INTERNET DRAFT
draft-shyam-real-ip-framework-56.txt
Intended status: Experimental
Expires: October 08, 2019

S. Bandyopadhyay
April 08, 2019

An Architectural Framework of the Internet for the Real IP World
draft-shyam-real-ip-framework-56.txt

Abstract

This document tries to propose an architectural framework of the internet in the real IP world. It describes how a three-tier mesh structured hierarchy can be established in a large address space based on fragmenting it into some regions and some sub regions inside each of them. It shows how to make a transition from private IP to real IP without making significant changes with the existing network. It introduces VLSM tree routing protocol. It introduces another protocol for host identification with provider independent address. With the useful works done through IPv6, it provides all necessary inputs based on which a specification of IP with 64 bit address space may be emerged.

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 October 08, 2019.

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

Table of Contents

1. Introduction....3
2. Background....3
3. A Three tier mesh structured hierarchical network....5
3.1. Route propagation....6
3.2. Determination of prefix lengths....8
3.2.1. A pseudo optimal distribution of prefixes in a 64 bit architecture....9
3.2.2. Whether to go for a two tier or three tier hierarchy 11
3.3. Issues related to Satellite communications....12
4. VLSM tree routing protocol....12
4.1. Setting default route inside VLSM tree....12
4.2. Router address space....14
4.3. Network management and support of explicit route option....15
4.3.1. VLSM tree routing protocol messages....16
4.3.1.1. The Hello packet....17
4.3.1.2. The Add Node packet....17
4.3.1.3. The Delete Node packet....18
4.3.1.4. The Link Down packet....18
4.3.1.5. The Link Up packet....19
4.3.1.6. The Get Subtree packet....19
4.3.1.7. The Acknowledgment packet....19
4.4. IP VPN with MPLS inside VLSM tree....20
4.4.1. Extension to RSVP-TE to support IP VPN inside VLSM tree....20
5. Provider Independent addressing, name services and multihoming..22
5.1. PI address Resolution....24
5.1.1. Record Format....27
5.1.2. Messages....29
5.1.3. Master file and data file....31
5.1.4. Zone maintenance and transfers....32
6. Issues related to IP mobility....33
6.1. Changes expected with the specifications related to IP mobility....35
7. Refinements over existing IPv6 specification....36
8. Distributed processing and Multicasting....39
9. Transition to real IP from private IP....39
10. IANA Consideration....40
11. Security Consideration....40
12. Acknowledgments....40
13. Normative References....41
14. Informative References....42
15. Author's Address....42

1. Introduction

Transition from IPv4 to IPv6 is in the process. Work has been done to upgrade individual nodes (workstations) from IPv4 to IPv6. Also, there are established documents to make routers/switches to work to support IPv4 as well as IPv6 packets simultaneously in order to make the transition possible [1]. CIDR[2] based hierarchical architecture in the existing 32-bit system is supposed to be continued in IPv6 too with a large address space. There are documents/concerns over BGP table entries to become too large in the existing system [3]. There are proposals to upgrade Autonomous System number to 32-bit from 16-bit to support the demand at the same time [4]. The challenge relies on how to make the transition smooth from IPv4 to a real IP world with least changes possible.

The term "real IP environment" is referred to an environment where hosts in a customer network will possess globally unique IP addresses and communicate with the rest of the world without the help of NAT[5]. This document reflects changes required with the BSD 4.4 source code where ever applicable.

2. Background

Existing system is in work with Autonomous System (AS) and inter-AS layer with the approach of CIDR. In order to meet the need within the 32-bit address space, Autonomous Systems of various sizes maintain CIDR based hierarchical architecture. With the help of NAT [5], a stub network can maintain an user ID space as large as a class A network and can meet its useful need to communicate with the rest of the world with very few real IP addresses. With the combination of CIDR and NAT applied in the entire space, most of the part of 32-bit address space gets effectively used as network ID.

With traditional CIDR based hierarchy, 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 till no further division is possible. The point worth noting is at each point the designer of the network has to preconceive the future expansion of the network with the concept in the mind that the resource can not be exhausted at any point of time. This phenomenon leads the designer to allocate resources much higher than whatever is needed which leads to a space of unused address space. The problem gets aggravated once resource gets exhausted by any chance. e.g. a node of prefix /16 can be divided with a number of nodes of prefixes /24. If any one of the nodes /24 gets exhausted, resources of other nodes of prefixes /24 can not be used even if they are available.

In IPv4 environment, there is a desperate attempt of the service providers to provide internet services with the help of NAT. e.g. a large educational institute meets its current requirement with 4 real IP addresses; one for its mail server, one for its web server, one for its ftp server and another one for its proxy server to provide web based services to all of its users. In general, these services are used by an organization of any size(it may be 400 or even 40000). In the current scenario, the CIDR based tree has been built using these components together. When private IP will be replaced with real IP, each customer network will require IP addresses based on its size and requirement.

Transitioning from private IP to real IP basically requires the following components:

- o A solution for site multihoming with provider assigned address space
- o A strategy to replace private IP to real IP
- o A solution to uniquely identify a host in a real IP environment
- o A solution to make individual nodes and routers/switches to work with IPv4 and next generation IP simultaneously.

Solution for site multihoming has been provided in a separate document [8]. Section 9 shows how to make a transition from private IP space to real IP space with provider assigned addresses with CIDR based approach itself without reorganization of the existing provider network. Section 5 provides a solution for identifying a host uniquely with a number in a real IP environment. RFC 4213 [1] has already described the transition mechanism from IPv4 to IPv6 for individual nodes and routers.

Transitioning to real IP will eliminate the extra routing entries associated with multihomed sites and thus will reduce the size of the BGP table substantially. Assignment of addresses requires an architectural framework. It may continue with the existing CIDR based architecture (provided transitioning to real IP will be good enough to handle all routing related issues for ever) or may come out with a different approach. Mesh structured hierarchy will reduce the growth of routing entries in a CIDR based environment as well as convenient for distribution of network resources in a suitable manner in the long run.

This document also tries to resolve and enhance several issues that were carried on as part of deployment of IPv6. It shows that a 64 bit address space is good enough for all practical purposes. With the useful works done through IPv6, it provides all necessary inputs based on which a specification of IP with 64 bit address space may be emerged.

3. A Three-tier mesh structured hierarchical network

As Autonomous Systems of various sizes are supported, Autonomous Systems and the nodes inside the Autonomous Systems can be viewed as graphically lying on the same plane within the address apace. If network can be viewed as lying on different planes, routing issues can be made simpler. If network is designed with a fixed length of prefix for the Autonomous System everywhere, routing information for the rest will get confined with the other part of the network prefix. Which means the maximum size of AS gets assigned to all irrespective of their actual sizes. This can be made possible with the advantage of using a large address space and dividing it into number of regions of fixed sizes inside it. Thus entire network can be viewed as a network of inter-AS layer nodes. Each node in the inter-AS layer can act either only as a router in the inter-AS layer or as a router in the inter-AS layer with an Autonomous System attached to it with a single point of attachment or as an Autonomous System with multiple Autonomous System border routers (ASBR) appearing like a mesh. Thus two tier mesh structured hierarchy gets established between AS layer and inter-AS layer with each AS having a fixed length of prefix.

Based on the definition of Autonomous System, it is a small area within the entire network that maintains its own independent identity that communicates with the rest of the world through some specific border routers. In the similar manner, if a larger area (say region or state) can be considered as network of Autonomous Systems, that can maintain its own identity by communicating with the rest of the world through some border routers (say, state border router), mesh structured hierarchy can be established within the inter-AS layer. The inter-AS layer will be split into inter-AS-top and inter-AS-bottom. To maintain this hierarchy, each node of inter-AS-top needs to have multiple regional or state border routers (say, SBR) through which each one will communicate with the rest of the world in the similar manner an Autonomous System maintains ASBR. Thus, entire network will appear as a network of nodes of inter-AS-top layer. To maintain hierarchy, each node of the inter-AS-top needs to have a fixed length of prefix. i.e. each node of the inter-AS top will be assigned a maximum (fixed) number of nodes of Autonomous Systems.

Thus, with three-tier mesh structured hierarchy in the network layer, network ID can be viewed as A.B.C. If pA, pB and pC be the prefix lengths of inter-AS-top, inter-AS-bottom and AS layers respectively, there will be 2^pA nodes at the topmost layer, 2^pB at the inter-AS-bottom layer and 2^pC nodes at the AS layer. Thus the entire space gets divided into a fixed number of regions and each region gets divided into fixed number of sub regions. This division is supposed to be made based on geography, population density and their demands and related factors.

Let nMaxInterASTopNodes be the possible maximum number of nodes assigned at the top most layer and nMaxInterASBottomNodes be that at the inter-AS-bottom layer and nMaxASNodes at the AS layer. Where nMaxInterASTopNodes <= 2^pA and nMaxInterASBottomNodes <= 2^pB and nMaxASNodes <= 2^pC.

