                   BGP Prefix Independent Convergence
                     draft-ietf-rtgwg-bgp-pic-00.txt

Abstract

In the network comprising thousands of iBGP peers exchanging millions
of routes, many routes are reachable via more than one path. Given
the large scaling targets, it is desirable to restore traffic after
failure in a time period that does not depend on the number of BGP
prefixes. In this document we proposed an architecture by which
traffic can be re-routed to ECMP or pre-calculated backup paths in a
timeframe that does not depend on the number of BGP prefixes. The
objective is achieved through organizing the forwarding chains in a
hierarchical manner and sharing forwarding elements among the maximum
possible number of routes. The proposed technique achieves prefix
independent convergence while ensuring incremental deployment,
complete transparency and automation, and zero management and
provisioning effort. It is noteworthy to mention that the benefits of
BGP-PIC are hinged on the existence of more than one path whether as
ECMP or primary-backup.

Status of this Memo

   Internet-Drafts are working documents of the Internet Engineering
   Task Force (IETF), its areas, and its working groups. Note that
   other groups may also distribute working documents as Internet-
   Drafts.

Table of Contents

1. Introduction

   As a path vector protocol, BGP is inherently slow due to the
   serial nature of reachability propagation. BGP speakers exchange
   reachability information about prefixes[2][3] and, for labeled
   address families, namely AFI/SAFI 1/4, 2/4, 1/128, and 2/128, an
   edge router assigns local labels to prefixes and associates the
   local label with each advertised prefix such as L3VPN [8], 6PE
   [9], and Softwire [7] using BGP label unicast technique[4]. A BGP
   speaker then applies the path selection steps to choose the best
   path. In modern networks, it is not uncommon to have a prefix
   reachable via multiple edge routers. In addition to proprietary
   techniques, multiple techniques have been proposed to allow for
   more than one path for a given prefix [6][11][12], whether in the
   form of equal cost multipath or primary-backup. Another more
   common and widely deployed scenario is L3VPN with multi-homed VPN
   sites.

   This document proposes a hierarchical and shared forwarding chain
   organization that allows traffic to be restored to pre-calculated
   alternative equal cost primary path or backup path in a time
   period that does not depend on the number of BGP prefixes. The
   technique relies on internal router behavior that is completely
   transparent to the operator and can be incrementally deployed and
   enabled with zero operator intervention.

   1.1. Conventions used in this document

   The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL
   NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL"
   in this document are to be interpreted as described in RFC-2119
   [1].

   In this document, these words will appear with that interpretation
   only when in ALL CAPS. Lower case uses of these words are not to
   be interpreted as carrying RFC-2119 significance.

1.2. Terminology

This section defines the terms used in this document. For ease of use, we will use terms similar to those used by L3VPN [8]

o BGP prefix: It is a prefix P/m (of any AFI/SAFI) that a BGP
   speaker has a path for.

o IGP prefix: It is a prefix P/m (of any AFI/SAFI) that is learnt
   via an Interior Gateway Protocol, such as OSPF and ISIS, has a
   path for. The prefix may be learnt directly through the IGP or
   redistributed from other protocol(s)

o CE: It is an external router through which an egress PE can
   reach a prefix P/m.

o Ingress PE, "iPE": It is a BGP speaker that learns about a
   prefix through another IBGP peer and chooses that IBGP peer as
   the next-hop for the prefix.

o Path: It is the next-hop in a sequence of unique connected
   nodes starting from the current node and ending with the
   destination node or network identified by the prefix.

o Recursive path: It is a path consisting only of the IP address
   of the next-hop without the outgoing interface. Subsequent
   lookups are needed to determine the outgoing interface.

o Non-recursive path: It is a path consisting of the IP address
   of the next-hop and one outgoing interface

o Primary path: It is a recursive or non-recursive path that can
   be used all the time. A prefix can have more than one primary
   path

o Backup path: It is a recursive or non-recursive path that can
   be used only after some or all primary paths become unreachable

o Leaf: A leaf is container data structure for a prefix or local
   label. Alternatively, it is the data structure that contains
   prefix specific information.

o IP leaf: Is the leaf corresponding to an IPv4 or IPv6 prefix

o Label leaf. It is the leaf corresponding to a locally allocated
   label such as the VPN label on an egress PE [8].

o Pathlist: It is an array of paths used by one or more prefix to
   forward traffic to destination(s) covered by a IP prefix. Each
   path in the pathlist carries its "path-index" that identifies
   its position in the array of paths. A pathlist may contain a
   mix of primary and backup paths

o OutLabel-Array: Each labeled prefix is associated with an
   OutLabel-Array. The OutLabel-Array is a list of one or more
   outgoing labels and/or label actions where each label or label
   action has 1-to-1 correspondence to a path in the pathlist. It
   is possible that the number of entries in the OutLabel-array is
   different from the number of paths in the pathlist and the ith
   Outlabel-Array entry is associated with the path whose path-
   index is "i". Label actions are: push the label, pop the label,
   or swap the incoming label with the label in the Outlabel-Array
   entry. The prefix may be an IGP or BGP prefix

o Adjacency: It is the layer 2 encapsulation leading to the layer
   3 directly connected next-hop

o Dependency: An object X is said to be a dependent or Child of
   object Y if Object Y cannot be deleted unless object X is no
   longer a dependent/child of object Y

o Route: It is a prefix with one or more paths associated with
   it. Hence the minimum set of objects needed to construct a
   route is a leaf and a pathlist.

[2](#). Constructing the Shared Hierarchical Forwarding Chain

   2.1. Databases

   The Forwarding Information Base (FIB) on a router maintains 3 basic
   databases

o Pathlist-DB: A pathlist is uniquely identified by the list of
   paths. The Pathlist DB contains the set of all shared pathlists

o Leaf-DB: A leaf is uniquely identified by the prefix or the label

o Adjacency-DB: An adjacency is uniquely identified by the outgoing
   layer 3 interface and the IP address of the next-hop directly
   connected to the layer 3 interface. Adjacency DB contains the
   list of all adjacencies

2.2. Constructing the forwarding chain from a downloaded route

1. A prefix with a list of paths is downloaded to FIB from BGP. For
   labeled prefixes, an OutLabel-Array and possibly a local label
   (e.g. for a VPN [8] prefix on an egress PE) are also downloaded

2. If the prefix does not exist, construct a new IP leaf from the
   downloaded prefix. If a local label is allocated, construct a
   label leaf from the local label

3. Construct an OutLabel-Array and attach the Outlabel array to the
   IP and label leaf

4. The list of paths attached to the route is looked up in the
   pathlist-DB

5. If a pathlist PL is found

    a. Retrieve the pathlist

6. Else

    a. Construct a new pathlist

    b. Insert the new pathlist in the pathlist-DB

    c. Resolve the paths of the pathlist as follows

    d. Recursive path:

        i. Lookup the next-hop in the leaf-DB

       ii. If a leaf with at least one reachable path is found, add
            the path to the dependency list of the leaf

      iii. Otherwise the path remains unresolved and cannot be used
            for forwarding

    e. Non-recursive path

        i. Lookup the next-hop and outgoing interface in the
           adjacency-DB

       ii. If an adjacency is found, add the path to the dependency
            list of adjacency

      iii. Otherwise, create a new adjacency and add the path to
            its dependency list

7. Attach the leaf(s) as (a) dependent(s) of the pathlist

As a result of the above steps, a forwarding chain starting with a leaf and ending with one or more adjacency is constructed. It is noteworthy to mention that the forwarding chain is constructed without any operator intervention at all.

