Network Working Group
Internet Draft
Intended status: Informational
Expires: February 2022

A. Bashandy, Ed.
Individual Contributor
C. Filsfils
Cisco Systems
P. Mohapatra
Sproute Networks

August 15, 2021

BGP Prefix Independent Convergence
draft-ietf-rtgwg-bgp-pic-14.txt

Abstract

In the network comprising thousands of iBGP peers exchanging millions of routes, many routes are reachable via more than one next-hop. 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 data structures 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 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.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

This Internet-Draft will expire on February 15, 2021.

Table of Contents

1. Introduction

As a path vector protocol, BGP propagates reachability serially. Hence BGP convergence speed is limited by the time taken to serially propagate reachability information from the point of failure to the device that must re-converge. BGP speakers exchange reachability information about prefixes[1][2] 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 [7], 6PE [8], and Softwire [6] using BGP label unicast technique[3]. 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 BGP to advertise more than one path for a given prefix [5][10][11], whether in the form of equal cost multipath or primary-backup. Another common and widely deployed scenario is L3VPN with multi-homed VPN sites with unique Route Distinguisher. It is advantageous to utilize the commonality among paths used by NLRIs to significantly improve convergence in case of topology modifications.

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. Terminology

This section defines the terms used in this document. For ease of use, we will use terms similar to those used by L3VPN [7].

o BGP prefix: A prefix P/m (of any AFI/SAFI) that a BGP speaker

has a path for.

- IGP prefix: 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)

- CE: An external router through which an egress PE can reach a prefix P/m.

- Egress PE, "ePE": A BGP speaker that learns about a prefix through an eBGP peer and chooses that eBGP as the next-hop for that prefix.

- Ingress PE, "iPE": A BGP speaker that learns about a prefix through a iBGP peer and chooses an egress PE as the next-hop for the prefix.

- Path: The next-hop in a sequence of nodes starting from the current node and ending with the destination node or network identified by the prefix. The nodes may not be directly connected.

- Recursive path: A path consisting only of the IP address of the next-hop without the outgoing interface. Subsequent lookups are necessary to determine the outgoing interface and a directly connected next-hop

- Non-recursive path: A path consisting of the IP address of a directly connected next-hop and outgoing interface

- Primary path: A recursive or non-recursive path that can be used all the time as long as a walk starting from this path can end to an adjacency. A prefix can have more than one primary path

- Backup path: A recursive or non-recursive path that can be used only after some or all primary paths become unreachable

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

- IP leaf: The leaf corresponding to an IPv4 or IPv6 prefix

- Label leaf. The leaf corresponding to a locally allocated label such as the VPN label on an egress PE [7].

- Pathlist: 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. "). In general, the value of the "path-index" stored in path may not necessarily has the same value of the location of the path in the pathlist. For example the 3rd path may carry path-index value of 1

- A pathlist may contain a mix of primary and backup paths

- OutLabel-List: Each labeled prefix is associated with an OutLabel-List. The OutLabel-List is an array 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. Label actions are: push the label, pop the label, swap the incoming label with the label in the Outlabel-Array entry, or don't push anything at all in case of "unlabeled". The prefix may be an IGP or BGP prefix

- Adjacency: The layer 2 encapsulation leading to the layer 3 directly connected next-hop

- Dependency: An object X is said to be a dependent or child of object Y if there is at least one forwarding chain where the forwarding engine must visits the object X before visiting the object Y in order to forward a packet. Note that if object X is a child of object Y, then Y cannot be deleted unless object X is no longer a dependent/child of object Y

- Route: 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. Overview

The idea of BGP-PIC is based on two pillars

- A shared hierarchical forwarding chain: It is not uncommon to see multiple destinations are reachable via the same list of next-hops. Instead of having a separate list of next-hops for each destination, all destinations sharing the same list of next-hops can point to a single copy of this list thereby allowing fast convergence by making changes to a single shared list of next-hops rather than possibly a large number of destinations. Because paths in a pathlist may be recursive, a hierarchy is formed between pathlist and the resolving prefix whereby the pathlist depends on the resolving prefix.

- A forwarding plane that supports multiple levels of indirection: A forwarding that starts with a destination and ends with an outgoing interface is not a simple flat structure. Instead a forwarding entry is constructed via multiple levels of dependency. A BGP NLRI uses a recursive next-hop, which in turn resolves via an IGP next-hop, which in turn resolves via an adjacency consisting of one or more outgoing interface(s) and next-hop(s).

Designing a forwarding plane that constructs multi-level forwarding chains with maximal sharing of forwarding objects allows rerouting a large number of destinations by modifying a small number of objects thereby achieving convergence in a time frame that does not depend on the number of destinations. For example, if the IGP prefix that resolves a recursive next-hop is updated there is no need to update the possibly large number of BGP NLRIs that use this recursive next-hop.

2.1. Dependency

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

2.1.1. 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.

2.1.2. Availability of more than one BGP next-hops

When the primary BGP next-hop fails, BGP-PIC depends on the availability of one or more pre-computed and pre-installed secondary BGP next-hop(s) 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 next-hops.

The BGP distribution of secondary next-hops is available thanks to the following BGP mechanisms: Add-Path [10], BGP Best-External [5], diverse path [11], 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.

2.2. BGP-PIC Illustration

To illustrate the two pillars above as well as the platform dependency, we will use an example of a simple multihomed L3VPN [7] prefix in a BGP-free core running LDP [4] or segment routing over MPLS forwarding plane [13].

