INTERNET DRAFT
draft-shyam-vlsmtrp-11.txt
Intended status: Experimental
Expires: September 30, 2026

S. Bandyopadhyay
March 30, 2026

VLSM Tree Routing Protocol
draft-shyam-vlsmtrp-11.txt

Abstract

This is a light weight routing protocol applicable inside a network that appears in the form of a tree and distribution of address space takes place with the approach of VLSM. It is based on setting default route inside VLSM tree. With this approach, routing information of the external world need not be passed down to the VLSM tree. Thus, load inside a router gets reduced substantially. This document includes IP-VPN with MPLS inside VLSM tree by extending RSVP-TE.

Status of this Memo

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

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

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

This Internet-Draft will expire on September 30, 2026.

Copyright Notice

Table of Contents

1. Introduction

This is a light weight routing protocol of provider network that appears in the form of a tree and distribution of address space takes place with the approach of VLSM. It is based on setting default route inside VLSM tree. Inside a VLSM tree, all the physical ports of a switch are configured with their associated domain (i.e. NetAddress/NetMask). Routing table will contain static routes based on the entries configured on these ports. With this approach, routing information of the external world need not be passed down to the VLSM tree. Thus, load inside a router gets reduced substantially. In order to support network management and explicit route option, root of the tree maintains an image of the entire tree. A section of the OSPF protocol without the SPF part is extended to get the image of the tree at the root. This protocol is intended to be used in a real IP environment (e.g. NAT free environment with IPv6 or any new generation IP that may be emerged), but, it makes use of existing 32 bits address space for illustration. It expects addressing architecture of real IP space to have separate address space assigned for the routers; e.g. section 3.2.1 of architectural specification[1] states that address space with prefix "111" will be assigned for the routers. This document includes IP-VPN with MPLS inside VLSM tree by extending RSVP-TE.

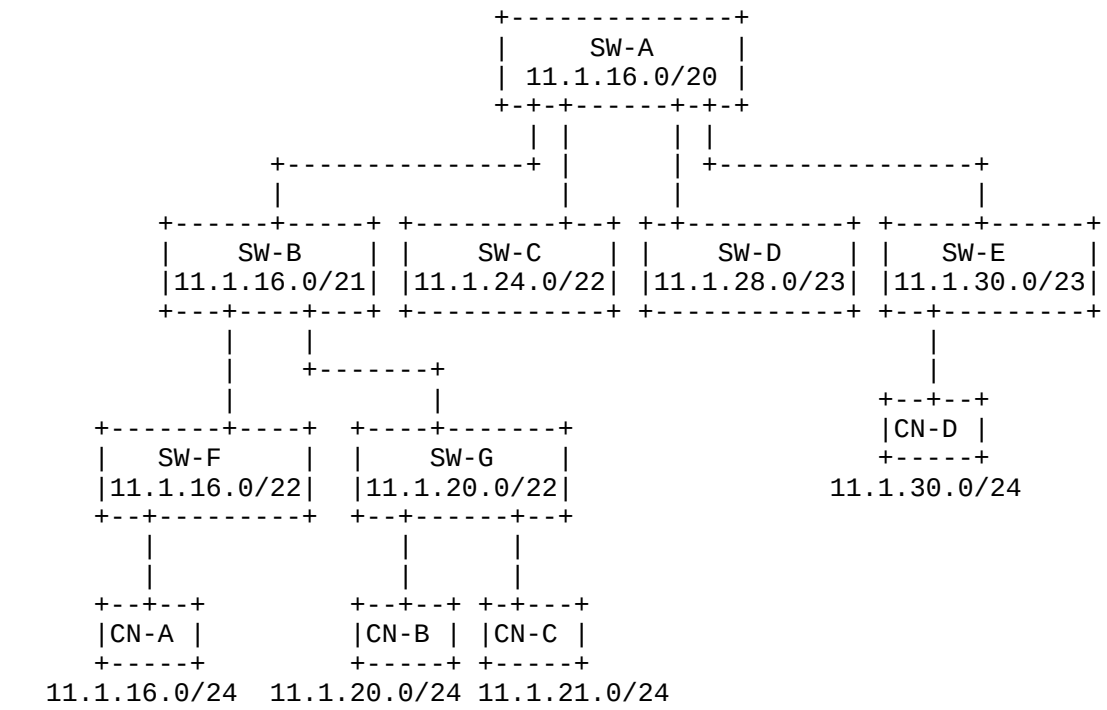
2. Setting default route inside VLSM tree