2.3. Examples

This section outlines three examples that we will use for illustration for the rest of the document. The first two examples use a standard multihomed VPN [8] prefix in a BGP-free core running LDP [5] or segment routing on MPLS [14]. The third example uses inter-AS option C [8] with 2 domains running segment routing [14] or LDP [5] in the core

The topology for the first two examples is depicted in Figure 1.
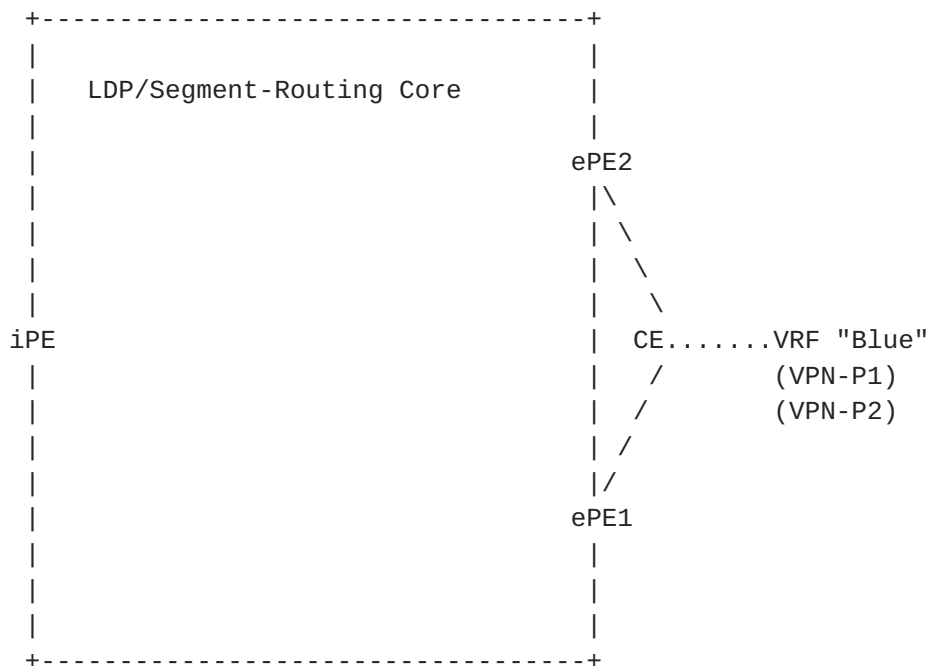
```
    +-----------------------------------+
    |                                   |
    |    LDP/Segment-Routing Core       |
    |                                   |
    |                              ePE2
    |                               |\
    |                               | \
    |                               |  \
    |                               |   \
     iPE                            |  CE.......VRF "Blue"
    |                               |   /       (VPN-P1)
    |                               |  /        (VPN-P2)
    |                               | /
    |                               |/
    |                              ePE1
    |                               |
    |                               |
    |                               |
    +-----------------------------------+
        Figure 1 VPN prefix reachable via multiple PEs
```

The first example is an illustration of ECMP while the second example is an illustration of primary-backup paths. The third example illustrate how to handle limited hardware capability.

2.3.1. Example 1: Forwarding Chain for iBGP ECMP

Consider the case of the ingress PE (iPE) in the multi-homed VPN prefixes depicted in Figure 1. Suppose the iPE receives route advertisements for the VPN prefixes VPN-P1 and VPN-P2 from two egress PEs, ePE1 and ePE2 with next-hop BGP-NH1 and BGP-NH2, respectively. Assume that ePE1 advertise the VPN labels VPN-L11 and

VPN-L12 while ePE2 advertise the VPN labels VPN-L21 and VPN-L22 for

VPN-P1 and VPN-P2, respectively. Suppose that BGP-NH1 and BGP-NH2
are resolved via the IGP prefixes IGP-P1 and IGP-P2, which also
happen to have 2 ECMP paths with IGP-NH1 and IGP-NH2 reachable via
the interfaces I1 and I2. Suppose that local labels (whether LDP[5]
or segment routing [14]) on the downstream LSRs for IGP-P1 and IGP-
P2 are assign the LDP labels LDP-L1 and LDP-L2 to the prefixes IGP-
P1 and IGP-P2. The forwarding chain on the ingress PE "iPE" for the
VPN prefixes is depicted in Figure 2.

```
     BGP OutLabel Array
         +---------+
         | VPN-L11 |
     +--->+---------+
     |    | VPN-L21 |
     |    +---------+                 IGP OutLabel Array
     |                                   +---------+
     |                                   | LDP-L11 |
     |                               +-->+---------+
     |                               |   | LDP-L21 |
  VPN-P1------+                      |   +---------+
            |                        |
            |                        |
            |                  IGP-P1-----+
            |                     ^       |
            |                     |       |
            V                     |       V  IGP Pathlist
         +--------+               |    +-------------+
         |BGP-NH1 |---------------+    | IGP-NH1, I1 |------>adj1
    BGP  +--------+                    +-------------+
  Pathlist |BGP-NH2 |----+               | IGP-NH2, I2 |------>adj2
         +--------+    |               +-------------+
             ^         |                     ^
             |         |                     |
             |         |                     |
             |     IGP-P2----------------+
             |         |
             |         |
  VPN-P2------+        |       +---------+
      |                |       | LDP-L12 |
      |                +--->+---------+
      |                       | LDP-L22 |
      |                       +---------+
      |     +---------+     IGP OutLabel Array
      |     | VPN-L12 |
    +--->+---------+
          | VPN-L22 |
          +---------+
       BGP OutLabel Array
```
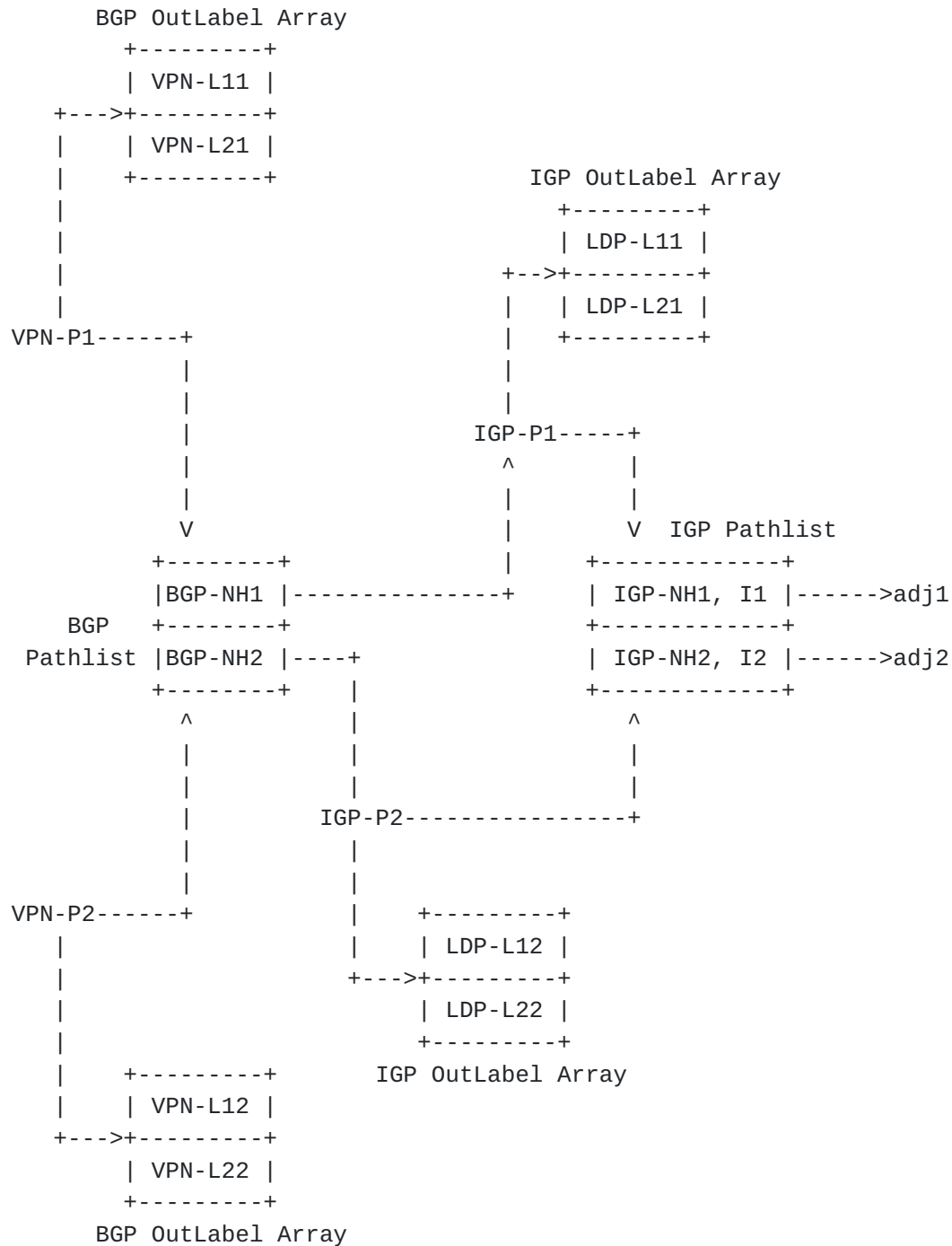
         Figure 2 Forwarding Chain for VPN Prefixes with iBGP ECMP