3.1. Route propagation

With hierarchy established, routing information that gets established inside a node of inter-AS-top, does not need to be propagated to another node of inter-AS-top. Entire routing information of inter-AS-top layer needs to be propagated to inter-AS-bottom layer. So, each router of inter-AS layer will have two tables of information, one for the inter-AS-top and another for the inter-AS-bottom of the inter-AS-top node that it belongs to. BGP (with little modification) will work very well with a trick applied at the SBRs. Each SBR will not propagate the routing information of inter-AS-bottom layer of its domain to another SBR of neighboring domain. i.e. SBR of one top layer node will propagate routing information only of inter-AS-top layer to SBR of another top layer node. Inside a node of inter-AS-top, routing information of inter-AS-top and inter-AS-bottom need to be propagated from one ASBR to another neighboring ASBR. Inside a top layer node A, routing information of another top layer node B will have two parts; one for the list of SBRs through which a packet will traverse from top layer node A to B and another for the list of ASBRs through which the packet will traverse from one AS to another inside A. In terms of BGP, AS_PATH attribute will be split into two parts; one for the information of the top layer and another for the bottom layer. Within the same node A routing information of one AS to another AS will not have any top layer information. i.e. the top layer information will be set to as NULL.

Similarly, each node of the AS layer will have three tables of routing entries. One for the inter-AS-top, one for the inter-AS-bottom and another for the routing information inside the Autonomous System itself.

Introduction of hierarchy at the inter-AS layer reduces the size of the routing table substantially. With the availability of hardware resources if flat address space is maintained at each layer, problems related to CIDR can be avoided. With flat address space, no hierarchical relationship needs to be established between any two nodes in the same layer. So, all the nodes inside each layer can be used till they get exhausted. With flat address space (i.e. without prefix reduction), BGP tables will have maximum nMaxInterASTopNodes + nMaxInterASBottomNodes entries.

IGP like OSPF has got provision to divide AS into smaller areas. OSPF

hides the topology of an area from the rest of the Autonomous System. This information hiding enables a significant reduction in routing traffic. With the support of subnetting, OSPF attaches an IP address mask to indicate a range of IP addresses being described by that particular route. With this approach it reduces the size of the routing traffic instead of describing all the nodes inside it, but introduces another level of hierarchy. If subnetting concept can be avoided from the AS layer(with the additional overhead of computation inside the SPF tree), each area can be configured from a free pool of addresses based on its requirement dynamically. So, an AS can be divided into number of areas of heterogeneous sizes with the nodes from a free pool of address space.

Similarly, the concept of area can be introduced in the inter-AS-bottom layer the way it works in OSPF. The area border routers in the inter-AS-bottom layer have to behave exactly in the similar manner the way an ABR behaves in OSPF. i.e. an area border router will hide the topology inside an area to the rest of the world and will distribute the collected information inside the area to the rest. It will distribute the collected routing information from outside to the nodes inside as well. In order to implement this, protocol running in the inter-AS layer (say BGP) will have to introduce a 'cost' factor. This cost factor can be interpreted as the cost of propagation of a packet from one AS to another. The protocols running inside AS layer (RIP/OSPF, etc) will have to the supply the cost information for a packet to travel from one ASBR to another. All the protocols must behave in unison for supplying this information. The cost factor is needed for a remote node while sending a packet to a node inside an area while more than one area border routers are equidistant from that remote node. Thus inter-AS-bottom layer (i.e. one inter-AS-top level node) can be divided into number of areas of heterogeneous sizes with nodes of AS from a free pool of address space. BGP adopts a technique called route aggregation. Along with route aggregation it reduces routing information within a message. In the similar manner, introduction of area inside inter-AS-bottom layer will not only reduce the complexity of the protocol, but will reduce the size of a BGP packet substantially.

With this architecture, each node(router) inside an AS is represented as A.B.C. Each node may or may not be attached with a network which acts as a leaf node (i.e. a network will not act as a transit). In order to make use of user-id space properly and to support customer networks of heterogeneous sizes, the user-ID space needs to be divided as subnet-ID and user-ID. Profoundly, a VLSM (variable length subnet mask) type of approach (in the form of a tree) has to be adopted at each node of an AS. So, each node of the AS layer will act as the root of a tree whose leaves are independent small customer networks which will act as stub. As the routing information of inter-

AS layer as well as AS layer need not be passed inside any node of the VLSM tree, each router inside the tree should maintain default route for any address outside of its network/domain. With this approach, load on each router of the service providers will become negligible. Protocols that supports VLSM with MPLS/VPN has to be implemented inside the tree. Inside the VLSM tree, all the physical ports of a switch have to be configured with the subnet mask. A light weight routing protocol can be developed on top of static routing table by setting default route inside VLSM tree.

The fundamental assumptions based on which this architecture lies can be summarized as follows:

i) Entire network can be viewed as a network of regions or states where each region or state can have its own identity by communicating with the rest of the world through some state border routers. Each region or state is a network of Autonomous Systems. Each region as well as each Autonomous System inside them will have a fixed (maximum) length of prefix.

ii) Availability of hardware resources is such that flat address space can be maintained at the inter-AS layer.

Introduction of mesh-structured hierarchy will have several advantages:

- Load at each router will get reduced substantially.
- Concept of CIDR style approach and complexity related to prefix reduction can be easily avoided.
- Mesh structured hierarchy will make traffic evenly distributed.
- Physical cable connection can be optimized.
- Administrative issues will become easier.

3.2. Determination of prefix lengths

With this architecture, IP address can be described as A.B.C.D where the D part represents the user id. Each router in the inter-AS layer will have two tables of information, one for the inter-AS-top and another for the inter-AS-bottom of the inter-AS-top node that it belongs to. Whereas, each node of the AS layer will have three tables of routing entries; one for the inter-AS-top, one for the inter-AS-bottom and another for the routing information inside the Autonomous System itself. In the worst case. a node inside an AS needs to maintain nMaxInterASTopNodes + nMaxInterASBottomNodes + nMaxASNodes entries in its routing table.

The dynamic nature of allocating an area from a free pool of address space is more frequent at the AS layer than at the inter-AS-bottom

layer. As OSPF supports all the features needed, it can be considered as default choice in the AS layer. Existing implementation of OSPF (Version 2) supports subnetting, by which an entire area can be represented as a combination of network address and subnet mask. With this approach, entire routing table gets reduced substantially. With the removal of subnetting, all the nodes inside an area will have an entry inside the routing table (OSPF Version 1). So the deterministic factor is what is the maximum number of nodes inside an AS OSPF can support once subnetting support gets removed. So the prefix length of AS layer will be determined by this factor of OSPF.

With the introduction of hierarchy in the inter-AS layer, number of entries in the BGP routing table will get reduced substantially. Even if pA and pB both are selected as 16, number of routing entries come within the admissible range of existing BGP protocol. But, it is the responsibility of IANA to come out with a scheme how nMaxInterASTopNodes and nMaxInterASBottomNodes are to be selected. Each top level node will have nMaxInterASBottomNodes nodes. It will be a waste of address space if each country gets assigned a top level nodes (e.g. china has got a population of 1,306,313,800 people where as Vatican City has got only 920 according to a census of 2006). So a moderate value of nMaxInterASBottomNodes is desirable, with which larger countries will have a number of top level nodes. e.g. each state of USA can be assigned a top level node. With the introduction of area in the inter-AS-bottom layer, each top level node can be divided into number of areas of heterogeneous sizes. So, a group of neighboring countries with less population can share the address space of a top level node. Similarly, user-id space has to be decided based on the largest area VLSM tree should be spanned through. All these issues are completely geo political and have to be decided by IANA.

3.2.1. A pseudo optimal distribution of prefixes in a 64 bit architecture

In order to have optimal use of cable connections, length of the VLSM tree is expected to be as short as possible. Also any single organization may prefer to have its user id space to be under the same network id. So, a 16 bit user-id may become insufficient for places like large university campus, where as 32 bit will become too large. Hence, 24 bit user-id will be a moderate one which is the class A address space in IPv4 (also used as the space for private IP). As published in 1998 [6], OSPF can support an area with 1600 routers and 30K external LSAs. So, 11 bits are needed to support this space. With the assumption that OSPF can support much more address space with the advancement of hardware technology as well as to keep the space open for future expansions, 12 bits are assigned for the AS layer. 16 bits are assigned for the inter-AS-bottom layer. So, if on

the average, 16 bit equivalent space gets used within the user-id space (i.e. one out of 256) and 8 bit equivalent nodes gets used inside an AS (16% of 1600), for a top level node (with 16 bit equivalent AS nodes), it will generate 2^40 IP addresses, which will give 8629 IP addresses per person in Japan (with a population of 127417200; Japan is at the 10th position from the top in the population list of the world). So, even if all the countries with population less than or equal to Japan are assigned a top level node and all the provinces/states of countries with larger population are assigned a top level node each, total number of nodes will come well under 1024. If a number of neighboring countries with lesser population shares a top level node, total number of top level nodes will come down further. This suggests that 62 bit equivalent (10(pA)+16(pB)+12(pC)+24(user-id)) space will be good enough for unicast addresses. This distribution expects OSPF to support 65K (64K+1K) external LSAs.