As it has been stated earlier, there is no need to pass down the routing information of the external world inside a VLSM tree that acts as a stub. Inside a VLSM tree, a node of higher prefix can be divided into number of nodes with lower prefixes. Each divided node can further be subdivided with nodes of further lower prefixes. This process can be continued as long as it is desired or no more division is further possible.

Following figure shows a typical arrangement of VLSM tree of a service provider’s network with IPv4 address space. Switch SW-A is connected to the outside world and maintains global routing table. It acts as the root of a VLSM tree that acts as a stub. It has been assigned an address block 11.1.16.0/20 which is distributed among its four children SW-B, SW-C, SW-D and SW-E with the approach of VLSM. Switch SW-B further divides its address space between switches SW-F and SW-G. Switch SW-F assigns an address block 11.1.16.0/24 to customer network CN-A. Switch SW-G assigns address block 11.1.20.0/24 and 11.1.21.0/24 to two customers CN-B and CN-C; where as switch SW-E assigns address block 11.1.30.0/24 to customer network CN-D.

Routing inside the tree takes place with the following principle.

Inside the tree, if a node (switch/router) that is assigned a domain (NetAddr/NetMask) receives a packet which is destined to somewhere outside of its domain, needs to forward the packet to its parent in the hierarchy.



If a host in CN-A wants to send a packet to an address 11.1.21.116, CE router of CN-A forwards it to SW-F. SW-F finds the destination address of the packet to be outside of its domain and forwards the packet to its parent SW-B. SW-B finds that a port that has been configured with the matching destination address and forwards it to its child SW-G. Switch SW-G sends the packet to customer network CN-B.

If a host in CN-B wants to send a packet to 11.1.17.120, CE router of CN-B forwards the packet to SW-G. SW-G finds the destination address of the packet to be outside of its domain and forwards the packet to its parent SW-B. SW-B finds that a port that has been configured with the matching destination address and forwards the packet to its child SW-F. SW-F finds the destination address to be within its domain, but no port has been configured with the matching destination address and generates ICMP UNREACHABLE.

If a host in CN-C wants to send a packet to 16.2.22.116, CE router of

CN-C forwards the packet to SW-G. SW-G finds the destination address of the packet to be outside its domain and forwards the packet to SW-B. SW-B forwards the packet to its parent SW-A. SW-A find the destination address of the packet to be outside its domain and consults with the global forwarding table and forwards the packet through the right port.

3. Router address space

Section 2.2.7 of RFC 1812 [2] states, "a router that has unnumbered point to point lines also has a special IP address, called a router-id in this memo. The router-id is one of the router's IP addresses (a router is required to have at least one IP address). This router-id is used as if it is the IP address of all unnumbered interfaces."

A router-id is selected based on the domain (NetAddress/NetMask) that it is associated with. The prefix of the domain gets embedded with in the router-id. The least significant bits of the router-id will contain the prefix. For a prefix of 'n' bits in a 32 bits address space there will be 32-'n' bits at the beginning of the address. Based on section 3.2.1 of the architectural specification[1], it starts with the prefix "1111", followed by set of '1' bits and ends with a '0' bit. Therefore, to get the prefix of the domain, router-id needs to be traced from the MSB towards LSB till it encounters a '0' bit. The rest of the bits till the end is the prefix. So, it expects prefix to be at most (32-5) i.e. 27 bits (5=first four bits as "1111" followed by '0'). So, minimum length of a domain that a router can be assigned is 32. With this approach, locators (i.e routers) and identifiers can be routed based on the same routing table. This can be defined as association between locators and identifiers. For a node acting as a child, the interface associated to its parent node gets assigned this ip address.