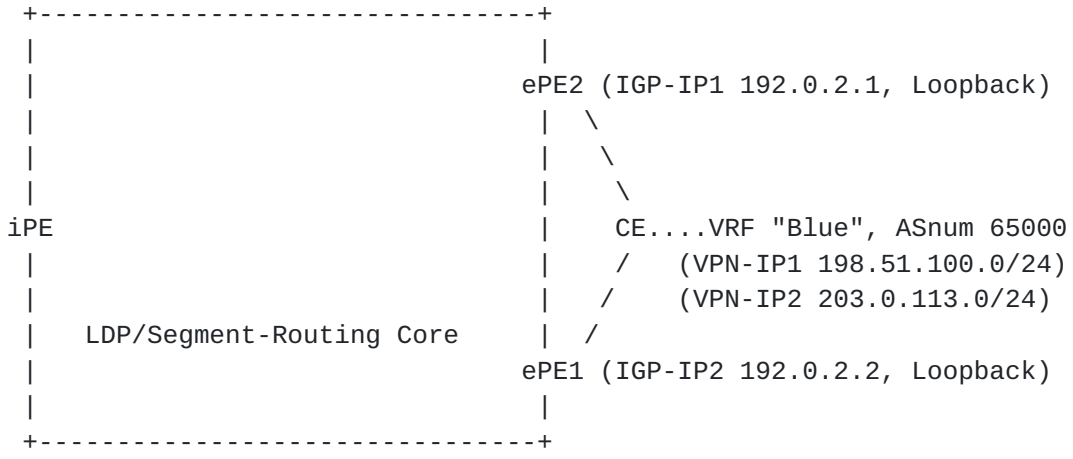

Figure 1 VPN prefix reachable via multiple PEs

Referring to Figure 1, suppose the iPE (the ingress PE) receives NLRIs for the VPN prefixes VPN-IP1 and VPN-IP2 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-IP1 and VPN-IP2, respectively. Suppose that BGP-NH1 and BGP-NH2 are resolved via the IGP prefixes IGP-IP1 and IGP-IP2, where each happen to have 2 ECMP paths with IGP-NH1 and IGP-NH2 reachable via the interfaces I1 and I2, respectively. Suppose that local labels (whether LDP [4] or segment routing [13]) on the downstream LSRs for IGP-IP1 are IGP-L11 and IGP-L12 while for IGP-IP2 are IGP-L21 and IGP-L22. As such, the routing table at iPE is as follows:

```
        65000: 198.51.100.0/24
           via ePE1 (192.0.2.1), VPN Label: VPN-L11
           via ePE2 (192.0.2.2), VPN Label: VPN-L21

        65000: 203.0.113.0/24
           via ePE1 (192.0.2.1), VPN Label: VPN-L12
           via ePE2 (192.0.2.2), VPN Label: VPN-L22

        192.0.2.1/32
           via Core, Label:  IGP-L11
           via Core, Label:  IGP-L12

        192.0.2.2/32
           via Core, Label:  IGP-L21
           via Core, Label:  IGP-L22
```

Based on the above routing table, a hierarchical forwarding chain can be constructed as shown in Figure 2.

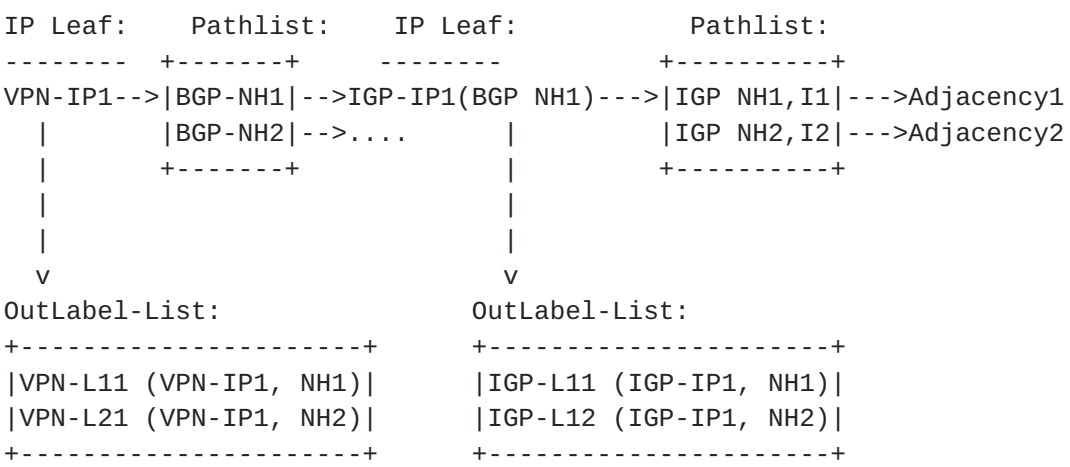

Figure 2 Shared Hierarchical Forwarding Chain at iPE

The forwarding chain depicted in Figure 2 illustrates the first pillar, which is sharing and hierarchy. We can see that the BGP pathlist consisting of BGP-NH1 and BGP-NH2 is shared by all NLRIs reachable via ePE1 and ePE2. As such, it is possible to make changes to the pathlist without having to make changes to the NLRIs. For example, if BGP-NH2 becomes unreachable, there is no need to modify any of the possibly large number of NLRIs. Instead only the shared pathlist needs to be modified. Likewise, due to the hierarchical structure of the forwarding chain, it is possible to make modifications to the IGP routes without having to make any changes to the BGP NLRIs. For example, if the interface "I2" goes down, only the shared IGP pathlist needs to be updated, but none of the IGP

prefixes sharing the IGP pathlist nor the BGP NLRIs using the IGP prefixes for resolution need to be modified.

Figure 2 can also be used to illustrate the second BGP-PIC pillar. Having a deep forwarding chain such as the one illustrated in Figure 2 requires a forwarding plane that is capable of accessing multiple levels of indirection in order to calculate the outgoing interface(s) and next-hops(s). While a deeper forwarding chain minimizes the re-convergence time on topology change, there will always exist platforms with limited capabilities and hence imposing a limit on the depth of the forwarding chain. Section 5 describes how to gracefully trade off convergence speed with the number of hierarchical levels to support platforms with different capabilities.

3. Constructing the Shared Hierarchical Forwarding Chain

Constructing the forwarding chain is an application of the two pillars described in Section 2. This section describes how to construct the forwarding chain in hierarchical shared manner.

3.1. Constructing the BGP-PIC forwarding Chain

The whole process starts when BGP downloads a prefix to FIB. The prefix contains one or more outgoing paths. For certain labeled prefixes, such as VPN [7] prefixes, each path may be associated with an outgoing label and the prefix itself may be assigned a local label. The list of outgoing paths defines a pathlist. If such pathlist does not already exist, then FIB creates a new pathlist, otherwise the existing pathlist is used. The BGP prefix is added as a dependent of the pathlist.