Distribution of address space will be finalized based on the consultation with IANA. Primarily, they may appear to be as follows:

64 bit address space may be divided into two 63 bits blocks:

i. Global unicast addresses with the most significant bit set to 0. This space is equally divided between provider assigned (PA) address space and provider independent (PI) address space.

a) Provider assigned address space with prefix 00.

b) Provider independent (PI) address space with prefix 01. Provider independent address space will be used for the customers who would like to retain their number even after changing their providers. As routing will be based on PA addresses, each PI address will be associated to at least one PA address. Most significant part of PI addressing is, it is independent of the architectural framework of the provider network; even if the architectural framework changes, same format of PI addressing can be maintained. Once implemented, PI address of a node will be the number that will be generally used by the common people. Section 5 describes issues related to PI addressing in detail.

ii. Address space with the MSB set to 1 will be distributed within the rest. Each of them will have a fixed prefix. This distribution will be based on the requirements and the work that have already been done in connection to IPv6:

a) Address space for multicasting with a prefix set to 1001.

b) Address space for link-local address: Link local addresses will

have a prefix 1010.

c) Router address space: Prefix 1111 will be used by routers inside VLSM trees and 1110 will be used by backbone routers connecting them.

d) Address space for private IP: Each customer network can maintain private address space to communicate within its users. This space will be distributed within all the customer sites of a corporate that can maintain VPN services. A 32 bit address space should be good enough for private IP. Private address space will have a 32 bit prefix with leading 4 bits are set to 1100 and the rest are set to 1.

Rest of the address space has been kept for future use.

3.2.2. Whether to go for a two-tier or three-tier hierarchy

Establishment of hierarchy in the inter-AS layer reduces the size of BGP entries to a great extent, but leads to an improper use of address space due to geo-political reason. If hierarchy in the inter-AS space gets removed, entire 26 bit (10+16) space will be available for a single layer and use of inter-AS space will be true to its sense, but will increase external LSA (and/or number of entries in the BGP table) dramatically. So, it depends on to what extent OSPF can support external LSAs. BGP expects the packet length to be limited to 4096 bytes. BGP manages to make it work with this limitation with the concept of prefix reduction in the CIDR based environment. As the number of inter-AS nodes increases, BGP has to change this limit in order to make it work in flat address space. The alternate will be to divide the inter-AS space into number of areas as defined in section 2.1. The area border routers will advertise the aggregated information to the rest of the world. BGP may have to incorporate both the options at the same time. As the number of nodes in the inter-AS layer increases, in order to reduce the number of entries in the routing table, inter-AS space has to be split into two separate planes. So, two-tier hierarchy can be considered as an interim state to go for three-tier hierarchy. If it so happen that current available data is good enough to support the present need, it will be worth to look for to what extent it can support in the future. Assignment of inter-AS nodes in two-tier hierarchy should be based on the geographical distribution as if it is part of three-tier hierarchy. Otherwise, introduction of three-tier hierarchy in the future will become another difficult task to go through. Based on the report of year 2011, BGP supports ~400,000 entries in the routing table. With this growing trend, BGP may have to change the limit of packet length even in a CIDR based environment. With the introduction of two-tier hierarchy, number of entries in the routing table will come down drastically and with the three-tier approach, it will come down further.

3.3. Issues related to Satellite communications

Establishment of hierarchy in the inter-AS layer expects the only way any two autonomous systems in two different top level nodes communicate is through their SBRs. If two autonomous systems inside the same top level node communicate through satellite, it will be considered as a direct link between them. Whenever autonomous system 'ASa' of top level node 'A' communicates with autonomous system 'ASb' of top level node 'B' through satellite, they have to go through their state border routers. i.e. satellite port inside 'A' that communicates with a satellite port inside 'B' will be considered as state border router. If multiple such ports exists inside node 'A', all of them will be equidistant from any port inside 'B'. Which expects any satellite port inside 'B' to have prior knowledge of list of autonomous systems that will be under the purview of any port inside 'A'. So, all the satellite ports of 'A' have to exchange such group of information with all the satellite ports of 'B' and vice versa. These group of autonomous systems can be considered as a cluster of autonomous systems inside an area of a top level node. If number of such ports is small, some heuristics can be applied while assigning AS numbers in order to reduce the processing time during the circuit establishment phase. It will become difficult to maintain such heuristics once the number of such ports becomes large. So, in case of satellite communication, the advantage of establishing hierarchy inside inter-AS layer diminishes as the number of satellite ports increases. If any private corporate maintains its own satellite channel to communicate between its offices at distant locations, all of these offices are going to be considered as under the user-id space of its network. Service providers that provide satellite services to the end-site customers, can operate in the usual manner as they will provide connection to customer networks which will act as stub.

4. VLSM tree routing protocol

This section describes a light weight routing protocol applicable inside VLSM tree. It is based on setting default route inside VLSM tree. Inside a VLSM tree, all the physical ports of a switch have to be configured with their associated domain (i.e. NetAddress/NetMask). Routing table will contain static routes based on the entries configured on these ports.

4.1 Setting default route inside VLSM tree

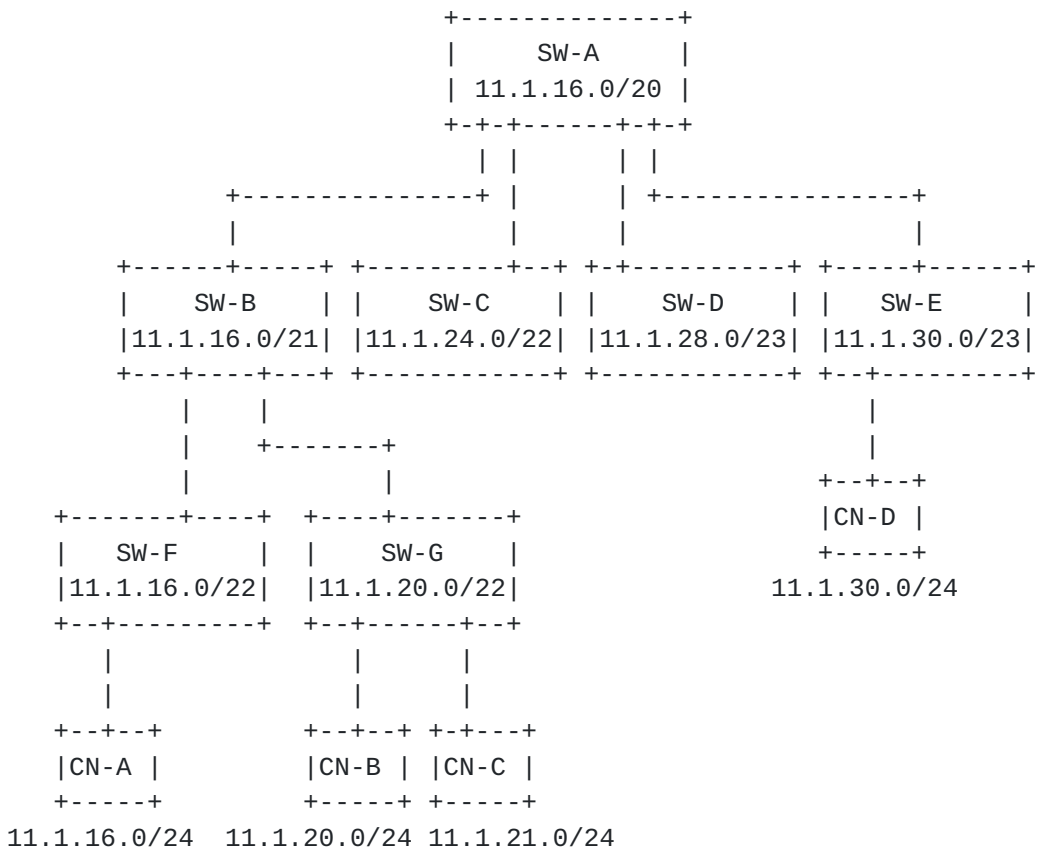
Section 3.1 describes that there is no need to pass down the routing information of the external world inside VLSM tree that acts as a stub. Inside a VLSM tree, a node of higher prefix can be divided into number of nodes with lower prefixes. Each divided node can further be

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.

4.2. Router address space

Section 2.2.7 of RFC 1812 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. It starts with the prefix "1111" (see section 3.2.1), 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 three 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.

4.3. Network management and support of explicit route option

Section 4.1. 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.

It is possible to construct this tree with the approach of network management system by means of pooling and generation of trap on network failure. This section shows how it is done 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.

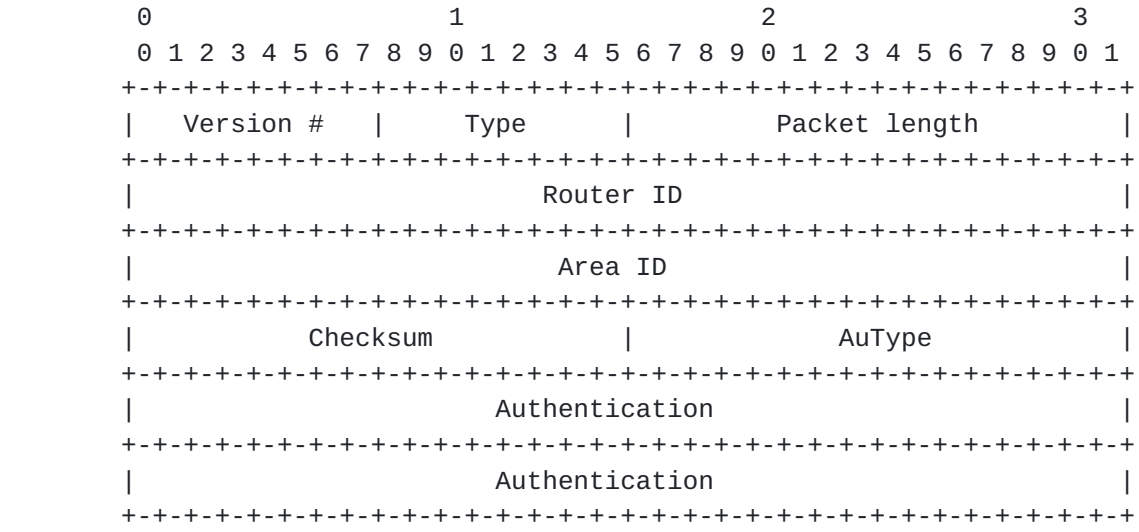
The router at the root constructs the tree the way it appears in the figure above. Every router in the tree is configured with the router-id of the root i.e. the domain of the tree it belongs to. Whenever a router adds a node (it may be a customer network or another router) as a child, it sends an "Add Node" message to its parent. The message gets propagated 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, "Delete Node" message gets propagated 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, "Link Down" message gets propagated 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 sends "Get Subtree" message to the node (if it happens to be a router) whose link was down 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. "Get Subtree" message sends the database of the entire Subtree in a recursively manner. Root can also send "Get Subtree" message to all of its child to build the entire tree at the time of transition from old protocol

to new or whenever required.