Add the following lines at the beginning of "ip forwarding" routine:

```
if destination address of the ip packet starts with 'router-id'
prefix {
    if prefix length of the prefix embedded inside the destination
    address of the ip packet is less than the prefix length of
    the prefix embedded inside the router-id of the router itself {
        forward the packet to the parent of the router;
    }
    else {
        find a temporary destination address 'tempDest'
        with the prefix embedded inside the destination
        address followed by '0' bits at the end.

        forward the packet the way 'tempDest' needs to be forwarded
```

```
            based on the forwarding rules as stated in section 2.
        }
    }
```

4. Network management and support of explicit route option

Section 2 has shown how routing is achieved using static route table based on the ports configured with their associated domain. Standard routing protocols usually advertise networks based on which routing table is constructed. There is no such need here. When a router tries to establish a circuit with another, it may contact a PCE to get the best possible route within a set of routes. On getting the best possible path, it sets the circuit using explicit route option. As there is only one path between any two nodes inside a tree, setting explicit route option does not make any sense to communicate between any two nodes within the same tree. It may be required to communicate a node in one VLSM tree to a node in another VLSM tree. To support this feature, root of a VLSM tree needs to maintain an image of the entire tree. A PCE can get this image by contacting the root of the tree. A network management system software also can get the status of the entire tree by communicating with the root of the tree.

This section shows how to construct the tree with the approach of routing protocol. It adopts "Hello protocol" and authentication mechanism of OSPF protocol leaving behind the SPF part and introducing new message types relevant to VLSM tree.

The router at the root constructs the tree the way it appears in the figure above. Every router in the tree is configured with the router-id of the root i.e. the domain of the tree it belongs to. Whenever a router adds a node (it may be a customer network or another router) as a child, it sends an "Add Node" message. The message is sent to the root. On getting an "Add Node" message, root traces the tree and identifies the node with "Router ID" as specified in the message and adds a node underneath. Similarly, whenever a node gets deleted, a "Delete Node" message is sent to the root. On getting "Delete Node" message, root deletes the entire sub-tree under that node in the tree. Whenever a link goes down, a "Link Down" message is sent to the root. On receiving "Link Down" message, root marks the link status as not active. Whenever a link comes up, on receiving "Link Up" message, root builds the subtree under the node whose link was down (if it happens to be a router) and sets the status of the link as active. This is to get the up-to-date status of the subtree whose link went down. Root calls "GetSubtree" routine recursively to build the subtree as follows:

```
void GetSubtree(struct TreeNode *node)
{
   send "Get Child Nodes" message to the router designated by node.
   for all the children under node, construct a TreeNode underneath.
   for all the children as a router call GetSubtree(&childNode).
}
```

Where TreeNode may be defined as:

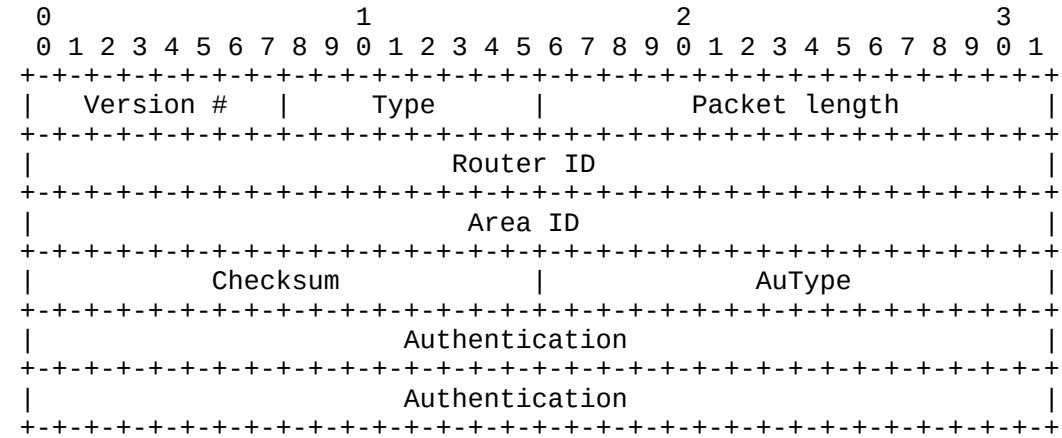
```
struct TreeNode{
   uint32 nodeId;       /* RouterId, 32 bits in IPv4 */
   uint16 nodeType      /* Customer Network (1)/Router(2) */
   uint16 noOfChildren; /* Number of children */
   struct TreeNode *parent;      /* pointer to the parent */
   struct TreeNode *childList;   /* List of child nodes */
   struct TreeNode *nextSibling; /* Next sibling in childList */
   uint16 linkStatus;   /* Link status with parent UP(1)/Down(2) */
}
```