The previous step constructs the upper part of the hierarchical forwarding chain. The forwarding chain is completed by resolving the paths of the pathlist. A BGP path usually consists of a next-hop. The next-hop is resolved by finding a matching IGP prefix.

The end result is a hierarchical shared forwarding chain where the BGP pathlist is shared by all BGP prefixes that use the same list of paths and the IGP prefix is shared by all pathlists that have a path resolving via that IGP prefix. It is noteworthy to mention that the forwarding chain is constructed without any operator intervention at all.

The remainder of this section goes over an example to illustrate the applicability of BGP-PIC in a primary-backup path scenario.

3.2. Example: Primary-Backup Path Scenario

Consider the egress PE ePE1 in the case of the multi-homed VPN prefixes in the BGP-free 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 3. We are only showing a single VPN prefix for simplicity. But all prefixes that are multihomed to ePE1 and ePE2 share the BGP pathlist.

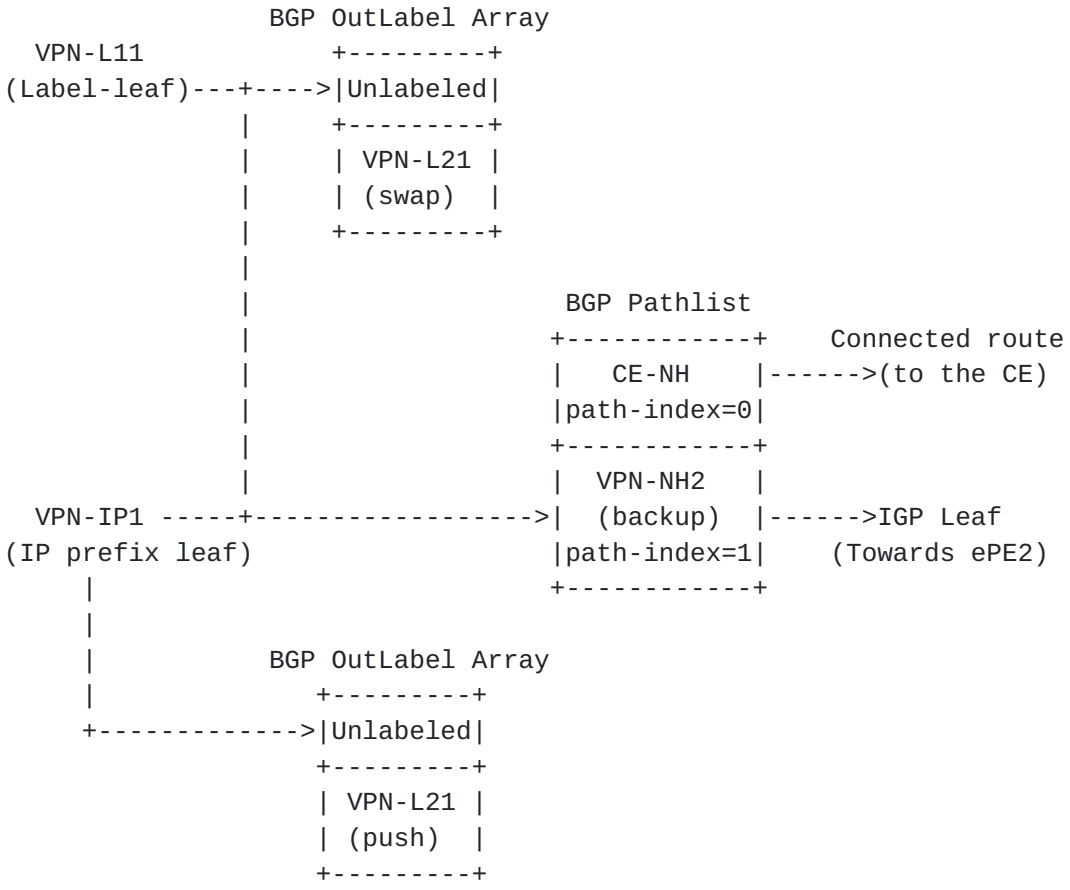

Figure 3 : VPN Prefix Forwarding Chain with eiBGP paths on egress PE

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-List 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 pathlist with the IP prefix.

4. Forwarding Behavior

This section explains how the forwarding plane uses the hierarchical shared forwarding chain to forward a packet.

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 "Pi" 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 (e.g. flow-preserving hash exploiting entropy within the MPLS stack and IP header). Let the "path-index" of the outgoing path "Pi" be "j".

4. If the prefix is labeled, use the "path-index" "j" to retrieve the jth label "Lj" stored the jth entry in the OutLabel-List 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"). As mentioned in Section 1.1 the value of the "path-index" stored in path may not necessarily be the same value of the location of the path in the pathlist.

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

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

7. If the chosen path is non-recursive move to its parent adjacency. Otherwise go to the next step.

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

Let's apply the above forwarding steps to the forwarding chain depicted in Figure 2 in Section 2. Suppose a packet arrives at ingress PE iPE from an external neighbor. Assume the packet matches the VPN prefix VPN-IP1. 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 I2 with the label stack "IGP-L12,VPN-L11".

5. Handling Platforms with Limited Levels of Hierarchy

This section describes the construction of the forwarding chain if a platform does not support the number of recursion levels required to resolve the NLRIs. There are two main design objectives.

- Being able to reduce the number of hierarchical levels from any arbitrary value to a smaller arbitrary value that can be supported by the forwarding engine.

- Minimal modifications to the forwarding algorithm due to such reduction.

5.1. Flattening the Forwarding Chain

Let's consider a pathlist associated with the leaf "R1" consisting of the list of paths <P1, P2,..., Pn>. Assume that the leaf "R1" has an Outlabel-list <L1, L2,..., Ln>. Suppose the path Pi is a recursive path that resolves via a prefix represented by the leaf "R2". The leaf "R2" itself is pointing to a pathlist consisting of the paths <Q1, Q2,..., Qm>.