4.3.1. 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 [19] 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 Subtree
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 [19].

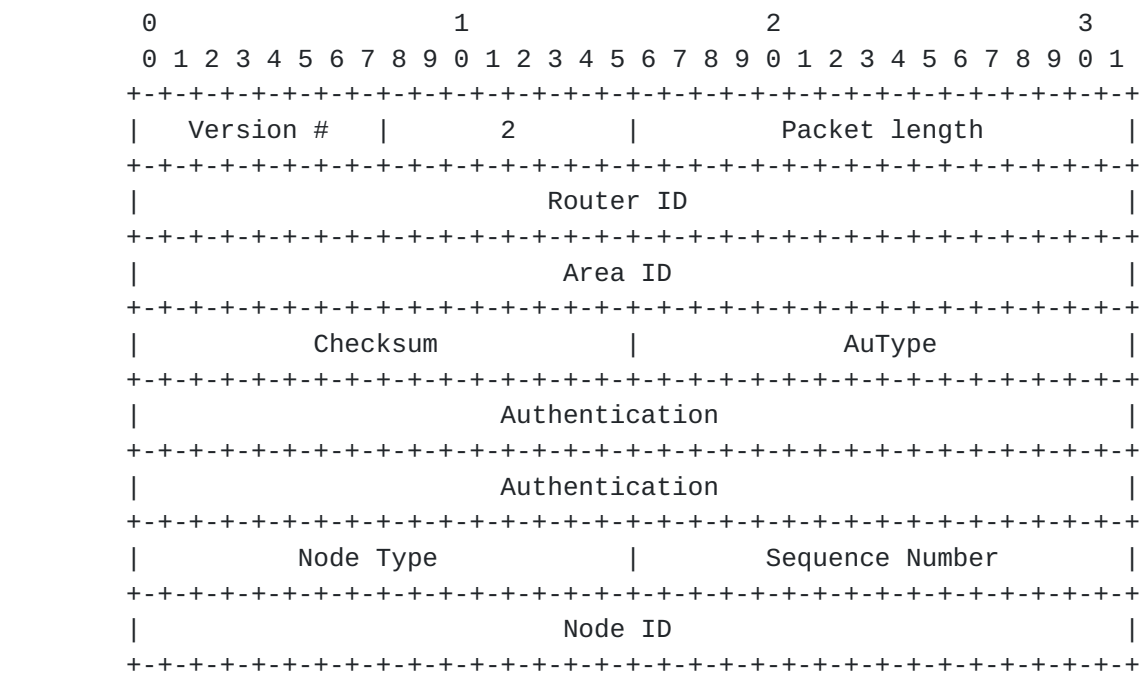
Authentication
A 64-bit field for use by the authentication scheme. See Appendix D of OSPF specification for details.

4.3.1.1. The Hello packet

Hello packet is just same as defined in OSPF protocol. Please follow Section A.3.2 of OSPF specification [19] for detail.

4.3.1.2. The Add Node packet

An "Add Node" packet is generated when a router adds a port to a customer network or attaches a new router. 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.

4.3.1.3. The Delete Node packet

"Delete Node" message gets generated by a router when a node gets deleted. The message gets transported 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" packets are same as an "Add Node" packets apart from the Type(3) field.

4.3.1.4. The Link Down packet

"Link Down" message gets generated once a router failed to get "Hello" from its neighbor and declares the link to be inactive. The message gets transported 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.

4.3.1.5. The Link Up packet

"Link Up" message gets generated once a router starts getting "Hello" messages from a neighbor which was marked as inactive. The message gets transported to the root. On receiving "Link Up" message, root sends "Get Subtree" message to the node (if it happens to a router) as designated by "Node ID". After getting the subtree database, root 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.

4.3.1.6. The Get Subtree packet

"Get Subtree" packet gets generated to get the database of a subtree. Database of a subtree is expressed recursively in the following manner.

Add Router ID of the root of the subtree (32 bits in IPv4) + Add Number of child of the subtree (16 bits) + for each child of the subtree {
Add Type of the child (Customer Network/Router) (16 bits) +
Add Router ID of the child (32 bits in IPv4) } for each child as a router of the subtree, call Get Subtree

Once router R1 (as master) sends "Get Subtree" packet to router R2 (as slave), router R2 collects database information from all the routers as child recursively and then exchange database information to R2 (the master). Exchange of database is just same as operation of "Database Description" packet of OSPF (See section A.3.3 of [19]). Format of "Get Subtree" packet is same as "Database Description" packet of OSPF with the "Type" field set as 6.

Amount of time that R2 takes to collect database information is based on its position at the tree. If it is at the bottom most position of the tree, it takes least amount of time, where as if it is just one level below the root of the tree, it takes the maximum amount of time. So, to get the first packet, R1 needs to wait maximum amount of time and subsequent packets will arrive at regular interval of time. If timer expires, R1 sends the same packet to R2. R2 ignores the packet if it has already started processing. The question arises how many times R1 will send the same packet to get the first response? The answer is if there is no link failure, it will get a response and if there is a link failure, it will receive "Link Down" message and halts operation. This process gets optimized using exponential back-off algorithm.

4.3.1.7. 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.

4.4. IP VPN with MPLS inside VLSM tree

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

RFC4364 [7] 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.

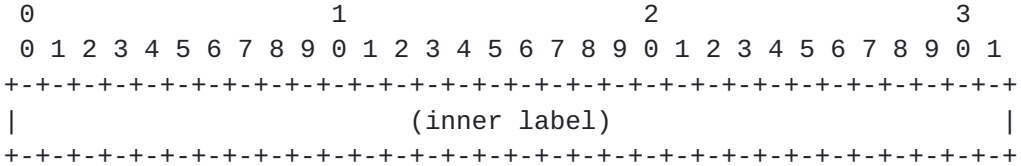
4.4.1. Extension to RSVP-TE to support IP VPN inside VLSM tree

This section describes extension to RSVP-TE[17] to support dynamic allocation of inner label of two-level label stack used to support VPN services.

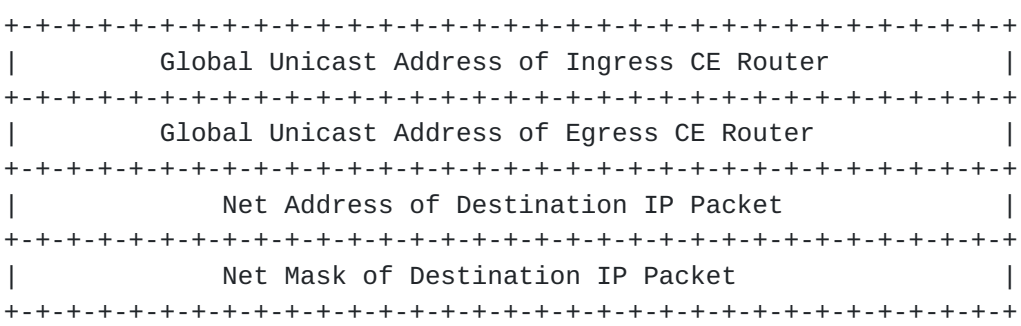
In order to establish LSP using RSVP-TE, ingress PE router sends Path message to the egress PE router. Path message is augmented with a LABEL_REQUEST object. Labels are allocated downstream and distributed (propagated upstream) by means of RSVP Resv message. For this purpose, the RSVP Resv message is 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.

5. Provider Independent addressing, name services and multihoming

Provider independent addressing can be conceived as naming a host with a number. It can be used by customer networks who would like to retain their number even after changing their service provider; also it is useful to designate a host uniquely if the customer network is multihomed. Just like in name services, as address corresponding to a name needs to be resolved first to initiate communication, the same is required for PI addressing. Each globally unique PI address will be associated to at least one global unicast provider assigned address. For a host with single interface, this number will be same as the number of service providers the customer network is associated with.

As either source or destination or both may be multihomed, there could be multiple paths to communicate between two hosts. This is required both for name services as well as for PI addressing.

A system call needs to be introduced to get the source address based on the destination address. If application program needs to use the destination address directly, it needs to use this system call.

```
int getcommaddr(int sockfd, struct in_addr *dst, struct addr_pair
*endpts);
```

'addr_pair' holds the addresses of communication end points as follows:

```
struct addr_pair {
    struct in_addr src;
    struct in_addr dst;
};
```

'getcommaddr'[8] returns the number of source-destination pairs for communication; the field 'endpt' will hold the array of these addresses. The array will be in sorted manner based on the best possible route. 'sockfd' is used to get the 'type of service' assigned. So, an application program needs to set its type of service before using this call.

