Achieving Sub-50 Milliseconds Recovery Upon BGP Peering Link Failures

Olivier Bonaventure, Clarence Filsfils and Pierre Francois

Abstract— **Recent measurements show that BGP peering links can fail as frequently as intradomain links and usually for short periods of time. We propose a new fast-reroute technique where routers are *prepared* to react quickly to interdomain link failures. For each of its interdomain links, a router precomputes a *protection tunnel*, i.e. an IP tunnel to an alternate nexthop which can reach the same destinations as via the protected link. We propose a BGP-based auto-discovery technique that allows each router to learn the candidate protection tunnels for its links. Each router selects the best protection tunnels for its links and when it detects an interdomain link failure, it immediately encapsulates the packets to send them through the protection tunnel. Our solution is applicable for the links between large transit ISPs and also for the links between multi-homed stub networks and their providers. Furthermore, we show that transient forwarding loops (and thus the corresponding packet losses) can be avoided during the routing convergence that follows the deactivation of a *protection tunnel* in BGP/MPLS VPNs and in IP networks using encapsulation.**

I. Introduction

The TCP/IP protocol suite was developed more than twenty years ago to serve the needs of researchers sending best-effort packets over a research network. Today, the same protocol suite has become the standard protocol suite in enterprise networks and the global Internet. Furthermore, Virtual Private Networks (VPN)[2], telephony and video services are now increasingly being deployed over an IP-based infrastructure.

To support those mission-critical applications, networks need to guarantee very stringent Service Level Agreements (SLA). Those SLAs typically require very low packet loss ratio, bounded delays through the network, high network availability (99.99 % or better) and a short restoration time after a failure. IP-based networks are being used to support almost any data transmission service including leased-line emulations [3]. For such stringent services, restoration times below 50 milliseconds are a common requirement [4].

When the network is stable and there are no link failures, buffer acceptance, marking and scheduling mechanisms implemented on today's routers [5] allow ISPs to provide the performance guarantees required by their customers. Unfortunately, the links used in IP networks are not 100% stable and measurements carried in operational networks indicate that link failures are common events [6], [7], [8], [9], [10]. Furthermore, many of those failures only last for a few seconds or tens of seconds.

This work was supported by Cisco Systems within the ICI project. Any opinions, findings, and conclusions or recommandations expressed in this paper are those of the authors and do not necessarily reflect the views of Cisco Systems. A first version of this paper was presented at CoNext 2005 [1]

ISPs typically use several techniques to quickly react to the failures of their intradomain links. A first solution is to rely on the convergence of the link-state intradomain routing protocols. In the past, this convergence was in the order of a few seconds, but recent improvements allow large networks to converge within less than one second [11]. Other techniques are required to achieve a faster convergence. For those "fast" techniques, the target is usually to restore a failure within 50 milliseconds. In some networks, the failures are handled by the SONET/SDH underlying layers [4]. In MPLS-based networks, fast-reroute and bypass tunnels [4] allow to protect failed links by locally rerouting packets around the failure. In pure IP networks, several solutions applicable to protect intradomain links are currently being discussed within the IETF [12].

In addition to affecting intradomain links, failures also affect BGP peering links between ASes or links between a BGP/MPLS VPN service provider and a customer site. In this case, ISPs depend on BGP to be able to recover from those failures. We analyse in section II-A measurements showing that the current state-of-the-art with current BGP routers is far from the 50 milliseconds target imposed by stringent real-time applications.

Several authors have proposed modifications to reduce the BGP convergence time in case of failures [13], [14]. Those techniques reduce the BGP convergence time by reducing the number of BGP messages that must be exchanged after a failure. However, as they depend on the exchange of messages, the achieved convergence time will always be much larger than the 50 milliseconds target for stringent real-time services.

In this paper, we propose a new fast-reroute technique that allows to provide sub-50 milliseconds restoration when a BGP peering link fails. We first assume that the failures of the interdomain links are detected by using a trigger from the physical layer such as a SONET loss of signal [4] or a protocol such as BFD [15]. This failure detection typically takes less than 15 milliseconds [16] on high-end routers. Instead of asking routers to *react* to the failure of their BGP peering links by starting an IGP or BGP convergence, our technique *prepares* the routers to quickly handle the failure of such links. For this, each router locates an *alternate nexthop* for each of its BGP peering links. We propose a BGP extension that allows a router to automatically discover the *alternate nexthops* for each of its BGP peering links. When a BGP peering link fails, the router that detects the failure immediately updates its Forwarding Information Base (FIB) to encapsulate the packets that were using the failed link and sends them to an *alternate nexthop* through an IP tunnel. The *alternate nexthop* will send the packets to their final destination without using the failed link. On high-end routers, we show how it is possible

to modify the FIB within the 50 milliseconds budget. The tunnel to the *alternate nexthop* allows to avoid packet losses by sending the packets on another path than the shortest path. After some time, the router attached to the failed link may need to announce the failure. This will cause a BGP convergence at least inside the local AS. For BGP/MPLS VPNs and IP networks using encapsulation, we show that no packet will be lost in the AS during this convergence.

The remainder of this paper is organised as follows. In section II, we analyse the impact of the failures of BGP peering links. In section III we first discuss the problem of protecting interdomain links and show that there are two different problems : the *stub* and the *parallel-links* problems. We then describe the principles of our solution in section IV. We show in sections V and VI how those two problems can be solved by using protection tunnels. Then, in section VII we discuss the conditions under which it is possible to remove an activated protection tunnel without causing packet losses or transient forwarding loops during the routing convergence that follows the deactivation of the protection tunnel. Finally, we compare our proposal with related work in section VIII.

II. Failures of BGP peering links

Several studies have analysed the performance of the global Internet and the impact of link failures. A common approach is to collect the link state packets exchanged by routers in a large network and infer the link failures from the reported changes. This method has been applied to several operational ISP networks [6], [7]. Those studies considered different networks, but they basically found three important results. First, *link failures are common events* that must be efficiently handled by the routing protocols. Second, a small number of links are responsible for a large fraction of the failures. This is the common but annoying problem of flapping links. Third, link failures are usually short-lived events. Very often, the duration of a link failure is around a few or a few tens of seconds. However, those analyses were focused on links inside ISP networks and it is not sure that similar results apply to BGP peering links.

In [1], we presented a detailed analysis of the BGP peering link failures in a transit AS containing 47 different eBGP nexthops. All of the peering relationships of the studied AS involved a single peering link with the neighbour AS, except for four neighbour ASes which were each interconnected via two peering links to the studied AS and one neighbour AS which had four peering links to the studied AS. During the studied three-month period, we found 9452 distinct failures. However, the failures were not equally spread among the peering links. In fact, 83% of the failures occurred on a single eBGP peering link. Discussions with the operator revealed that this link had indeed problems at the physical layer which explained the large amount of failures of this link. Four other links had more than 100 failures during the three month period and some links did not fail at all.

We also looked at the durations of the BGP peering link failures. For this, we focused on the 42 BGP peering links that were the most stable. This analysis revealed several interesting points. First, 22% of the eBGP peering link failures lasted less than 1 second. During such short failures, packets can be lost, but BGP does not usually have enough time to detect the failure and start a convergence. Second, 82% of the failures lasted less than 180 seconds. This is similar to the study of intradomain link failures reported in [17],where about 70% of the failures lasted less than 3 minutes.

Another study of interdomain routing failures was performed recently by Wang et al. [18]. This study collected BGP routes on a route monitor attached to 52 routers in a large ISP network. They also observed that most of the routing failures are short lived. In their study, more than 60% of the routing failures last less than 100 seconds. This is similar to the findings we reported in [1]. However, as they relied on BGP messages to detect the routing failures, the very short failures lasting less than one second were probably masked by the BGP routers in their study.

A. Impact of eBGP session failure

The previous section has shown that eBGP peering failures are common events. Another important point to be considered concerning those events is their impact on the data packets. When an eBGP peering link fails in an AS, how much time is necessary to allow all the routers of the AS to completely recover from this failure ?

Several authors recently proposed measurements addressing this issue in different networks. Pei and Van der Merwe analysed the BGP convergence in a BGP/MPLS VPN network [19]. For this, they analysed different types of data including BGP data and `syslog` messages from an operational network. One of the failure scenarios that they consider is when an eBGP session comes down and the traffic needs to be rerouted to another eBGP session. In this case, their measurements show that the recovery time is longer than 20 seconds for 17% of the failure events they consider. Those long recovery times are attributed to the failure detection time and the time required to distribute the iBGP messages during the iBGP convergence. Wang et al. analyse in [20] the convergence time and the impact on data traffic after the withdrawal or announcement of a BGP route by a BGP beacon.

To evaluate the impact of the failure of an eBGP session on the transmission of interdomain packets, we performed several blackbox measurements in a lab. For these measurements, we considered several high-end router models from different vendors. All the routers that we considered were fitted with several OC-48 interfaces and are typical of the routers found in large ISP networks. For each experiment, we used three routers and two Agilent router-testers to model a very simple AS with a full-mesh of iBGP sessions between the three routers as shown in figure 1. Router $R1$ and $R2$ maintain an eBGP session with the router-tester $T2$ that announces 100,000 different IPv4 prefixes. Routers $R1$ and $R2$ are both connected to router $R3$. Due to the IGP weights used on the $R3-R1$ and $R3-R2$ links, router $R3$ prefers to use router $R2$ to reach the prefixes advertised by $T2$. Finally, $T1$ is another router-tester that we configured to send a constant rate of one million packets per second towards all the prefixes advertised by $T2$.

In this network, all packets reach the prefixes advertised $T2$ through the $R2-T2$ link. Router $R3$ has learned the routes from both $R2$ and $R1$, but it has only installed the best routes learned from $R2$ inside its FIB. When link $R2-T2$ fails, router $R2$ must inform all the routers of the AS of the failure. In our network, $R2$ advertised the prefix of link $R2-T2$ inside its link state packet. Thus, when link $R2-T2$ fails, $R2$ simply sends a new link-state packet without the prefix of link $R2-T2$. Upon reception of this packet, router $R3$ learns that all the interdomain prefixes that it learned from $R2$ are unreachable. Since at that time router $R3$ already received all the prefixes learned from $R1$, it can immediately start to update its FIB. It should be noted that such a network is a best-case since no BGP message needs to be exchanged before updating the FIB. In a real network, router $R3$ would probably not know all the alternate routes [20] due to routing policies or route reflectors and this would further delay the reaction time.