If the platform supports the number of hierarchy levels of the forwarding chain, then a packet that uses the path "Pi" will be forwarded as follows:

1. The forwarding engine is now at leaf "R1".

2. So it moves to its parent pathlist, which contains the list <P1, P2,..., Pn>.

3. The forwarding engine applies a hashing algorithm and picks the path "Pi". So now the forwarding engine is at the path "Pi".

4. The forwarding engine retrieves the label "Li" from the outlabel-list attached to the leaf "R1" and applies the label action.

5. The path "Pi" uses the leaf "R2".

6. The forwarding engine walks forward to the leaf "R2" for resolution.

7. The forwarding plane performs a hash to pick a path among the pathlist of the leaf "R2", which is <Q1, Q2,..., Qm>.

8. Suppose the forwarding engine picks the path "Qj".

9. Now the forwarding engine continues the walk to the parent of "Qj".

Suppose the platform cannot support the number of hierarchy levels in the forwarding chain. FIB needs to reduce the number of hierarchy levels. The idea of reducing the number of hierarchy levels is to "flatten" two chain levels into a single level. The "flattening" steps are as follows

1. FIB wants to reduce the number of levels used by "Pi" by 1.

2. FIB walks to the parent of "Pi", which is the leaf "R2".

3. FIB extracts the parent pathlist of the leaf "R2", which is <Q1, Q2,..., Qm>.

4. FIB also extracts the OutLabel-list(R2) associated with the leaf "R2". Remember that OutLabel-list(R2) = <L1, L2,..., Lm>.

5. FIB replaces the path "Pi", with the list of paths <Q1, Q2,..., Qm>.

6. Hence the path list <P1, P2,..., Pn> now becomes "<P1, P2,...,Pi-1, Q1, Q2,..., Qm, Pi+1, Pn>.

7. The path index stored inside the locations "Q1", "Q2", ..., "Qm" must all be "i" because the index "i" refers to the label "Li" associated with leaf "R1".

8. FIB attaches an OutLabel-list with the new pathlist as follows: <Unlabeled,..., Unlabeled, L1, L2,..., Lm, Unlabeled, ..., Unlabeled>. The size of the label list associated with the flattened pathlist equals the size of the pathlist. Hence there is a 1-1 mapping between every path in the "flattened" pathlist and the OutLabel-list associated with it.

It is noteworthy to mention that the labels in the outlabel-list associated with the "flattened" pathlist may be stored in the same memory location as the path itself to avoid additional memory access. But that is an implementation detail that is beyond the scope of this document.

The same steps can be applied to all paths in the pathlist <P1, P2,..., Pn> so that all paths are "flattened" thereby reducing the number of hierarchical levels by one. Note that that "flattening" a pathlist pulls in all paths of the parent paths, a desired feature to utilize all ECMP/UCMP paths at all levels. A platform that has a limit on the number of paths in a pathlist for any given leaf may choose to reduce the number paths using methods that are beyond the scope of this document.

The steps can be recursively applied to other paths at the same levels or other levels to recursively reduce the number of

hierarchical levels to an arbitrary value so as to accommodate the capability of the forwarding engine.

Because a flattened pathlist may have an associated OutLabel-list the forwarding behavior has to be slightly modified. The modification is done by adding the following step right after step 4 in Section 4.

5. If there is an OutLabel-list associated with the pathlist, then if the path "Pi" is chosen by the hashing algorithm, retrieve the label at location "i" in that OutLabel-list and apply the label action of that label on the packet.

In the next subsection, we apply the steps in this subsection to a sample scenario.

5.2. Example: Flattening a forwarding chain.

This example uses a case of inter-AS option C [7] 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 [3] instead of redistributing them 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-IP1 and VPN2-IP2.