   The structure depicted in Figure 2 illustrates the two important
   properties discussed in this memo: sharing and hierarchy. We can
   see that the both the BGP and IGP pathlists are shared among
   multiple BGP and IGP prefixes, respectively. At the same time, the
   forwarding chain objects depend on each other in a child-parent
   relation instead of being collapsed into a single level.

[2.3.2](#). **Example 2: Primary Backup Paths**

   Consider the egress PE ePE1 in the case of the multi-homed VPN
   prefixes in the BGP-free LDP core depicted in Figure 1. Suppose ePE1
   determines that the primary path is the external path but the backup
   path is the iBGP path to the other PE ePE2 with next-hop BGP-NH2.
   ePE2 constructs the forwarding chain depicted in Figure 1. We are
   only showing a single VPN prefix for simplicity. But all prefixes
   that are multihomed to ePE1 and ePE2 share the BGP pathlist


```
                        BGP OutLabel Array
    VPL-L11                +---------+
  (Label-leaf)---+---->|Unlabeled|
                 |       +---------+
                 |       | VPN-L21 |
                 |       | (swap)  |
                 |       +---------+
                 |            ^
                 |            |          BGP Pathlist
                 |            |        +------------+     Connected route
                 |            |        |   CE-NH    |------>(to the CE)
                 |            |        |path-index=0|
                 |            |        +------------+
                 V            |        |  VPN-NH2   |
     VPN-P1 ------------------+------>|  (backup)  |------>IGP Leaf
   (IP prefix leaf)                   |path-index=1|    (Towards ePE2)
                                      +-----+------+
```
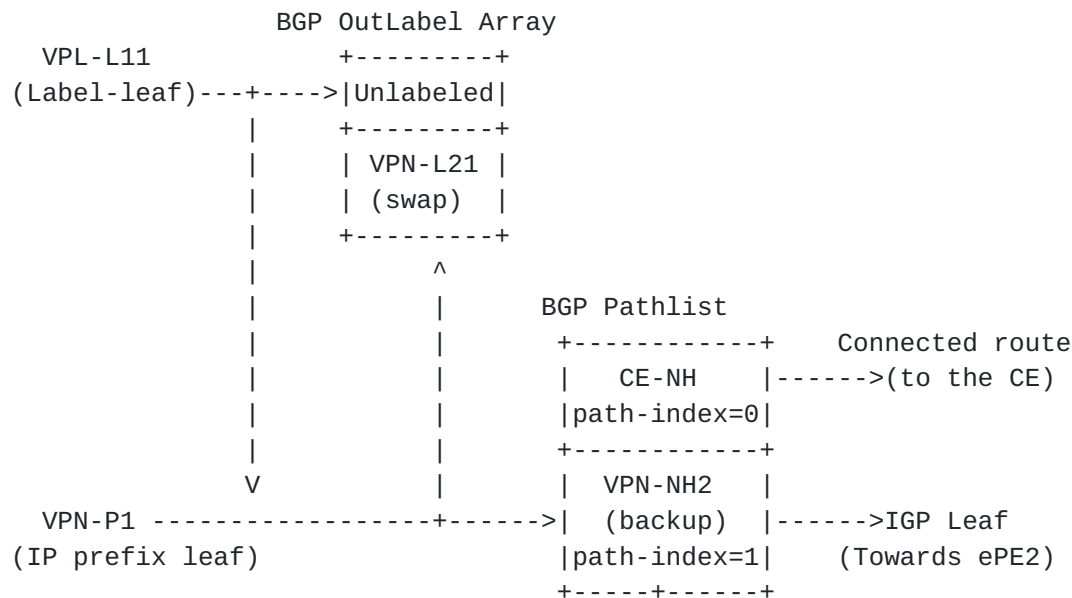
   Figure 3 : VPN Prefix Forwarding Chain with eiBGP paths on egress PE

   The example depicted in Figure 3 differs from the example in Figure
   2 in two main aspects. First as long as the primary path towards the
   CE (external path) is useable, it will be the only path used for
   forwarding while the OutLabel-Array contains both the unlabeled
   label (primary path) and the VPN label (backup path) advertised by
   the backup path ePE2. The second aspect is presence of the label
   leaf corresponding to the VPN prefix. This label leaf is used to
   match VPN traffic arriving from the core. Note that the label leaf
   shares the OutLabel-Array and the pathlist with the IP prefix.

[2.3.3](#). **Example 3: Platforms with Limited Levels of Hierarchy**

   This example uses a case of inter-AS option C [8] where there are 3
   levels of hierarchy. Figure 4 illustrates the sample topology. To
   force 3 levels of hierarchy, the ASBRs on the ingress domain (domain
   1) advertise the core routers of the egress domain (domain 2) to the
   ingress PE (iPE) via BGP-LU [4] instead of redistributing then into