To evaluate the time required to perform this update of the FIB, we capture the packets received by router-tester $T2$. Since router tester $T1$ is sending one million packets per second towards all the 100,000 prefixes advertised by $T2$, we are in fact sampling the state of $R3$'s FIB for each prefix ten times per second. When router $R3$ receives a packet destined to prefix p, the packet will only reach its destination (i.e. $T2$) if $R3$ has already updated its FIB for this prefix. Otherwise, the packet will be sent to $R2$ that will either drop it (if its own FIB has not yet been updated) or return it to $R3$. Figure 2 plots the percentage of the packets sent by $T1$ that are received by router-tester $T2$ in function of time. At time t=7 seconds, router-tester $T2$ does not receive any packet, indicating the failure of link $R2-T2$. Between time=8 seconds and time=20 seconds, the percentage of packets received by router-tester $T2$ slowly increases. This slow increase is because router $R3$ needs some time to update each FIB entry. A closer look at the duration of the FIB update shows that router $R3$ needs 11.3 seconds to completely update its FIB. Given that this FIB contains 100,000 prefixes, this implies that router $R3$ needs on average 113 microseconds to update one entry inside its BGP FIB. This amount of time is similar to the measurements presented in [11] for the update of the IGP FIB. We performed the same measurements with other router models from several vendors. The router models that we evaluated required between one hundred and a few hundred of microseconds on average to update one entry in their FIB.

III. Problem statement

There are several ways of interconnecting ASes together [21]. To design our fast reroute technique, we first assume that if ASx considers that a BGP peering link with ASy is valuable enough to be protected, then there should at least be a second link between ASx and ASy. This is a very reasonable requirement from an operational viewpoint.

This type of interconnection is very common between transit ISPs and when stub ASes are connected with redundant links to their provider. For such multi-connected ASes, the failure of one interdomain link can be naturally handled by redirecting the packets sent on the protected link to another link with the same AS. For example, in figure 3, if link $R1-X1$ fails, then $R1$ should be able to immediately reroute the packets that were using the failed link to $X2$ via $R2$. This redirection of the packets is possible provided that the same destinations are reachable via the two parallel links. This is a common requirement for peering links [22] and can be a design guideline to provide sub-50 milliseconds recovery in case of failures.

A similar interconnection is also used in BGP/MPLS VPNs (right part of figure 3). For important customer sites, it is common to attach two *customer edge* (CE) routers from this site to two different *provider edge* (PE) routers of the service provider. In the right part of figure 3, if link $PE1-CE1$ fails, then $PE1$ should be able to immediately reroute the packets that were using the failed link to $CE2$ via $PE2$.

We call the problem of protecting such links the *parallel-links* problem in the remainder of this document. To be deployable, a solution to the *parallel-links* problem will need to meet four requirements.

1) The same solution should be applicable for **both directions** of the interdomain link.
2) As a router controls its outgoing traffic, it should be able to protect it **without any cooperation** with BGP routers outside its AS. This implies that if a tunnel is used, the packet de-encapsulation should be performed in the same AS. A cooperation between routers in neighbouring ASes may improve the performance of the solution, but it should not be required.
3) Links between distinct routers may fail at the same time [4], [6] because they use a shared physical infrastructure (fibre, physical or datalink devices). The set of links that share the same physical infrastructure is usually called a **Shared Risk Link Group (SRLG)**. The solution to the *parallel-links* problem should take into account those SRLGs.
4) The solution should also take into account the **BGP policies** [23] used for the interdomain links. In most cases when there are multiple links between two ASes, the same BGP policy (e.g. *shared cost peering* or *customer-provider*) is used over all these links. However, the routing policies used between large transit ASes can be more complex. For example, a tier-2 ISP may be a customer of a tier-1 ISP in the US and a peer of the same ISP in Asia. Another example is a corporate network that advertises different prefixes over the multiple links with its provider.

At this point, it is important to note that the second requirement prohibits the utilisation of the MPLS-based fast-reroute and bypass tunnels [4] to protect a peering link. Indeed, to establish such tunnels, the peering routers must be able to find an alternate path and exchange RSVP-TE messages to signal the bypass or fast-reroute tunnel.

While requiring the utilisation of parallel-links is reasonable for large ASes, it could be too strong for small multi-homed stub ASes. A solution should also be developed to allow a multi-homed stub AS to protect its interdomain links (figure 4) when it is attached with a single link to each of its providers.

We call this problem the *stub* problem in the remainder of this paper. In the *stub* problem, there are two different sub-problems. In the *outgoing stub* problem, the stub AS needs to protect its outgoing packet flow. The solution developed to solve this problem should meet the same requirements as the solution to the *parallel-links* problem as the stub can reach all destinations via either of its two providers.

The second sub-problem is called the *incoming stub* problem. In this case, the stub AS wishes to protect the incoming direction of an interdomain link. The solution developed to solve this problem will require a cooperation between the stub AS and its providers. This cooperation is not a problem as the utilisation of a fast recovery technique can be part of the contract between the stub and its provider. Furthermore, it should be possible to use the proposed technique to protect one link and not the others. For this, no mutual cooperation between the providers should be required. For example, in figure 4 it should be possible for router $Z2$ to protect link $Z2 \rightarrow X2$ without any change to router $Y1$. In figure 4, when link $Z2 \rightarrow X2$ fails router $Z2$ should be able to immediately reroute the packets so that they reach the stub without waiting for a BGP convergence.

IV. Principle of our solution

In this section, we briefly describe the key elements of our proposed solution based on a simple example. Additional details will be provided in the remaining sections. We consider the two ISPs shown in figure 5 and focus on the packets flowing from the upstream AS to the downstream AS. We assume that the downstream AS advertises the same prefixes over both links and that the routing policies on $X1$ and $X2$ are configured such that $X2 \rightarrow R2$ is used to forward packets while $X1 \rightarrow R1$ is only a backup link. This configuration can be achieved by setting a low `local-pref` value on the BGP routes learned by $X1$ for example.

To quickly react to a failure of directed link $X2 \rightarrow R2$, router $X2$ must be able to quickly update its FIB to send the packets affected by the failure via an alternate path. We describe in section IV-A a technique that allows the FIB to be updated in less than 50 milliseconds. In figure 5, the alternate path is clearly through the $X1 \rightarrow R1$ link. Let us assume in this section that router $X2$ was manually configured with this alternate path. We will discuss later mechanisms that allow router $X2$ to automatically discover this alternate path. To forward the packets affected by the failure through the $X1 \rightarrow R1$ link, router $X2$ cannot simply send them on its interface towards $X3$ as $X3$'s BGP table indicates that the nexthop for those prefixes is router $X2$. We show in section IV-B that by using *protection* tunnels it is possible to avoid such loops.

A. A fast update of the FIB

The update of the FIB after the failure is a key implementation issue to achieve the sub-50 milliseconds target. The FIB is a data structure that associates a BGP prefix to a nexthop and an outgoing interface. Figure 6 shows the conceptual view of such a FIB as two tables. In such a FIB, the outgoing interface is obtained from the IGP routing table. Detailed measurements performed on high-end routers revealed that the time required to update one entry of such a FIB was on average around 110 microseconds per entry [11]. This implies that less than 5000 FIB entries can be updated within the sub-50 milliseconds target on such routers.

To achieve the sub-50 milliseconds target it is necessary to reduce the number of FIB entries that must be modified after the detection of a failure. There are several possible methods to reroute packets towards many destinations without changing a large number of entries in the FIB. Some commercial routers already support such mechanisms [24]. The exact organisation of the FIB strongly depends on the hardware capabilities of the concerned router. The details of those FIB organisations are outside the scope of this paper. We show, conceptually, one possible organisation of the FIB to illustrate the possibility of achieving this fast FIB update.

The new organisation of the FIB is illustrated in figure 7. Conceptually, this FIB is organised as two tables. The first table contains the BGP prefixes and the BGP nexthops are *pointers* to a table (noted *P(...)*) of all nexthop entries. Each nexthop entry in the second table contains the address of the nexthop, a flag that indicates whether the link to the nexthop is up or down and two outgoing interfaces (OIF) : a primary OIF and a secondary OIF. The OIF is in fact a data structure that contains all the information required to forward packets on this interface. For a point-to-point interface, this data structure will contain the layer 2 encapsulation to be used (e.g. PPP or Packet over SONET). For a point-to-multipoint interfaces, the data structure will contain the layer 2 encapsulation and the layer 2 address of the nexthop router. For a virtual interface such as a tunnel, the FIB will contain the IP address of the tunnel endpoint and the tunnel specific parameters. Those parameters are useful notably for L2TP [25] or MPLS over IP tunnels [26].

With this new FIB, when the router consults its nexthop table, it uses the primary OIF if the flag is set to "Up" and the backup OIF otherwise. This means that when a peering link fails, a *single* modification to the Nexthops Table is sufficient to reroute *all* affected prefixes over a *protection* tunnel. This clearly meets the sub-50 milliseconds target.

B. The protection tunnels

As explained earlier, a solution is required to allow router $X2$ to reroute the packets immediately to router $X1$ even if the routing tables of $X3$ and $X1$ still point to $X2$ as their nexthop. For this, two different types of tunnels can be envisaged :

- A tunnel from the *primary egress* router (`X2`) to another egress router (e.g. `X1`) of the upstream AS that peers with the same downstream AS. We call this tunnel a *primary egress - secondary egress* or *pe-se* tunnel and will use such tunnels to solve the *parallel-link* problem.
- A tunnel from the *primary egress* router (`X2`) to another ingress router in the downstream AS (e.g. `R1`). We call this tunnel a *primary egress - secondary ingress* or *pe-si* tunnel and will use such tunnels to solve the *stub* problem.

The *pe-se* and *pe-si* protection tunnels are "pre-defined" before the link failure. At the *primary egress* router, a protection

tunnel is defined by two parameters : an encapsulation header and an outgoing interface. At the secondary *ingress* or *egress*, the definition of the protection tunnel is simply the ability to de-encapsulate the packets received over the tunnel.

Several types of protection tunnels exist : IP over IP, GRE, IPSec, L2TP, MPLS over IP, ... However, not all encapsulation types are suitable for *pe-se* tunnels. Consider again figure 5. When link $X2 \rightarrow R2$ fails, router $X2$ will encapsulate the packets towards router $X1$. If $X2$ uses IP-in-IP encapsulation, then router $X1$ will use its FIB to forward the de-encapsulated packets. Unfortunately, $X1$'s FIB may still use $X2$ as the nexthop to reach the affected prefixes.

To avoid this problem, we require the utilisation of an encapsulation scheme that contains a label such as L2TP [25] or MPLS over IP [26]. This label is assigned by the *secondary egress* router. When it receives an encapsulated packet, it uses the label as a key to forward the de-encapsulated packet over the appropriate secondary link *without consulting its BGP FIB*. This ensures that the *secondary egress* will not return the received encapsulated packets to the *primary egress* router even if this *primary egress* is the current BGP nexthop according to the FIB of the secondary egress router.

Using IP-based tunnels usually raises two immediate questions. The first one is the cost of encapsulation and de-encapsulation. In the past, those operations were performed on the central CPU of the router and were costly from a performance viewpoint [27]. Today, the situation is completely different and high-end routers are able to perform encapsulation or de-encapsulation at line rate. Furthermore, many large ISPs have deployed MPLS to support BGP/MPLS VPNs and some rely on L2TP or GRE-based encapsulation [28]. The second question is the problem of fragmenting packets whose size exceeds the MTU. On current Packet over SONET interfaces used by high-end routers, this issue becomes a design problem : the network must be designed to ensure that the MTU is large enough. The design guidelines developed for GRE-based tunnels in [28] would ensure that fragmentation is avoided when IP-based protection tunnels are used.