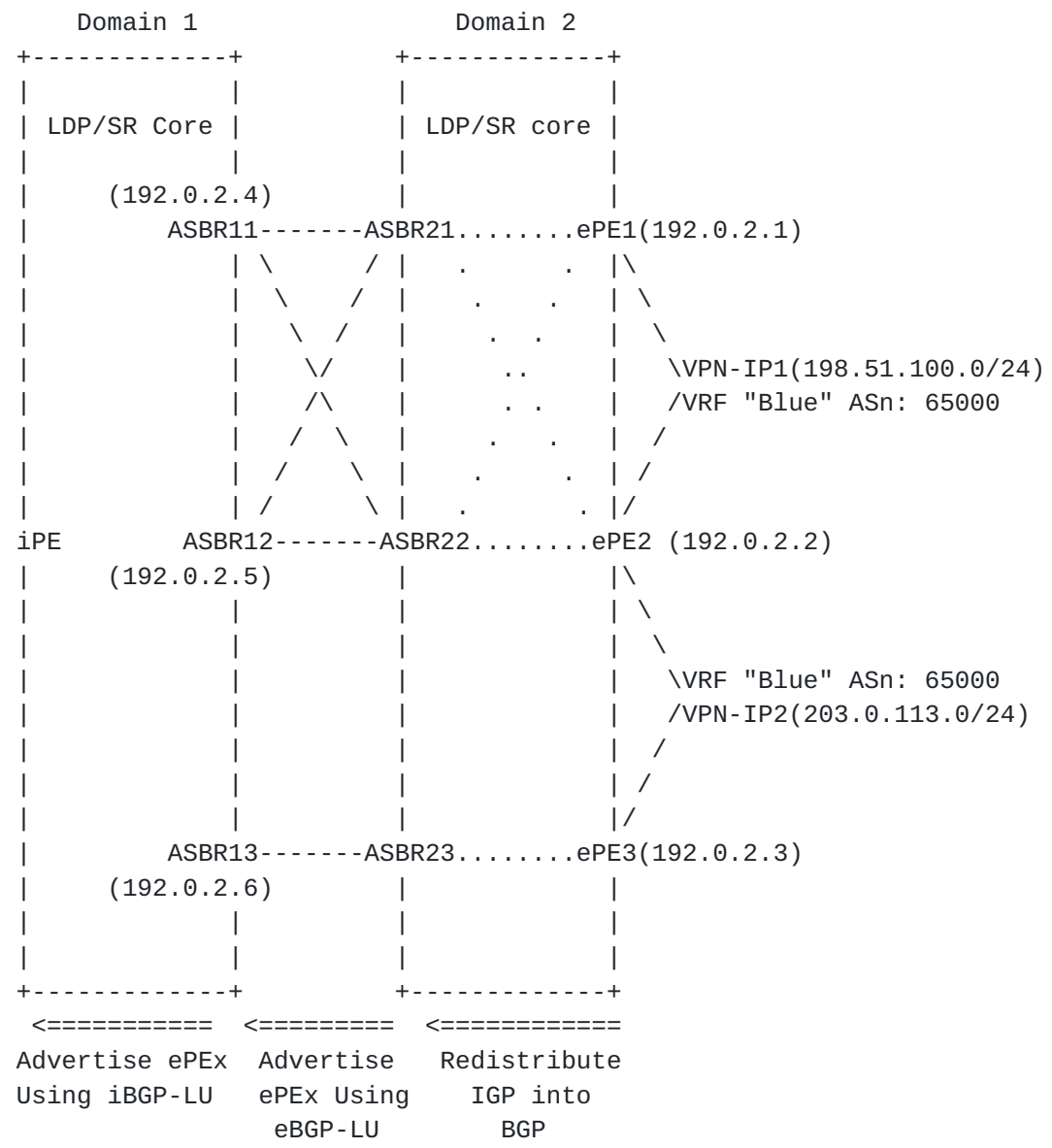

Figure 4 : Sample 3-level hierarchy topology

We will make the following assumptions about connectivity

- In "domain 2", both ASBR21 and ASBR22 can reach both ePE1 and ePE2 using the same distance.

- In "domain 2", only ASBR23 can reach ePE3.

- In "domain 1", iPE (the ingress PE) can reach ASBR11, ASBR12, and ASBR13 via IGP using the same distance.

We will make the following assumptions about the labels

- The VPN labels advertised by ePE1 and ePE2 for prefix VPN-IP1 are VPN-L11 and VPN-L21, respectively.

- The VPN labels advertised by ePE2 and ePE3 for prefix VPN-IP2 are VPN-L22 and VPN-L32, respectively.

- The labels advertised by ASBR11 to iPE using BGP-LU [3] for the egress PEs ePE1 and ePE2 are LASBR111(ePE1) and LASBR112(ePE2), respectively.

- The labels advertised by ASBR12 to iPE using BGP-LU [3] for the egress PEs ePE1 and ePE2 are LASBR121(ePE1) and LASBR122(ePE2), respectively.

- The label advertised by ASBR13 to iPE using BGP-LU [3] for the egress PE ePE3 is LASBR13(ePE3).

- The IGP labels advertised by the next hops directly connected to iPE towards ASBR11, ASBR12, and ASBR13 in the core of domain 1 are IGP-L11, IGP-L12, and IGP-L13, respectively.

- Both the routers ASBR21 and ASBR22 of Domain 2 advertise the same label LASBR21 and LASBR22 to the egress PEs ePE1 and ePE2, respectively, to the routers ASBR11 and ASBR22 of Domain 1.

- The router ASBR23 of Domain 2 advertises the label LASBR23 for the egress PE ePE3 to the router ASBR13 of Domain 1.

Based on these connectivity assumptions and the topology in Figure 4, the routing table on iPE is

```
        65000: 198.51.100.0/24
           via ePE1 (192.0.2.1), VPN Label: VPN-L11
           via ePE2 (192.0.2.2), VPN Label: VPN-L21
        65000: 203.0.113.0/24
           via ePE1 (192.0.2.2), VPN Label: VPN-L22
           via ePE2 (192.0.2.3), VPN Label: VPN-L32

        192.0.2.1/32 (ePE1)
           Via ASBR11, BGP-LU Label: LASBR111(ePE1)
           Via ASBR12, BGP-LU Label: LASBR121(ePE1)
        192.0.2.2/32 (ePE2)
           Via ASBR11, BGP-LU Label: LASBR112(ePE2)
           Via ASBR12, BGP-LU Label: LASBR122(ePE2)
        192.0.2.3/32 (ePE3)
           Via ASBR13, BGP-LU Label: LASBR13(ePE3)

        192.0.2.4/32 (ASBR11)
           via Core, Label:  IGP-L11
        192.0.2.5/32 (ASBR12)
           via Core, Label:  IGP-L12
        192.0.2.6/32 (ASBR13)
           via Core, Label:  IGP-L13
```

The diagram in Figure 5 illustrates the forwarding chain in iPE 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:

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

- The index inside the pathlist entry indicates the label that will be picked from the Outlabel-List associated with the child leaf if that path is chosen by the forwarding engine hashing function.