the IGP of domain 1. The end result is that the ingress PE (iPE) has
2 levels of recursion for the VPN prefix VPN-P1 and VPN2-P2.

```
              Domain 1                    Domain 2
         +----------------+          +-------------+
         |                |          |             |
         |   LDP/SR Core  |          | LDP/SR core |
         |                |          |             |
         |            ASBR11------ASBR21.......PE21\
         |                |  \    /  |   .   .    |  \
         |                |   \  /   |    . .     |   \
         |                |    \/    |     ..     |    \VPN-P1
         |                |    /\    |     . .    |    /
         |                |   /  \   |    .   .   |   /
         |                |  /    \  |   .      . |  /
        iPE            ASBR12------ASBR22.......PE22
         |                |          |            |  \
         |                |          |            |   \
         |                |          |            |    \
         |                |          |            |    /VPN-P2
         |                |          |            |   /
         |                |          |            |  /
         |            ASBR13------ASBR23.......PE23/
         |                |          |            |
         |                |          |            |
         +----------------+          +-------------+
          <==============  <========= <============
          Advertise PE2x   Advertise  Redistribute
           Using iBGP-LU  PE2x Using   IGP into
                            eBGP-LU       BGP
```
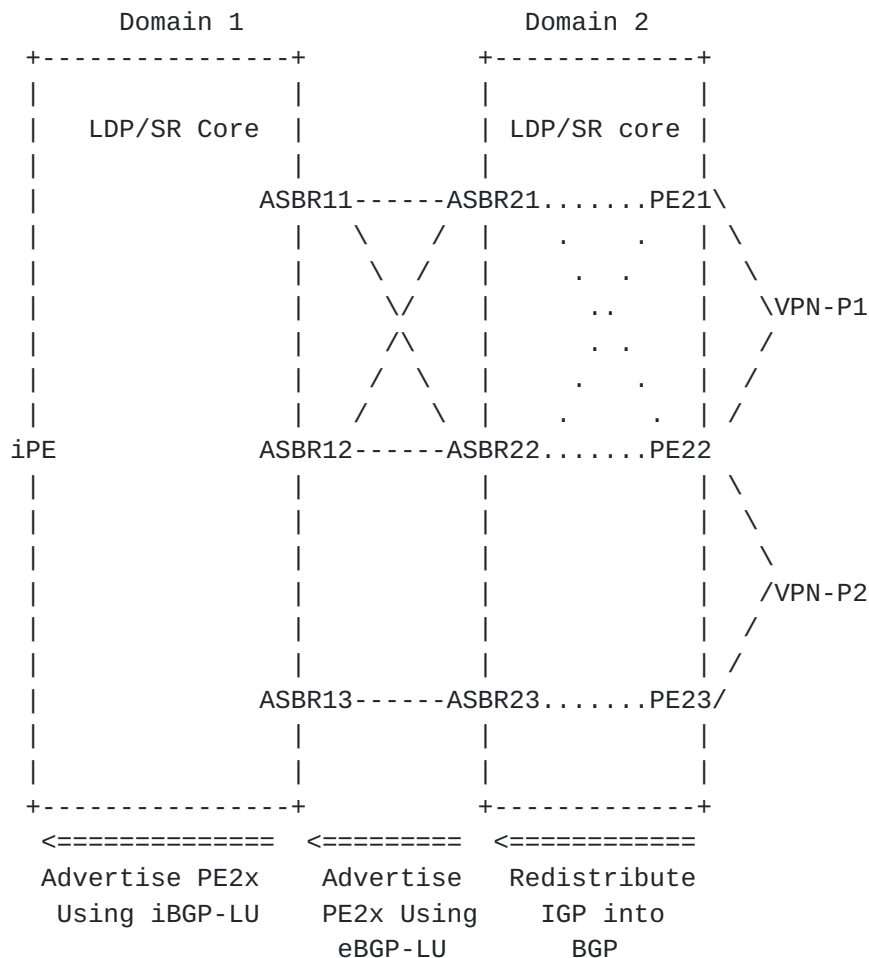
            Figure 4 Sample 3-level hierarchy topology

   We will make the following assumptions about connectivity

   o In "domain 2", both ASBR21 and ASBR22 can reach both PE21 and
      PE22 using the same distance

   o In "domain 2", only ASBR23 can reach PE23

   o In "domain 1", iPE (the ingress PE) can reach ASBR1, ASBR12, and
      ASBR13 via IGP using the same distance

   We will make the following assumptions about the labels

   o The VPN labels advertised by PE21 and PE22 for prefix VPN-P1 are
      VPN-PE21(P1) and VPN-PE22(P1), respectively

o The VPN labels advertised byPE22 and PE23 for prefix VPN-P2 are
   VPN-PE22(P2) and VPN-PE23(P2), respectively

o The labels for advertised to iPE by ASBR11 using BGP-LU [4] for
   the egress PEs PE21 and PE22 are LASBR11(PE21) and LASBR11(PE22),
   respectively.

o The labels for advertised by ASBR12 to iPE using BGP-LU [4] for
   the egress PEs PE21 and PE22 are LASBR12(PE21) and LASBR12(PE22),
   respectively

o The label for advertised by ASBR11 to iPE using BGP-LU [4] for
   the egress PE PE23 is LASBR13(PE23)

o The local labels of the next hops from the ingress PE iPE towards
   ASBR11, ASBR12, and ASBR13 in the core of domain 1 are L11, L12,
   and L13, respectively.

The diagram in Figure 5 illustrates the forwarding chain assuming
that the forwarding hardware in iPE supports 3 levels of hierarchy.
The leaves corresponding to the ABSRs on domain 1 (ASBR11, ASBR12,
and ASBR13) are at the bottom of the hierarchy. There are few
important points

o Because the hardware supports the required depth of hierarchy,
   the sizes of a pathlist equal the size of the label array
   associated with the leaves using this pathlist

