

Network Working Group	M. Shand	
Internet-Draft	S. Bryant	
Intended status: Experimental	S. Previdi	
Expires: September 6, 2010	Cisco Systems	
	March 05, 2010	

[TOC](#)

IP Fast Reroute Using Not-via Addresses **draft-ietf-rtgwg-ipfrr-notvia-addresses-05**

Abstract

This draft describes a mechanism that provides fast reroute in an IP network through encapsulation to "not-via" addresses. A single level of encapsulation is used. The mechanism protects unicast, multicast and LDP traffic against link, router and shared risk group failure, regardless of network topology and metrics.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119 \(Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels," March 1997.\)](#) [RFC2119].

Status of this Memo

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

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

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

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/ietf/lid-abstracts.txt>.

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

This Internet-Draft will expire on September 6, 2010.

Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the BSD License.

Table of Contents

- [1.](#) Introduction
- [2.](#) Overview of Not-via Repairs
 - [2.1.](#) Use of Equal Cost Multi-Path
 - [2.2.](#) Use of LFA repairs
- [3.](#) Not-via Repair Path Computation
 - [3.1.](#) Computing not-via repairs in routing vector protocols
- [4.](#) Operation of Repairs
 - [4.1.](#) Node Failure
 - [4.2.](#) Link Failure
 - [4.2.1.](#) Loop Prevention Under Node Failure
 - [4.3.](#) Multi-homed Prefixes
 - [4.4.](#) Installation of Repair Paths
- [5.](#) Compound Failures
 - [5.1.](#) Shared Risk Link Groups
 - [5.1.1.](#) Use of LFAs with SRLGs
 - [5.2.](#) Local Area Networks
 - [5.2.1.](#) Simple LAN Repair
 - [5.2.2.](#) LAN Component Repair
 - [5.2.3.](#) LAN Repair Using Diagnostics
 - [5.3.](#) Multiple Independent Failures
 - [5.3.1.](#) Looping Repairs
 - [5.3.2.](#) Outline Solution
 - [5.3.3.](#) Looping Repairs
 - [5.3.3.1.](#) Dropping Looping Packets
 - [5.3.3.2.](#) Computing non-looping Repairs of Repairs
 - [5.3.3.3.](#) N-level Mutual Loops
 - [5.3.4.](#) Mixing LFAs and Not-via
- [6.](#) Optimizing not-via computations using LFAs
- [7.](#) Multicast
- [8.](#) Fast Reroute in an MPLS LDP Network.
- [9.](#) Encapsulation
- [10.](#) Routing Extensions

11.	Incremental Deployment
12.	IANA Considerations
13.	Security Considerations
14.	Acknowledgements
15.	References
15.1.	Normative References
15.2.	Informative References
§	Authors' Addresses

1. Introduction

[TOC](#)

When a link or a router fails, only the neighbors of the failure are initially aware that the failure has occurred. In a network operating IP fast reroute [\[RFC5714\] \(Shand, M. and S. Bryant, "IP Fast Reroute Framework," January 2010.\)](#), the routers that are the neighbors of the failure repair the failure. These repairing routers have to steer packets to their destinations despite the fact that most other routers in the network are unaware of the nature and location of the failure. A common limitation in most IPFRR mechanisms is an inability to indicate the identity of the failure and to explicitly steer the repaired packet round the failure. The extent to which this limitation affects the repair coverage is topology dependent. The mechanism proposed here is to encapsulate the packet to an address that explicitly identifies the network component that the repair must avoid. This produces a repair mechanism, which, provided the network is not partitioned by the failure, will always achieve a repair.

2. Overview of Not-via Repairs

[TOC](#)

When a link or a router fails, only the neighbors of the failure are initially aware that the failure has occurred. In a network operating IP fast reroute [\[RFC5714\] \(Shand, M. and S. Bryant, "IP Fast Reroute Framework," January 2010.\)](#), the routers that are the neighbors of the failure repair the failure. These repairing routers have to steer packets to their destinations despite the fact that most other routers in the network are unaware of the nature and location of the failure. A common limitation in most IPFRR mechanisms is an inability to indicate the identity of the failure and to explicitly steer the repaired packet round the failure. The extent to which this limitation affects the repair coverage is topology dependent. The mechanism proposed here is to encapsulate the packet to an address that explicitly identifies the network component that the repair must avoid.

This produces a repair mechanism, which, provided the network is not partitioned by the failure, will always achieve a repair.

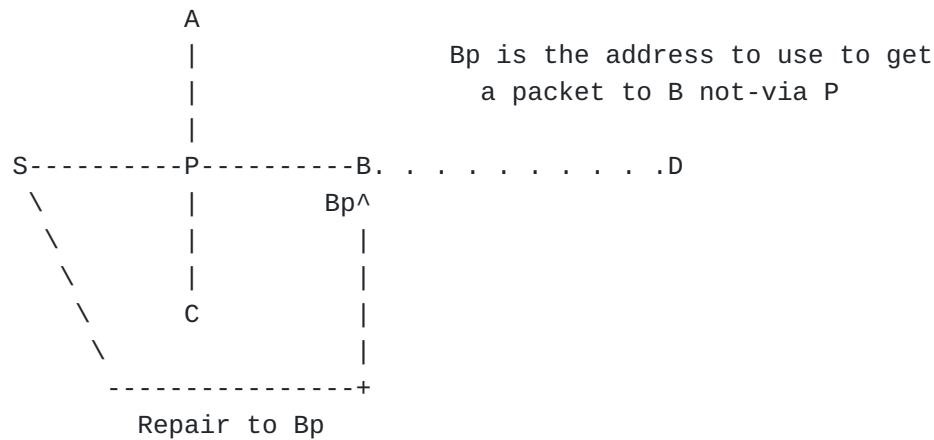


Figure 1: Not-via repair of router failure

Assume that S has a packet for some destination D that it would normally send via P and B, and that S suspects that P has failed. S encapsulates the packet to Bp. The path from S to Bp is the shortest path from S to B not going via P. If the network contains a path from S to B that does not transit router P, i.e. the network is not partitioned by the failure of P, then the packet will be successfully delivered to B. When the packet addressed to Bp arrives at B, B removes the encapsulation and forwards the repaired packet towards its final destination.

Note that if the path from B to the final destination includes one or more nodes that are included in the repair path, a packet may back track after the encapsulation is removed. However, because the decapsulating router is always closer to the packet destination than the encapsulating router, the packet will not loop.

For complete protection, all of P's neighbors will require a not-via address that allows traffic to be directed to them without traversing P. This is shown in [Figure 2 \(The set of Not-via P Addresses \)](#).

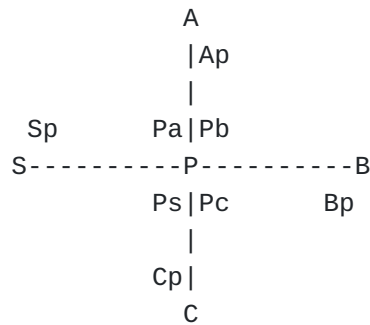


Figure 2: The set of Not-via P Addresses

2.1. Use of Equal Cost Multi-Path

[TOC](#)

A router can use an equal cost multi-path (ECMP) repair in place of a not-via repair.