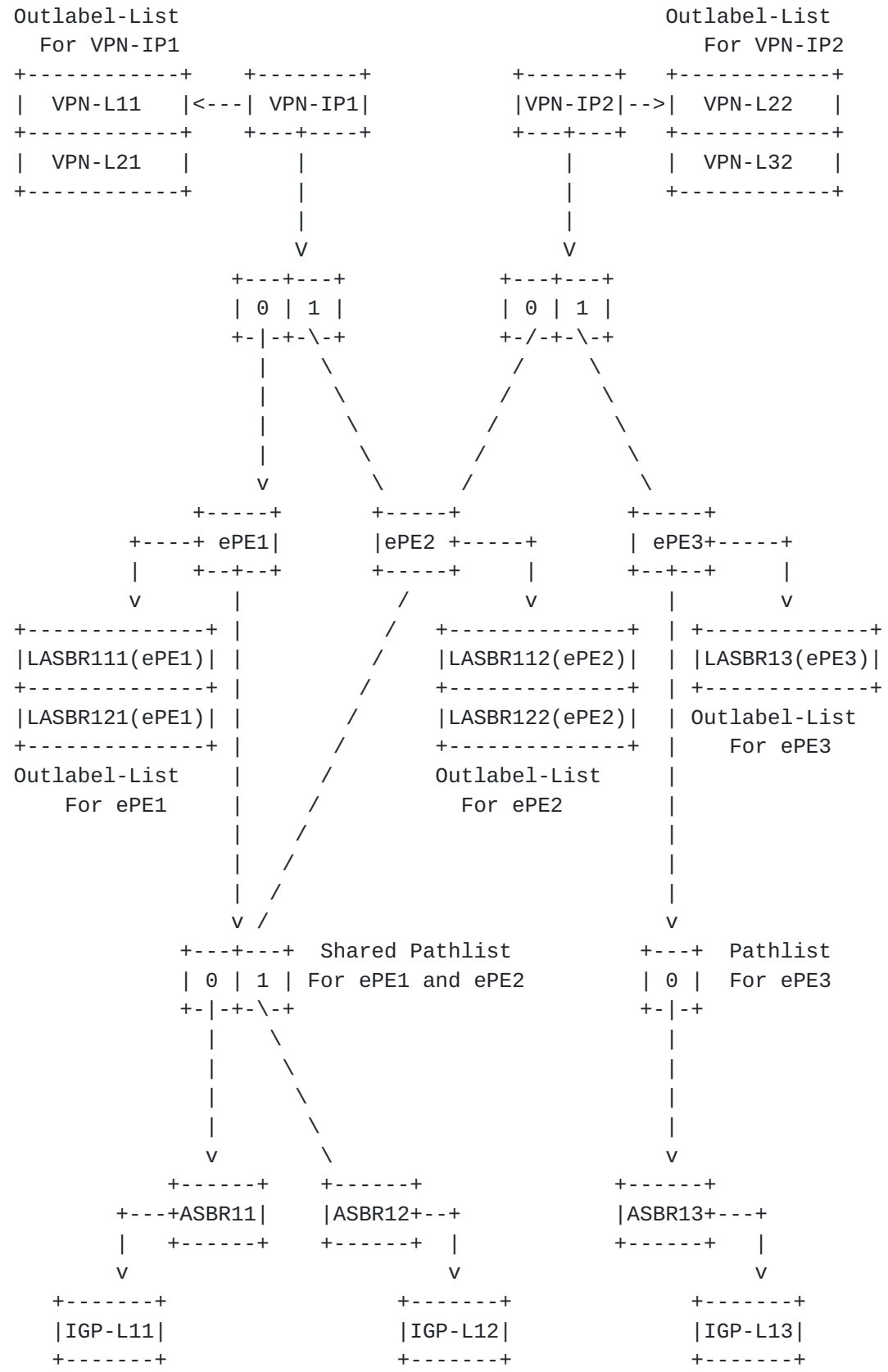

Figure 5 : Forwarding Chain for hardware supporting 3 Levels

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 "flattened" into 2 levels only.