Root can also call "GetSubtree" routine for all of its child to build the entire tree at the time of transition from old protocol to new or whenever required.

4.1. VLSM tree routing protocol messages

It maintains same message format of OSPF protocol such that existing source code can be directly ported. This section describes new message types along with Hello message of OSPF. Please follow section A.3.1 of OSPF specification [3] for OSPF message format.

Every message starts with a standard 24 byte header.



Version #
The version number. This specification documents version 1 of the protocol.

Type
The message types are as follows.

Type	Description
1	Hello
2	Add Node
3	Delete Node
4	Link Down
5	Link Up
6	Get Child Nodes
7	Acknowledgment

Packet length
The length of the protocol packet in bytes. This length includes the standard header.

Router ID
The Router ID of the packet's source.

Area ID
This is not relevant here but has been retained to make use of existing OSPF source code with least modification.

Checksum
The standard IP checksum of the entire contents of the packet, starting with the packet header but excluding the 64-bit authentication field. This checksum is calculated as the 16-bit one's complement of the one's complement sum of all the 16-bit words in the packet, excepting the authentication field. If the packet's length is not an integral number of 16-bit words, the packet is padded with a byte of zero before checksumming. The checksum is considered to be part of the packet authentication procedure; for some authentication types the checksum calculation is omitted.

AuType
Identifies the authentication procedure to be used for the packet. Authentication is discussed in Appendix D of OSPF specification [3].

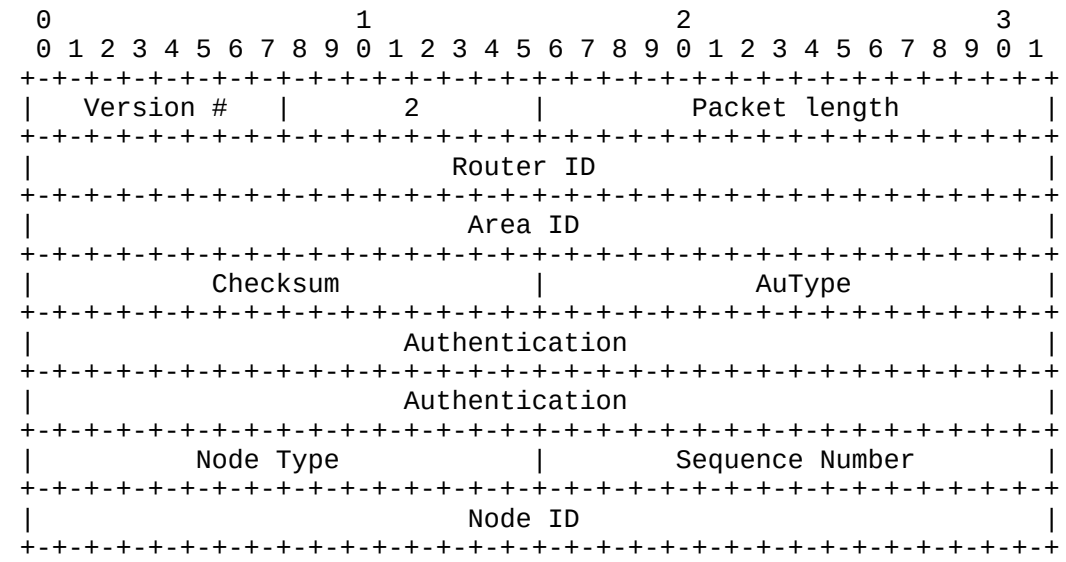
Authentication
A 64-bit field for use by the authentication scheme. See Appendix D of OSPF specification for details.