In a production network, allowing routers to process encapsulated packets may cause security problems unless the routers have a way to verify that the packets come from legitimate sources. For the *pe-se* tunnels, the tunnel source belongs to the same ISP as the tunnel destination. In this case, IP-based filters such as those already deployed by ISPs [29] would be sufficient. For the *pe-si* tunnels, the *secondary ingress* should be able to verify the validity of the received encapsulated packets. A possible solution could be to use IPSec for those tunnels. Another solution would be to use filters.

To define a *pe-se* (resp. *pe-si*) protection tunnel, the *primary egress* router must thus determine the IP address of the appropriate *secondary egress* (resp. *secondary ingress*) router and the tunnel type to be used. We propose in the following sections techniques to select the endpoints of the protection tunnels.

V. THE *parallel-links* PROBLEM

To solve the *parallel-links* problem, we utilise *pe-se* protection tunnels. Such tunnels could be configured manually on the *primary-egress* router. For example, the network operator could configure on this router the addresses of the candidate *secondary-egress* routers and the parameters of the *pe-se* tunnel to be used. This manual configuration would be sufficient in the common case where a small stub AS is connected to its provider via two interdomain links. However, in a large network, an auto-discovery mechanism is required to simplify the configuration and more importantly to allow the routers to automatically adapt the protection tunnels to topology changes.

To build this auto-discovery mechanism, we first consider the simple case of two physically independent parallel links and assume that the same prefixes are advertised by the downstream AS over those links. In this case, the main problem for the *primary egress* router is to locate the appropriate *secondary egress* router. To discover the *secondary egress* router, the *primary egress* router cannot simply consult its BGP table as it may not have alternate routes for the affected prefixes. For example, in figure 5, router $X2$ does not learn any route advertised by the downstream AS from router $X1$ due to the `local-pref` settings on this router. A similar situation could occur in a large AS, where due to the utilisation of BGP confederations or route reflectors, routers only receive a single route towards some destinations.

To solve this auto-discovery problem, we propose to allow each egress router to advertise via iBGP the "characteristics" of its currently active eBGP sessions by using a new type of BGP routes called *protection routes*. A *protection route* contains the following information :

- the Network Layer Reachability Information (NLRI) is the local IP address on the peering link with the downstream AS.
- the AS-Path attribute contains only the downstream AS
- a tunnel attribute containing the parameters of the protection tunnel to be established

The IP address used in the NLRI must be routable and unique, at least within the upstream AS. The uniqueness of the NLRI information is necessary to ensure that the *protection route* will be distributed to all the routers inside the upstream AS. If the same NLRI was used for several protection routes, then a route reflector could run the BGP decision process to advertise only one of them to its clients. By using a unique NLRI for each protection route, we ensure that the protection route is distributed throughout the AS even if there are route reflectors or confederations. The tunnel attribute indicates the supported type of tunnel (GRE, L2TP or MPLS over IP tunnels) and the optional parameters such as the label for MPLS over IP encapsulation.

It is important to note that a router advertises *one* protection route for each of its active eBGP sessions. A *protection route* is only advertised when the corresponding BGP peering link is active. When a peering link fails, the corresponding *protection route* is withdrawn. Furthermore, the protection routes are only distributed inside the local AS. For these reasons, the iBGP load due to the protection routes is negligible compared to the normal iBGP load.

When the *primary egress* router needs to select a *pe-se* tunnel endpoint for a primary link, it considers as candidate

secondary egress routers all the protection routes whose AS-Path is equal to the downstream AS and whose tunnel endpoint is reachable according to its IGP routing table. In practice, the closest *secondary egress* would often be the best one.

However, as discussed in section III, the solution should also be able to protect from SRLG failures. To be able to correctly handle SRLG failures, the routers need to know the SRLG associated with each BGP peering link. For example, considering figure 8, router $R2$ should not be selected as a *secondary egress* to protect link $R1 \rightarrow X1$ as link $R2 \rightarrow X1$ also terminates at router $X1$. In practice, a BGP peering link can be characterised by a set of SRLG values specified by the network operator [4]. A BGP peering link is composed of two half-links, one half in the upstream AS and the other in the downstream AS. It will thus be characterised by SRLG values managed by the downstream AS and SRLG values managed by the upstream AS. The SRLG values can be manually configured on a per eBGP session basis by encoding each value as a pair `AS#:SRLG-value` of 32 bits integers[1] where `AS#` is the AS number of the AS that allocated the SRLG value.

Another problem to be considered is when different BGP policies are used over the parallel-links. As an example, consider the network topology shown in figure 8. Assume that *primary egress* router `R1` needs to create a protection tunnel for directed link `R1`→`X1` and that `R1` and `R3` receive all routes known by `AS2` (full routing) while `R2` only receives the routes from the clients of `AS2`. In this case, router `R1` should select `R3` as its *secondary egress* since `R3` receives the same routes as `R1`.

To solve this problem, each egress router must know the BGP policy used by its peer. This is because the packets that are sent on the *primary-egress → primary ingress* link depend on the BGP routes advertised by the *primary ingress* router. For this, we propose to add to the configuration of each eBGP session an integer, called *eBGP session type* that identifies the BGP policy (customer, peer, ...) used by the peer on the eBGP session. In practice, this identifier would usually correspond to the peer-group used in the BGP configuration to specify the export filter [21]. We propose to reserve value 0 for the eBGP session type corresponding to an eBGP session over which a full BGP routing table or a default route is advertised. Each egress router should thus be configured with the BGP policy used by its peer. To reduce the amount of manual configuration, the eBGP session type could be exchanged during the establishment of the BGP session by encoding this information inside the BGP capabilities. If required, BGP capabilities can also be updated during the lifetime of the BGP session. The SRLG values could be exchanged over the eBGP session by using the same technique.

Coming back to the example of figure 8, `R3` will advertise a protection route for an eBGP session of type 0 and `R2` a protection route for an eBGP session of type 1. `R1` will select the protection route of type 0 and `R3` will be the endpoint of the *pe-se* protection tunnel.

Finally, parallel links between ASes can have different bandwidth. When the endpoint of a protection tunnel is chosen, it should be possible to select as tunnel endpoint a secondary egress router with sufficient capacity. For this, the protection route can optionally contain the bandwidth extended community defined in [30]. Table I summarises the content of protection routes.

When the *primary egress* router needs to select a *pe-se* tunnel endpoint to protect a primary link, it will consider all protection routes whose tunnel endpoint is reachable according to its IGP routing table. The selection of the best protection route among those candidates will be done as follows.

1) Remove from consideration the protection routes that contain one of the SRLG values associated to the primary link to be protected.
2) Build a set $\mathcal{P}$ containing all the protection routes with the same upstream AS and the same eBGP session type as the primary eBGP session. If the set $\mathcal{P}$ is not empty, go to the last step, otherwise :
3) Build a set $\mathcal{P}$ containing all the protection routes with the same upstream AS as the primary eBGP session and an eBGP session type equal to 0.
4) If there are several candidate protection routes inside $\mathcal{P}$, break the ties by using the IGP cost to reach the tunnel endpoint and, if available, the link bandwidth extended community.

The first step is used to remove the protection routes that use the same SRLG as the primary link. The second step is used to protect a primary link with another link with the same eBGP session type with the same AS. Finally, the third step allows to protect a primary eBGP session with a restrictive BGP policy by using a protection route towards an eBGP session with the same AS but over which a default route is advertised.

In order to guarantee that the packets pushed on a protection tunnel are correctly forwarded to their destination, we must ensure some routing properties between the BGP speakers that are the end-points of the protection tunnel.

Let us consider again the left part of figure 3 assume that link $R1 \rightarrow X1$ between `AS1` and `AS2` is protected by using a *pe-se* tunnel $R1 \rightarrow R2$. In this case, $R2$ will send packets towards router $X2$ inside `AS2` when the tunnel is activated. To avoid transient loops, we must ensure that the BGP policy used by router $X2$ will not cause $X2$ to send the packets back to $R2$.

Loops will be avoided if `AS2` uses the classical BGP routing policies (customer-provider and shared-cost peering) described in [23] provided that all routers of `AS2` advertise[2] the same prefixes over all eBGP sessions of the same type. This is a common requirement for BGP peering agreements [22]. It must also be ensured that for each prefix p, the path advertised by $X2$ to $R2$ does not contain `AS1` otherwise this

[1]The Traffic Engineering extensions to OSPF and IS-IS already encode SRLG values as 32 bits integers.

[2]It should be noted that the attributes of the paths such as the AS-Path length or the Multi-Exit Discriminator do not have to be equal.

would prevent $R2$ from using it [3]

Let us first consider that AS2 is a provider of $AS1$. In this case, as $R1$ was using $R1 \rightarrow X1$ to reach prefix p before the utilization of the protection tunnel, this implies that AS1 did not learn a more preferred (i.e. learned from a customer) path to reach p. As AS1 does not learn a customer path to reach p, router $R2$ does not advertise this prefix to $X2$. Thus, router $X2$ will not return to router $R2$ the packets towards p.

Let us now consider that AS2 is a customer of AS1. In this case, as $R1$ was using $R1 \rightarrow X1$ to reach prefix p, this implies that either p belongs to AS2 or to a customer of AS2. In this case, the routers of AS2, including $X2$ will never select a path learned from their provider to reach p as there is a customer path.

If AS1 and AS2 are two shared cost peers, then if $R1$ was using $R1 \rightarrow X1$ to reach p, then p must belong to AS2 or one of its customers. Since router $R1$ selected the path to p advertised by $X1$, this implies that AS1 did not learn any customer path to reach p. Thus, router $R2$ cannot have advertised a path towards p to router $X2$.

VI. The *stub* problems

To solve the stub problems, we have to consider the two directions of the packet flow. For the *outgoing stub* problem, we note that in this case the stub receives either a default route or full BGP routing tables from its providers. Thus, the same destinations are reachable over all links with the providers. For this reason, we propose the utilisation of a *pe-se* protection tunnel to solve the *outgoing stub* problem. For the *incoming stub* problem, we will utilise a *pe-si* protection tunnel.

A. The outgoing stub *problem*

To protect the *stub→provider* packet flow on an interdomain link, we note that from the stub's viewpoint, the providers can be considered as equivalent as they can be used to reach any destination. Thus, the *outgoing stub* problem is similar to the *parallel links* problem. We slightly change the criteria to select the best *secondary egress* router for the protection tunnel and add rule 3b. This rule allows a stub to protect a link with a default route or full routing table with another similar link to another provider. The selection of the best protection route among those candidates is done as follows :

1 Remove from consideration the protection routes that contain one of the SRLG values associated to the primary link to be protected.
2 Build a set $\mathcal{P}$ containing all the protection routes with the same upstream AS and the same eBGP session type as the primary eBGP session. If the set $\mathcal{P}$ is not empty, go to the last step, otherwise :
3a Build a set $\mathcal{P}$ containing all the protection routes with the same upstream AS and an eBGP session type equal to 0. If the set $\mathcal{P}$ is not empty, go to the last step, otherwise :
3b Build a set $\mathcal{P}$ containing all the protection routes with an eBGP session type equal to 0.
4 If there are several candidate protection routes inside $\mathcal{P}$, break the ties by using the IGP cost to reach the tunnel endpoint and, if available, the link bandwidth extended community.