o The index inside the pathlist entry indicates the label that will
   be picked from the Outlabel-array if that path is chosen by the
   forwarding engine hashing function.

```
    Outlabel Array                                Outlabel Array
      For VPN-P1                                    For VPN-P2
   +------------+    +-------+          +-------+    +------------+
   |VPN-PE21(P1)|<---| VPN-P1|          | VPN-P2|-->|VPN-PE22(P2)|
   +------------+    +---+---+          +---+---+    +------------+
   |VPN-PE22(P1)|        |                  |        |VPN-PE23(P2)|
   +------------+        |                  |        +------------+
                        |                  |
                        V                  V
                 +---+---+          +---+---+
                 | 0 | 1 |          | 0 | 1 |
                 +-|-+-\-+          +-/-+-\-+
                   |    \            /     \
                   |     \          /       \
                   |      \        /         \
                   |       \      /           \
                   v        \    /             \
               +-----+       +-----+          +-----+
         +----+ PE21|        |PE22 +-----+     | PE23+-----+
         |    +--+--+        +-----+     |     +--+--+     |
         v       |            /          v        |        v
   +------------+ |          /    +------------+   | +-------------+
   |LASBR11(PE21)| |         /     |LASBR11(PE22)|   | |LASBR13(PE23)|
   +------------+ |        /      +------------+   | +-------------+
   |LASBR12(PE21)| |       /       |LASBR12(PE22)|   | Outlabel Array
   +------------+ |      /        +------------+   |     For PE23
   Outlabel Array  |     /         Outlabel Array    |
       For PE21    |    /              For PE22       |
                   |   /                              |
                   |  /                               |
                   | /                                |
                   v /                                v
            +---+---+  Shared Pathlist        +---+  Pathlist
            | 0 | 1 | For PE21 and PE22       | 0 |  For PE23
            +-|-+-\-+                         +-|-+
              |    \                            |
              |     \                           |
              |      \                          |
              |       \                         |
              v        \                        v
     +---+      +------+    +------+    +---+    +------+    +---+
     |L11|<--->|ASBR11|    |ASBR12+--->|L12|    |ASBR13+--->|L13|
     +---+      +------+    +------+    +---+    +------+    +---+
```
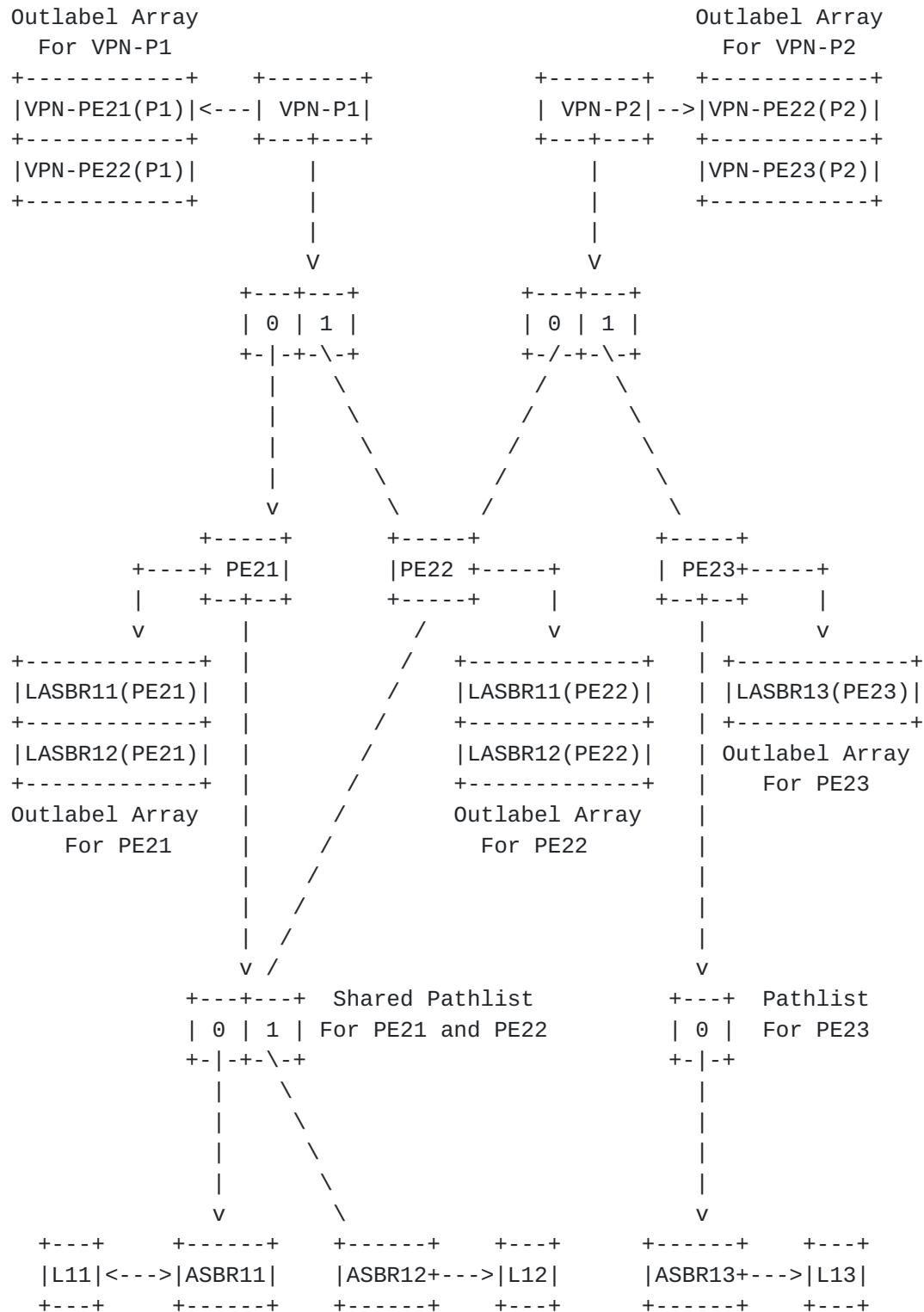
                Figure 5 : Forwarding Chain for hardware supporting 3 Levels

Now suppose the hardware on iPE (the ingress PE) supports 2 levels
of hierarchy only. In that case, the 3-levels forwarding chain in
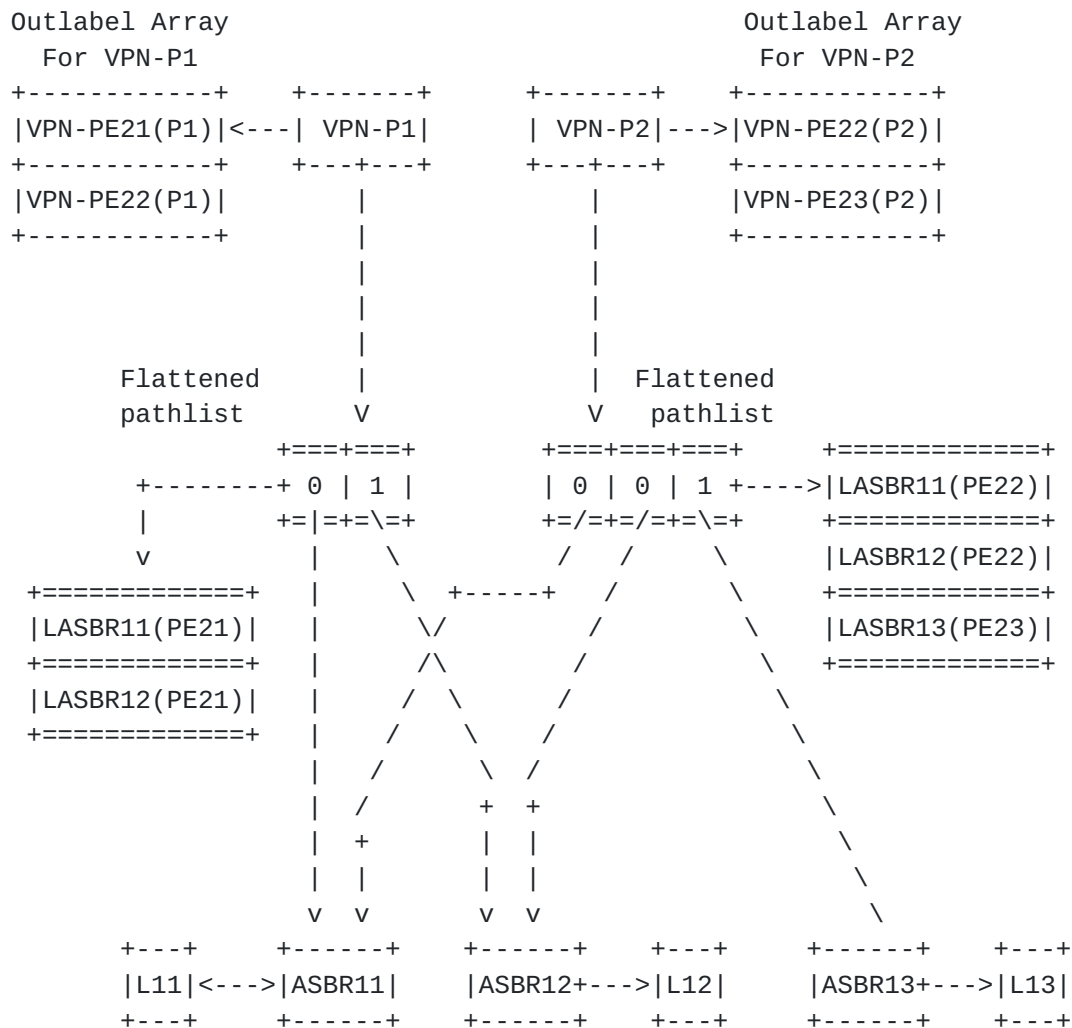Figure 5 needs to be "flattended" into 2 levels only.

```
Outlabel Array                                Outlabel Array
  For VPN-P1                                    For VPN-P2
+------------+    +-------+      +-------+    +------------+
|VPN-PE21(P1)|<---| VPN-P1|      | VPN-P2|--->|VPN-PE22(P2)|
+------------+    +---+---+      +---+---+    +------------+
|VPN-PE22(P1)|        |             |        |VPN-PE23(P2)|
+------------+        |             |        +------------+
                      |             |
                      |             |
                      |             |
     Flattened        |             |  Flattened
     pathlist         V             V  pathlist
               +===+===+      +===+===+===+    +=============+
      +--------+ 0 | 1 |      | 0 | 0 | 1 +---->|LASBR11(PE22)|
      |        +=|=+=\=+      +=/=+=/=+=\=+    +=============+
      V         |    \        /   /     \      |LASBR12(PE22)|
 +=============+|     \  +-----+  /       \     +=============+
 |LASBR11(PE21)||      \/        /         \    |LASBR13(PE23)|
 +=============+|      /\        /           \  +=============+
 |LASBR12(PE21)||     /  \      /             \
 +=============+|    /    \    /               \
               |   /      \  /                 \
               | /         + +                  \
               | +         | |                   \
               | |         | |                    \
               V V         V V                     \
      +---+    +------+    +------+    +---+    +------+    +---+
      |L11|<-->|ASBR11|    |ASBR12+--->|L12|    |ASBR13+--->|L13|
      +---+    +------+    +------+    +---+    +------+    +---+
```