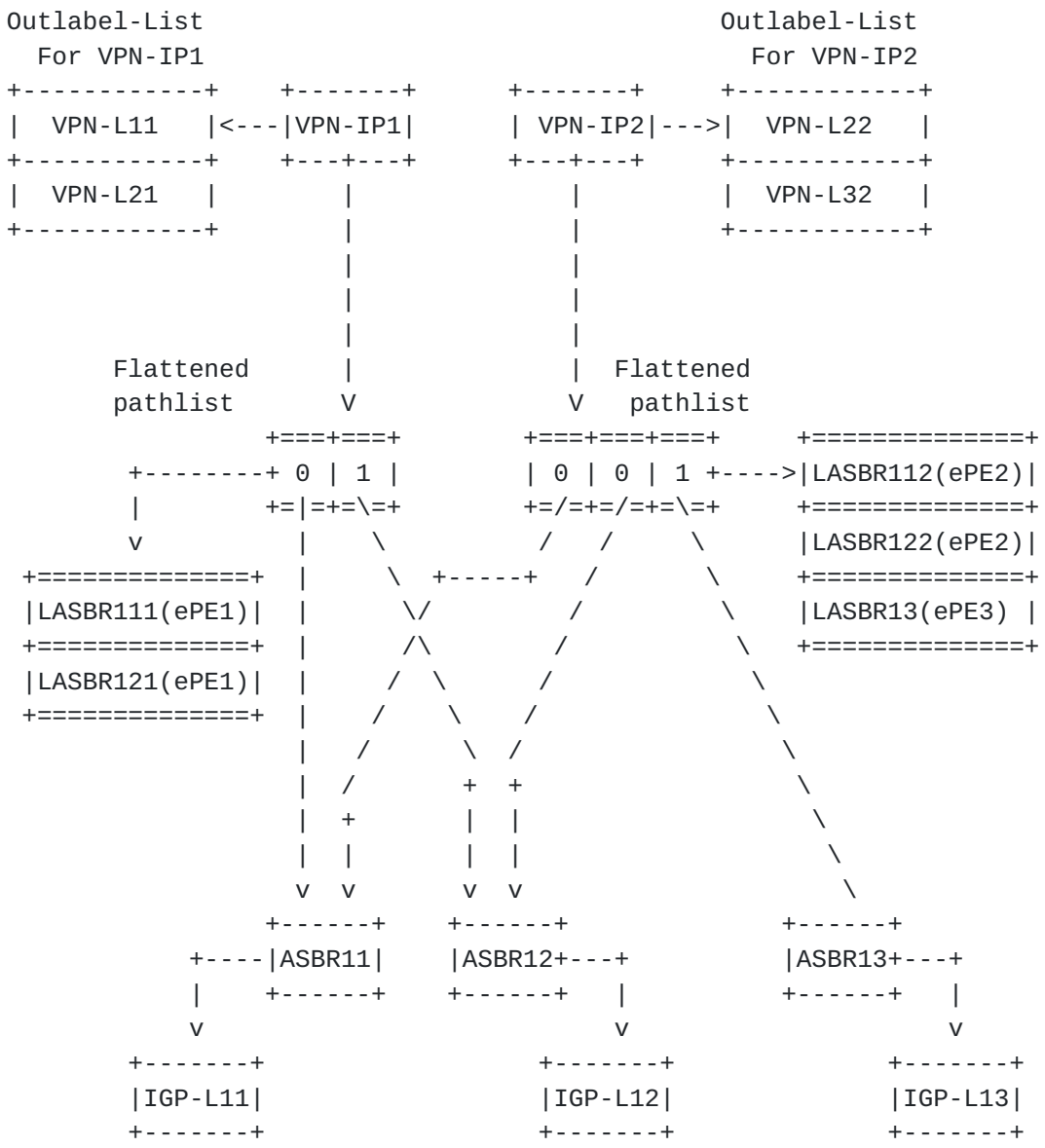

Figure 6 : Flattening 3 levels to 2 levels of Hierarchy on iPE

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

- As mentioned in Section 5.1 a flattened pathlist may have label lists associated with them. The size of the label list associated with a flattened pathlist equals the size of the pathlist. Hence it is possible that an implementation includes these label lists in the flattened pathlist itself.

- Again as mentioned in Section 5.1, the size of a flattened pathlist may not be equal to the size of the OutLabel-lists of leaves using the flattened pathlist. So the indices inside a flattened pathlist still indicate the label index in the Outlabel-Lists of the leaves using that pathlist. Because the size of the flattened pathlist may be different from the size of the OutLabel-lists of the leaves, the indices may be repeated.

- Let's take a look at the flattened pathlist used by the prefix "VPN-IP2", The pathlist associated with the prefix "VPN-IP2" has three entries.

    - The first and second entry have index "0". This is because both entries correspond to ePE2. 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-L22", which is the label advertised by ePE2 for the prefix "VPN-IP2".

    - The third entry has the index "1". This is because the third entry corresponds to ePE3. 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-L32", which is the label advertised by "ePE3" for the prefix "VPN-IP2".

Now let's try and apply the forwarding steps in Section 4 together with the additional step in Section 5.1 to the flattened forwarding chain illustrated in Figure 6.

- Suppose a packet arrives at "iPE" and matches the VPN prefix "VPN-IP2".

- The forwarding engine walks to the parent of the "VPN-IP2", which is the flattened pathlist and applies a hashing algorithm to pick a path.

- Suppose the hashing by the forwarding engine picks the second path in the flattened pathlist associated with the leaf "VPN-IP2".

- Because the second path has the index "0", the label "VPN-L22" is pushed on the packet.

- Next the forwarding engine picks the second label from the Outlabel-Array associated with the flattened pathlist. Hence the next label that is pushed is "LASBR122(ePE2)".

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

- So the packet is forwarded towards the ASBR "ASBR12" and the IGP label at the top will be "L12".

Based on the above steps, a packet arriving at iPE and destined to the prefix VPN-L22 reaches its destination as follows:

- iPE sends the packet along the shortest path towards ASBR12 with the following label stack starting from the top: {L12, LASBR122(ePE2), VPN-L22}.

- The penultimate hop of ASBR12 pops the top label "L12". Hence the packet arrives at ASBR12 with the label stack {LASBR122(ePE2), VPN-L22} where "LASBR12(ePE2)" is the top label.

- ASBR12 swaps "LASBR122(ePE2)" with the label "LASBR22(ePE2)", which is the label advertised by ASBR22 for the ePE2 (the egress PE).

- ASBR22 receives the packet with "LASBR22(ePE2)" at the top.

- Hence ASBR22 swaps "LASBR22(ePE2)" with the IGP label for ePE2 advertised by the next-hop towards ePE2 in domain 2, and sends the packet along the shortest path towards ePE2.

- The penultimate hop of ePE2 pops the top label. Hence ePE2 receives the packet with the top label VPN-L22 at the top

- ePE2 pops "VPN-L22" and sends the packet as a pure IP packet towards the destination VPN-IP2.

6. Forwarding Chain Adjustment at a Failure

The hierarchical and shared structure of the forwarding chain explained in the previous section 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 BGP prefix independent convergence.

6.1. 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/or 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.

When a local link fails, FIB manager detects the failure almost immediately. The FIB manager marks the impacted path(s) as unusable so that only useable paths are used to forward packets. Hence only IGP pathlists with paths using the failed local link need to be modified. All other pathlists are not impacted. Note that in this particular case there is actually no need even to backwalk to IGP leaves to adjust the OutLabel-Lists because FIB can rely on the path-index stored in the useable paths in the pathlist 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, assuming a backup path has been precomputed, within the timeframe of local detection (e.g. 50ms). Examples of solutions that pre-computing backup paths are IP FRR [15] remote LFA [16], Ti-LFA [14] and MRT [17] or eBGP path having a backup path [9].

Let's apply the procedure mentioned in this subsection to the forwarding chain depicted in Figure 2. Suppose a remote link failure occurs and impacts the first ECMP IGP path to the remote BGP next-hop. Upon IGP convergence, the IGP pathlist used by the BGP next-hop is updated to reflect the new topology (one path instead of two). As soon as the IGP convergence is effective for the BGP next-hop 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 next-hop 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 next-hop and a backup path was pre-computed and installed in the pathlist, upon the local interface failure, the LFA backup path is immediately activated (e.g. sub-50msec) and thus protection benefits all the

depending BGP traffic through the hierarchical forwarding dependency between the routes.

6.2. BGP-PIC edge

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

6.2.1. 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:

- FIB manager deletes the IGP leaf corresponding to the failed edge node
- FIB manager backwalks to all dependent BGP pathlists and marks that path using the deleted IGP leaf as unresolved
- Note that there is no need to modify the possibly large number of BGP leaves because each path in the pathlist carries its path index and hence the correct outgoing label will be picked. Consider for example the forwarding chain depicted in Figure 2. If the 1st BGP path becomes unresolved, then the forwarding engine will only use the second path for forwarding. Yet the path index of that single resolved path will still be 1 and hence the label VPN-L12 will be pushed.

6.2.2. 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 applies the following steps:

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

- Hence traffic will be forwarded used the backup path towards ePE2.