For example, consider in figure 9 that AS1 is a stub and that P1, P2 and P3 are its providers. Assume that P2 and P1 advertise a default route and P3 only regional routes. In this case, R2 will advertise inside AS1 two protection routes :

- a protection route with NLRI=2.0.2.2, AS Path=P2, and eBGP session type=0
- a protection route with NLRI=3.0.3.1, AS Path=P3, and eBGP session type=17

To protect link R1→RX, R1 would select IP address 2.0.2.2 as the endpoint of the protection tunnel.

B. The incoming stub *problem*

To quickly recover the *provider→stub* packet flow when an interdomain link to a stub fails, we propose to utilise a *pe-si* protection tunnel. This tunnel is established between the *primary egress* router located inside one provider and a *secondary ingress* router inside the stub. The advantage of using a *pe-si* tunnel in this case is that the routers of the secondary provider are not involved in the activation of the protection tunnel or the de-encapsulation of the packets.

As for the *pe-se* protection tunnel, the best *secondary ingress* router and the parameters of the protection tunnel to be used can be manually configured on the *primary egress* router. This manual configuration is probably acceptable for a small dual-homed stub AS, but it increases the complexity of the configuration that must be maintained by the provider. A better solution is to use BGP to auto-configure the required *pe-si* protection tunnels.

For this, we propose to allow each ingress router in the stub AS to advertise over the eBGP session with its provider the *secondary ingress* routers inside the stub that could be used as candidate endpoints for *pe-si* protection tunnels. This information can be advertised by the *primary ingress* router as *protection routes*[4]. In those protection routes, the NLRI is set to the IP address of the *secondary ingress* router and the tunnel attribute contains the supported tunnel type and the associated tunnel parameters.

A key issue for the utilisation of a *pe-si* protection tunnel is that the *primary egress* router must still be able to reach the *secondary ingress* router even if it was using the failed link to the *primary ingress* router to reach all the IP prefixes advertised by the stub. This reachability can be guaranteed provided that the IP address of the *secondary ingress* router belongs to an IP prefix allocated to and advertised by the secondary provider and not to an IP prefix advertised by the stub. This is a common practice among ISPs and could become

[3] It should be noted that very particular BGP policies have to be used on both ASes in order not to verify this property. Indeed, when this property is not verified for a prefix p, the routing state before the failure was such that some routers of AS1 were using $R1 \rightarrow X1$ to reach p via AS2, and some routers of AS2 were using another link between AS2 and AS1 to reach p via AS1.

[4] The NO_ADVERTISE BGP community is attached to the protection routes advertised over eBGP sessions as they do not need to be distributed beyond the *primary egress* router.

a design rule when *pe-si* tunnels are required. For example, in figure 9, router RX learns prefix 11.0.0.0/8 from router R1. If link RX → R1 fails, router RX can still reach the *secondary egress*, R2, by sending encapsulated packets to IP addresses 2.0.2.2 or 3.0.0.3.1.

The *protection routes* that are advertised by the *primary ingress* router can be manually configured, but a better solution is to use the protection routes that are distributed inside the stub to solve the *outgoing stub* problem.

For this, each ingress router of the stub AS will filter the *protection routes* that it receives via iBGP. The ingress router will only advertise over its eBGP session the protection routes containing the same eBGP session type as the session type of the primary link and different SRLG values than the SRLG values associated with the primary link.

The *primary egress* router will select, among the protection routes that it receives over its eBGP session, the best endpoint for the *pe-si* protection tunnel.

For example, consider the stub AS1 attached to providers P1, P2 and P3 in figure 9. Assume now that the three providers advertise a default route to AS1. R1 will receive via iBGP two protection routes from router R2 :

- a protection route with NLRI=2.0.2.2, AS Path=P2, and eBGP session type=0
- a protection route with NLRI=3.0.3.1, AS Path=P3, and eBGP session type=0

On its eBGP session with RX, R1 will advertise these two *protection routes* with 3.0.3.1 and 2.0.2.2 as tunnel endpoints. Based on the received candidate protection routes, RX will select 2.0.2.2 as the tunnel endpoint to protect the RX→R1 link.

VII. BGP CONVERGENCE AFTER DEACTIVATION OF A PROTECTION TUNNEL

Once activated, a protection tunnel can be used to forward the packets that were using the failed link over an alternate path. However, when the protection tunnel is used, the packet flow inside the network is not optimal anymore. If the failure lasts for a few seconds, this is not a problem, but using a protection tunnel for several minutes or hours could create congestion inside the network. The measurements discussed in section II have shown that most of the failures of eBGP peering links are short.

When a *primary egress* router detects the failure of a protected link, it should immediately activate the protection tunnel. Given the short duration of most failures, it should wait some time[5] before advertising the failure of its peering link via BGP or its IGP. If the failure is short enough, the peering link will come back while the protection tunnel is still active. At that time, the primary egress router simply needs to modify its FIB to deactivate the protection tunnel. Otherwise, the advertisement of the failure will trigger the exchange of iBGP messages and the update of the FIBs of many routers. To meet the requirements expressed in section III, we must ensure that no packet will be lost during this BGP convergence. We show in this section that this is possible with *pe-se* tunnels for BGP/MPLS VPNs services and in ASes using encapsulation.

[5]The exact value of this timer should depend on the duration of the eBGP link failures in the considered network. An operator could measure the average duration of eBGP link failures inside his network to tune this timer. For example, in the network considered in [1], there were 9452 eBGP peering link failures during 3 months. With a waiting time of three minutes, 9216 of these failures would have been protected locally without being advertised inside the network.

A. Deactivation of a pe-se tunnel

To illustrate the potential problems caused by the iBGP convergence, let us consider the network topology shown in figure 10 and focus on the packets sent to destination *D*. In this topology, R1−X1 is the primary link between AS1 and AS2 and R3−X3 a backup link. This backup link is implemented by configuring a low local−pref attribute in the import filter of router R3. When link R1−X1 fails, the *pe-se* tunnel reroutes the packet via link R3−X3. However, the utilisation of this tunnel is not optimal since the packets that enter AS1 at router R2 will pass twice through the R1−R2 link. After some time, router R1 will need to remove the *pe-se* protection tunnel. If router R1 sends a BGP withdraw message (W_{R1}) to indicate that destination *D* is not reachable anymore, router R3 will react to this message by updating its FIB and sending a BGP update indicating its own route (U_{R3}). Depending on the processing order of those messages by the routers, several transient losses of connectivity to destination D are possible. In table II, we show the eight possible orderings of the exchanges of BGP messages inside AS1. We use the notation $Rx : W_{R1}$ (resp. $Ry : U_{R3}$) to indicate that message W_{R1} (resp. U_{R3}) has been processed by router Rx (resp. Ry). As shown by this table, only one ordering of the updates of the FIBs ensures the reachability of D during the convergence. For five of the possible orderings, D becomes unreachable during a short period of time and a transient loop between $R1$ and $R2$ appears for two of the possible orderings.

Thus, two different problems must be solved to allow a *pe* router to remove a *pe-se* protection tunnel without causing packets losses :

- All the destinations that are currently reached via the protection tunnel must remain reachable during the entire routing convergence (*the convergence preserves reachability*)
- No transient packet forwarding loops are caused by the update of the FIBs of the routers inside the AS (*the convergence does not cause transient loops*)

To preserve reachability and avoid transient loops, we need to consider how packets are forwarded inside an autonomous system. This problem was discussed early during the development of BGP [27] and two techniques have emerged. The first solution, proposed in 1990, is to use encapsulation [31], i.e. the ingress border router encapsulates the interdomain packets inside a tunnel towards the egress border router chosen by its BGP decision process. In the early nineties, the existing routers were not capable of performing encapsulation at line rate [27]. Today, high-end routers are capable of performing encapsulation or de-encapsulation at line rate when using MPLS or IP-based tunnels [28]. We discuss the routing convergence

in BGP/MPLS VPNs in section VII-A.1 and in autonomous systems using encapsulation in section VII-A.2.

The second technique, called *Pervasive BGP* by [32] is to use BGP on all (border and non-border) routers inside the transit autonomous system. This technique is commonly used in pure IP-based transit networks. We explain in section VII-A.3 the difficulty of avoiding transient forwarding loops inside autonomous systems with *Pervasive BGP*.

1) BGP/MPLS VPN: In a network providing BGP/MPLS VPNs (right part of figure 3), iBGP is used to distribute the VPN routes to the PE routers [2]. A VPN route is composed of two parts : a *Route Distinguisher (RD)* and an IP prefix. The *RD* is used to allow sites belonging to different customers to use the same IP addresses (e.g. RFC1918 private addresses). A VPN route is considered as an opaque bit string by the BGP routers that distribute the routes. A service provider can either use the same *RD* for all VPN routes belonging to the same VPN or a different *RD* for each PE-CE link. Furthermore, a *route target (RT)* is associated to each VPN route. A *RT* is encoded as a BGP extended community. It is used, in combination with filters on the PE routers, to ensure that a VPN route from a given customer is only distributed to the PE routers that are attached to CE routers belonging to the same VPN. This utilisation of the *RT* reduces the size of the VPN routing tables on the PE routers [2].

To avoid packet losses during the BGP convergence in this environment, the service provider simply needs to configure its PE routers to use a different *RD* for each PE−CE link. Using a different *RD* ensures that each PE router will receive via iBGP all the VPN routes for the prefixes that are reachable over the PE−CE links. This remains true even if the service provider network is divided in confederations or uses BGP route reflectors as VPN routes with different *RD* are considered as different *opaque* prefixes by the BGP decision process. When a PE router sends BGP withdraw messages due to the failure of a parallel-link, those messages will reach distant PE routers where an alternate VPN route (with a different *RD*) is already available. As this alternate route uses an MPLS tunnel, it is loop-free. The same reasoning applies if the service provider uses IP tunnels instead of MPLS tunnels.

For example, consider in figure 3 the failure of link PE1−CE1. PE1 first activates the *pe-se* protection tunnel to reach CE2 via PE2. At that time, PE3 uses an MPLS tunnel to send via PE1 the VPN packets from CE3 to CE1. Then, PE1 sends a BGP withdraw message. When this message reaches PE3, it updates its VPN routing table and uses the loop-free MPLS tunnel to PE2 to reach CE2 and CE1.

2) AS using encapsulation: In an AS using encapsulation, to ensure that the *protected* destinations remain reachable during the iBGP convergence, we propose to allow the *primary egress* router to send a special BGP message to indicate that the destinations that are reached via the *pe-se* tunnel will soon become unreachable. For this special iBGP advertisement, we propose to reserve a low local−pref value, e.g. 0, to indicate a route that will be removed later. A route with a local−pref attribute set to 0 is considered as the worst route by the standard BGP decision process. Thus, a router will only use this route if this is the only route that it knows for this prefix.

