent. If traffic is to be sent with a tree-identifying label(-stack), it MUST include a Tree Label Stack sub-TLV.

- MPLS, MPLS in GRE, MPLS in UDP: like the Any-Encapsulation case, except that specifically MPLS (native or in GRE/UDP) tunneling is used.

- Segment List: This is for sending traffic via an explicit SR path represented by the segment list encoded in the tunnel. The first segment of the list MUST be a Prefix/Adjacent/Binding SID that enables the node to send replicated packet towards the downstream node. A Tunnel Egress Endpoint sub-TLV MAY be included but only for informational purpose and not used for deriving forwarding information. If tree-identifying label(-stack) is needed, a Tree Label Stack sub-TLV MAY be included, or the label(-stack) MAY be encoded as the last one or two or three segments.

- Load-balancing: This is for a downstream node to be reached via one of the member tunnels listed in a Load-balancing tunnel.

Other tunnel types and sub-TLVs may also be used but not specified here.

4.3. Receiving Replication State Routes

Each potential tree node MUST be (auto-)configured with an IP Address Specific Route Target to import Replication State Routes targeted at itself. The Global Administration Field MUST be set to a local address known to be used by the controller to identify the node, and the Local Administration Field MUST set to 0.

When a BGP speaker receives a Replication State Route and the attached Route Target matches its (auto-)configured Route Target to import the route, it MUST stops re-advertising the route further. Otherwise, normal BGP route propagation rules apply.

If an imported Replication State Route carries an Extended Community derived from a Route Target for a local VRF, the route is imported into that VRF. Otherwise, it is imported into the default routing instance.

For the same <tree, node>, there may be multiple routes imported,
from the same or different controllers. BGP best route selection
process selects one of them as the active path. Without considering
the RD field, all routes with the same NLRI as the active path MUST
be considered together to create forwarding state on the node for the
tree. Recall that multiple such routes may be advertised when it is
desired to signal a large set of replication branches via multiple
routes.

The forwarding state includes two parts – the "nexthop" part that has
the replication branches and the "route" part (with the key being a
label or (x,g) tuple), just like how a unicast IP route points to a
nexthop in a RIB/FIB.

4.3.1. Compiling Replication Branches

The receiving node goes through the tunnels in the TEAs in all the
relevant routes as described above, and build the nexthop as a
collection of replication branches.

* If a tunnel has a RPF sub-TLV and the tree is unidirectional, it
   is skipped.

* If a tunnel is of type Segment List, the replication branch is
   constructed from the Segment List sub-TLV and optionally a Tree
   Label Stack sub-TLV if that is included. Even for an (x,g) IP
   multicast tree, the segment list may be used to identify both the
   tunnel to reach the node and/or tree-identifying label(-stack).
   For example, an incoming IP multicast packet can be replicated out
   of some branches as native IP packets and some other branches with
   label stacks. Those label stacks may just forward/tunnel the
   packets to the downstream/upstream node, or may include tree-
   identifying label(-stack) to allow the receiving node to forward
   based on incoming label(-stack) instead of (x,g) prefix.

* If a tunnel is of type Any-Encapsulation, it must have a Tunnel
   Egress Endpoint sub-TLV.

   – If the egress endpoint address is a local interface address,
      the interface is the replication branch. The interface could
      be a loopback, indicating that traffic needs to be delivered
      locally off the tree, e.g.:

      o To an application running on the node, or,

      o To be further routed in a VRF, e.g., when this tree is a
         provider tunnel for MVPN.

- Otherwise, the forwarding state for the replication branch is constructed as "pushing tree-identifying labels in the Tree Label Stack sub-TLV if it is present, and then pushing any encapsulation that can be used to reach the node as encoded in the Tunnel Egress Endpoint sub-TLV".

* If a tunnel is of type MPLS, MPLS in GRE or MPLS in UDP, it is similar to the Any-Encapsulation case. The difference is that MPLS, MPLS in GRE or MPLS in UDP MUST be used to reach the downstream node.

* If a tunnel is of type Load-Balancing, then each of the member tunnels in the Load-Balancing tunnel is examined to construct the branch that comprises the set of Load-Balancing members, so that a replicated copy will be sent out of one of Load-Balancing members.

4.3.2. Installing Forwarding State

The above procedures build a nexthop to be pointed to by some label or (x,g) routes. The routes are determined by checking the tree identification in the NLRI and tunnels in the TEA.

If the tree is a bidirectional (*,g) IP multicast, a (*,g) route is installed, pointing to the previously constructed nexthop.