         Figure 6 : Flattening 3 levels to 2 levels of Hierarchy on iPE

Figure 6 represents one way to "flatten" a 3 levels hierarchy into
two levels. There are few important points.

o The flattened pathlists have label arrays associated with them.
   The size of the label array associated with the flattened
   pathlist equals the size of the pathlist. Hence it is possible
   that an implementation includes these label arrays in the
   flattened pathlist itself

o Because of "flattening", the size of a flattened pathlist may not
   be equal to the size of the label arrays of leaves using the
   flattened pathlist.

o The indices inside a flattened pathlist still indicate the label
   index in the Outlabel-Arrays of the leaves using that pathlist.
   Because the size of the flattened pathlist may be different from
   the size of the label arrays of the leaves, the indices may be
   repeated

o Let's take a look at the flattened pathlist used by the prefix
   "VPN-P2", The pathlist associated with the prefix "VPN-P2" has
   three entries.

   o The first and second entry have index "0". This is because
     both entries correspond to PE22. Hence when hashing performed
     by the forwarding engine results in using first or the second
     entry in the pathlist, the forwarding engine will pick the
     correct VPN label "VPN-PE22(P2)", which is the label
     advertised by PE22 for the prefix "VPN-P2"

   o The third entry has the index "1". This is because the third
     entry corresponds to PE23. Hence when the hashing is
     performed by the forwarding engine results in using the third
     entry in the flattened pathlist, the forwarding engine will
     pick the correct VPN label "VPN-PE22(P2)", which is the label
     advertised by "PE23" for the prefix "VPN-P2"

3. **Forwarding Behavior**

   When a packet arrives at a router, it matches a leaf. A labeled
   packet matches a label leaf while an IP packet matches an IP prefix
   leaf. The forwarding engines walks the forwarding chain starting
   from the leaf until the walk terminates on an adjacency. Thus when a
   packet arrives, the chain is walked as follows:

   1. Lookup the leaf based on the destination address or the label at
      the top of the packet

   2. Retrieve the parent pathlist of the leaf

   3. Pick the outgoing path from the list of resolved paths in the
      pathlist. The method by which the outgoing path is picked is
      beyond the scope of this document (i.e. flow-preserving hash
      exploiting entropy within the MPLS stack and IP header). Let the
      "path-index" of the outgoing path be "i".

4. If the prefix is labeled, use the "path-index" "i" to retrieve
   the ith label "Li" stored the ith entry in the OutLabel-Array and
   apply the label action of the label on the packet (e.g. for VPN
   label on the ingress PE, the label action is "push").

5. Move to the parent of the chosen path "i"

6. If the chosen path "i" is recursive, move to its parent prefix
   and go to step 2

7. If the chosen path "i" is non-recursive move to its parent
   adjacency

8. Encapsulate the packet in the L2 string specified by the
   adjacency and send the packet out.

Let's applying the above forwarding steps to the example described
in Figure 1 Section 2.3.1. Suppose a packet arrives at ingress PE
iPE from an external neighbor. Assume the packet matches the VPN
prefix VPN-P1. While walking the forwarding chain, the forwarding
engine applies a hashing algorithm to choose the path and the
hashing at the BGP level yields path 0 while the hashing at the IGP
level yields path 1. In that case, the packet will be sent out of
interface I1 with the label stack "LDP-L12,VPN-L21".

Now let's try and apply the above steps to the flattened forwarding
chain illustrated in Figure 6.

o Suppose a packet arrives at "iPE" and matches the VPN prefix
   "VPN-P2"

o The forwarding engine walks to the parent of the "VPN_P2", whiuch
   is the flattened pathlist and applies a hashing algorithm to pick
   a path

o Suppose the hashing by the forwarding engine picks the second
   entry in the flattened pathlist associated with the leaf "VPN-
   P2".

o Because the second entry has the index "0", the label "VPN-
   PE22(P2)" is pushed on the packet

o At the same time, the forwarding engine picks the second label
   from the Outlabel-Array associated with the flattened pathlist.
   Hence the next label that is pushed is "LASBR12(PE22)"

o The forwarding engine now moves to the parent of the flattened
   pathlist corresponding tgo the second entry. The parent is the
   IGP label leaf corresponding to "ASBR12"

o So the packet is forwarded towards the ASBR "ASBR12" and the
   SR/LDP label at the top will be "L12"

The packet is arriving at iPE reaches its destination as follows

o iPE sends the packet along the shortest path towards ASBR12 with
   the following label stack starting from the top: {L12,
   LASBR12(PE22), VPN-PE22(P2)}.

o The penultimate hop of ASBR12 pops the top label "L12". Hence the
   packet arrives at ASBR12 with the label stack {LASBR12(PE22),
   VPN-PE22(P2)} where "LASBR12(PE22)" is the top label.

o ASBR12 swaps "LASBR12(PE22)" with the label "LASBR22(PE22)",
   which is the label advertised by ASBR22 for the PE22 (the egress
   PE).

o ASBR22 receives the packet with "LASBR22(PE22)" at the top.

o Hence ASBR22 swaps "LASBR22(PE22)" with the LDP/SR label of PE22,
   pushes the label of the next-hop towards PE22 in domain 2, and
   sends the packet along the shortest path towards PE22.

o The penultimate hop of PE22 pops the top label. Hence PE22
   receives the packet with the top label VPN-PE22(P2) at the top

o PE22 pops "VPN-PE22(P2)" and sends the packet as a pure IP packet
   towards the destination VPN-PE22.

[4]. Forwarding Chain Adjustment at a Failure

The hierarchical and shared structure of the forwarding chain
explained in [Section 2]. allows modifying a small number of
forwarding chain objects to re-route traffic to a pre-calculated
equal-cost or backup path without the need to modify the possibly
very large number of BGP prefixes. In this section, we go over
various core and edge failure scenarios to illustrate how FIB
manager can utilize the forwarding chain structure to achieve prefix
independent convergence.

4.1. BGP-PIC core

This section describes the adjustments to the forwarding chain when
a core link or node fails but the BGP next-hop remains reachable.

There are two case: remote link failure and attached link failure.
Node failures are treated as link failures.

When a remote link or node fails, IGP on the ingress PE receives
advertisement indicating a topology change so IGP re-converges to

either find a new next-hop and outgoing interface or remove the path completely from the IGP prefix used to resolve BGP next-hops. IGP and/or LDP download the modified IGP leaves with modified outgoing labels for labeled core. FIB manager modifies the existing IGP leaf by executing the steps outlined in Section 2.2.

When a local link fails, FIB manager detects the failure almost immediately. The FIB manager marks the impacted path(s) as unuseable so that only useable paths are used to forward packets. Note that in this particular case there is actually no need even to backwalk to IGP leaves to adjust the OutLabel-Arrays because FIB can rely on the path-index stored in the useable paths in the loadinfo to pick the right label.