A router computing a not-via repair path MAY subject the repair to ECMP.

2.2. Use of LFA repairs

[TOC](#)

The not-via approach provides complete repair coverage and therefore may be used as the sole repair mechanism. There are, however, advantages in using not-via in combination with loop free alternates (LFA) and or downstream paths as documented in [\[RFC5286\] \(Atlas, A. and A. Zinin, "Basic Specification for IP Fast Reroute: Loop-Free Alternates," September 2008.\)](#).

LFAs are computed on a per destination basis and in general, only a subset of the destinations requiring repair will have a suitable LFA repair. In this case, those destinations which are repairable by LFAs are so repaired and the remainder of the destinations are repaired using the not-via encapsulation. This has the advantage of reducing the volume of traffic that requires encapsulation. On the other hand, the path taken by an LFA repair may be less optimal than that of the equivalent not-via repair for traffic destined to nodes close to the far end of the failure, but may be more optimal for some other traffic. The description in this document assumes that LFAs will be used where available, but the distribution of repairs between the two mechanisms is a local implementation choice.

3. Not-via Repair Path Computation

[TOC](#)

The not-via repair mechanism requires that all routers on the path from S to B ([Figure 1 \(Not-via repair of router failure\)](#)) have a route to Bp. They can calculate this by failing node P, running an SPF, and finding the shortest route to B.

A router has no simple way of knowing whether it is on the shortest path for any particular repair. It is therefore necessary for every router to calculate the path it would use in the event of any possible router failure. Each router therefore "fails" every router in the network, one at a time, and calculates its own best route to each of the neighbors of that router. In other words, with reference to [Figure 1 \(Not-via repair of router failure\)](#), some router X will consider each router in turn to be P, fail P, and then calculate its own route to each of the not-via P addresses advertised by the neighbors of P. i.e. X calculates its route to Sp, Ap, Bp, and Cp, in each case, not via P.

To calculate the repair paths a router has to calculate n-1 SPFs where n is the number of routers in the network. This is expensive to compute. However, the problem is amenable to a solution in which each router (X) proceeds as follows. X first calculates the base topology with all routers functional and determines its normal path to all not-via addresses. This can be performed as part of the normal SPF computation. For each router P in the topology, X then performs the following actions:-

1. Removes router P from the topology.
2. Performs an incremental SPF [\[ISPF\] \(McQuillan, J., Richer, I., and E. Rosen, "ARPANET Routing Algorithm Improvements", 1978.\)](#) on the modified topology. The iSPF process involves detaching the sub-tree affected by the removal of router P, and then re-attaching the detached nodes. However, it is not necessary to run the iSPF to completion. It is sufficient to run the iSPF up to the point where all of the nodes advertising not-via P addresses have been re-attached to the SPT, and then terminate it.
3. Reverts to the base topology.

This algorithm is significantly less expensive than a set of full SPFs. Thus, although a router has to calculate the repair paths for n-1 failures, the computational effort is much less than n-1 SPFs. Experiments on a selection of real world network topologies with between 40 and 400 nodes suggest that the worst-case computational complexity using the above optimizations is equivalent to performing

between 5 and 13 full SPF's. Further optimizations are described in section 6.

3.1. Computing not-via repairs in routing vector protocols

[TOC](#)

While this document focuses on link state routing protocols, it is equally possible to compute not-via repairs in distance vector (e.g. RIP) or path vector (e.g. BGP) routing protocols. This can be achieved with very little protocol modification by advertising the not-via address in the normal way, but ensuring that the information about a not-via address P_s is not propagated through the node S . In the case of link protection this simply means that the advertisement from P to S is suppressed, with the result that S and all other nodes compute a route to P_s which doesn't traverse S , as required.

In the case of node protection, where P is the protected node, and N is some neighbor, the advertisement of N_p must be suppressed not only across the link $N \rightarrow P$, but also across any link to P . The simplest way of achieving this is for P itself to perform the suppression of any address of the form X_p .

4. Operation of Repairs

[TOC](#)

This section explains the basic operation of the not-via repair of node and link failure.

4.1. Node Failure

[TOC](#)

When router P fails ([Figure 2 \(The set of Not-via P Addresses \)](#)) S encapsulates any packet that it would send to B via P to B_p , and then sends the encapsulated packet on the shortest path to B_p . S follows the same procedure for routers A and C in [Figure 2 \(The set of Not-via P Addresses \)](#). The packet is decapsulated at the repair target (A , B or C) and then forwarded normally to its destination. The repair target can be determined as part of the normal SPF by recording the "next-next-hop" for each destination in addition to the normal next-hop. Notice that with this technique only one level of encapsulation is needed, and that it is possible to repair ANY failure regardless of link metrics and any asymmetry that may be present in the network. The only exception to this is where the failure was a single point of failure that partitioned the network, in which case ANY repair is clearly impossible.

4.2. Link Failure

[TOC](#)

The normal mode of operation of the network would be to assume router failure. However, where some destinations are only reachable through the failed router, it is desirable that an attempt be made to repair to those destinations by assuming that only a link failure has occurred. To perform a link repair, S encapsulates to Ps (i.e. it instructs the network to deliver the packet to P not-via S). All of the neighbors of S will have calculated a path to Ps in case S itself had failed. S could therefore give the packet to any of its neighbors (except, of course, P). However, S should preferably send the encapsulated packet on the shortest available path to P. This path is calculated by running an SPF with the link SP failed. Note that this may again be an incremental calculation, which can terminate when address Ps has been reattached.

4.2.1. Loop Prevention Under Node Failure

[TOC](#)

It is necessary to consider the behavior of IPFRR solutions when a link repair is attempted in the presence of node failure. In its simplest form the not-via IPFRR solution prevents the formation of loops forming as a result of mutual repair, by never providing a repair path for a not-via address. The repair of packets with not-via addresses is considered in more detail in [Section 5.3 \(Multiple Independent Failures\)](#). Referring to [Figure 2 \(The set of Not-via P Addresses\)](#), if A was the neighbor of P that was on the link repair path from S to P, and P itself had failed, the repaired packet from S would arrive at A encapsulated to Ps. A would have detected that the AP link had failed and would normally attempt to repair the packet. However, no repair path is provided for any not-via address, and so A would be forced to drop the packet, thus preventing the formation of loop.

4.3. Multi-homed Prefixes

[TOC](#)

A multi-homed Prefix (MHP) is a prefix that is reachable via more than one router in the network. Some of these may be repairable using LFAs as described in [\[RFC5286\] \(Atlas, A. and A. Zinin, "Basic Specification for IP Fast Reroute: Loop-Free Alternates," September 2008.\)](#). Only those without such a repair need be considered here.

When IPFRR router S ([Figure 3 \(Multi-homed Prefixes\)](#)) discovers that P has failed, it needs to send packets addressed to the MHP X, which is normally reachable through P, to an alternate router, which is still able to reach X.