The transmission of this iBGP message will cause an iBGP convergence. This iBGP convergence will not render the prefix advertised in the iBGP message unreachable as all routers will always have at least this route in their Adj-RIB-In.

In an AS using encapsulation, this iBGP convergence will not cause loops provided that, first, the AS is stable and loop-free[6] and, second, no eBGP messages concerning the *protected* destinations are received during the iBGP convergence[7]. The assumption that no eBGP messages are received is reasonable for two reasons. First, measurement studies have shown that, although a lot of BGP messages are exchanged in the global Internet the BGP routes over which a large amount of traffic is sent are stable [34], [35]. Second, the failure of a link will not force routers in other ASes to send relevant BGP messages as the routers that are not affected by the failure will keep their previously selected routes while the routers that are affected by the failure may only send new routes to the considered AS.

To explain the absence of transient loops, we have to consider the types of encapsulation. As explained in section IV-B, it is possible to avoid transient loops with encapsulation schemes such as MPLS or L2TP where only the ingress border router consults its BGP routing table to forward a transit packet. Those encapsulation schemes rely on two "levels" of encapsulation. With MPLS, the top label is used to reach the egress border router and the bottom label indicates the interdomain link to be used to reach the nexthop [36]. With L2TP, or MPLS over IP, the first encapsulation level is the added IP header whose destination is set to the IP address of the egress border router and the second encapsulation level is the label that indicates the interdomain link. With those encapsulation schemes, only the ingress border router consults its BGP routing table to forward a received interdomain packet. All the other routers inside the AS will rely on their IGP routing tables or their label forwarding table to forward the packet.

When a router receives an iBGP message, it may modify its FIB and select a new nexthop (and thus a new tunnel) to reach the destination. The first level of encapsulation cannot cause a loop due to the arrival of the iBGP message as it only depends on the intradomain routing tables that are assumed to be stable and loop-free. The second level of encapsulation will neither cause a transient loop inside the AS as the received label simply indicates the interdomain link over which the de-encapsulated packet should be forwarded.

As there is at least an alternate path to reach the destination via the *secondary egress* router, at least one alternate path will be eventually advertised and all the BGP routers inside the network will update their FIB and stop using the *pe-se* protection tunnel. The *primary egress* router will only send the BGP withdraw message once no packets are using the *pe-se* protection tunnel.

For example, consider the network topology shown in

[6]Those conditions imply that either no intradomain link changes occur or that the updates of the intradomain routing tables are ordered as proposed in [33] to avoid intradomain loops.

[7]Otherwise, an eBGP convergence is taking place and transient forwarding loops between ASes are possible during this eBGP convergence.

figure 11 where $R1 - X1$ is the primary link protected by a *pe-se* protection tunnel between $R1$ and $R3$. If a full iBGP mesh is used inside AS1 and $X1$ and $X3$ advertise routes with a MED set to the IGP cost to the nexthop, then $R1$ does not receive any alternate route towards destination D from $R2$ or $R3$ as those routes have a longer AS-Path or a lower MED. To remove the *pe-se* tunnel, $R1$ first sends an iBGP update with local-pref=0. This message will cause an iBGP convergence inside AS1. Two interdomain paths will be advertised inside AS1 : the first by $R2$ with an AS-Path of AS4:AS3:ASD and the second by $R3$ with an AS-Path of AS2:ASD and a MED value of 2. The path advertised by $R3$ will be selected as the best by all routers inside AS1. Once all AS1 routers have updated their FIB, router $R1$ will stop receiving packets towards D. At that time, router $R1$ can safely update its FIB, send an iBGP withdraw for destination D and remove the *pe-se* protection tunnel as no router inside AS1 is using it.

3) AS using Pervasive BGP: In autonomous systems using pervasive BGP, the solution described above is unfortunately not applicable. The main problem in such a network is that each iBGP message that causes a change in the FIB of one router may cause a transient forwarding loop. Such forwarding loops have been detected in large ISP networks [37].

To illustrate the problem, let us consider again the deactivation of a *pe-se* protection tunnel in the topology shown in figure 10. If AS1 is using pervasive BGP and we modify the primary egress router to send an iBGP update with the local-pref attribute set to 0 to deactivate the *pe-se* tunnel, then destination D always remains reachable. However, during the iBGP convergence, the ordering of the updates of the FIBs is important. In table III, we summarise what happens during the eight possible orderings of the FIB updates. In this table, $Rx : U^0_{Ry}$ indicates that router Rx has updated its FIB after the arrival of the iBGP messages with local-pref set to 0. Out of the eight possible orderings, only three are always loop-free.

Avoiding transient loops in autonomous systems using pervasive BGP is a difficult problem. Autonomous systems willing to use *pe-se* protection tunnels to protect their interdomain links should consider the utilisation of encapsulation techniques (e.g. MPLS or L2TP) between all of their border routers. In addition to avoiding BGP-induced transient forwarding loops, encapsulation allows border routers to have more flexibility in the selection of the BGP nexthop that they use to reach each external destination.

B. Deactivation of a pe-si protection tunnel

When a *pe-si* protection tunnel has been activated, the router that is using the tunnel may wish to remove it if the failure lasts too long. Ideally, the removal of this tunnel should not cause packet losses or transient loops. Unfortunately, removing a *pe-si* protection tunnel could cause a complete BGP convergence on the Internet for all the prefixes learned over the failed interdomain link. This BGP convergence may potentially affect all BGP routers in all ASes and cause packet losses or transient loops.

To reduce the amount of lost packets, the *primary egress* router should not immediately send a BGP withdraw messages for the routes learned over the failed links. Without changing the currently deployed BGP, the only possible solution is to send a new BGP update message with the local-pref attribute set to 0 and a prepended AS-Path attribute. Given the diameter of the Internet, prepending 7 times would be a reasonable choice[8]. In this BGP message, the setting of the local-pref attribute is used to force the selection of an alternate path in the upstream AS. If there is no alternate path, then the prepended AS-Path will be propagated to other ASes. A possible improvement to this scheme would be to define a new standard BGP community that requests each AS receiving the route to set its local-pref attribute to 0. Unfortunately, this implies that all routers in the Internet must be updated or re-configured to support this new community.

VIII. Related work

Several fast reroute techniques have been proposed and are deployed in MPLS networks. A survey of these techniques may be found in [4]. Several ISPs have started to deploy interdomain MPLS tunnels. Extensions to RSVP-TE to allow those tunnels to be protected on the interdomain links have been proposed recently [38]. The main advantage of our solution is that it allows to quickly recover from the failure of PE-CE links in BGP/MPLS VPNs although no MPLS tunnel is used on those links.

In pure IP networks, fast reroute techniques have been recently proposed to recover from the failure of intradomain links [12]. These techniques assume that the routers are using a link-state intradomain routing protocol and know the entire network topology. To our knowledge, our solution is the first fast reroute technique that allows to protect interdomain links. In [39] an extension of the O2 routing protocol [40] was proposed to recover from the failure of interdomain links. However, this solution assumes both a new routing protocol and that the *primary* and *secondary egress* routers are directly connected.

Gummadi et al. propose in [9] a source routing technique that allows endsystems to reroute packets around failures by using intermediate nodes as relays. Measurements with a prototype implementation reveal that this technique allows to recover from 56% of network failures. This end-to-end recovery technique is characterised by a recovery time of at least several seconds. Our fast-reroute mechanism only allows to recover from a failed BGP link, but those links are key in today's Internet. Our technique is also applicable for the BGP/MPLS VPNs that are increasingly used to replace frame relay and ATM-based networks.

Several modifications to BGP have been proposed to reduce the BGP convergence time. To our knowledge, the closest solution to our interdomain tunnels is the Fast Scoped Rerouting proposed for BGP in [41]. With this approach, BGP routers try to find an alternate path for each destination affected by

[8]It would also be possible to modify the BGP extension proposed by Pei et al. in [13] to support such graceful changes. However, this extension has not yet been implemented or deployed in the Internet.

a failure and exchange messages with the routers on this alternate path. As BGP messages must be exchanged *after* the failure to find an alternate path, the recovery time of this BGP extension will be longer than with our solution. The Root Cause Notification proposed in [13] adds to the BGP messages an information about the reason for the BGP message. Another method to tag BGP messages was proposed in [42]. Our solution is orthogonal to those BGP extensions and could benefit from them if implemented and deployed. Our solution allows the protection tunnel to be used immediately after the failure without requiring the exchange of any BGP message.

IX. Conclusion

BGP peering links are important in both the global Internet and in BGP/MPLS VPNs. We have shown that upon failure of a peering link current BGP routers are not able to react immediately to the failure.

In this paper, we have proposed a new technique to ensure that the packet flow on failed eBGP peering links can be recovered within 50 milliseconds. Our solution relies on two types of *protection tunnels*. Its main advantages are that it can be incrementally deployed, does not require major changes to the BGP protocol and is applicable for both normal BGP peering links and for the links to customer sites in BGP/MPLS VPNs.

The *primary egress-secondary egress* protection tunnels can be used when there are several parallel links between two ASes. We have proposed simple BGP extensions that allow border routers to automatically discover the best *pe-se* protection tunnel to use to protect each of their interdomain links. In autonomous systems using encapsulation and in networks providing BGP/MPLS VPN service, our solution also avoids packet losses during BGP convergence that follows the deactivation of the protection tunnel.

The *primary egress-secondary ingress* protection tunnels can be used to protect the interdomain links that attach providers to a multi-homed stub AS. We have proposed a simple extension to BGP that allows the routers of the stub AS to automatically indicate to their provider's routers the best *pe-si* tunnels to use.

Acknowledgements

We would like to thank Bruno Quoitin and the anonymous reviewers for their suggestions and constructive comments. We would also like to thank Peter De Vriendt and Kris Michielsen who performed the measurements presented in section II-A.

References

[1] O. Bonaventure, C. Filsfils, and P. Francois, "Achieving sub-50 milliseconds recovery upon BGP peering link failures," in *Co-Next 2005*, Toulouse, France, October 2005.

[2] E. Rosen and Y. Rekhter, "BGP/MPLS VPNs," March 1999, RFC2547.

[3] S. Bryant and P. Pate, "Pseudo Wire Emulation Edge-to-Edge (PWE3) Architecture," March 2005, RFC3985.

[4] J.-P. Vasseur, M. Pickavet, and P. Demeester, *Network Recovery: Protection and Restoration of Optical, SONET-SDH, and MPLS*. Morgan Kaufmann, 2004.

[5] C. Filsfils and J. Evans, "Deploying Diffserv in IP/MPLS backbone networks for Tight SLA control," *IEEE Internet Computing*, vol. 9, no. 1, pp. 58–65, 2005.

[6] A. Markopoulou, G. Iannaccone, S. Bhattacharyya, C. Chuah, and C. Diot, "Characterization of failures in an IP backbone," in *IEEE Infocom2004*, Hong Kong, March 2004.

[7] D. Watson, F. Jahanian, and C. Labovitz, "Experiences with monitoring OSPF on a regional service provider network," in *Proceedings of the 23rd International Conference on Distributed Computing Systems*. IEEE Computer Society, 2003, p. 204.