- For labeled traffic

  - The Outlabel-List attached to the BGP leaf already contains an entry corresponding to the backup path.

  - The label entry in OutLabel-List corresponding to the internal path to backup egress PE has swap action to the label advertised by backup egress PE.

  - For an arriving label packet (e.g. VPN), the top label is swapped with the label advertised by backup egress PE and the packet is sent towards that backup egress PE.

- For unlabeled traffic, packets are simply redirected towards backup egress PE.

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 6.2.1.

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 [9].

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 swapped with the label VPN-L21 and the packet is forwarded towards ePE2.

6.3. Handling Failures for Flattened Forwarding Chains

As explained in the in Section 5 if the number of hierarchy levels of a platform cannot support the native number of hierarchy levels of a recursive forwarding chain, 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 in domain 1. If the platform supports the full hierarchy depth, the forwarding chain is the one depicted in Figure 5 and hence the FIB manager needs to backwalk one level to the pathlist shared by "ePE1" and "ePE2" 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 updates both of them.

The main observation is that the loss of convergence speed due to the loss of hierarchy depth depends on the structure of the forwarding chain itself. To illustrate this fact, let's take two extremes. Suppose the forwarding objects in level i+1 depend on the forwarding objects in level i. If every object on level i+1 depends on a separate object in level i, then flattening level i into level i+1 will not result in loss of convergence speed. Now let's take the other extreme. Suppose "n" objects in level i+1 depend on 1 object in level i. Now suppose FIB flattens level i into level i+1. If a topology change results in modifying the single object in level i, then FIB has to backwalk and modify "n" objects in the flattened level, thereby losing all the benefit of BGP-PIC. Experience shows that flattening forwarding chains usually results in moderate loss of BGP-PIC benefits. Further analysis is needed to corroborate and quantify this statement.

7. Properties

7.1. 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 next-hop as described in Section 6. 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.

7.1.1. A remote failure on the path to a BGP next-hop

Upon IGP convergence, the IGP leaf for the BGP next-hop is updated upon IGP convergence and all the BGP depending routes leverage the new IGP forwarding state immediately. Details of this behavior can be found in Section 6.1.

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

7.1.2. A local failure on the path to a BGP next-hop

Upon LFA protection, the IGP leaf for the BGP next-hop is updated to use the precomputed LFA backup path and all the BGP depending routes

leverage this LFA protection. Details of this behavior can be found in Section 6.1.

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

7.1.3. A remote iBGP next-hop fails

Upon IGP convergence, the IGP leaf for the BGP next-hop 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. Details about this behavior can be found in Section 6.2.1.

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

7.1.4. A local eBGP next-hop fails

Upon local link failure detection, the adjacency to the BGP next-hop is deleted and all the depending BGP pathlists are updated to either use the remaining ECMP BGP best-paths or if none remains available to activate precomputed backups. Details about this behavior can be found in Section 6.2.2.

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

7.2. Performance

When the failure is local (a local IGP next-hop failure or a local eBGP next-hop 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 next-hop or a remote BGP next-hop 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 next-hop even if millions of BGP routes are impacted.

Appendix A puts the BGP-PIC benefits in perspective by providing some results using actual numbers.

7.3. 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:

- Extension of the BGP Best Path to compute more than one primary ([10]and [11]) or backup BGP next-hop ([5] and [12]).
- Sharing of BGP Path-list across BGP destinations with same primary and backup BGP next-hop.
- Hierarchical indirection and dependency between BGP pathlist and IGP pathlist.

7.4. 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.

8. Security Considerations

The behavior described in this document is internal functionality to a router that result in significant improvement to convergence time as well as reduction in CPU and memory used by FIB while not showing change in basic routing and forwarding functionality. As such no additional security risk is introduced by using the mechanisms proposed in this document.

9. IANA Considerations

No requirements for IANA

10. Conclusions

This document proposes a hierarchical and shared forwarding chain structure that allows achieving BGP 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 thereby supporting smooth and incremental deployment.

11. References

11.1. Normative References

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

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

[3] Y. Rekhter and E. Rosen, " Carrying Label Information in BGP-4", RFC 8277, October 2017

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

11.2. Informative References

[5] 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.

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

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

[8] 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

[9] 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

[10] D. Walton, A. Retana, E. Chen, J. Scudder, "Advertisement of Multiple Paths in BGP", RFC 7911, July 2016

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

[12] 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

[13] A. Bashandy, C. Filsfils, S. Previdi, B. Decraene, S. Litkowski, M. Horneffer, R. Shakir, "Segment Routing with MPLS data plane", RFC 8660, December 2019

[14] S. Litkowski, A. Bashandy, C. Filsfils, B. Decraene, P. Francois, D. Voyer, F. Clad, P. Camarillo" Topology Independent Fast Reroute using Segment Routing", draft-ietf-rtgwg-segment-routing-ti-lfa-02 (work in progress), January 2020

[15] M. Shand and S. Bryant, "IP Fast Reroute Framework", RFC 5714, January 2010

[16] S. Bryant, C. Filsfils, S. Previdi, M. Shand, N So, " Remote Loop-Free Alternate (LFA) Fast Reroute (FRR)", RFC 7490 April 2015

[17] A. Atlas, C. Bowers, G. Enyedi, " An Architecture for IP/LDP Fast-Reroute Using Maximally Redundant Trees", RFC 7812, June 2016

12. 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
Individual Contributor
Email: abashandy.ietf@gmail.com

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

Prodosh Mohapatra
Sproute Networks
Email: mpradosh@yahoo.com

Appendix A. Perspective

The following table puts the BGP-PIC benefits in perspective assuming

- 1M impacted BGP prefixes
- IGP convergence ~ 500 msec
- local protection ~ 50msec
- FIB Update per BGP destination ~ 100usec conservative,
  ~ 10usec optimistic
- 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 next-hop failure or remote IGP next-hop failure, the existing primary BGP next-hop is intact and usable hence the resiliency only depends on the ability of the FIB mechanism to reflect the new path to the BGP next-hop 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 next-hop failure or remote BGP next-hop 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.