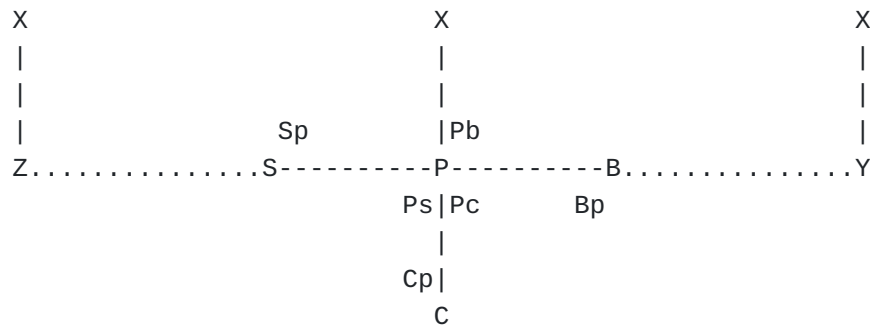


Figure 3: Multi-homed Prefixes

S should choose the closest router that can reach X during the failure as the alternate router. S determines which router to use as the alternate while running the SPF with P failed. This is accomplished by the normal process of re-attaching a leaf node to the core topology (this is sometimes known as a "partial SPF").

First, consider the case where the shortest alternate path to X is via Z. S can reach Z without using the failed router P. However, S cannot just send the packet towards Z, because the other routers in the network will not be aware of the failure of P, and may loop the packet back to S. S therefore encapsulates the packet to Z (using a normal address for Z). When Z receives the encapsulated packet it removes the encapsulation and forwards the packet to X.

Now consider the case where the shortest alternate path to X is via Y, which S reaches via P and B. To reach Y, S must first repair the packet to B using the normal not-via repair mechanism. To do this S encapsulates the packet for X to Bp. When B receives the packet it removes the encapsulation and discovers that the packet is intended for MHP X. The situation now reverts to the previous case, in which the shortest alternate path does not require traversal of the failure. B therefore follows the algorithm above and encapsulates the packet to Y (using a normal address for Y). Y removes the encapsulation and forwards the packet to X.

It may be that the cost of reaching X using local delivery from the alternate router (i.e. Z or Y) is greater than the cost of reaching X via P. Under those circumstances, the alternate router would normally forward to X via P, which would cause the IPFRR repair to loop. To prevent the repair from looping the alternate router must locally deliver a packet received via a repair encapsulation. This may be

specified by using a special address with the above semantics. Note that only one such address is required per node. Notice that using the not-via approach, only one level of encapsulation was needed to repair MHPs to the alternate router.

4.4. Installation of Repair Paths

[TOC](#)

The following algorithm is used by node S ([Figure 3 \(Multi-homed Prefixes\)](#)) to pre- calculate and install repair paths in the FIB, ready for immediate use in the event of a failure. It is assumed that the not-via repair paths have already been calculated as described above. For each neighbor P, consider all destinations which are reachable via P in the current topology:-

1. For all destinations with an ECMP or LFA repair (as described in [\[RFC5286\] \(Atlas, A. and A. Zinin, "Basic Specification for IP Fast Reroute: Loop-Free Alternates," September 2008.\)](#)) install that repair.
2. For each destination (DR) that remains, identify in the current topology the next-next-hop (H) (i.e. the neighbor of P that P will use to send the packet to DR). This can be determined during the normal SPF run by recording the additional information. If S has a path to the not-via address Hp (H not via P), install a not-via repair to Hp for the destination DR.
3. Identify all remaining destinations (M) which can still be reached when node P fails. These will be multi-homed prefixes that are not repairable by LFA, for which the normal attachment node is P, or a router for which P is a single point of failure, and have an alternative attachment point that is reachable after P has failed. One way of determining these destinations would be to run an SPF rooted at S with node P removed, but an implementation may record alternative attachment points during the normal SPF run. In either case, the next best point of attachment can also be determined for use in step (4) below.
4. For each multi-homed prefix (M) identified in step (3):-
 - A. Identify the new attachment node (as shown in [Figure 3 \(Multi-homed Prefixes\)](#)). This may be:-
 - a. Y, where the next hop towards Y is P, or
 - b. Z, where the next hop towards Z is not P.

If the attachment node is Z, install the repair for M as a tunnel to Z' (where Z' is the address of Z that is used to force local forwarding).

- B. For the subset of prefixes (M) that remain (having attachment point Y), install the repair path previously installed for destination Y.

For each destination (DS) that remains, install a not-via repair to Ps (P not via S). Note, these are destinations for which node P is a single point of failure, and can only be repaired by assuming that the apparent failure of node P was simply a failure of the S-P link. Note that, if available, a downstream path to P may be used for such a repair. This cannot generate a persistent loop in the event of the failure of node P, but if one neighbor of P uses a not-via repair and another uses a downstream path, it is possible for a packet sent on the downstream path to be returned to the sending node inside a not-via encapsulation. Since packets destined to not-via addresses are not repaired, the packet will be dropped after executing a single turn loop.

5. Compound Failures

[TOC](#)

The following types of failures involve more than one component:

1. Shared Risk Link Groups
2. Local Area Networks
3. Multiple Independent Failures

The considerations that apply in each of the above situations are described in the following sections.

5.1. Shared Risk Link Groups

[TOC](#)

A Shared Risk Link Group (SRLG) is a set of links whose failure can be caused by a single action such as a conduit cut or line card failure. When repairing the failure of a link that is a member of an SRLG, it must be assumed that all the other links that are also members of the SRLG have also failed. Consequently, any repair path must be computed

to avoid not just the adjacent link, but also all the links which are members of the same SRLG.

In [Figure 4 \(Shared Risk Link Group\)](#) below, the links S-P and A-B are both members of SRLG "a". The semantics of the not-via address Ps changes from simply "P not-via the link S-P" to be "P not-via the link S-P or any other link with which S-P shares an SRLG" In [Figure 4 \(Shared Risk Link Group\)](#) this is the links that are members of SRLG "a". I.e. links S-P and A-B. Since the information about SRLG membership of all links is available in the Link State Database, all nodes computing routes to the not-via address Ps can infer these semantics, and perform the computation by failing all the links in the SRLG when running the iSPF.

Note that it is not necessary for S to consider repairs to any other nodes attached to members of the SRLG (such as B). It is sufficient for S to repair to the other end of the adjacent link (P in this case).

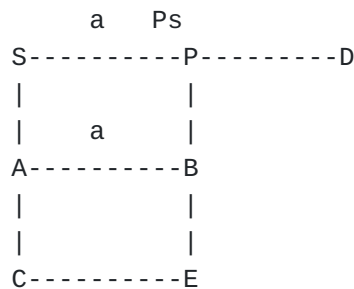


Figure 4: Shared Risk Link Group

In some cases, it may be that the links comprising the SRLG occur in series on the path from S to the destination D, as shown in [Figure 5 \(Shared Risk Link Group members in series\)](#). In this case, multiple consecutive repairs may be necessary. S will first repair to Ps, then P will repair to Dp. In both cases, because the links concerned are members of SRLG "a" the paths are computed to avoid all members of SRLG "a".