It is noteworthy to mention that because FIB manager modifies the forwarding chain starting from the IGP leaves only, BGP pathlists and leaves are not modified. Hence traffic restoration occurs within the time frame of IGP convergence, and, for local link failure, within the timeframe of local detection. Thus it is possible to achieve sub-50 msec convergence as described in [10] for local link failure

Let's apply the procedure to the forwarding chain depicted in Figure 2 Section 2.3.1. Suppose a remote link failure occurs and impacts the first ECMP IGP path to the remote BGP nhop. Upon IGP convergence, the IGP pathlist of the BGP nhop is updated to reflect the new topology (one path instead of two). As soon as the IGP convergence is effective for the BGP nhop entry, the new forwarding state is immediately available to all dependent BGP prefixes. The same behavior would occur if the failure was local such as an interface going down. As soon as the IGP convergence is complete for the BGP nhop IGP route, all its BGP depending routes benefit from the new path. In fact, upon local failure, if LFA protection is enabled for the IGP route to the BGP nhop and a backup path was pre-computed and installed in the pathlist, upon the local interface failure, the LFA backup path is immediately activated (sub-50msec) and thus protection benefits all the depending BGP traffic through the hierarchical forwarding dependency between the routes.

4.2. BGP-PIC edge

This section describes the adjustments to the forwarding chains as a result of edge node or edge link failure

[4.2.1](#). **Adjusting forwarding Chain in egress node failure**

   When an edge node fails, IGP on neighboring core nodes send route
   updates indicating that the edge node is no longer reachable. IGP
   running on the iBGP peers instructs FIB to remove the IP and label
   leaves corresponding to the failed edge node from FIB. So FIB
   manager performs the following steps:

   o FIB manager deletes the IGP leaf corresponding to the failed edge
      node

   o FIB manager backwalks to all dependent BGP pathlists and marks
      that path using the deleted IGP leaf as unresolved

   o Note that there is no need to modify BGP leaves because each path
      in the pathlist carries its path index and hence the correct
      outgoing label will be picked. So for example the forwarding
      chain depicted in Figure 2, if the 1st path becomes unresolved,
      then the forwarding engine will only use the second path path for
      forwarding. Yet the pathindex of that single resolved path will
      still be 1 and hence the label VPN-L21 or VPN-L22 will be pushed

[4.2.2](#). **Adjusting Forwarding Chain on PE-CE link Failure**

   Suppose the link between an edge router and its external peer fails.
   There are two scenarios (1) the edge node attached to the failed
   link performs next-hop self and (2) the edge node attached to the
   failure advertises the IP address of the failed link as the next-hop
   attribute to its iBGP peers.

   In the first case, the rest of iBGP peers will remain unaware of the
   link failure and will continue to forward traffic to the edge node
   until the edge node attached to the failed link withdraws the BGP
   prefixes. If the destination prefixes are multi-homed to another
   iBGP peer, say ePE2, then FIB manager on the edge router detecting
   the link failure performs the following tasks

   o FIB manager backwalks to the BGP pathlists marks the path through
      the failed link to the external peer as unresolved

   o Hence traffic will be forwarded used the backup path towards ePE2

   o For labeled traffic

       o The Outlabel-Array attached to the BGP leaves already
         contains an entry corresponding to the path towards ePE2.

       o The label entry in OutLabel-Arrays corresponding to the
         internal path to ePE2 has swap action and the label
         advertised by ePE2

   o For an arriving label packet (e.g. VPN), the top label is
      swapped with the label advertised by ePE2

o For unlabeled traffic, packets is simply redirected towards ePE2.
   To avoid loops, ePE2 MUST treat any core facing path as a backup
   path, otherwise ePE2 may redirect traffic arriving from the core
   back to ePE1 causing a loop.

In the second case where the edge router uses the IP address of the
failed link as the BGP next-hop, the edge router will still perform
the previous steps. But, unlike the case of next-hop self, IGP on
failed edge node informs the rest of the iBGP peers that IP address
of the failed link is no longer reachable. Hence the FIB manager on
iBGP peers will delete the IGP leaf corresponding to the IP prefix
of the failed link. The behavior of the iBGP peers will be identical
to the case of edge node failure outlined in Section 4.2.1.

It is noteworthy to mention that because the edge link failure is
local to the edge router, sub-50 msec convergence can be achieved as
described in [10].

Let's try to apply the case of next-hop self to the forwarding chain
depicted in Figure 3. After failure of the link between ePE1 and CE,
the forwarding engine will route traffic arriving from the core
towards VPN-NH2 with path-index=1. A packet arriving from the core
will contain the label VPN-L11 at top. The label VPN-L11 is swaped
with the label VPN-L21 and the packet is forwarded towards ePE2

4.3. Handling Failures for Flattended Forwarding Chains

As explained in the Example in Section 2.3.3. , if the number of
hierarchy levels of a platform cannot support the number of
hierarchy levels of a recursive dependency, the instantiated
forwarding chain is constructed by flattening two or more levels.
Hence a 3 levels chain in Figure 5 is flattened into the 2 levels
chain in Figure 6.

While reducing the benefits of BGP-PIC, flattening one hierarchy
into a shallower hierarchy does not always result in a complete loss
of the benefits of the BGP-PIC. To illustrate this fact suppose
ASBR12 is no longer reachable. If the platform supports the full
hierarchy depth, the forwarding chain is depicted in Figure 5 and
hence the FIB manager needs to backwalk one level to the pathlist
shared by "PE21" and "PE222" and adjust it. If the platform supports
2 levels of hierarchy, then a useable forwarding chain is the one
depicted in Figure 6. In that case, if ASBR12 is no longer
reachable, the FIB manager has to backwalk to the two flattened
pathlists and update both of them.

Hence if the platform supports the "unflattened" forwarding chain,
then a single pathlist needs to be updated while if the platform
supports a shallower forwarding chain, then two pathlists need to be
updated. In the latter case, convergence is still independent of the
number of leaves due to the fact that the flattened pathlists
continue to be shared among possibly a large number of leaves

5. **Properties**

5.1 Coverage

All the possible failures, except CE node failure, are covered,
whether they impact a local or remote IGP path or a local or remote
BGP nhop as described in Section 4. This section provides details
for each failure and now the hierarchical and shared FIB structure
proposed in this document allows recovery that does not depend on
number of BGP prefixes

5.1.1 A remote failure on the path to a BGP nhop

Upon IGP convergence, the IGP leaf for the BGP nhop is updated upon
IGP convergence and all the BGP depending routes leverage the new
IGP forwarding state immediately.

This BGP resiliency property only depends on IGP convergence and is
independent of the number of BGP prefixes impacted.

5.1.2 A local failure on the path to a BGP nhop

Upon LFA protection, the IGP leaf for the BGP nhop is updated to use
the precomputed LFA backup path and all the BGP depending routes
leverage this LFA protection.

This BGP resiliency property only depends on LFA protection and is
independent of the number of BGP prefixes impacted.

5.1.3 A remote iBGP nhop fails

Upon IGP convergence, the IGP leaf for the BGP nhop is deleted and
all the depending BGP Path-Lists are updated to either use the
remaining ECMP BGP best-paths or if none remains available to
activate precomputed backups.

This BGP resiliency property only depends on IGP convergence and is
independent of the number of BGP prefixes impacted.

5.1.4 A local eBGP nhop fails

Upon local link failure detection, the adjacency to the BGP nhop is
deleted and all the depending BGP Path-Lists are updated to either
use the remaining ECMP BGP best-paths or if none remains available
to activate precomputed backups.

This BGP resiliency property only depends on local link failure
detection and is independent of the number of BGP prefixes impacted.