If the tree is a unidirectional (x,g) IP multicast, one of the tunnels MUST have the RPF sub-TLV (referred to as the RPF tunnel) with a Tunnel Egress Endpoint sub-TLV with a local interface address. If it has no Receiving MPLS Label Stack sub-TLV, an (x,g) route MUST be installed with the corresponding interface as the expected incoming interface and the route points to the previously constructed nexthop. The route MAY be installed even if there is a receiving MPLS Label Stack sub-TLV in the tunnel – this is to allow native IP multicast packets to be put onto the tree at this node.

If the tree is unidirectional, only one of the tunnels MAY contain a Receiving MPLS Label Stack sub-TLV. If it is bidirectional, multiple tunnels MAY contain the Receiving MPLS Label Stack sub-TLV. For each tunnel with the Receiving MPLS Label Stack sub-TLV:

* If the sub-TLV includes only one label (which is allocated from SRGB or the node's SRLB), a label forwarding entry for that label is installed in the default label forwarding table, pointing at the previously constructed nexthop.

   * If the sub-TLV includes two labels and the first label is locally
      allocated for a label forwarding table, a label forwarding entry
      for the second label is installed in the label forwarding table
      for which the first label is allocated, pointing to the previously
      constructed nexthop.

   * If the sub-TLV includes two labels and the first label is not
      allocated for a label forwarding table, then it is assumed to be
      for a particular neighbor. A Label forwarding entry for the first
      label is installed in the default label forwarding table with the
      forwarding behavior "pop the label and save it for later
      comparison", and a label forwarding entry for the second label is
      installed in the default label forwarding table, pointing to the
      previously constructed nexthop, with additional RPF check such
      that packets are forwarded only if the popped and saved preceding
      label match the first (neighbor-identifying) label in the sub-TLV.

   * If the sub-TLV includes three labels and the first label is
      locally allocated for a label forwarding table, a Label forwarding
      entry for the second label is installed in the label forwarding
      table identified by the first label with the forwarding behavior
      "pop the label and save it for later comparison", and a label
      forwarding entry for the third label is installed in the label
      forwarding table identified by the first label, pointing to the
      previously constructed nexthop, with additional RPF check such
      that packets are forwarded only if the popped and saved preceding
      label match the second (neighbor-identifying) label in the sub-
      TLV.

   * Otherwise, the TEA is considered semantically incorrect and a
      negative acknowledgement MUST be sent back to the controller - see
      Section 4.3.3.

4.3.3. Acknowledgement to Controller

   After processing a received Replication State Route, the node MUST
   send an acknowledgement back to the controller. It originates a
   route with the same NLRI, except that the Originating Router's IP
   Address is set to match the Tree Node's IP Address. It attaches a IP
   Address Specific Route Target, with the Global Administration Field
   set to match the Originating Router's IP Address in the receive
   route, and the Local Administration Field set to 0.

   If the processing is not successful (e.g., due to unsupported tunnels
   or missing/conflicting/inappropriate sub-TLVs in the TEA), an MCAST
   NACK Extended Community MUST be attached.

5. Security Considerations

   This document does not introduce new security implications beyond
   typical BGP-based controller-to-node signaling of forwarding state.

6. IANA Considerations

   IANA has assigned the following code points:

   * "Any-Encapsulation" tunnel type 78 from "BGP Tunnel Encapsulation
      Attribute Tunnel Types" registry

   * "RPF" sub-TLV type 124 and "Tree Label Stack" sub-TLV type 125
      from "BGP Tunnel Encapsulation Attribute Sub-TLVs" registry

   This document makes the following additional IANA requests:

   * Assign the following tunnel types from the "BGP Tunnel
      Encapsulation Attribute Tunnel Types" registry:

      - Load-balancing: TBD1

      - Segment List: TBD2

   * Assign the following sub-TLV types from the "BGP Tunnel
      Encapsulation Attribute Sub-TLVs" registry:

      - Member Tunnels: TBD3, from range 128-255

      - Backup Tunnels: TBD4, from range 128-255

      - Receiving MPLS Label Stack: TBD5

   * Assign "Context Label TLV" type TBD6 from the "BGP-LS Node
      Descriptor, Link Descriptor, Prefix Descriptor, and Attribute
      TLVs" registry.

   * Assign "Replication State" route type TBD7 from the "BGP MCAST-
      TREE Route Types" registry.

   * Create a "Tree Type Registry for MCAST-TREE Replication State
      Route", with the following initial assignments:

      - 2: P2MP Tree with Label as Identification

      - 3: IP Multicast

      - 0x43: mLDP

The registration procedure is "Standards Action".

* Assign type TBD8 from the BGP Transitive Extended Community Types
  registry for the MCAST Extended Community.