'getcommaddr needs to call a routine 'getmappedaddr' to resolve the mapped provider assigned addresses of a provider independent address.

```
int getmappedaddr(struct in_addr *piaddr, struct in_addr *mpiaddr);
```

'getmappedaddr' will return number of mapped addresses and 'mpiaddr' will hold their values.

Users may use name instead of IP address to reach the destination. A new system call needs to be introduced 'gethostbynamewithsrcaddr', which is an extension to 'gethostbyname' as follows:

```
struct hostent *gethostbynamewithsrcaddr(int sockfd,const char *name,
               int *nroutes, struct addr_pair *endpts);
```

'gethostbynamewithsrcaddr'[8] takes 'name' and 'sockfd' as input parameters and finds out the best possible route to reach the destination. It returns the pointer to the 'hostent' structure as returned by 'gethostbyname' system call. The parameter 'nroutes' gets the number of possible routes to be used and the corresponding source and destination addresses gets assigned to 'endpts' in sorted manner. 'sockfd' is used to get the 'type of service' assigned. So, an application program needs to set its type of service before using this call.

An application program needs to use these source addresses from the

top (i.e. the 0th) to establish connection with the destination. It needs to bind source address 'src' and then connect with the destination address 'dst'.

5.1. Host identification with Provider Independent address

This section tries to come up with a solution for PI address resolution with the approach of DNS[10] with necessary differences. Just like name space in DNS, entire address range with prefix 01 will be the address space used by PI addresses. Servers that will hold the information of mapping between PI addresses and corresponding PA addresses will be called as PIMapServers and the programs that will be used to resolve addresses will be called as PIMapResolvers.

In case of DNS where name is used in hierarchical format to resolve the addresses, PI address resolution will be based on the prefix of the PI address used for resolution. The prefix is determined based on the architectural model used for the internet. Based on the prefix information addresses of a list of servers can be found out that will act as regional servers which will be used to resolve mapped PA addresses corresponding to that PI address. A prefix will serve a fixed address space within entire PI address space. Address space belonging to a prefix will be distributed within customer networks of heterogeneous sizes. Address space allocation and the mapping of associated PA address(es) will be assigned by a regional authority. The regional authority will be fully responsible for the operation of regional servers in that region.

Like DNS, there are some root servers which will have some fixed addresses, under which there are some prefixes which will act as top-level-domains. In case of CIDR based hierarchy, these prefixes may be of different prefix lengths which are selected based on the requirements. Each prefix in a top level domain can further be split into number of prefixes with the approach of CIDR. This tree structured hierarchy will be kept on growing till we get prefixes associated with regional servers. Each prefix associated with a regional server will be distributed amongst customer networks of various sizes as well as prefixes that will again be associated with some regional servers with the approach of CIDR. These regional servers can be considered as equivalent to the authoritative name servers of DNS which are associated with zones. As stated earlier, prefixes starting with "00" will be assigned for provider assigned addresses and prefix starting with "01" will be assigned for provider independent addresses where as prefix starting with "1" will be assigned for addresses of all other types.

As inherent hierarchy is involved in "Mesh structured hierarchy", this hierarchy goes up to two levels. As usual, there will be some

root servers with fixed assigned addresses. Each root server will have prefixes with "01.A" that will act like top level domain. Under each top level domain, there will be entries with prefixes "01.A.B". Within a region "A.B", every global PA address is represented as "00.A.B.C.user-id". In order to support customer networks of heterogeneous sizes with the approach of VLSM, the "user-id" portion is further divided as "subnet-id.user-id". So, the effective network prefix of a customer network in PA address space is "00.A.B.C.pa-subnet-id". Within an "A.B", entire PI address space with prefix "01.A.B" will be distributed within customer networks of heterogeneous sizes. So, effective network prefix of a customer network with PI address will be "01.A.B.pi-subnet-id". A particular prefix "01.A.B.pi-subnet-id" will be mapped to at least one provider assigned prefix of same prefix length. For a multihomed customer network within "A.B" that receives services from two service providers will have prefixes "00.A.B.C1.pa-subnet-id1" and "00.A.B.C2.pa-subnet-id2". A PI address prefix "01.A.B.pi-subnet-id" of same length will be mapped to both these prefixes of PA address space. Every region "A.B" will have regional server and backup server(s) with a maximum limit (say 4) with net addresses "00.A.B.server1", "00.A.B.server2", "00.A.B.server3" and "00.A.B.server4".

Each PIMapServer will have a database of records that will have information to resolve PI addresses. In memory copy of a region will have an array of records where each record will have the following format:

NetAddress	NetMask	Type	TTL	NAddr	Addr(1-4)

First two fields "NetAddress/NetMask" represents the PI address range of a network. "Type" will be either Domain/Referral/Individual/ SingleEntry/Default based on which a query and rest of the fields of a record have to be processed. A PI address can have maximum four mapped PA addresses. "Addr1", "Addr2", "Addr3", "Addr4" will hold the corresponding PA addresses and "NAddr" will hold the number of such addresses. The field "TTL" is a 32 bit integer measured in seconds which will hold same meaning and approach as defined in the specification of DNS[10]. When a server receives a query for an address "X", it extracts the record of the network based on "NetAddress/NetMask" and "X" from its database. If no matching record is found, a negative response is sent. Based on the "Type" of the record, the query is processed in the following manner.

Type=Domain:

This is the most common type. If a customer network would not like to maintain a map server opts for this option. In this case there will be one to one mapping between a PI address and corresponding PA addresses. The fields "Addr1"/"Addr2"/"Addr3"/"Addr4" will hold the PA Net Addresses corresponding to the PI address of the network. Server will send the matching record to the resolver with Type=Domain. Resolver will extract the user-id portion of "X" and find the corresponding mapped PA addresses based on "Addr1"/"Addr2"/...etc.

Theoretically, "A.B" portion of a PI address need not match with the "A.B" portion of the corresponding PA addresses. Consider a large corporate that has its corporate office and a branch office within the same region of a particular "A.B" and some other offices with different values of "A.B". The corporate can maintain a contiguous range of PI addresses for the ease of its operation. It needs to split entire PI address range based on its offices and assign the corresponding PA addresses. In order to minimize the path of a query it is desirable that "A.B" of a PI address and its corresponding mapped PA addresses belong to the same region.

Type=Referral:

This is used when an address within the domain "NetAddress"/"NetMask" has to be processed by another map server. The map server may itself be another regional server or a server within a customer network.

When a customer network would like to have a direct control for the mapping of its addresses it needs to opt for this option. "Addr1"/"Addr2"/"Addr3"/"Addr4" of the database entry will hold the pointer to the information associated to each map server. "NAddr" will hold the number of map servers that can be referred. Information of each server will hold the following values: PI address of the map server + Number of PA addresses to reach the map server + PA addresses of the map server. Any one of these map servers need to be queried for further processing. A server may act either in recursive mode or in iterative mode based on its implementation just like in DNS. A large corporate may have different offices and each (or some of them) may maintain a map server based on their policies.

When a server needs to handle a particular address separately, it needs to set "NetAddress" with that particular address and all the bits of "NetMask" will be set to "1". The "Type" field has to be set as "SingleEntry"(which is similar to the Type Address(A) in terms of DNS). If some of its addresses need to be handled separately but for the rest common rule may apply (like Type=Domain), records of the individual entries should be processed first and then for the rest. In these cases "Type" has to be set as "Default". So, a server of a

customer network may have database entries with Type=Domain/Referral /SingleEntry/Default. It makes sense for a server (or a master file) to have entries with Type=Default, but from the point of a resolver, it does not make any sense. So a server needs to extract the PA addresses and form a record with Type=SingleEntry and send it back to the resolver.

For a host having multiple interfaces, each interface may be assigned PA addresses supplied by all the service providers, but it is desirable that PI address gets mapped to only one of them (preferably for a CE router, the interface which will have the shortest path will be mapped PI address with the PA address associated with that CE router).

Type=Individual:

This is meant for the individual users opting for services like telephonic services that need to maintain PI address. With this option a mobile user may maintain its PI address after changing its service provider. A map server needs to maintain some networks with a range of PI addresses in its database. When a query for an address "X" is received, server needs to get the corresponding record where "Addr1" will hold the pointer to a open file descriptor (or pointer to the in memory copy) of a separate data file where there will be one to one mapping between PI address and its corresponding PA address of all the assigned PI addresses. These networks and assignment of individual PI addresses have to be done by the regional authority.

As with Type=Default, Type=Individual does not make any sense to a resolver. So, server needs to extract PA address and form a record with Type=SingleEntry and send it back to the resolver.

As stated above, this solution is based on the approach of DNS. For the ease of implementation and to make use of the existing source code related to DNS (e.g. BIND) most of the features have been taken from DNS. DNS supports multiple entry output, but they appear in a sequential manner. In order to make processing easier, they are arranged in a structured manner in this document.

IANA has assigned a port <IANA_TBD5> for its UDP/TCP based implementation.

5.1.1. Record Format

Each record (the way they will appear in a master file or will be used for communication) will have the following format:

NetAddress/NetMask + Type (8 bit unsigned int) + <TTL> + RDATA (Type specific information)

Record types are primarily the types of records as described above along with three other types: SOA (Start of a zone of authority), MPS (host with Type=SingleEntry that acts as a Map server for this zone) and DFL (Data File). These types are mainly useful in the context of processing AXFR/IXFR/NOTIFY/DFAXFR/DFIXFR messages.