5.2 Performance

When the failure is local (a local IGP nhop failure or a local eBGP
nhop failure), a pre-computed and pre-installed backup is activated
by a local-protection mechanism that does not depend on the number
of BGP destinations impacted by the failure. Sub-50msec is thus
possible even if millions of BGP routes are impacted.

When the failure is remote (a remote IGP failure not impacting the
BGP nhop or a remote BGP nhop failure), an alternate path is
activated upon IGP convergence. All the impacted BGP destinations
benefit from a working alternate path as soon as the IGP convergence
occurs for their impacted BGP nhop even if millions of BGP routes
are impacted.

5.2.1 Perspective

The following table puts the BGP PIC benefits in perspective
assuming

o 1M impacted BGP prefixes

o IGP convergence ~ 500 msec

o local protection ~ 50msec

o FIB Update per BGP destination ~ 100usec conservative,

                              ~ 10usec optimistic

o BGP Convergence per BGP destination ~ 200usec conservative,

                              ~ 100usec optimistic

	Without PIC	With PIC
Local IGP Failure	10 to 100sec	50msec
Local BGP Failure	100 to 200sec	50msec

   Remote IGP Failure   10 to 100sec   500msec

   Local BGP Failure   100 to 200sec   500msec

   Upon local IGP nhop failure or remote IGP nhop failure, the existing
   primary BGP nhop is intact and usable hence the resiliency only
   depends on the ability of the FIB mechanism to reflect the new path
   to the BGP nhop to the depending BGP destinations. Without BGP PIC,
   a conservative back-of-the-envelope estimation for this FIB update
   is 100usec per BGP destination. An optimistic estimation is 10usec
   per entry.

   Upon local BGP nhop failure or remote BGP nhop failure, without the
   BGP PIC mechanism, a new BGP Best-Path needs to be recomputed and
   new updates need to be sent to peers. This depends on BGP processing
   time that will be shared between best-path computation, RIB update
   and peer update. A conservative back-of-the-envelope estimation for
   this is 200usec per BGP destination. An optimistic estimation is
   100usec per entry.

   5.3 Automated

   The BGP PIC solution does not require any operator involvement. The
   process is entirely automated as part of the FIB implementation.

   The salient points enabling this automation are:

   o Extension of the BGP Best Path to compute more than one primary
      ([11]and [12]) or backup BGP nhop ([6] and [13]).

   o Sharing of BGP Path-list across BGP destinations with same
      primary and backup BGP nhop

   o Hierarchical indirection and dependency between BGP Path-List and
      IGP-Path-List

   5.4 Incremental Deployment

   As soon as one router supports BGP PIC solution, it benefits from
   all its benefits without any requirement for other routers to
   support BGP PIC.

6. Dependency

   This section describes the required functionality in the forwarding
   and control planes to support BGP-PIC described in this document

6.1 Hierarchical Hardware FIB

BGP PIC requires a hierarchical hardware FIB support: for each BGP
forwarded packet, a BGP leaf is looked up, then a BGP Pathlist is
consulted, then an IGP Pathlist, then an Adjacency.

An alternative method consists in "flattening" the dependencies when
programming the BGP destinations into HW FIB resulting in
potentially eliminating both the BGP Path-List and IGP Path-List
consultation. Such an approach decreases the number of memory
lookup's per forwarding operation at the expense of HW FIB memory
increase (flattening means less sharing hence duplication), loss of
ECMP properties (flattening means less pathlist entropy) and loss of
BGP PIC properties.

6.2 Availability of more than one primary or secondary BGP next-hops

When the primary BGP next-hop fails, BGP PIC depends on the
availability of a pre-computed and pre-installed secondary BGP next-
hop in the BGP Pathlist.

The existence of a secondary next-hop is clear for the following
reason: a service caring for network availability will require two
disjoint network connections hence two BGP nhops.

The BGP distribution of the secondary next-hop is available thanks
to the following BGP mechanisms: Add-Path [11], BGP Best-External
[6], diverse path [12], and the frequent use in VPN deployments of
different VPN RD's per PE. It is noteworthy to mention that the
availability of another BGP path does not mean that all failure
scenarios can be covered by simply forwarding traffic to the
available secondary path. The discussion of how to cover various
failure scenarios is beyond the scope of this document

6.3 Pre-Computation of a secondary BGP nhop

[13] describes how a secondary BGP next-hop can be precomputed on a
per BGP destination basis.

7. Security Considerations

No additional security risk is introduced by using the mechanisms
proposed in this document

8. IANA Considerations

No requirements for IANA

9. Conclusions

This document proposes a hierarchical and shared forwarding chain structure that allows achieving prefix independent convergence, and in the case of locally detected failures, sub-50 msec convergence. A router can construct the forwarding chains in a completely transparent manner with zero operator intervention. It supports incremental deployment.

10. References

10.1. Normative References

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

[2]   Rekhter, Y., Li, T., and S. Hares, "A Border Gateway Protocol 4 (BGP-4), RFC 4271, January 2006

[3]   Bates, T., Chandra, R., Katz, D., and Rekhter Y., "Multiprotocol Extensions for BGP", RFC 4760, January 2007

[4]   Y. Rekhter and E. Rosen, " Carrying Label Information in BGP-4", RFC 3107, May 2001

[5]   Andersson, L., Minei, I., and B. Thomas, "LDP Specification", RFC 5036, October 2007

10.2. Informative References

[6]   Marques,P., Fernando, R., Chen, E, Mohapatra, P., Gredler, H., "Advertisement of the best external route in BGP", draft-ietf-idr-best-external-05.txt, January 2012.

[7]   Wu, J., Cui, Y., Metz, C., and E. Rosen, "Softwire Mesh Framework", RFC 5565, June 2009.

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

[9]   De Clercq, J. , Ooms, D., Prevost, S., Le Faucheur, F., "Connecting IPv6 Islands over IPv4 MPLS Using IPv6 Provider Edge Routers (6PE)", RFC 4798, February 2007

[10]   O. Bonaventure, C. Filsfils, and P. Francois. "Achieving sub-50 milliseconds recovery upon bgp peering link failures, " IEEE/ACM Transactions on Networking, 15(5):1123-1135, 2007

   [11] D. Walton, A. Retana, E. Chen, J. Scudder, "Advertisement of
         Multiple Paths in BGP", draft-ietf-idr-add-paths-12.txt,
         November 2015

   [12] R. Raszuk, R. Fernando, K. Patel, D. McPherson, K. Kumaki,
         "Distribution of diverse BGP paths", RFC 6774.txt, November
         2012

   [13] P. Mohapatra, R. Fernando, C. Filsfils, and R. Raszuk, "Fast
         Connectivity Restoration Using BGP Add-path", draft-pmohapat-
         idr-fast-conn-restore-03, Jan 2013

   [14] C. Filsfils, S. Previdi, A. Bashandy, B. Decraene, S.
         Litkowski, M. Horneffer, R. Shakir, J. Tansura, E. Crabbe
         "Segment Routing with MPLS data plane", draft-ietf-spring-
         segment-routing-mpls-02 (work in progress), October 2015

11. Acknowledgments

   Special thanks to Neeraj Malhotra, Yuri Tsier for the valuable
   help

   Special thanks to Bruno Decraene for the valuable comments

   This document was prepared using 2-Word-v2.0.template.dot.

Authors' Addresses

   Ahmed Bashandy
   Cisco Systems
   170 West Tasman Dr, San Jose, CA 95134, USA
   Email: bashandy@cisco.com

   Clarence Filsfils
   Cisco Systems
   Brussels, Belgium
   Email: cfilsfil@cisco.com

   Prodosh Mohapatra
   Sproute Networks
   Email: mpradosh@yahoo.com