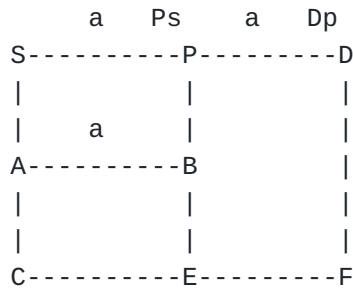


Figure 5: Shared Risk Link Group members in series

While the use of multiple repairs in series introduces some additional overhead, these semantics avoid the potential combinatorial explosion of not-via addresses that could otherwise occur.

Note that although multiple repairs are used, only a single level of encapsulation is required. This is because the first repair packet is decapsulated before the packet is re-encapsulated using the not-via address corresponding to the far side of the next link which is a member of the same SRLG. In some cases the de-capsulation and re-encapsulation takes place (at least notionally) at a single node, while in other cases, these functions may be performed by different nodes.

This scenario is illustrated in [Figure 6 \(Shared Risk Link Group members in series\)](#) below.

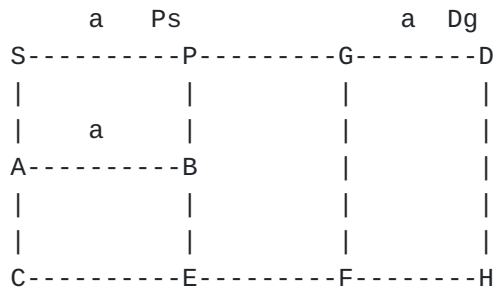


Figure 6: Shared Risk Link Group members in series

In this case, S first encapsulates to Ps, and node P decapsulates the packet and forwards it "native" to G using its normal FIB entry for destination D. G then repairs the packet to Dg.

It can be shown that such multiple repairs can never form a loop because each repair causes the packet to move closer to its destination.

It is often the case that a single link may be a member of multiple SRLGs, and those SRLGs may not be isomorphic. This is illustrated in [Figure 7 \(Multiple Shared Risk Link Groups \)](#) below.

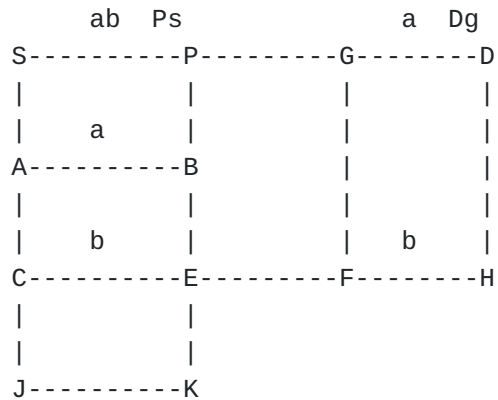


Figure 7: Multiple Shared Risk Link Groups

The link SP is a member of SRLGs "a" and "b". When a failure of the link SP is detected, it must be assumed that BOTH SRLGs have failed. Therefore the not-via path to Ps must be computed by failing all links which are members of SRLG "a" or SRLG "b". I.e. the semantics of Ps is now "P not-via any links which are members of any of the SRLGs of which link SP is a member". This is illustrated in [Figure 8 \(Topology used for repair computation for link S-P\)](#) below.

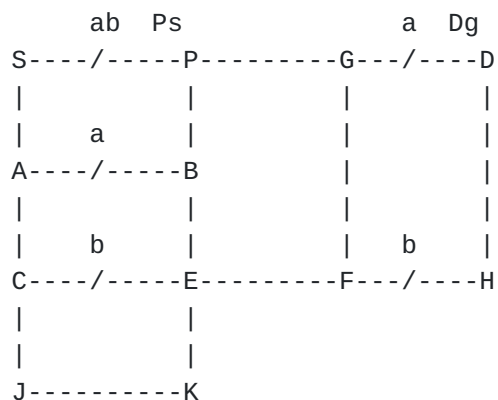


Figure 8: Topology used for repair computation for link S-P

In this case, the repair path to Ps will be S-A-C-J-K-E-B-P. It may appear that there is no path to D because GD is a member of SRLG "a" and FH is a member of SRLG "b". This is true if BOTH SRLGs "a" and "b" have in fact failed. But that would be an instance of multiple uncorrelated failures which are out of scope for this design. In practice it is likely that there is only a single failure, i.e. either SRLG "a" or SRLG "b" has failed, but not both. These two possibilities are indistinguishable from the point of view of the repairing router S and so it must repair on the assumption that both are unavailable. However, each link repair is considered independently. The repair to Ps delivers the packet to P which then forwards the packet to G. When the packet arrives at G, if SRLG "a" has failed it will be repaired around the path G-F-H-D. This is illustrated in [Figure 9 \(Topology used for repair computation for link G-D\)](#) below. If, on the other hand, SRLG "b" has failed, link GD will still be available. In this case the packet will be delivered as normal across the link GD.

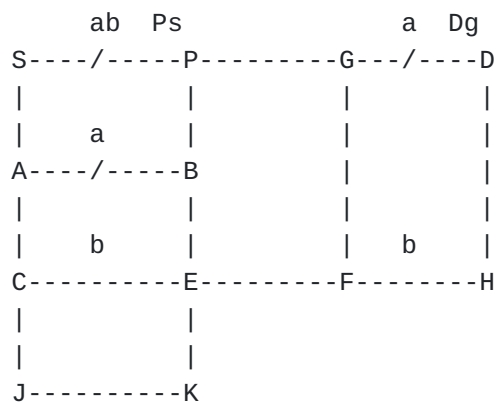


Figure 9: Topology used for repair computation for link G-D

A repair strategy that assumes the worst-case failure for each link can often result in longer repair paths than necessary. In cases where only a single link fails, rather than the full SRLG, this strategy may occasionally fail to identify a repair even though a viable repair path exists in the network. The use of sub-optimal repair paths is an inevitable consequence of this compromise approach. The failure to identify any repair is a serious deficiency, but is a rare occurrence in a robustly designed network. This problem can be addressed by:-

1. Reporting that the link in question is irreparable, so that the network designer can take appropriate action.
2. Modifying the design of the network to avoid this possibility.

3. Using some form of SRLG diagnostic (for example, by running BFD over alternate repair paths) to determine which SRLG member(s) has actually failed and using this information to select an appropriate pre-computed repair path. However, aside from the complexity of performing the diagnostics, this requires multiple not-via addresses per interface, which has poor scaling properties.
 4. Using the mechanism described in [Section 5.3 \(Multiple Independent Failures \)](#)
-

5.1.1. Use of LFAs with SRLGs

[TOC](#)

Section 4.1 above describes the repair of links which are members of one or more SRLGs. LFAs can be used for the repair of such links provided that any other link with which S-P shares an SRLG is avoided when computing the LFA. This is described for the simple case of "local-SRLGs" in [\[RFC5286\] \(Atlas, A. and A. Zinin, "Basic Specification for IP Fast Reroute: Loop-Free Alternates," September 2008.\)](#).

5.2. Local Area Networks

[TOC](#)

LANs are a special type of SRLG and are solved using the SRLG mechanisms outlined above. With all SRLGs there is a trade-off between the sophistication of the fault detection and the size of the SRLG. Protecting against link failure of the LAN link(s) is relatively straightforward, but as with all fast reroute mechanisms, the problem becomes more complex when it is desired to protect against the possibility of failure of the nodes attached to the LAN as well as the LAN itself.