Types are defined as follows:

Types	values	comments
SEN (SingleEntry)	1	same as type A(address) in DNS
MPS (MapServer)	2	Map server
DMN (Domain)	3	
DEF (Default)	4	
REF (Referral)	5	
SOA (Start of a zone)	6	
IND (Individual)	7	
DFL (Data File)	8	

RDATA of different types will appear as follows:

Type=SOA:
PI address of server+SERIAL+REFRESH+RETRY+EXPIRE+MINIMUM (meaning and values of SERIAL/REFRESH/RETRY/EXPIRE/MINIMUM are same as they were defined in section 3.3.13 of RFC 1035[11])

Type=(SEN/MPS):
NAddr(Number of addresses) + corresponding PA addresses

Type=(DMN/DEF):
NAddr(Number of addresses) + corresponding Net addresses

Type=REF:
NAddr(Number of map server) + for each map server (PI address of map server + NAddr(Number of addresses of map server) + corresponding PA addresses))

Type=IND:
NAddr(=1) + full path name of the data file

Type=DFL:
Data file name + SERIAL + Number of records in the data file(32 bit unsigned int)

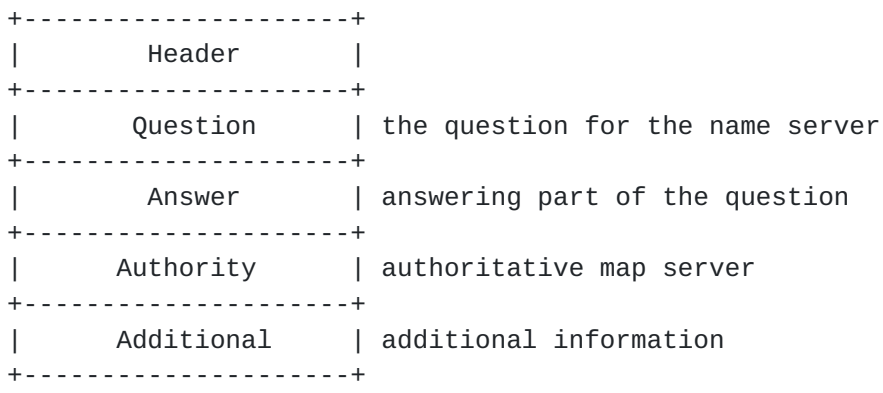
While used in communication data file name is used as its length (8 bit unsigned int) followed by the octets of the string.

TTL value of a record has to be set to 0 if it is not relevant or to accept the value associated with the record of SOA.

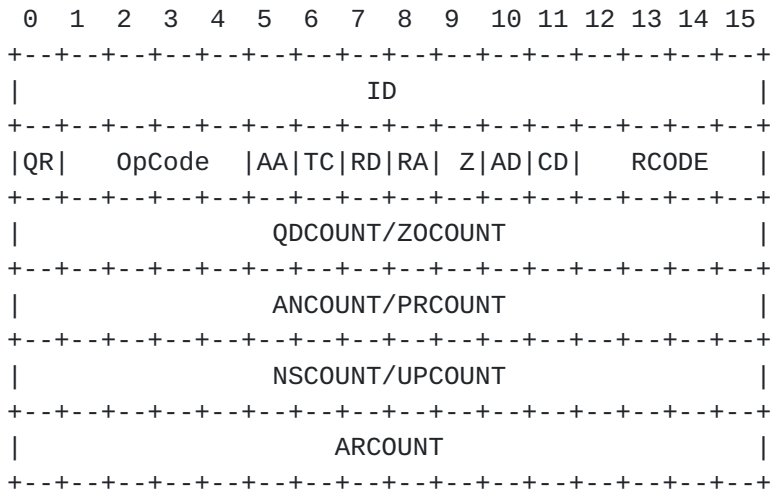
5.1.2. Messages

In order to support most of the features of DNS, message format has been retained almost same as that of DNS. So, all the relevant fields will be processed exactly in the same manner as that have been done in DNS and all the irrelevant issues have to be ignored. Rest of this section describes where and how changes have to be made.

As defined in RFC 1035, the top level format of message is divided into 5 sections (some of which are empty in certain cases) shown below:



The header section has been retained as defined in RFC 5395[12] as follows:



The question section will have two parts:
QType(one octet unsigned int)+QData.

Query types are defined as follows:

QTypes	values	comments
SEN	1	query for mapped PA address
SOA	6	query information related to SOA
DFL	8	query information related to data file
DFXFR	249	data file transfer
DFIXFR	250	incremental data file transfer
IXFR	251	incremental authoritative data file xfr
AXFR	252	authoritative data file transfer

QData will hold values based on QType.

Following section describes issues related to QType=SEN. Issues related to all other QTypes (i.e. related to file transfer) will be discussed afterwords.

For QType=SEN(1): QData=PI address that needs to be resolved.

The answer section, authority section and additional section will have a number of resource records where the number will be specified in the header.

On receiving a query, map server will return the matching record from its database. If response is address, the answer section will hold the record of any one of these two types: SEN/DMN.

If Type=DMN, resolver needs to extract the mapped addresses as described in section 5.1.

If Type=DMN, entire address range will appear in the form of NetAddress/NetMask. This will have advantages while catching data for any particular address, but getting the information of the entire address range.

If the response is referral, answer section will be empty and the authoritative section will hold the record with Type=REF.

If server supports recursion, for each iterative process that it receives a record with Type=REF, it needs to push the record to the additional section of the message that needs to be sent to the resolver. So, additional section will hold the records of Type=REF of the chain of the tree through which PA addresses have been resolved.

5.1.3. Master file and data file

Section 5 of RFC 1035 states:

"Master files are text files that contain RRs in text form. Since the contents of a zone can be expressed in the form of a list of RRs a master file is most often used to define a zone, though it can be used to list a cache's contents."

Section 5.1 of RFC 1035 states:

"The format of these files is a sequence of entries. Entries are predominantly line-oriented, though parentheses can be used to continue a list of items across a line boundary, and text literals can contain CRLF within the text. Any combination of tabs and spaces act as a delimiter between the separate items that make up an entry. The end of any line in the master file can end with a comment. The comment starts with a ";" (semicolon)."

Master files follow the same approach and format in the line of DNS as described in section 5 of RFC 1035 with necessary differences.

An example master file may look like as follows:

```
@ "PI NetAddr"/"Net Mask"  SOA  "PI address of primary server" (
                                 20     ; SERIAL
                                 7200   ; REFRESH
                                 600    ; RETRY
                                 3600000; EXPIRE
                                 60)    ; MINIMUM
"PI NetAddr"/"Net Mask"    MPS  0  NAddr "PA addresses"
"PI NetAddr"/"Net Mask"    SEN  0  NAddr "PA addresses"
"PI NetAddr"/"Net Mask"    DMN  0  NAddr "Net addresses"
"PI NetAddr"/"Net Mask"    DEF  0  NAddr "Net addresses"
"PI NetAddr"/"Net Mask"    IND  0  NAddr(=1) "Data file name"
```

A data file contains a sequence of entries where each entry appears in a separate line. Each entry is a mapping between a PI address and its associated PA address separated by space(s). Entries are generally sorted with PI address. As in case of master file comments can be inserted with the start of a ";" (semicolon) that will end at the end of the line. Data files are commonly associated with the map servers maintained by regional authority, but they are not generally associated with the map servers maintained by individual customer networks. A data file entry may appear to be as follows:

"PI Address" NAddr "PA Addresses"

A map server may have a number of data files. These files have to be defined in another file (a supporting file, the way boot file "named.boot" is used in BIND) that will have information of each of them. An entry in that file will follow the same format of a record (Type=DFL) and will have the following fields:

"PI NetAddr"/"NetMask" Type(DFL) TTL "Data File Name" SERIAL "Number of records".

This file will be used to process message with QType=DFL which will be used to support data file transfer/incremental data file transfer.

For QType=DFL(8): QData="PI NetAddr"/"NetMask" of the desired network
For QType=SOA(6): QData="PI NetAddr"/"NetMask" of the desired zone

A map server will return a record of Type=DFL on receiving a query with QType=DFL where as it will return a record of Type=SOA on receiving a query with QType=SOA.

5.1.4. Zone maintenance and transfers

Section 4.3.5 of RFC 1034 states:

"The general model of automatic zone transfer or refreshing is that one of the name servers is the master or primary for the zone. Changes are coordinated at the primary, typically by editing a master file for the zone. After editing, the administrator signals the master server to load the new zone. The other non-master or secondary servers for the zone periodically check for changes (at a selectable interval) and obtain new zone copies when changes have been made.

To detect changes, secondaries just check the SERIAL field of the SOA for the zone. In addition to whatever other changes are made, the SERIAL field in the SOA of the zone is always advanced whenever any change is made to the zone."

Section 1.2 of RFC 5936 states:

"A DNS implementation is not required to support AXFR, IXFR, and NOTIFY, but it should have some means for maintaining name server coherency. A general-purpose DNS implementation will likely support AXFR (and in the same vein IXFR and NOTIFY), but turnkey DNS implementations may exist without AXFR."

Zone maintenance and transfer will follow the same approach as DNS with few minor updates. Frequency of update of data files will be high compared to the frequency of update of master file. That is why

transfer(/incremental transfer) of data file has been treated separately from the transfer(/incremental transfer) of master file.