[8] N. Feamster, D. Andersen, H. Balakrishnan, and M. Kaashoek, "Measuring the effects of Internet path faults on reactive routing," in *ACM SIGMETRICS*, San Diego, CA (USA), June 2003.

[9] K. Gummardi, H. Madhyastha, S. Gribble, H. Leby, and D. Wetherall, "Improving the reliability of Internet paths with One-hop Source Routing," in *USENIX OSDI'04*, 2004.

[10] A. Feldmann, O. Maennel, M. Mao, A. Berger, and B. Maggs, "Locating internet routing instabilities," in *ACM SIGCOMM2004*, August 2004.

[11] P. Francois, C. Filsfils, J. Evans, and O. Bonaventure, "Achieving subsecond IGP convergence in large IP networks," *SIGCOMM Comput. Commun. Rev.*, vol. 35, no. 3, pp. 35–44, 2005.

[12] M. Shand and S. Bryant, "IP Fast Reroute Framework," June 2007, internet draft, draft-ietf-rtgwg-ipfrr-framework-07.txt, work in progress.

[13] D. Pei, M. Azuma, N. Nguyen, J. Chen, D. Massey, and L. Zhang, "BGP-RCN: Improving BGP convergence through Root Cause Notification," *Computer Networks*, vol. 48, no. 2, pp. 175–194, June 2005, 2005.

[14] T. Griffin and B. Presmore, "An experimental analysis of BGP convergence time," in *ICNP 2001*. IEEE Computer Society, November 2001, pp. 53–61.

[15] D. Katz and D. Ward, "Bidirectional Forwarding Detection," March 2007, internet draft, draft-ietf-bfd-base-07.txt, work in progress.

[16] C. Filsfils, "IGP and BGP fast convergence," December 2004, networkers'2004, Cannes, France.

[17] G. Iannaccone, C.-N. Chuah, R. Mortier, S. Bhattacharyya, and C. Diot, "Analysis of link failures over an IP backbone," in *ACM SIGCOMM Internet Measurement Workshop*, Marseilles, France, Nov. 2002.

[18] F. Wang, L. Gao, J. Wang, and J. Qiu, "On understanding of transient interdomain routing failures," in *ICNP*, Boston, MA, USA, November 2005.

[19] D. Pei and J. V. der Merwe, "BGP convergence in Virtual Private Networks," in *Internet Measurement Conference*, Rio de Janeiro, BrazilB, October 2006.

[20] F. Wang, Z. Mao, J. Wang, L. Gao, and R. Bush, "A measurement study on the impact of routing events on end-to-end Internet path performance," in *ACM SIGCOMM*, Pisa, Italy, September 2006, pp. 375–387.

[21] R. White, D. McPherson, and S. Sangli, *Practical BGP*. Addison Wesley, 2004.

[22] N. Feamster, Z. Mao, and J. Rexford, "BorderGuard: Detecting Cold Potatoes from Peers," in *ACM Internet Measurement Conference*, Taormina, Italy, October 2004.

[23] L. Gao and J. Rexford, "Stable internet routing without global coordination," in *SIGMETRICS*, 2000.

[24] Cisco, "Prefix and Tunnel Independent FRR," November 2004, http://www.cisco.com/en/US/products/ps5763/prod_release_note09186a008033575a.html#wp98916.

[25] J. Lau, M. Townsley, and I. Goyret, "Layer two tunneling protocol - version 3 (L2TPv3)," December 2004, internet draft, draft-ietf-l2tpext-l2tp-base-15.txt, work in progress.

[26] T. Worster, Y. Rekhter, and E. Rosen, "Encapsulating MPLS in IP or Generic Routing Encapsulation (GRE)," March 2005, RFC 4023.

[27] Y. Rekhter, "Constructing intra-AS path segments for an inter-AS path," *SIGCOMM Comput. Commun. Rev.*, vol. 21, no. 1, pp. 44–57, 1991.

[28] S. Gross, "Modern L2 VPNs : Implementing network convergence," Feb 2005, presentation at NANOG33.

[29] B. Greene and P. Smith, *Cisco ISP Essentials*. Cisco Press, 2002.

[30] S. Sangli, D. Tappan, and Y. Rekhter, "BGP extended communities attribute," February 2006, RFC4320.

[31] J. C. Honig, D. Katz, M. Mathis, Y. Rekhter, and J. Y. Yu, "Application of the Border Gateway Protocol in the Internet," Request for Comments 1164, June 1990.

[32] Y. Rekhter and P. Gross, "Application of the Border Gateway Protocol in the Internet," RFC1655, July 1994.

[33] P. Francois and O. Bonaventure, "Avoiding transient loops during IGP convergence in IP networks," in *IEEE INFOCOM'2005*, Miami, Florida, USA, March 2005.

[34] J. Rexford, J. Wang, Z. Xiao, and Y. Zhang, "BGP routing stability of popular destinations," in *Proc. Internet Measurement Workshop*, November 2002.
[35] S. Uhlig, V. Magnin, O. Bonaventure, C. Rapier, and L. Deri, "Implications of the topological properties of internet traffic on traffic engineering," in *ACM Symposium on Applied Computing*, March 2004.
[36] B. Davie and Y. Rekhter, *MPLS: technology and applications*. Morgan Kaufmann, 2000.
[37] U. Hengartner, S. Moon, R. Mortier, and C. Diot, "Detection and analysis of routing loops in packet traces," in *Proceedings of the second ACM SIGCOMM Workshop on Internet measurment*. ACM Press, 2002, pp. 107–112.
[38] S. D. Cnodder and C. Pelsser, "Protection for inter-AS MPLS tunnels," July 2004, work in progress, draft-decnodder-ccamp-interas-protection-00.txt.
[39] C. Reichert, "IP-protection for fast inter-domain resilience," May 2004, presented at IDRWS'04.
[40] G. Schollmeier, J. Charzinski, A. Kirstodter, C. Reichert, K. Schrodi, Y. Glickman, and C. Winkler, "Improving the resilience in IP networks," in *IEEE HPSR2003*, Torino, Italy, June 2003, pp. 91–96.
[41] R. Bless, G. Lichtwald, M. Schmidt, and M. Zitterbart, "Fast Scoped Rerouting for BGP," in *International Conference on Networks*. IEEE, September 2003, pp. 25–30.
[42] J. Chandrashekar, Z. Duan, Z. Zhang, and J. Krasky., "Limiting path exploration in BGP," in *IEEE INFOCOM*, Miami, Florida, March 2005.

Olivier Bonaventure (M'92/ACM'92) received his Ph.D. degree from the University of Liège, Belgium. He is currently a professor at Université catholique de Louvain (UCL), Belgium where he leads the IP Networking Lab (http://inl.info.ucl.ac.be). He received the Wernaers and Alcatel prizes awarded by the Belgian National Fund for Scientific Research in 2001. His current research interests include routing, traffic engineering, and the evolution of the Internet architecture.

Clarence Filsfils is a Distinguished Engineer at Cisco Systems. He has been playing a key role in defining, productizing, marketing and deploying the Quality of Service and Fast Routing Convergence technology at Cisco Systems. Clarence is a regular speaker at conferences. He published several journal articles, a book on QoS SP deployment and holds over 30 patents on QoS and Routing mechanisms.

Pierre Francois is currently finishing his Ph. D. at the Université catholique de Louvain (UCL), Belgium. He obtained his MS degree in computer science from the University of Namur, Belgium. He received the Infocom best paper award in 2007. His research interests include intra- and interdomain routing. (http://inl.info.ucl.ac.be/) e-mail: pierre.francois@uclouvain.be