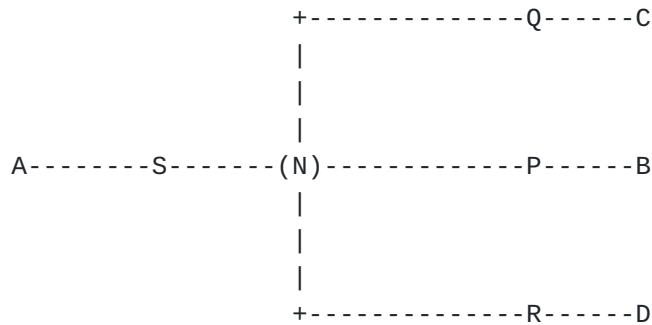


Figure 10: Local Area Networks

Consider the LAN shown in [Figure 10 \(Local Area Networks\)](#). For connectivity purposes, we consider that the LAN is represented by the pseudonode (N). To provide IPFRR protection, S must run a connectivity check to each of its protected LAN adjacencies P, Q, and R, using, for example BFD [\[I-D.ietf-bfd-base\] \(Katz, D. and D. Ward, "Bidirectional Forwarding Detection," January 2010.\)](#).

When S discovers that it has lost connectivity to P, it is unsure whether the failure is:

- *its own interface to the LAN,
- *the LAN itself,
- *the LAN interface of P,
- *the node P.

5.2.1. Simple LAN Repair

[TOC](#)

A simple approach to LAN repair is to consider the LAN and all of its connected routers as a single SRLG. Thus, the address P not via the LAN (P1) would require P to be reached not-via any router connected to the LAN. This is shown in [Figure 11 \(Local Area Networks - LAN SRLG\)](#).

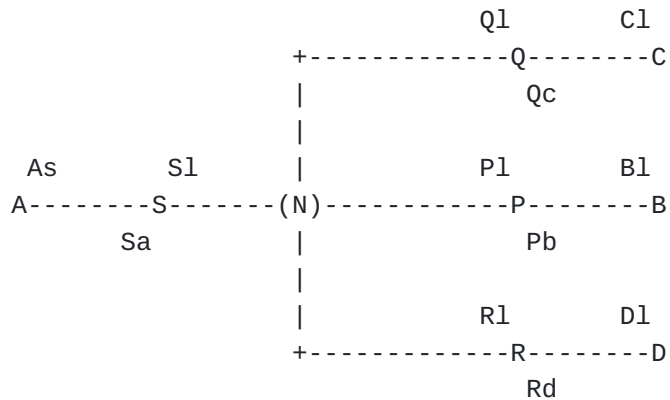


Figure 11: Local Area Networks - LAN SRLG

In this case, when S detected that P had failed it would send traffic reached via P and B to B not-via the LAN or any router attached to the LAN (i.e. to B1). Any destination only reachable through P would be addressed to P not-via the LAN or any router attached to the LAN (except of course P).

Whilst this approach is simple, it assumes that a large portion of the network adjacent to the failure has also failed. This will result in the use of sub-optimal repair paths and in some cases the inability to identify a viable repair.

5.2.2. LAN Component Repair

[TOC](#)

In this approach, possible failures are considered at a finer granularity, but without the use of diagnostics to identify the specific component that has failed. Because S is unable to diagnose the failure it must repair traffic sent through P and B, to B not- via P,N (i.e. not via P and not via N), on the conservative assumption that both the entire LAN and P have failed. Destinations for which P is a single point of failure must as usual be sent to P using an address that avoids the interface by which P is reached from S, i.e. to P not-via N. Similarly for routers Q and R.

Notice that each router that is connected to a LAN must, as usual, advertise one not-via address for each neighbor. In addition, each router on the LAN must advertise an extra address not via the pseudonode (N).

Notice also that each neighbor of a router connected to a LAN must advertise two not-via addresses, the usual one not via the neighbor and an additional one, not via either the neighbor or the pseudonode. The required set of LAN address assignments is shown in [Figure 12 \(Local Area Networks\)](#) below. Each router on the LAN, and each of its

neighbors, is advertising exactly one address more than it would otherwise have advertised if this degree of connectivity had been achieved using point-to-point links.

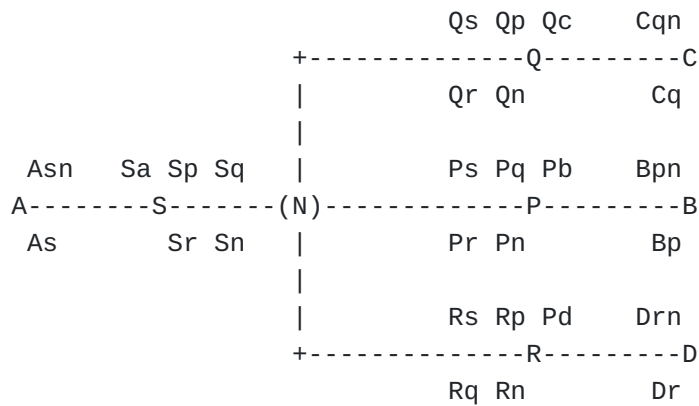


Figure 12: Local Area Networks

5.2.3. LAN Repair Using Diagnostics

[TOC](#)

A more specific LAN repair can be undertaken by using diagnostics. In order to explicitly diagnose the failed network component, S correlates the connectivity reports from P and one or more of the other routers on the LAN, in this case, Q and R. If it lost connectivity to P alone, it could deduce that the LAN was still functioning and that the fault lay with either P, or the interface connecting P to the LAN. It would then repair to B not via P (and P not-via N for destinations for which P is a single point of failure) in the usual way. If S lost connectivity to more than one router on the LAN, it could conclude that the fault lay only with the LAN, and could repair to P, Q and R not-via N, again in the usual way.

5.3. Multiple Independent Failures

[TOC](#)

IPFRR repair of multiple simultaneous failures which are not members of a known SRLG is complicated by the problem that the use of multiple concurrent repairs may result in looping repair paths. As described in [Section 4.2.1 \(Loop Prevention Under Node Failure\)](#), the simplest method

of preventing such loops, is to ensure that packets addressed to a not-via address are not repaired but instead are dropped. It is possible that a network may experience multiple simultaneous failures. This may be due to simple statistical effects, but the more likely cause is unanticipated SRLGs. When multiple failures which are not part of an anticipated group are detected, repairs are abandoned and the network reverts to normal convergence. Although safe, this approach is somewhat draconian, since there are many circumstances where multiple repairs do not induce loops.

This section describes the properties of multiple unrelated failures and proposes some methods that may be used to address this problem.

5.3.1. Looping Repairs

[TOC](#)

Let us assume that the repair mechanism is based solely on not-via repairs. LFA or downstream routes may be incorporated, and will be dealt with later.