For all the messages of QType=AXFR/DFXFR/IXFR/DFIXFR, QData="PI NetAddr"/"NetMask" of the desired zone or the desired network. NOTIFY message needs to include which file has been updated followed by the related information. So, if master file has been changed, NOTIFY message with query type SOA will be sent and query type DFL will be sent if a data file has been changed.

Transfer of master file will be same as transfer of master file in DNS followed by transfer of all the data files. i.e. processing of AXFR will have the same approach as DNS followed by DFXFR for all the data files. In order to make this happen, at the end of transferring the contents of the master file, server (of AXFR message) needs to send NOTIFY message for all of the data files belonging to that zone to the client(i.e. the secondary server). Processing of NOTIFY of a data file by the secondary server needs to send DFIXFR to the primary if data file already exist; otherwise it needs to send DFXFR. Incremental update of master file (IXFR) will be same as IXFR in DNS with a minor update. If client of IXFR finds a new data file gets introduced, it calls DFXFR corresponding to that data file. Similarly if an entry of a data file gets deleted, client deletes corresponding data file.

Processing of DFXFR will have same approach of AXFR in DNS. Similarly processing of DFIXFR will have same approach as IXFR in DNS. While transferring a data file record, an equivalent record of type SEN needs to be sent with the values of PI address and mapped PA address(es) from the record of data file. Where ever a record of type SOA is sent while processing AXFR/IXFR in case of DNS, record of type DFL needs to be sent while processing DFXFR/DFIXFR.

For AXFR, IXFR and NOTIFY in DNS, one needs to follow RFC 5936[13], RFC 1995[14] and RFC 1996[15] respectively.

6. Issues related to IP mobility

An interface of a customer network may have several IP addresses (e.g. for a multihomed customer site, each interface will have multiple global unicast addresses also it may have private addresses). For a mobile node that has been moved to a customer network which gets service from a service provider and maintains private IP addresses, will have at least three IP addresses; provider assigned unicast address, private address and its permanent "Home Address". The "Home Address" will be aliased with the provider assigned address (i.e. the co-located care-of address). So the interface structure needs to have an additional field to hold the

value of care-of address. The PCB structure will have an additional field 'inp_lcladdr'. So 'inp_lcladdr' will have the current provider assigned address that a foreign node needs to use for communication. The field 'inp_laddr' that is used to hold the value of local address will hold the value of "Home Address" of a mobile node. Similarly, PCB needs to introduce another field 'inp_fcladdr' to support the destination address to be mobile. The existing field 'inp_faddr' which is used to address a foreign address will hold the value of "Home Address" of the mobile node. Customers with PI address who would like to have mobility support, the mapped address will be considered as the "Home Address" of the mobile node.

An outgoing packet from a mobile node in a foreign site needs to be stacked with the associated care-of address. While initiating communication, the 'bind' system call needs to go through the interface list and fetch the associated structure to check whether the source address is aliased or not and needs to fill the value of 'inp_lcladdr' of PCB accordingly.

When TCP receives a SYN for connection establishment, it allocates a PCB and assigns the values for 'inp_laddr', and related fields. During this phase, TCP also needs to check whether the local address is aliased or not (based on the fields of interface structure; which is applicable for a mobile node at foreign site) and needs to fill the values of 'inp_lcladdr' accordingly. Similarly if destination address is found to be aliased, based on the stacking type, it needs to fill up the field 'inp_fcladdr'.

IP address stacking can be performed with the approach introduced in section 6.4 of RFC6275[9]. RFC6275 talks about the stacking of IP addresses for a destination address (Let us call it as type 0 stacking). Two more types of stacking need to be introduced; type 1 stacking where only source address will appear in the stack and type 2 stacking where both source address and destination address will appear in the stack with a particular type of ordering.

Protocol output routine like 'tcp_output' or 'udp_output' needs to fill the IP packet in the following manner.

If the socket contains a valid 'inp_lcladdr', use 'inp_lcladdr' as the source address and 'inp_laddr' will appear in the stack. If the socket contains a valid 'inp_fcladdr' use 'inp_fcladdr' as the destination address and 'inp_faddr' will appear in the stack. If only 'inp_fcladdr' contains a valid address where as 'inp_lcladdr' is NULL, use type 0 stacking. If only 'inp_lcladdr' contains a valid address where as 'inp_fcladdr' is set as NULL, use type 1 stacking. If both 'inp_lcladdr' and 'inp_fcladdr' contains valid addresses, use type 2 stacking.

Protocol input routine like 'tcp_input' or 'udp_input' needs to process the packet in the reverse order based on the type of stacking. For type 0 stacking, use the address in the stack as the destination address; for type 1 stacking, use the address in the stack as the source address; for type 2 stacking use both source address and destination address from the stack.

6.1. Changes expected with the specifications related to IP mobility

RFC6275 demands correspondent node binding from mobile nodes for route optimization. This binding is required when a connection gets established as well as when the mobile node changes it address space. There are application like HTTP which opens up multiple connections on the run time which are very short lived. If mobile nodes need to send binding messages for all the connections, network will be unnecessarily congested. This congestion can be avoided with the establishment of binding at the time of connection establishment itself. So, if TCP server happens to be mobile, it will set the value of 'inp_lcladdr' in the stack while sending SYN+ACK. TCP client which initiates communication through 'connect' needs to set 'inp_fcladdr' field on receiving TCP+ACK. With this approach correspondent node binding messages need to be sent only when a mobile node changes its position from one address space to another.

Route optimization is not applicable to applications which are of multicast type. In these cases packets need to be forwarded with the mechanism of reverse tunneling with the approach of "IP Encapsulation within IP" as defined in RFC2003. In order to support packet delivery with route optimization method as well as with "Encapsulating Delivery Style" based on the application type the protocol control block needs to introduce another field 'inp_hagentaddr' to hold the address of the home agent of the mobile node. The interface structure also needs to have same field. The 'bind' system call needs to go through the interface list to fetch 'inp_hagentaddr' to the PCB along with 'inp_lcladdr' as described earlier. So, protocol output routines like 'tcp_output', 'udp_output' need to fill up the packets based on the application type. In "Encapsulating Delivery Style" packets need to be formed in the following manner.

The inner IP header will contain
   Source Address: Home address of the mobile node
   (i.e. 'inp_laddr')
   Destination address: Address of the correspondent node
   (i.e. 'inp_faddr')
The outer IP header will contain
   Source Address: co-located care of address of the mobile node
   (i.e. 'inp_lcladdr')

Destination Address: Address of the home agent of the mobile node (i.e. 'inp_hagentaddr')
Protocol field: IP in IP

7. Refinements over existing IPv6 specification

As IPv6 was envisioned long before some of the newer technologies e.g. MPLS came into picture, some refinements can be made over the existing specification. These considerations are related to bandwidth usages and performance inside switches. Experimental results show that smaller packet size gives better result for the processing of RT packets. So, it is desirable to have IP packet header to be as small as possible.

As described earlier, evaluation of the parameters nMaxInterASTopNodes, nMaxInterASBottomNodes and nMaxASNodes is geo-political and have to be decided by IANA. Once these parameters are determined with mutual agreements, values of pA, pB, pC and prefix length of user id can be determined. With 64 bit address space, IP header will be reduced by 16 bytes.

The 'flow label' field of IPv6 packet header may not be of any use with MPLS is in use. ATM used to have 4 priority classes. The first specification of IPv6 RFC-1883 used a 4 bit type of service field along with a 24 bit flow label field. These two were modified to a 8 bit type of service field and a 20 bit flow label field in the current spec RFC-2460. Too many priority classes may increase complexities to process inside switches. If type of service field of IPv6 header may be reduced to be of 4 bit length as it was stated in RFC-1883 and 'flow label' field gets removed, another three bytes may be reduced from the IPv6 header.

The field 'Hop Limit' has got a 8 bit value in the existing spec. The role of this field needs to be discussed properly with a large address space.

RFC4862[16] introduces the concept of "Stateless auto configuration" with the goal in mind that no manual configuration is required by individual machines before connecting them to the network. It generates a link local address with a link-local prefix and the link address (e.g. Ethernet/E.164 for ISDN) first. This link local address is used to configure global unicast address and any other configurable parameters based on router advertisement. Global unicast addresses are generated by the prefix supplied by the router advertisement and the link specific interface identifier. This identifier can be as large as 64 bit length. So irrespective of the size of the network (it may be 10000 or 100 or even less than that) every subnet of a customer network will consume a 64 bit equivalent

addresses. This seems to be a huge blunder. What is expected is the length of the interface identifier is equivalent to support the number of nodes supported by that subnet. In order to achieve this, the router itself or a server in that subnet needs to maintain a storage which will generate the interface identifier based on the request from individual hosts. It may be desirable that interface identifiers are generated from DHCP servers. With the option of generating interface identifier through DHCP, changes in the auto configuration process can be looked at as follows:

From the point of view of a host, it can be considered as a two step process. Host needs to send Router Solicitations message to find out the presence of a router. Router Advertisement message should include an option field which will inform whether prefix information should be configured through Router Advertisement or through DHCP. Host needs to send a request message to get the interface identifier. If both the information needs to be obtained from a DHCP server they can be obtained through a single message.