4.1.1. The Hello packet

Hello packet is just same as defined in OSPF protocol. Please follow Section A.3.2 of OSPF specification [3] for detail.

4.1.2. The Add Node packet

An "Add Node" packet is generated when a router adds a node as its child. A node can be a customer network or a router. The message gets transported to the root. The receiving router sends back an "Acknowledgment" message by changing the "Type" field as Acknowledgment. The "Sequence Number" and "Router ID" field gets verified on receiving the acknowledgment back. On receiving an "Add Node" message, root adds a new node to the tree under the node designated by "Router ID".



Node Type
Node type is Customer Network (1)/Router (2)

Sequence Number
Whenever a router generates an Add Node message it uses a Sequence Number. Usually it increments the Sequence Number on completion of the transaction.

Node ID
Node ID is the router ID of the domain associated with the router/customer network.

4.1.3. The Delete Node packet

"Delete Node" message gets generated by a router when a child node gets deleted. The message is sent to the root. On receiving "Delete Node" message, root deletes the node (i.e. the entire subtree) under the node designated as "Node ID". All the fields of a "Delete Node" packet are same as an "Add Node" packet apart from the Type(3) field.

4.1.4. The Link Down packet

"Link Down" message gets generated once a router fails to get "Hello" from any of its child and declares the link to be as inactive. The message is sent to the root. On receiving "Link Down" message root marks the link in the tree to be inactive. All the fields of a "Link Down" packet are same as an "Add Node" packet apart from the Type(4) field.

4.1.5. The Link Up packet

"Link Up" message gets generated once a router starts getting "Hello" messages from a child which was marked as inactive. The message is sent to the root. On receiving "Link Up" message, root calls "GetSubtree" routine for the node as designated by "Node ID" (if it happens to be a router). It updates changes in the subtree and marks the link as active. All the fields of a "Link Up" packet are same as an "Add Node" packet apart from the Type(5) field.

4.1.6. The Get Child Nodes packet

"Get Child Nodes" packet gets generated by root to get all the children under a router. Contents of the router is expressed as follows:

```
Router ID of the router (32 bits in IPv4) +
Number of children of the router (16 bits) +
for each child of the router {
  Type of the child (Customer Network/Router) (16 bits) +
  Router ID of the child (32 bits in IPv4)
}
```

Exchange of router information is just same as the operation of "Database Description" packet of OSPF (See section A.3.3 of [3]). Format of "Get Child Nodes" packet is same as "Database Description" packet of OSPF with the "Type" field set as 6.

4.1.7. The Acknowledgment packet

An "Acknowledgment" packet is sent to acknowledge that an "Add

Node"/"Delete Node"/"Link Up"/"Link Down" message has been received to the sender. All the fields of an "Acknowledgment" packet are same as an "Add Node" packet apart from the Type(7) field.

5. When a parent node gets connected to some of its child nodes through a shared media

Suppose a parent router Rp gets connected to some of its child routers Rc1, Rc2 ...Rcn through a shared media. Let us consider the parent port that gets attached to the shared media is assigned a domain NetAddressParentPort/NetMaskParentPort. This is the domain that will be assigned to all routers/hosts attached to the shared media. So, domain of routers Rc1, Rc2 ...Rcn will be a subset of this domain. As stated in section 3 above, interface of a child router that gets attached to the shared media will be assigned ip address with its router-id. Interface of the parent node associated to the shared media will be assigned an ip address that is router-id equivalent to NetAddressParentPort/NetMaskParentPort. ARP request for these IP addresses will return their associated hardware address. A child router needs to have a routing entry for each one of its siblings; For a sibling, the route entry will map the routing domain of that sibling with its router-id. Basically, a router needs to advertise its router-id and the parent router has to be set as the default router in the usual manner.