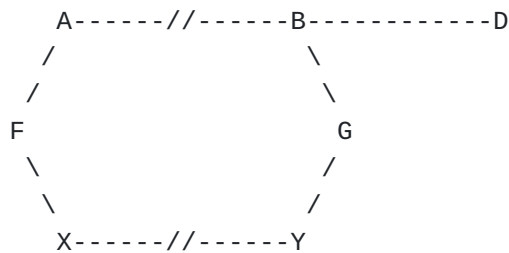


Figure 13: The General Case of Multiple Failures

The essential case is as illustrated in [Figure 13 \(The General Case of Multiple Failures\)](#). Note that depending on the repair case under consideration, there may be paths present in [Figure 13 \(The General Case of Multiple Failures\)](#), that are in addition to those shown in the figure. For example there may be paths between A and B, and/or between X and Y. These paths are omitted for graphical clarity. There are three cases to consider:

- 1) Consider the general case of a pair of protected links A-B and X-Y as shown in the network fragment shown [Figure 13 \(The General Case of Multiple Failures\)](#). If the repair path for A-B does not traverse

X-Y and the repair path for X-Y does not traverse A-B, this case is completely safe and will not cause looping or packet loss.

A more common variation of this case is shown in [Figure 14 \(Concatenated Repairs\)](#), which shows two failures in different parts of the network in which a packet from A to D traverses two concatenated repairs.

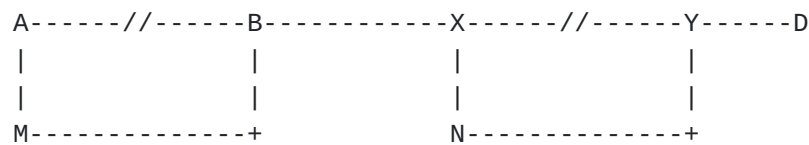


Figure 14: Concatenated Repairs

2) In [Figure 13 \(The General Case of Multiple Failures\)](#), the repair for A-B traverses X-Y, but the repair for X-Y does not traverse A-B. This case occurs when the not-via path from A to B traverses link X-Y, but the not-via path from X to Y traverses some path not shown in [Figure 13 \(The General Case of Multiple Failures\)](#). Without the multi-failure mechanism described in this section the repaired packet for A-B would be dropped when it reached X-Y, since the repair of repaired packets would be forbidden. However, if this packet were allowed to be repaired, the path to D would be complete and no harm would be done, although two levels of encapsulation would be required.

3) The repair for A-B traverses X-Y AND the repair for X-Y traverses A-B. In this case unrestricted repair would result in looping packets and increasing levels of encapsulation.

The challenge in applying IPFRR to a network that is undergoing multiple failures is, therefore, to identify which of these cases exist in the network and react accordingly.

5.3.2. Outline Solution

[TOC](#)

When A is computing the not-via repair path for A-B (i.e. the path for packets addressed to Ba, read as "B not-via A") it is aware of the list

of nodes which this path traverses. This can be recorded by a simple addition to the SPF process, and the not-via addresses associated with each forward link can be determined. If the path were A, F, X, Y, G, B, ([Figure 13 \(The General Case of Multiple Failures\)](#)) the list of not-via addresses would be: Fa, Xf, Yx, Gy, Bg. Under standard not-via operation, A would populate its FIB such that all normal addresses normally reachable via A-B would be encapsulated to Ba when A-B fails, but traffic addressed to any not-via address arriving at A would be dropped. The new procedure modifies this such that any traffic for a not-via address normally reachable over A-B is also encapsulated to Ba unless the not-via address is one of those previously identified as being on the path to Ba, for example Yx, in which case the packet is dropped.

The above procedure allows cases 1 and 2 above to be repaired, while preventing the loop which would result from case 3.

Note that this is accomplished by pre-computing the required FIB entries, and does not require any detailed packet inspection. The same result could be achieved by checking for multiple levels of encapsulation and dropping any attempt to triple encapsulate. However, this would require more detailed inspection of the packet, and causes difficulties when more than 2 “simultaneous” failures are contemplated. So far we have permitted benign repairs to coexist, albeit sometimes requiring multiple encapsulation. Note that in many cases there will be no performance impact since unless both failures are on the same node, the two encapsulations or two decapsulations will be performed at different nodes. There is however the issue of the MTU impact of multiple encapsulations.

In the following sub-section we consider the various strategies that may be applied to case 3 - mutual repairs that would loop.

5.3.3. Looping Repairs

[TOC](#)

In case 3, the simplest approach is to simply not install repairs for repair paths that might loop. In this case, although the potentially looping traffic is dropped, the traffic is not repaired. If we assume that a hold-down is applied before reconvergence in case the link failure was just a short glitch, and if a loop free convergence mechanism further delays convergence, then the traffic will be dropped for an extended period. In these circumstances it would be better to “abandon all hope” (AAH) [[draft-bryant-francois-shand-ipfrr-aah-00.txt](#)] and immediately invoke normal re-convergence.

Note that it is not sufficient to expedite the issuance of an LSP reporting the failure, since this may be treated as a permitted simultaneous failure by the oFIB algorithm. It is therefore necessary to explicitly trigger an oFIB AAH.

5.3.3.1. Dropping Looping Packets

[TOC](#)

One approach to case 3 is to allow the repair, and to experimentally discover the incompatibility of the repairs if and when they occur. With this method we permit the repair in case 3 and trigger AAH when a packet drop count on the not-via address has been incremented. Alternatively, it is possible to wait until the LSP describing the change is issued normally (i.e. when X announces the failure of X-Y). When the repairing node A, which has precomputed that X-Y failures are mutually incompatible with its own repairs receives this LSP it can then issue the AAH. This has the disadvantage that it doesn't overcome the hold-down delay, but it requires no "data-driven" operation, and it still has the required effect of abandoning the oFIB which is probably the longer of the delays (although with signalled oFIB this should be sub-second).

Whilst both of the experimental approaches described above are feasible, they tend to induce AAH in the presence of otherwise feasible repairs, and they are contrary to the philosophy of repair pre-determination that has been applied to existing IPFRR solutions.

5.3.3.2. Computing non-looping Repairs of Repairs

[TOC](#)

An alternative approach to simply dropping the looping packets, or to detecting the loop after it has occurred, is to use secondary SRLGs. With a link state routing protocol it is possible to precompute the incompatibility of the repairs in advance and to compute an alternative SRLG repair path. Although this does considerably increase the computational complexity it may be possible to compute repair paths that avoid the need to simply drop the offending packets. This approach requires us to identify the mutually incompatible failures, and advertise them as "secondary SRLGs". When computing the repair paths for the affected not-via addresses these links are simultaneously failed. Note that the assumed simultaneous failure and resulting repair path only applies to the repair path computed for the conflicting not-via addresses, and is not used for normal addresses. This implies that although there will be a longer repair path when there is more than one failure, if there is a single failure the repair path length will be "normal".

Ideally we would wish to only invoke secondary SRLG computation when we are sure that the repair paths are mutually incompatible. Consider the case of node A in [Figure 13 \(The General Case of Multiple Failures\)](#). A first identifies that the repair path for A-B is via F-X-Y-G-B. It then explores this path determining the repair path for each link in the path. Thus, for example, it performs a check at X by running an SPF

rooted at X with the X-Y link removed to determine whether A-B is indeed on X's repair path for packets addressed to Yx.