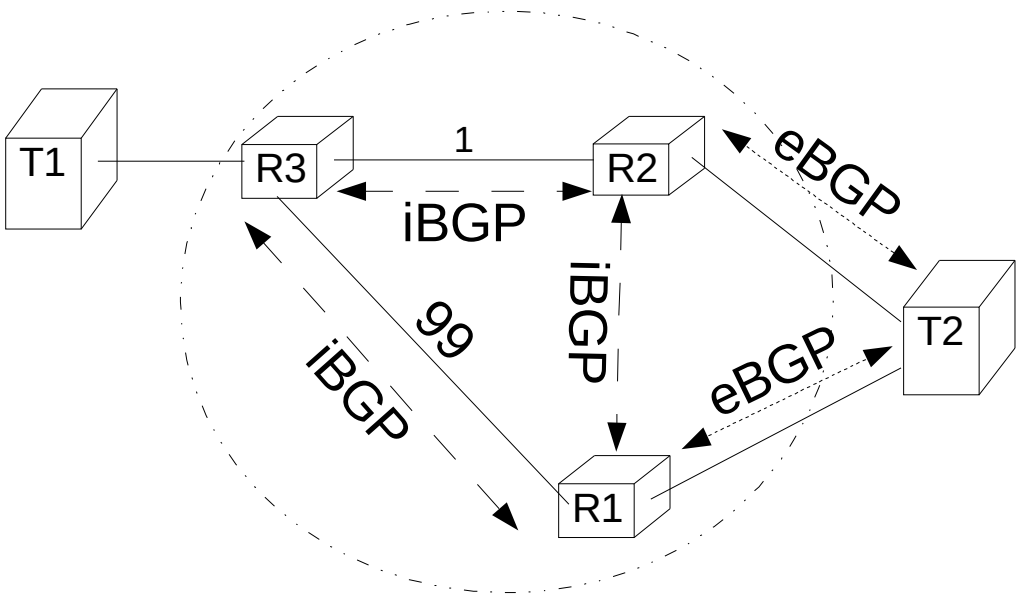

Fig. 1. Lab testbed

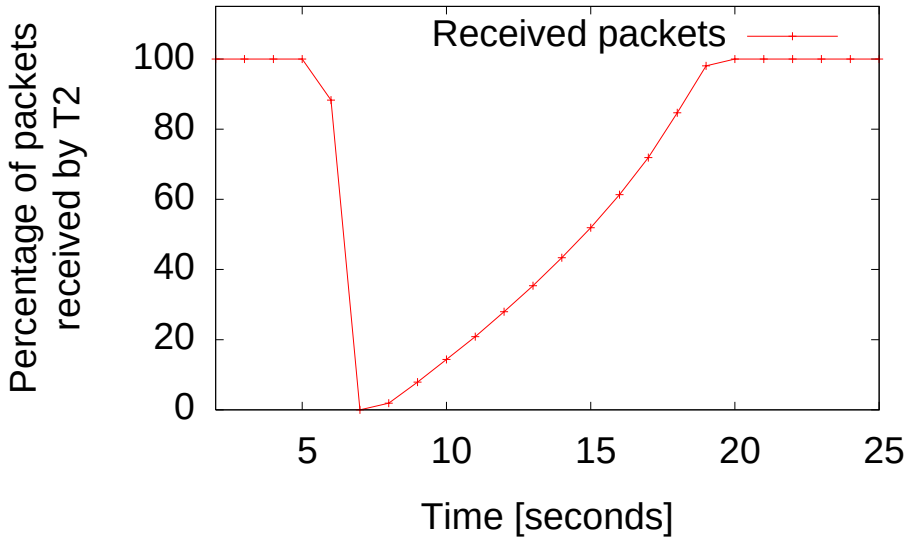

Fig. 2. Results of the blackbox measurements

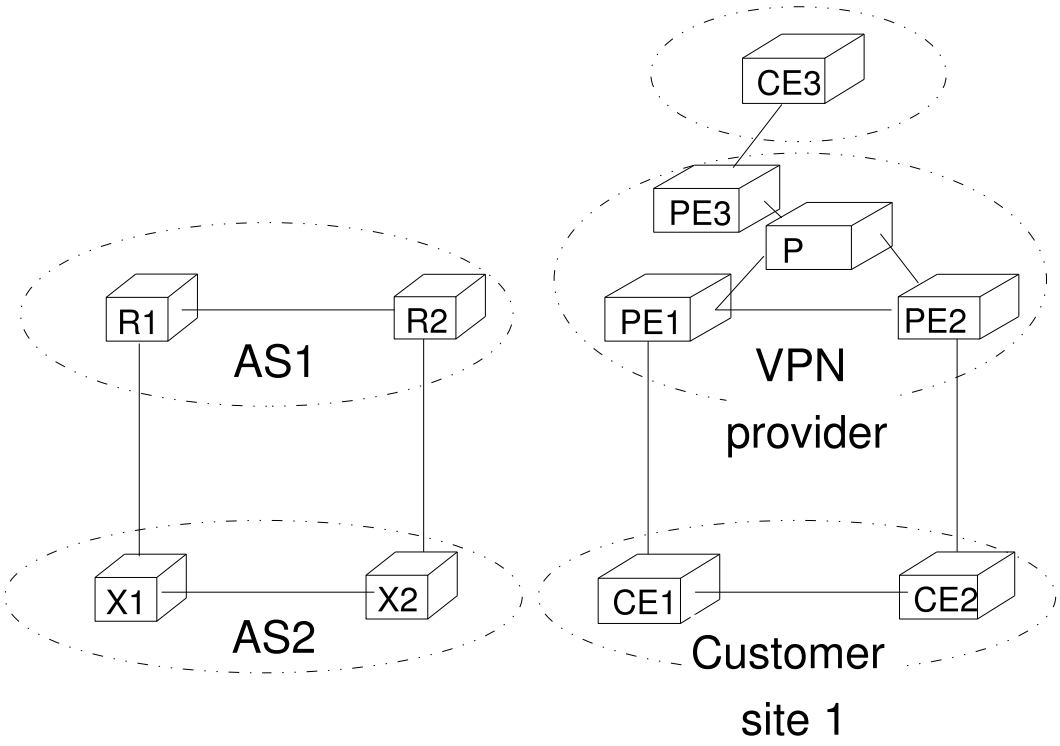

Fig. 3. The *parallel-links* problem for peering links and BGP/MPLS VPNs

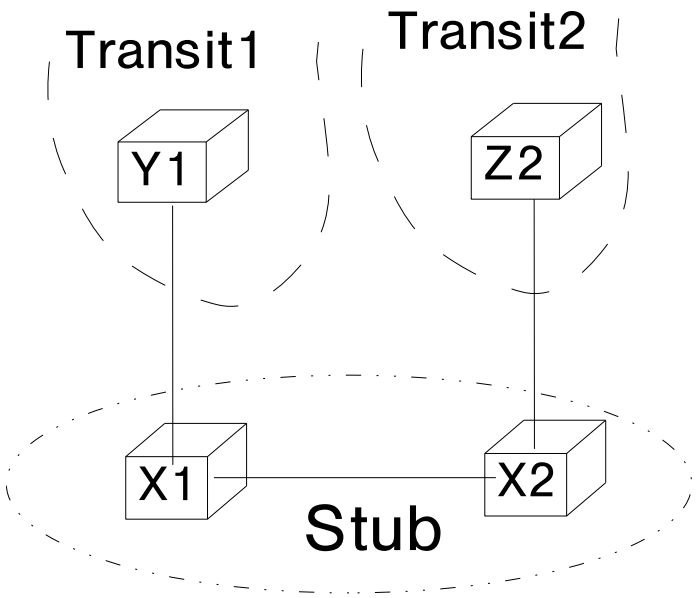

Fig. 4. The *stub* problem

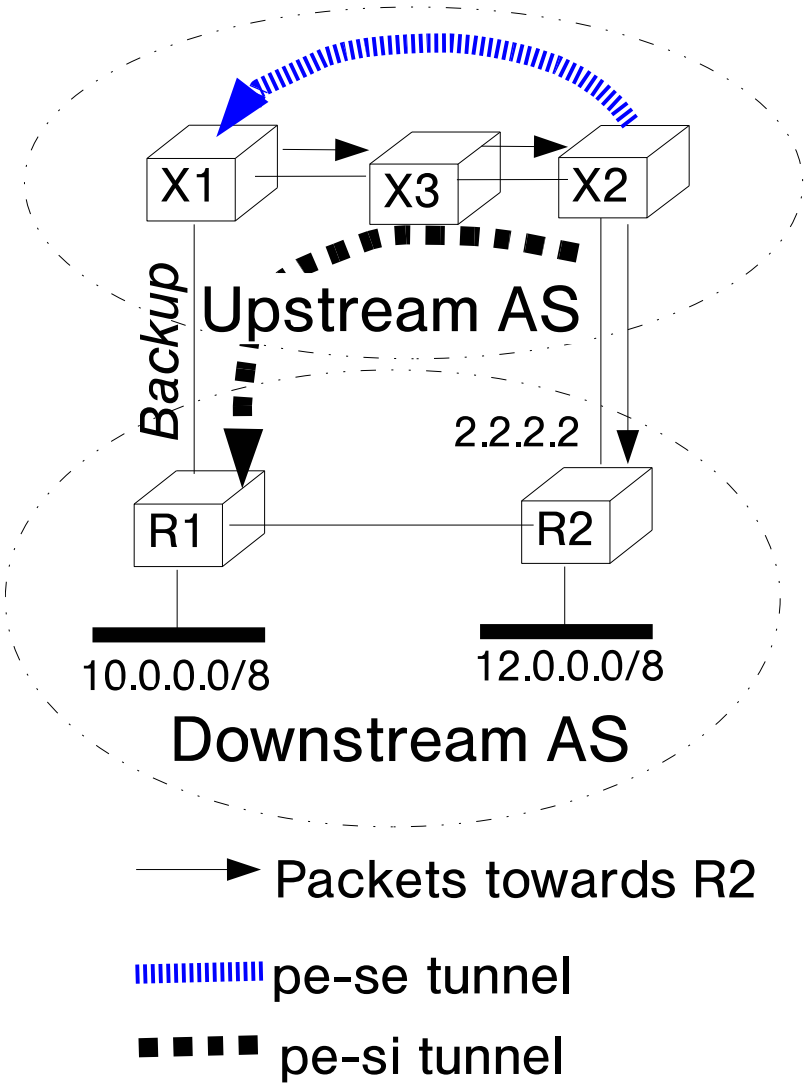

Fig. 5. Reference network

IGP Table

Prefix	Interface
2.2.2.2/32	South
3.3.3.3/32	West
...	

BGP Table

Prefix	Nexthop	Interface
10.0.0.0/8	2.2.2.2	South
11.0.0.0/8	3.3.3.3	West
12.0.0.0/8	2.2.2.2	South
...		