This problem gets resolved by introducing a new ICMP "Child Router Discovery" message. When a router gets initialized, it sends a "Child Router Discovery" message with a code set as "Router Solicitation" to the all-routers multicast address, 224.0.0.2 or the limited-broadcast address, 255.255.255.255. All routers listen to "Child Router Discovery" message and update their routing table if no entry associated to the router-id of the advertising router already exists and reply back to the advertising router with "Child Router Discovery" message with a code set as "Router Advertisement". For each new router-id of a child router, one entry gets added to the routing table as stated above; for the parent router, two new entries get added to the routing table, the first one will be the domain of the parent port that will have direct delivery. This entry gets added to support any host getting attached to the shared media along with the child routers. As domain of all the child routers are subset of the domain of the parent port, this has to be the last entry prior to the default route. The last entry will have a default route to the parent port router-id.

When a host gets initialized, it generates "Child Router Solicitation" message with a code set as "Router Solicitation". On receiving a message from a host, routers reply back in the usual manner with "Router Advertisement" without making changes to their

routing table. On receiving "Router Advertisement" from a router, hosts update their routing table in the same manner as the routers do. Hosts process "Router Solicitation" message from a router and update their routing table in the same manner that routers do without generating "Router Advertisement" message. Hosts do not process "Router Solicitation" message from any other host.

So, if we summarize the route entries in the routing table associated to the shared media, for a host, it will have entries associated to all the child routers followed by the entries related to the parent port router-id; for a child router, it will have entries related to its siblings followed by entries related to parent port router-id; for the parent router, it will have entries related to all of its child routers first followed by an entry associated to the domain of NetAddressParentPort/NetMaskParentPort which will have direct delivery linked to the associated port. If parent router gets associated to multiple shared media, it will follow the same order for each one of them and its default route will be the route that will point to its parent in the hierarchy.

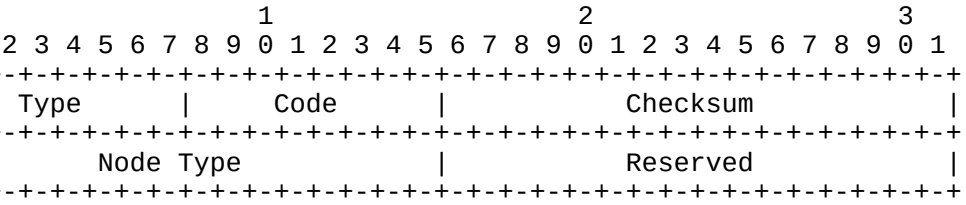
A child router, needs to maintain two sets of routing entries; the first set belongs to all the ports within its domain and the second set belongs its siblings and parent port router-id associated to the shared media. So, forwarding routine will look like as follows:

```
If destination address of the packet is part of its domain {
   If destination address falls within the domain of any port that
   has been configured, forward the packet through that port;
   else send ICMP UNREACHABLE;
} else {
   Forward the packet based on the entries that have been learned
   through the shared media;
}
```

"Keep Alive" messages are part of "Hello protocol". Every router has to send ICMP "Child Router Discovery" message with the code set as "Keep Alive" periodically at an interval of "KeepAliveTimeInterval". A time stamp has to be added associated to a router-id that has to be updated whenever "Keep Alive" message is received. System has to maintain another timer which gets activated at an interval typically, 10*"KeepAliveTimeInterval" to check the liveliness of the routers. If no "Keep Alive" message was received within that time interval, route entries associated to that router-id gets deleted.