Some optimizations are possible in this calculation, which appears at first sight to be order hk (where h is the average hop length of repair paths and k is the average number of neighbours of a router). When A is computing its set of repair paths, it does so for all its k neighbours. In each case it identifies a list of node pairs traversed by each repair. These lists may often have one or more node pairs in common, so the actual number of link failures which require investigation is the union of these sets. It is then necessary to run an SPF rooted at the first node of each pair (the first node because the pairings are ordered representing the direction of the path), with the link to the second node removed. This SPF, while not an incremental, can be terminated as soon as the not-via address is reached. For example, when running the SPF rooted at X, with the link X-Y removed, the SPF can be terminated when Yx is reached. Once the path has been found, the path is checked to determine if it traverses any of A's links in the direction away from A. Note that, because the node pair XY may exist in the list for more than one of A's links (i.e. it lies on more than one repair path), it is necessary to identify the correct list, and hence link which has a mutually looping repair path. That link of A is then advertised by A as a secondary SRLG paired with the link X-Y. Also note that X will be running this algorithm as well, and will identify that XY is paired with A-B and so advertise it. This could perhaps be used as a further check.

The ordering of the pairs in the lists is important. i.e. X-Y and Y-X are dealt with separately. If and only if the repairs are mutually incompatible, we need to advertise the pair of links as a secondary SRLG, and then ALL nodes compute repair paths around both failures using an additional not-via address with the semantics not-via A-B AND not-via X-Y.

A further possibility is that because we are going to the trouble of advertising these SRLG sets, we could also advertise the new repair path and only get the nodes on that path to perform the necessary computation. Note also that once we have reached Q space with respect to the two failures we need no longer continue the computation, so we only need to notify the nodes on the path that are not in Q-space. One cause of mutually looping repair paths is the existence of nodes with only two links, or sections of the network which are only bi-connected. In these cases, repair is clearly impossible - the failure of both links partitions the network. It would be advantageous to be able to identify these cases, and inhibit the fruitless advertisement of the secondary SRLG information. This could be achieved by the node detecting the requirement for a secondary SRLG, first running the not-via computation with both links removed. If this does not result in a path, it is clear that the network would be partitioned by such a failure, and so no advertisement is required.

5.3.3.3. N-level Mutual Loops

[TOC](#)

[Editors' Note: This section needs to be reviewed before final publication]

It is tempting to conclude that the mechanism described above can be applied to the general case of N failures. If we use the approach of assuming that repairs are not mutual, and catching the loops and executing AAH when they occur, then we can attempt repairs in the case of N failures.

If we use the approach of avoiding potentially mutual repairs and creating secondary SRLG, then we have to explore N levels of repair, where N is the number of simultaneous failures we wish to repair.

5.3.4. Mixing LFAs and Not-via

[TOC](#)

So far in this section we have assumed that all repairs use not-via tunnels. However, in practise we may wish to use LFAs or downstream routes where available. This complicates the issue, because their use results in packets which are being repaired, but NOT addressed to not-via addresses. If BOTH links are using downstream routes there is no possibility of looping, since it is impossible to have a pair of nodes which are both downstream of each other [\[RFC5286\] \(Atlas, A. and A. Zinin, "Basic Specification for IP Fast Reroute: Loop-Free Alternates," September 2008.\)](#).

Loops can however occur when LFAs are used. An obvious example is the well known node repair problem with LFAs [\[RFC5286\] \(Atlas, A. and A. Zinin, "Basic Specification for IP Fast Reroute: Loop-Free Alternates," September 2008.\)](#). If one link is using a downstream route, while the other is using a not-via tunnel, the potential mechanism described above would work provided it were possible to determine the nodes on the path of the downstream route. Some methods of computing downstream routes do not provide this path information. If the path information is however available, the link using a downstream route will have a discard FIB entry for the not-via address of the other link. The consequence is that potentially looping packets will be discarded when they attempt to cross this link.

In the case where the mutual repairs are both using not-via repairs, the loop will be broken when the packet arrives at the second failure. However packets are unconditionally repaired by means of a downstream routes, and thus when the mutual pair consists of a downstream route and a not-via repair, the looping packet will only be dropped when it gets back to the first failure. i.e. it will execute a single turn of the loop before being dropped.

There is a further complication with downstream routes, since although the path may be computed to the far side of the failure, the packet may

“peel off” to its destination before reaching the far side of the failure. In this case it may traverse some other link which has failed and was not accounted for on the computed path. If the A-B repair (Figure 1) is a downstream route and the X-Y repair is a not-via repair, we can have the situation where the X-Y repair packets encapsulated to Yx follow a path which attempts to traverse A-B. If the A-B repair path for “normal” addresses is a downstream route, it cannot be assumed that the repair path for packets addressed to Yx can be sent to the same neighbour. This is because the validity of a downstream route must be ascertained in the topology represented by Yx, i.e. that with the link X-Y failed. This is not the same topology that was used for the normal downstream calculation, and use of the normal downstream route for the encapsulated packets may result in an undetected loop. If it is computationally feasible to check the downstream route in this topology (i.e. for any not-via address Qp which traverses A-B we must perform the downstream calculation for that not-via address in the topology with link Q-P failed.), then the downstream repair for Yx can safely be used. These packets cannot re-visit X-Y, since by definition they will avoid that link. Alternatively, the packet could be always repaired in a not-via tunnel. i.e. even though the normal repair for traffic traversing A-B would be to use a downstream route, we could insist that such traffic addressed to a not-via address MUST use a tunnel to Ba. Such a tunnel would only be installed for an address Qp if it were established that it did not traverse Q-P (using the rules described above).

6. Optimizing not-via computations using LFAs

[TOC](#)

If repairing node S has an LFA to the repair endpoint it is not necessary for any router to perform the incremental SPF with the link SP removed in order to compute the route to the not-via address Ps. This is because the correct routes will already have been computed as a result of the SPF on the base topology. Node S can signal this condition to all other routers by including a bit in its LSP or LSA associated with each LFA protected link. Routers computing not-via routes can then omit the running of the iSPF for links with this bit set.

When running the iSPF for a particular link AB, the calculating router first checks whether the link AB is present in the existing SPT. If the link is not present in the SPT, no further work is required. This check is a normal part of the iSPF computation.

If the link is present in the SPT, this optimization introduces a further check to determine whether the link is marked as protected by an LFA in the direction in which the link appears in the SPT. If so the iSPF need not be performed. For example, if the link appears in the SPT

in the direction A->B and A has indicated that the link AB is protected by an LFA no further action is required for this link.

If the receipt of this information is delayed, the correct operation of the protocol is not compromised provided that the necessity to perform a not-via computation is re-evaluated whenever new information arrives. This optimization is not particularly beneficial to nodes close to the repair since, as has been observed above, the computation for nodes on the LFA path is trivial. However, for nodes upstream of the link SP for which S-P is in the path to P, there is a significant reduction in the computation required.

7. Multicast

[TOC](#)

Multicast traffic can be repaired in a similar way to unicast. The multicast forwarder is able to use the not-via address to which the multicast packet was addressed as an indication of the expected receive interface and hence to correctly run the required RPF check.

In some cases, all the destinations, including the repair endpoint, are repairable by an LFA. In this case, all unicast traffic may be repaired without encapsulation. Multicast traffic still requires encapsulation, but for the nodes on the LFA repair path the computation of the not-via forwarding entry is unnecessary since, by definition, their normal path to the repair endpoint is not via the failure.