From the server's point of view, it needs to maintain a database for a mapping of the link-layer address and subnet specific interface identifier. Lifetime of an interface identifier has to be processed in the usual manner the way existing DHCP implementation treats IP addresses.

There seem to be another possible danger to obtain prefix information through Router Advertisement. As the Router Advertisement comes in the form of ICMP messages, once it is received by the ICMP layer, it looses information from which interface the message has been received (This problem arises for hosts that are having multiple interfaces and not all of them are attached to the same subnet). So, auto configuration of a host has to be performed one interface at a time by making all other interfaces disabled. Once configuration of all the interfaces are done, all of them have to be enabled.

If it is expected that hosts should reconfigure their addresses dynamically based on Router Advertisement message, Router Advertisement needs to generate a special message for a certain amount of time that needs to include old prefix and the corresponding new prefix in the message.

In order to support multihoming[8], prefix information needs to include the fields 'default router' and 'next hop address' to reach the default router for each of the prefixes.

In a 64 bit architecture, link-local address can be formed with a link-local prefix and link-layer address in a suitable manner; say it can be formed with a 4 bit link-local prefix followed by a 60 bit

link-layer address. IPv6 supports Modified EUI-64 format for hardware that supports 48 bit addressing by inserting a padding of 16 bit (FF FE) in between company_id and manufacturer selected extension identifier. In order to make things work, this padding has to be reduced to 12 bit. For hardware that support E.164 format, uses a 15 digits number in BCD format followed by a padding of four bits set to 1111. Thus in this case, link local address can be formed with the link-local prefix followed by the most significant 60 bit of E.164 format.

Section 3.1 of RFC 7421[18] states "It is sometimes suggested that assigning a prefix such as /48 or /56 to every user site (including the smallest) as recommended by [RFC6177] is wasteful. In fact, the currently released unicast address space, 2000::/3, contains 35 trillion /48 prefixes ((2**45 = 35,184,372,088,832), of which only a small fraction have been allocated. Allowing for a conservative estimate of allocation efficiency, i.e., an HD-ratio of 0.94 [RFC4692], approximately 5 trillion /48 prefixes can be allocated. Even with a relaxed HD-ratio of 0.89, approximately one trillion /48 prefixes can be allocated. Furthermore, with only 2000::/3 currently committed for unicast addressing, we still have approximately 85% of the address space in reserve. Thus, there is no objective risk of prefix depletion by assigning /48 or /56 prefixes even to the smallest sites."

So, each customer network can be assigned a /48 prefix, i.e 80 bits address space.

In IPv4, class A(24 bits), class B(16 bits) and class C(8 bits) networks were classified with the thoughts in mind that there will be very few large networks (class A), a large number of mid sized networks (class B) and a very large number of small sized networks (class C). If we go back to the assignment of address space in IPv4, before the emergence of CIDR, class B address space were getting exhausted very fast. Moreover, it was realized that 16 bits class B address space is way too large compared to the requirement of most of the mid sized networks [2]. So, if we look at the actual need of customer networks, on the average, it needs less than 16 bits (say, m bits) address space.

So, if 80 bits address space is used for each customer network in IPv6, more than 64 bits will remain unused on the average. In effect, out of 128 bits, less than 64 bits will be of actual use. i.e. if RFC 7421 justifies 128 bits address space as good enough for the need of this world, 64 bits address space will satisfy the need of this world when customer networks are assigned address space based on their sizes.

Where ever one network gets satisfied with 80 bits address space based on RFC 7421, 2^(16-m) networks get satisfied with 16 bits address space if customer networks are assigned address space based on their sizes. If total M networks with /48 prefixes can be satisfied with 128 bits address space based on RFC 7421, total M*2^(16-m) networks will be satisfied with 64 bits address space once networks are assigned address space based on their sizes.

8. Distributed processing and Multicasting

With the inherent hierarchy involved in this architecture, distributed applications can also be structured in a suitable manner. Say, for a commonly used web based application a master level server will be there at every top level node. Any change that might happen in the application, has to be synchronized within these master level servers first. There might be servers at the middle layer (inside each inter-AS-bottom) inside each top level node. Once the changes get reflected at the master node, all the servers at the middle layer needs to update themselves with their master level node. This will reduce network traffic substantially. Inherent hierarchy in the architecture will also help establishing multicast tree in the similar manner. Work on these issues can be progressed only after this architecture gets approved.

9. Transition to real IP from private IP

Both CIDR and mesh structured hierarchy expects a VLSM tree at the bottom. In VLSM, in real IP space with provider assigned (PA) addresses, assignment of network resources has to be associated with the address space to be used with the type of service. Within a typical switch supporting multiple types of ports, a line card of strength OC48 can be replaced with 4 line cards of strength OC12. An OC12 card may also be replaced with 4 OC3 cards. An OC12 card may be attached to another switch with DS3 ports and so on. When it reaches to the customer network port density of a switch has to be directly proportional to the address block that a customer network will be assigned to. i.e. each customer network has to be assigned a block of address space (say, 128, 256, 512, 1K, 2K etc). Within the switch these ports have to be assigned net address/net mask the way VLSM works.

In IPv4 environment, providers have provided services in terms of bandwidth of the ports say, 2 Mbps/4 Mbps/1 Gbps line etc. If these ports were assigned addresses based on the number of users of the customer network, transition from private IP to real IP is simple. Consider a switch that has supplied 2 Mbps line to a set of customers with number of users within 1K to 2k, each of them will be assigned a block of 2K each. But if number of users are not proportional to the

bandwidth used, say same 2 Mbps line were used to customers of sizes 1K, 2K 4K and 16K respectively reorganization will be needed if possible. This rearrangement may be possible within the switch itself or by connecting ports of appropriate sizes from different switch, otherwise each of them has to be assigned an address block of 16K each or with the way VLSM works whatever is suitable. So, address block assignment in the VLSM tree has to grow in a bottom up approach.

Thus, transition of existing provider network without (or very little) rearrangement to a real IP space with CIDR based approach is apparently not a difficult job. In a CIDR based approach, sizes of the VLSM trees are heterogeneous that leads to number of routing entries to be very high. Mesh structured hierarchy is convenient to reduce the routing overhead as well as for distribution of network resources in a suitable manner in the long run. To covert CIDR based approach to mesh structured hierarchy requires reorganization mainly in the routing domain and by splitting trees of very large sizes (>24 bit address space) at the top.

Mesh structured hierarchy makes use of a large address space and distributes the entire space into some regions and sub regions inside each region by maintaining flat address space in each layer for the convenience of routing and distribution. It shows that 64 bit address space is good enough for all practical purposes. If address space gets assigned based on the actual need of the customer networks, there will be lots of unused address space within 64 bit address space. If CIDR based hierarchy is maintained, unused address space will be much higher.

10. IANA Consideration

IANA has assigned protocol number <IANA_TBD1> for VLSM tree routing protocol. IANA has 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. IANA has assigned a port number <IANA_TBD5> and service name <IANA_TBD6> for PI address resolution for both TCP and UDP.

11. Security Consideration

This document does not include any security related issues.

12. Acknowledgments

The author would like to thank to Professor Amitava Datta of University of Western Australia for his review and constructive

comments.

13. Normative References

[1] Nordmark, E. and R. Gilligan, "Basic Transition Mechanisms for IPv6 Hosts and Routers", RFC 4213, October 2005.

[2] Fuller V., Li. T., "Classless Inter-Domain Routing (CIDR): The Internet Address Assignment and Aggregation Plan", RFC 4632, August 2006.

[3] Huston, G., "Commentary on Inter-Domain Routing in the Internet", RFC 3221, December 2001.

[4] Q. Vohra, E. Chen., "BGP Support for Four-octet AS Number Space", RFC 4893, May 2007.

[5] Srisuresh, P. and K. Egevang, "Traditional IP Network Address Translator (Traditional NAT)", RFC 3022, January 2001.

[6] J. Moy., "OSPF Standardization Report", RFC 2329, April 1998

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

[8] S. Bandyopadhyay, "Solution for Site Multihoming in a Real IP Environment", <draft-shyam-site-multi-44> work in progress.

[9] C. Perkins, Ed., D. Johnson, J. Arkko, "Mobility Support in IPv6" RFC 6275, July 2011.

[10] P.V. Mockapetris., "Domain names - concepts and facilities", RFC 1034, November 1987.

[11] P.V. Mockapetris, "Domain names - implementation and specification", RFC 1035, November 1987.

[12] D. Eastlake 3rd, "Domain Name System (DNS) IANA Considerations", RFC 5395, November 2008.

[13] E. Lewis, A. Hoenes, Ed., "DNS Zone Transfer Protocol (AXFR)", RFC 5936, June 2010.

[14] M. Ohta, "Incremental Zone Transfer in DNS", RFC 1995, August 1996.

[15] P. Vixie, "A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY)", RFC 1996, August 1996.

[16] S. Thomson, T. Narten, T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, September 2007.

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

[18] B. Carpenter, Ed., T. Chown, F. Gont, S. Jiang, A. Petrescu, A. Yourtchenko, "Analysis of the 64-bit Boundary in IPv6 Addressing", RFC 7421, January 2015.

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

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

14. Informative References

[21] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.

[22] Rekhter, Y., and T., Li, "A Border Gateway Protocol 4 (BGP-4)",RFC 1771, March 1995.

[23] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification, RFC 1883, December 1995.

[24] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

[25] Rosen, E., Viswanathan, A. and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, January 2001.

15. Author's Address

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