5.1. Message Format

ICMP Child Router Discovery Message



ICMP Fields:

Type	<IANA_TBD1>
Code	0 (Router Solicitation)/ 1 (Router Advertisement)/ 2 (Keep Alive)
Checksum	The 16-bit one's complement of the one's complement sum of the ICMP message, starting with the ICMP Type. For computing the checksum, the Checksum field is set to 0.
Node Type	Parent (0)/Child(1)/Host(2); type of node of the router that is sending this message

IP Fields:

Source Address	router-id of the child router from which this message is sent. For parent router, it is the router-id associated to the parent port. For a host it is the ip address assigned to the shared media.
Destination Address	All router multicast address 224.0.0.2 or limited broadcast address 255.255.255.255. (for code = "Router Solicitation" or for code = "Keep Alive") router-id of the router or ip address of the host where this message is being sent. (for code = "Router Advertisement")
Time-to-Live	1 if the Destination Address is an IP multicast address; at least 1 otherwise.

6. IP VPN with MPLS inside VLSM tree

This section describes how to make IP VPN work inside VLSM tree without using BGP.

RFC4364 [4] describes "IP VPN" with BGP/MPLS. To support VPN, PE routers maintain per-site forwarding table. When a packet arrives from an associated CE router, PE router consults with this forwarding table to forward the packet. If the packet is supposed to be forwarded to another site of VPN through the backbone, it uses two-level label stack. The upper label is used to forward the packet from ingress PE router to the egress PE router; where as, the inner label is used for the egress PE router to identify the associated CE router where the packet is supposed to be forwarded. BGP is used by the Service Provider to exchange the routes of a particular VPN among the PE routers that are attached to that VPN. Configuration takes place on PE routers of both the sides of LSP. The simplest way to achieve this is to configure these attributes manually on PE routers. In order to have dynamic allocation of inner label, MPLS signaling protocols (in place of BGP) need to be extended. Allocation of inner label has to be done by the egress PE router. Same message that is used for the assignment of upper label may be used for the assignment of inner label. Inside the forwarding table, each entry contains the forwarding destination address based on a set of destination addresses (NetAddress/NetMask) of the IP packets received from ingress CE router. While establishing inner label, ingress PE router needs to send these attributes with the signaling message and the egress PE router needs to validate those before assigning label.

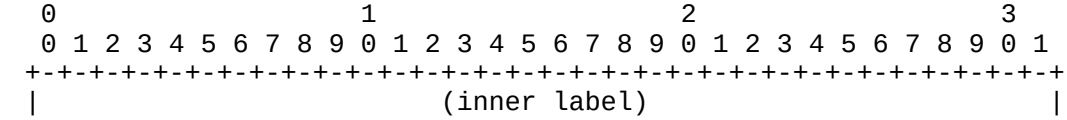
6.1. Extension to RSVP-TE to support IP VPN inside VLSM tree

This section describes extension to RSVP-TE[5] to support dynamic allocation of inner label of two-level label stack used to support VPN services.

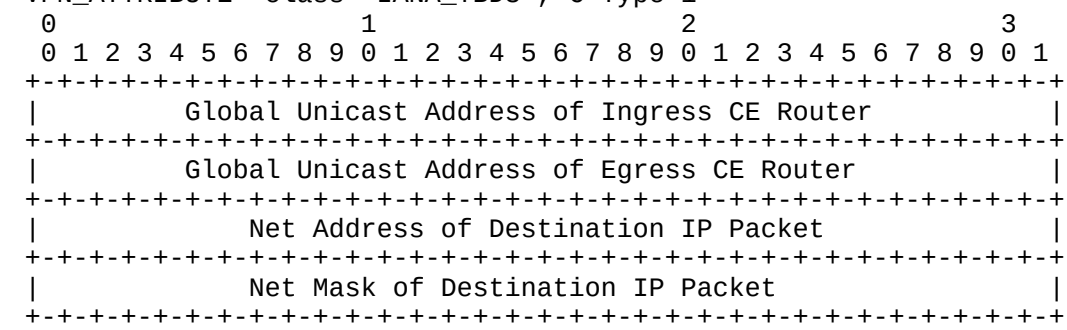
In order to establish LSP using RSVP-TE, ingress PE router sends Path message to the egress PE router. Path message is augmented with a LABEL_REQUEST object. Labels are allocated downstream and distributed (propagated upstream) by means of RSVP Resv message. For this purpose, the RSVP Resv message is extended with a special LABEL object. In order to support VPN to establish the inner label, Path message is augmented with a VPN_ATTRIBUTE label. Similarly, RSVP Resv message is extended with a VPN_LABEL object. When an egress PE router receives a Path message, it checks the presence of VPN_ATTRIBUTE object. On finding this object, egress PE router checks the viability of assignment of VPN label with the parameters from the VPN_ATTRIBUTE object and the attributes that are already configured with the egress PE router. If the test is positive, it assigns a VPN label and does