A more complete description of multicast operation is for further study.

8. Fast Reroute in an MPLS LDP Network.

[TOC](#)

Not-via addresses are IP addresses and LDP [\[RFC5036\] \(Andersson, L., Minei, I., and B. Thomas, "LDP Specification," October 2007.\)](#) will distribute labels for them in the usual way. The not-via repair mechanism may therefore be used to provide fast re-route in an MPLS network by first pushing the label which the repair endpoint uses to forward the packet, and then pushing the label corresponding to the not-via address needed to effect the repair. Referring once again to [Figure 1 \(Not-via repair of router failure\)](#), if S has a packet destined for D that it must reach via P and B, S first pushes B's label for D. S then pushes the label that its next hop to Bp needs to reach Bp.

Note that in an MPLS LDP network it is necessary for S to have the repair endpoint's label for the destination. When S is effecting a link repair it already has this. In the case of a node repair, S either needs to set up a directed LDP session with each of its neighbor's neighbors, or it needs to use the next-next hop label distribution

mechanism proposed in [\[I-D.shen-mpls-ldp-nnhop-label\]](#) (Shen, N., "Discovering LDP Next-NextHop Labels," May 2005.).

9. Encapsulation

[TOC](#)

Any IETF specified IP in IP encapsulation may be used to carry a not-via repair. IP in IP [\[RFC2003\]](#) (Perkins, C., "IP Encapsulation within IP," October 1996.), GRE [\[RFC1701\]](#) (Hanks, S., Li, T., Farinacci, D., and P. Traina, "Generic Routing Encapsulation (GRE)," October 1994.) and L2TPv3 [\[RFC3931\]](#) (Lau, J., Townsley, M., and I. Goyret, "Layer Two Tunneling Protocol - Version 3 (L2TPv3)," March 2005.), all have the necessary and sufficient properties. The requirement is that both the encapsulating router and the router to which the encapsulated packet is addressed have a common ability to process the chosen encapsulation type. When an MPLS LDP network is being protected, the encapsulation would normally be an additional MPLS label. In an MPLS enabled IP network an MPLS label may be used in place of an IP in IP encapsulation in the case above.

10. Routing Extensions

[TOC](#)

IPFRR requires IGP extensions. Each IPFRR router that is directly connected to a protected network component must advertise a not-via address for that component. This must be advertised in such a way that the association between the protected component (link, router or SRLG) and the not-via address can be determined by the other routers in the network.

It is necessary that not-via capable routers advertise in the IGP that they will calculate not-via routes.

It is necessary for routers to advertise the type of encapsulation that they support (MPLS, GRE, L2TPv3 etc). However, the deployment of mixed IP encapsulation types within a network is discouraged.

11. Incremental Deployment

[TOC](#)

Incremental deployment is supported by excluding routers that are not calculating not-via routes (as indicated by their capability information flooded with their link state information) from the base topology used for the computation of repair paths. In that way repairs may be steered around islands of routers that are not IPFRR capable. Routers that are protecting a network component need to have the

capability to encapsulate and decapsulate packets. However, routers that are on the repair path only need to be capable of calculating not-via paths and including the not-via addresses in their FIB i.e. these routers do not need any changes to their forwarding mechanism.

12. IANA Considerations

[TOC](#)

There are no IANA considerations that arise from this draft.

13. Security Considerations

[TOC](#)

The repair endpoints present vulnerability in that they might be used as a method of disguising the delivery of a packet to a point in the network. The primary method of protection should be through the use of a private address space for the not-via addresses. These addresses MUST NOT be advertised outside the area, and SHOULD be filtered at the network entry points. In addition, a mechanism might be developed that allowed the use of the mild security available through the use of a key [\[RFC1701\]](#) (Hanks, S., Li, T., Farinacci, D., and P. Traina, "Generic Routing Encapsulation (GRE)," October 1994.) [\[RFC3931\]](#) (Lau, J., Townsley, M., and I. Goyret, "Layer Two Tunneling Protocol - Version 3 (L2TPv3)," March 2005.). With the deployment of such mechanisms, the repair endpoints would not increase the security risk beyond that of existing IP tunnel mechanisms. An attacker may attempt to overload a router by addressing an excessive traffic load to the de-capsulation endpoint. Typically, routers take a 50% performance penalty in decapsulating a packet. The attacker could not be certain that the router would be impacted, and the extremely high volume of traffic needed, would easily be detected as an anomaly. If an attacker were able to influence the availability of a link, they could cause the network to invoke the not-via repair mechanism. A network protected by not-via IPFRR is less vulnerable to such an attack than a network that undertook a full convergence in response to a link up/down event.

14. Acknowledgements

[TOC](#)

The authors would like to acknowledge contributions made by Alia Atlas and John Harper.

15. References

[TOC](#)

15.1. Normative References

[TOC](#)

[RFC2119]	Bradner, S. , " Key words for use in RFCs to Indicate Requirement Levels ," BCP 14, RFC 2119, March 1997 (TXT , HTML , XML).
-----------	--

15.2. Informative References

[TOC](#)

[I-D.ietf-bfd-base]	Katz, D. and D. Ward, " Bidirectional Forwarding Detection ," draft-ietf-bfd-base-11 (work in progress), January 2010 (TXT).
[I-D.shen-mpls-ldp-nnhop-label]	Shen, N., " Discovering LDP Next-NextHop Labels ," draft-shen-mpls-ldp-nnhop-label-02 (work in progress), May 2005 (TXT).
[ISPF]	McQuillan, J., Richer, I., and E. Rosen, "ARPANET Routing Algorithm Improvements", BBN Technical Report 3803, 1978.
[RFC1701]	Hanks, S. , Li, T. , Farinacci, D. , and P. Traina , " Generic Routing Encapsulation (GRE) ," RFC 1701, October 1994 (TXT).
[RFC2003]	Perkins, C. , " IP Encapsulation within IP ," RFC 2003, October 1996 (TXT , HTML , XML).
[RFC3931]	Lau, J., Townsley, M., and I. Goyret, " Layer Two Tunneling Protocol - Version 3 (L2TPv3) ," RFC 3931, March 2005 (TXT).
[RFC5036]	Andersson, L., Minei, I., and B. Thomas, " LDP Specification ," RFC 5036, October 2007 (TXT).
[RFC5286]	Atlas, A. and A. Zinin, " Basic Specification for IP Fast Reroute: Loop-Free Alternates ," RFC 5286, September 2008 (TXT).
[RFC5714]	Shand, M. and S. Bryant, " IP Fast Reroute Framework ," RFC 5714, January 2010 (TXT).

Authors' Addresses

[TOC](#)

	Mike Shand
	Cisco Systems
	250, Longwater Avenue.
	Reading, Berks RG2 6GB

	UK
Email:	mshand@cisco.com
	Stewart Bryant
	Cisco Systems
	250, Longwater Avenue.
	Reading, Berks RG2 6GB
	UK
Email:	stbryant@cisco.com
	Stefano Previdi
	Cisco Systems
	Via Del Serafico, 200
	00142 Rome,
	Italy
Email:	sprevidi@cisco.com