* Create an MCAST Extended Community Sub-Type registry with the
  following initial assignments:

  - 0x00-0x02: Reserved

  - TBD9: NACK (Negative Acknowledgement).

  The registration procedure is First Come First Served.

7. Acknowledgements

The authors Eric Rosen for his questions, suggestions, and help
finding solutions to some issues like the neighbor based explicit RPF
checking. The authors also thank Lenny Giuliano, Sanoj Vivekanandan
and IJsbrand Wijnands for their review and comments.

8. References

8.1. Normative References

[I-D.ietf-bess-bgp-multicast]
          Zhang, Z. J., Giuliano, L., Patel, K., Wijnands, I.,
          Mishra, M. P., and A. Gulko, "BGP Based Multicast", Work
          in Progress, Internet-Draft, draft-ietf-bess-bgp-
          multicast-07, 2 December 2023,
          <https://datatracker.ietf.org/doc/html/draft-ietf-bess-
          bgp-multicast-07>.

[I-D.ietf-idr-rt-derived-community]
          Zhang, Z. J., Haas, J., and K. Patel, "Extended
          Communities Derived from Route Targets", Work in Progress,
          Internet-Draft, draft-ietf-idr-rt-derived-community-00, 7
          March 2023, <https://datatracker.ietf.org/doc/html/draft-
          ietf-idr-rt-derived-community-00>.

[I-D.ietf-idr-segment-routing-te-policy]
          Previdi, S., Filsfils, C., Talaulikar, K., Mattes, P., and
          D. Jain, "Advertising Segment Routing Policies in BGP",
          Work in Progress, Internet-Draft, draft-ietf-idr-segment-
          routing-te-policy-26, 23 October 2023,
          <https://datatracker.ietf.org/doc/html/draft-ietf-idr-
          segment-routing-te-policy-26>.

   [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>.

   [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>.

   [RFC9012] Patel, K., Van de Velde, G., Sangli, S., and J. Scudder,
              "The BGP Tunnel Encapsulation Attribute", RFC 9012,
              DOI 10.17487/RFC9012, April 2021,
              <https://www.rfc-editor.org/info/rfc9012>.

8.2. Informative References

   [RFC5331] Aggarwal, R., Rekhter, Y., and E. Rosen, "MPLS Upstream
              Label Assignment and Context-Specific Label Space",
              RFC 5331, DOI 10.17487/RFC5331, August 2008,
              <https://www.rfc-editor.org/info/rfc5331>.

   [RFC6388] Wijnands, IJ., Ed., Minei, I., Ed., Kompella, K., and B.
              Thomas, "Label Distribution Protocol Extensions for Point-
              to-Multipoint and Multipoint-to-Multipoint Label Switched
              Paths", RFC 6388, DOI 10.17487/RFC6388, November 2011,
              <https://www.rfc-editor.org/info/rfc6388>.

   [RFC6513] Rosen, E., Ed. and R. Aggarwal, Ed., "Multicast in MPLS/
              BGP IP VPNs", RFC 6513, DOI 10.17487/RFC6513, February
              2012, <https://www.rfc-editor.org/info/rfc6513>.

   [RFC6514] Aggarwal, R., Rosen, E., Morin, T., and Y. Rekhter, "BGP
              Encodings and Procedures for Multicast in MPLS/BGP IP
              VPNs", RFC 6514, DOI 10.17487/RFC6514, February 2012,
              <https://www.rfc-editor.org/info/rfc6514>.

   [RFC7752] Gredler, H., Ed., Medved, J., Previdi, S., Farrel, A., and
              S. Ray, "North-Bound Distribution of Link-State and
              Traffic Engineering (TE) Information Using BGP", RFC 7752,
              DOI 10.17487/RFC7752, March 2016,
              <https://www.rfc-editor.org/info/rfc7752>.

   [RFC7761] Fenner, B., Handley, M., Holbrook, H., Kouvelas, I.,
              Parekh, R., Zhang, Z., and L. Zheng, "Protocol Independent
              Multicast - Sparse Mode (PIM-SM): Protocol Specification
              (Revised)", STD 83, RFC 7761, DOI 10.17487/RFC7761, March
              2016, <https://www.rfc-editor.org/info/rfc7761>.

   [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>.

Authors' Addresses

   Zhaohui Zhang
   Juniper Networks
   Email: zzhang@juniper.net

   Robert Raszuk
   Arrcus
   2077 Gateway Place
   San Jose, CA 95110
   United States of America
   Email: robert@raszuk.net

   Dante Pacella
   Verizon
   Email: dante.j.pacella@verizon.com

   Arkadiy Gulko
   Edward Jones Wealth Management
   Email: arkadiy.gulko@edwardjones.com