the rest of the processing of LSP label assignment and sends the RSVP Resv message with the extension of VPN_LABEL object towards the ingress PE router. On receiving Resv message with VPN_LABEL object, ingress PE router assigns VPN label along with the rest of the processing of Resv message and completes the operation. VPN_ATTRIBUTE and VPN_LABEL objects are described below.

VPN_LABEL class=<IANA_TBD2>, C-Type=1



VPN_ATTRIBUTE class=<IANA_TBD3>, C-Type=1



The format of the Path message is as follows:

```
<Path Message> ::=       <Common Header> [ <INTEGRITY> ]
                         <SESSION> <RSVP_HOP>
                         <TIME_VALUES>
                         [ <EXPLICIT_ROUTE> ]
                         <LABEL_REQUEST>
                         [ <VPN_ATTRIBUTE> ]
                         [ <SESSION_ATTRIBUTE> ]
                         [ <POLICY_DATA> ... ]
                         <sender descriptor>

<sender descriptor> ::=  <SENDER_TEMPLATE> <SENDER_TSPEC>
                         [ <ADSPEC> ]
                         [ <RECORD_ROUTE> ]
```

The format of the Resv message is as follows:

```
<Resv Message> ::=       <Common Header> [ <INTEGRITY> ]
                         <SESSION>  <RSVP_HOP>
                         <TIME_VALUES>
```

```
                              [ <RESV_CONFIRM> ]  [ <SCOPE> ]
                              [ <POLICY_DATA> ... ]
                              [ <VPN_LABEL> ]
                              <STYLE> <flow descriptor list>

      <flow descriptor list> ::= <FF flow descriptor list>
                              | <SE flow descriptor>

      <FF flow descriptor list> ::= <FLOWSPEC> <FILTER_SPEC> <LABEL>
                              [ <RECORD_ROUTE> ]
                              | <FF flow descriptor list>
                              <FF flow descriptor>

      <FF flow descriptor> ::= [ <FLOWSPEC> ] <FILTER_SPEC> <LABEL>
                              [ <RECORD_ROUTE> ]

      <SE flow descriptor> ::= <FLOWSPEC> <SE filter spec list>

      <SE filter spec list> ::= <SE filter spec>
                              | <SE filter spec list> <SE filter spec>

      <SE filter spec> ::=     <FILTER_SPEC> <LABEL> [ <RECORD_ROUTE> ]
```

Egress router generates an error with Error Code = 24, sub-code = <IANA_TBD4> (VPN label allocation error) if the operation fails.

7. IANA Consideration

IANA has assigned a new ICMP message type <IANA_TBD1>. IANA has also assigned RSVP class number <IANA_TBD2> for the object VPN_LABEL and RSVP class number <IANA_TBD3> for VPN_ATTRIBUTE. IANA has also assigned an error sub-code <IANA_TBD4> for VPN label allocation error under Error Code = 24.

8. Security Consideration

This document does not include any security related issues.

9. Normative References

[1] S. Bandyopadhyay, "An Architectural Framework of the Internet for the Real IP World" <draft-shyam-real-ip-framework-61.txt> (work in progress).

[2] F. Baker, Ed.., "Requirements for IP Version 4 Routers", RFC 1812, June 1995.

[3] Moy, J., "OSPF Version 2", STD 54, RFC 2328, April 1998.

[4] E. Rosen, Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, February 2006.

[5] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, December 2001.

10. Author’s Address

Shyamaprasad Bandyopadhyay
HL No 205/157/7, Kharagpur 721305, India
Phone: +91 3222 225137 (r) / +91 8759454400 (m)
e-mail: shyamb66@gmail.com