Fig. 6. Classical conceptual organisation of the FIB

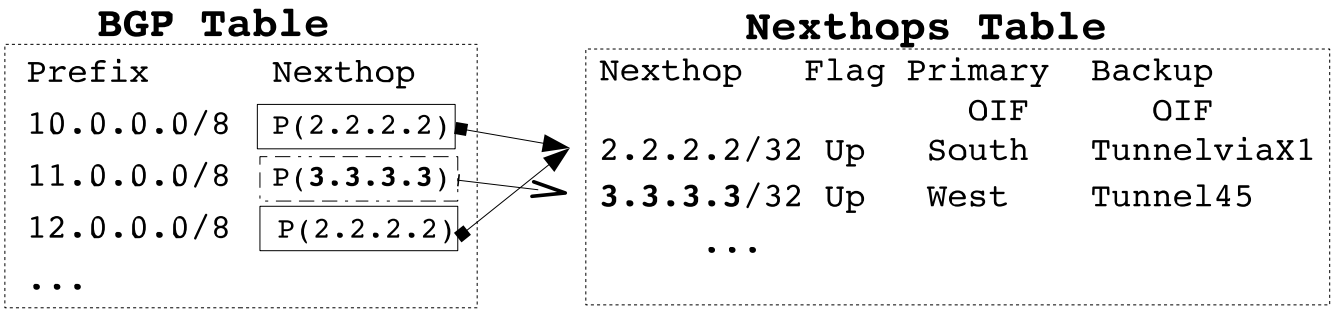

Fig. 7. Improved conceptual organisation of the FIB

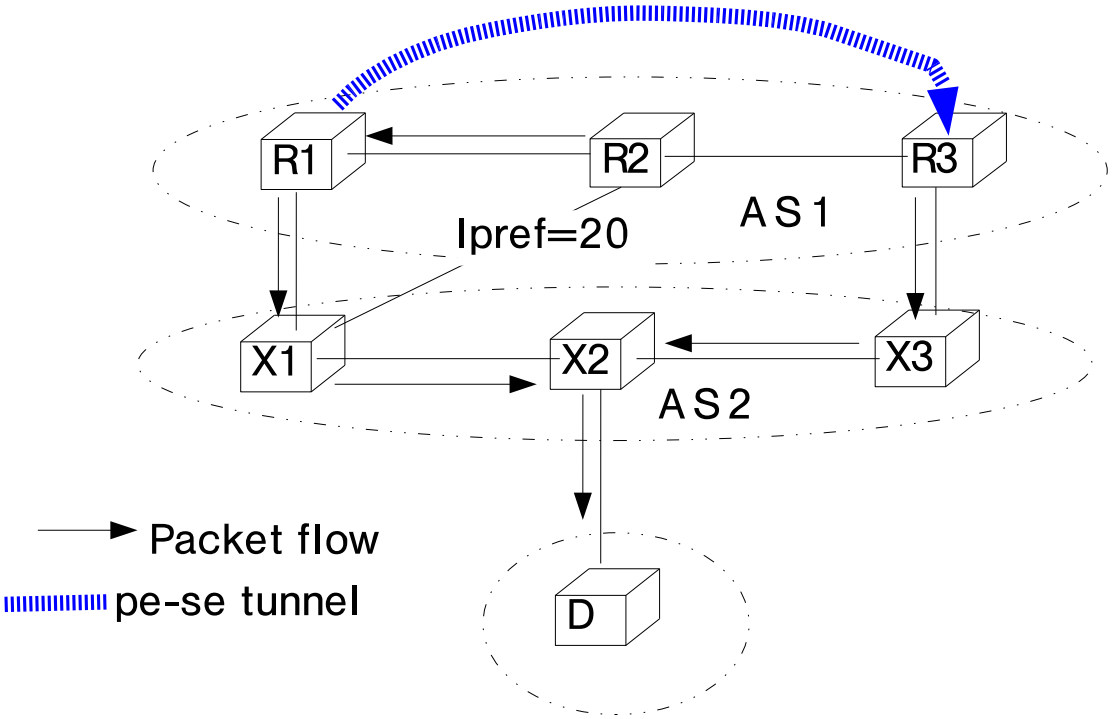

Fig. 8. Utilisation of a *pe-se* protection tunnel

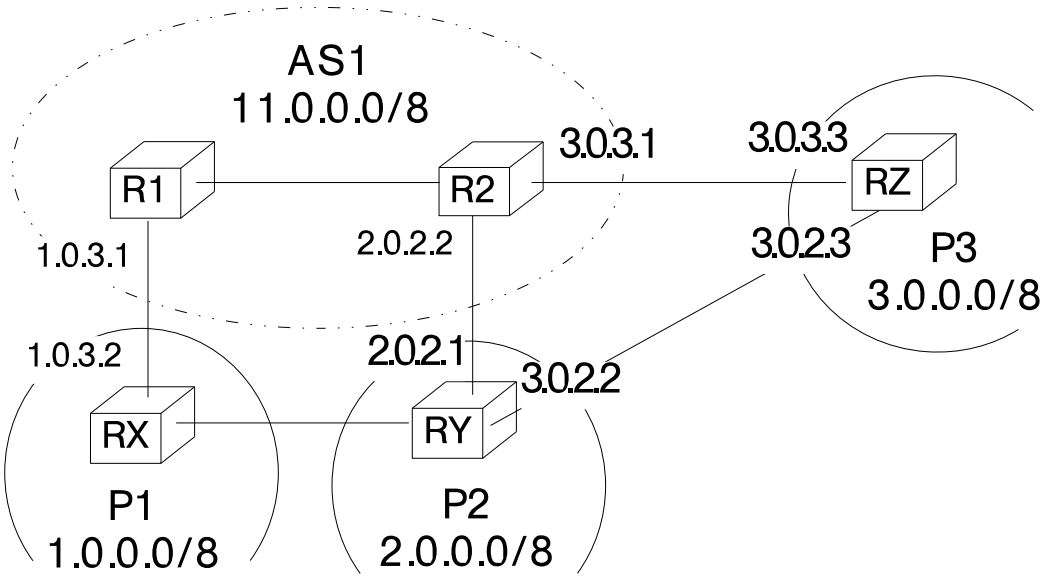

Fig. 9. A stub AS attached to three providers

Parameter	Comment
NLRI	IP address of egress router on peering link
AS-Path	Downstream AS
eBGP session type	32 bits unsigned integer
Tunnel attribute	Type and optional parameters for the tunnel
SRLG	optional list of pairs `AS#:SRLG-value`
Link bandwidth	optional extended community

TABLE I

PROPOSED PROTECTION ROUTES

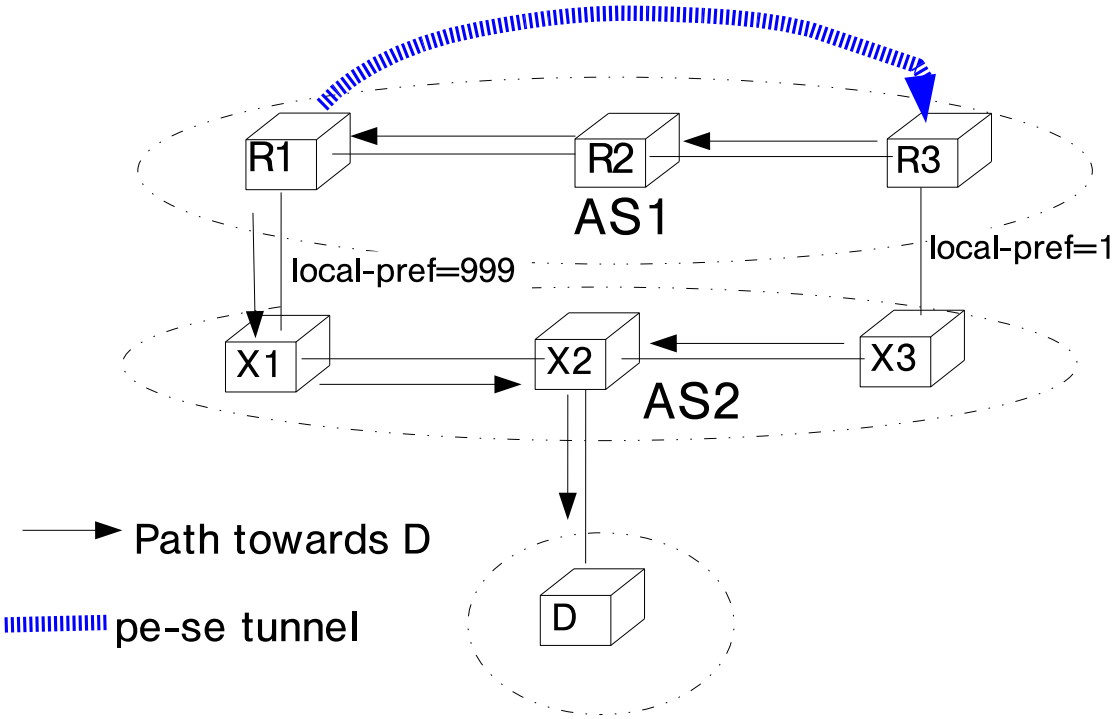

Fig. 10. Example topology for the deactivation of a *pe-se* tunnel

TABLE II

PROCESSING ORDER OF THE IBGP MESSAGES INSIDE AS1 AFTER THE TRANSMISSION OF A BGP WITHDRAW

First BGP message	Second BGP message	Third BGP message	Fourth BGP message	Comment
$R2 : W_{R1}$	$R3 : W_{R1}$	$R2 : U_{R3}$	$R1 : U_{R3}$	D unreachable from R2 between first and third message
$R2 : W_{R1}$	$R3 : W_{R1}$	$R1 : U_{R3}$	$R2 : U_{R3}$	D unreachable from R2 between first and fourth message
$R3 : W_{R1}$	$R2 : W_{R1}$	$R2 : U_{R3}$	$R1 : U_{R3}$	D unreachable from R2 between second and third message
$R3 : W_{R1}$	$R2 : W_{R1}$	$R1 : U_{R3}$	$R2 : U_{R3}$	D unreachable from R2 between second and fourth message
$R3 : W_{R1}$	$R2 : U_{R3}$	$R2 : W_{R1}$	$R1 : U_{R3}$	D always reachable during convergence
$R3 : W_{R1}$	$R2 : U_{R3}$	$R1 : U_{R3}$	$R2 : W_{R1}$	transient loop R1−R2 between third and fourth message
$R3 : W_{R1}$	$R1 : U_{R3}$	$R2 : W_{R1}$	$R2 : U_{R3}$	D unreachable from R2 between third and fourth message
$R3 : W_{R1}$	$R1 : U_{R3}$	$R2 : U_{R3}$	$R2 : W_{R1}$	transient loop R1−R2 between second and fourth message

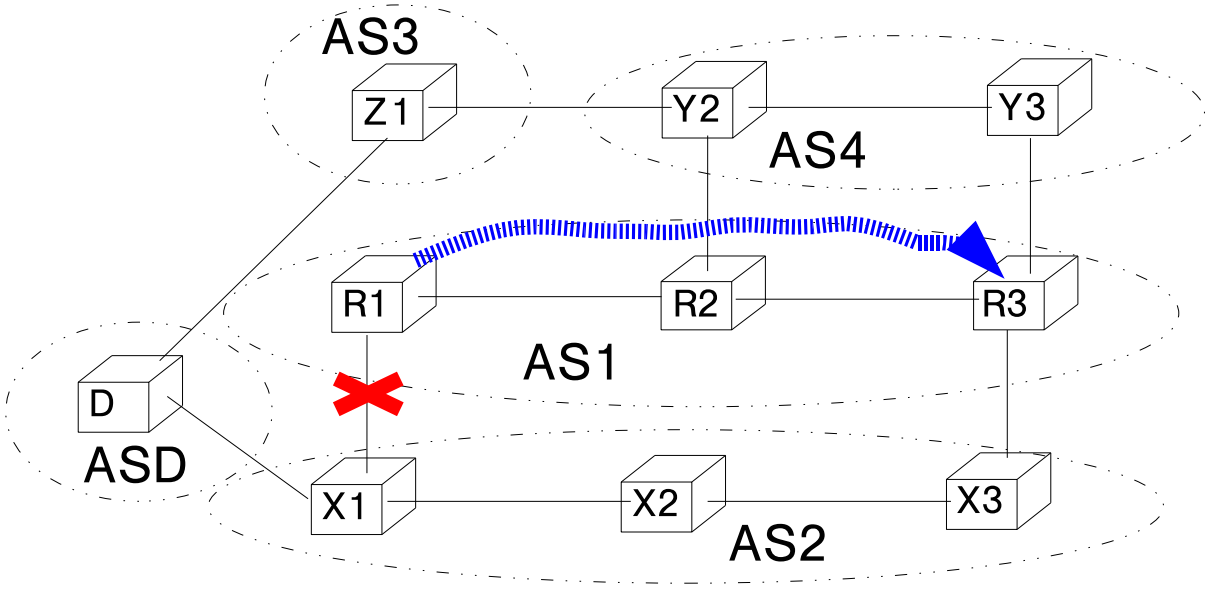

Fig. 11. BGP convergence with a *pe-se* protection tunnel

TABLE III

TRANSIENT LOOPS CAUSED BY THE UPDATES OF THE FIBS WITH PERVASIVE BGP

First BGP message	Second BGP message	Third BGP message	Fourth BGP message	Comment
$R2 : U^0_{R1}$	$R3 : U^0_{R1}$	$R2 : U_{R3}$	$R1 : U_{R3}$	D always reachable without loops during convergence
$R2 : U^0_{R1}$	$R3 : U^0_{R1}$	$R1 : U_{R3}$	$R2 : U_{R3}$	transient loop `R1-R2` between third and fourth message
$R3 : U^0_{R1}$	$R2 : U^0_{R1}$	$R2 : U_{R3}$	$R1 : U_{R3}$	D always reachable without loops during convergence
$R3 : U^0_{R1}$	$R2 : U^0_{R1}$	$R1 : U_{R3}$	$R2 : U_{R3}$	transient loop `R1-R2` between third and fourth message
$R3 : U^0_{R1}$	$R2 : U_{R3}$	$R2 : U^0_{R1}$	$R1 : U_{R3}$	D always reachable without loops during convergence
$R3 : U^0_{R1}$	$R2 : U_{R3}$	$R1 : U_{R3}$	$R2 : U^0_{R1}$	transient loop `R1-R2` between third and fourth message
$R3 : U^0_{R1}$	$R1 : U_{R3}$	$R2 : U^0_{R1}$	$R2 : U_{R3}$	transient loop `R1-R2` between second and fourth message
$R3 : U^0_{R1}$	$R1 : U_{R3}$	$R2 : U_{R3}$	$R2 : U^0_{R1}$	transient loop `R1-R2` between second and fourth message