Network Working Group
Internet Draft
Intended status: Standards Track
Expires: September 7, 2011

A. Csaszar
G. Enyedi
S. Kini
Ericsson

March 7, 2011

IP Fast Re-Route with Fast Notification
draft-csaszar-ipfrr-fn-00.txt

Status of this Memo

Copyright Notice

Abstract

This document describes a mechanism that provides IP fast reroute (IPFRR) by using a failure notification (FN) to nodes beyond the ones that first detect the failure (i.e. nodes that are directly connected to the failure point). The paths used when IPFRR-FN is active are in most cases identical to those used after Interior Gateway Protocol (IGP) convergence. The proposed mechanism can address all single node and link failures in an area and has been designed to allow traffic recovery traffic to happen quickly (The goal being to recover in under 50msec).

Table of Contents

1. Introduction.....2
2. Overview of current IPFRR Proposals based on Local Repair.....5
3. Requirements of an Explicit Failure Signaling Mechanism.....6
4. Conceptual Operation of IPFRR relying on Fast Notification....7
4.1. Preparation Phase.....7
4.2. Failure Reaction Phase.....7
4.2.1. Activating Failure Specific Backups.....8
5. Operation Details.....9
5.1. Message Handling and Encoding.....9
5.1.1. Failure Identification Message for OSPF.....10
5.1.2. Failure Identification TLV for ISIS.....12
5.2. Bypassing Legacy Nodes.....12
5.3. Capability Advertisement.....13
6. Protection against Replay Attacks.....13
6.1. Calculating LSDB Digest.....14
7. Security Considerations.....14
8. IANA Considerations.....15
9. References.....15
9.1. Normative References.....15
9.2. Informative References.....15
10. Acknowledgments.....16
Appendix A. Memory needs of a Naive Implementation.....17
A.1. An Example Implementation.....17
A.2. Estimation of Memory Requirements.....18

1. Introduction

Convergence of link-state IGPs, such as OSPF or IS-IS, after a link or node failure is known to be relatively slow. While this may be sufficient for many applications, some network SLAs and applications require faster reaction to network failures.

IGP convergence time is composed mainly of:

1. Failure detection at nodes adjacent to the failure

2. Advertisement of the topology change

3. Calculation of new routes

4. Installing new routes to linecards

Traditional Hello-based failure detection methods of link-state IGPs are relatively slow, hence a new, optimized, Hello protocol has been standardized [BFD] which can reduce failure detection times to the range of 10ms even if no lower layer notices the failure quickly (like loss of signal, etc.).

Even with fast failure detection, reaction times of IGPs may take several seconds, and even with a tuned configuration it may take at least a couple of hundreds of milliseconds.

To decrease fail-over time even further, IPFRR techniques [RFC5714], can be introduced. IPFRR solutions compliant with [RFC5714] are targeting fail-over time reduction of steps 2-4 with the following design principles:

IGP		IPFRR
2. Advertisement of the topology change	==>	No explicit advertisement, only local repair
3. Calculation of new routes	==>	Pre-computation of new routes
4. Installing new routes to linecards	==>	Pre-installation of backup routes

Pre-computing means that the way of bypassing a failed resource is computed before any failure occurs. In order to limit complexity, IPFRR techniques typically prepare for single link, single node and single Shared Risk Link Group (SRLG) failures, which failure types are undoubtedly the most common ones. The pre-calculated backup routes are also downloaded to linecards in preparation for the failure, in this way sparing the lengthy communication between control plane and data plane when a failure happens.

The principle of local rerouting requires forwarding a packet along a detour even if only the immediate neighbors of the failed resource know the failure. IPFRR methods observing the local rerouting principle do not explicitly propagate the failure information.

Unfortunately, packets on detours must be handled in a different way than normal packets as otherwise they might get returned to the failed resource. Rephrased, a node not having *any* sort of information about the failure may loop the packet back to the node from where it was rerouted - simply because its default routing/forwarding configuration dictates that. As an example, see the following figure. Assuming a link failure between A and Dst, A needs to drop packets heading to Dst. If node A forwarded packets to Src, and if the latter had absolutely no knowledge of the failure, a loop would be formed between Src and A.

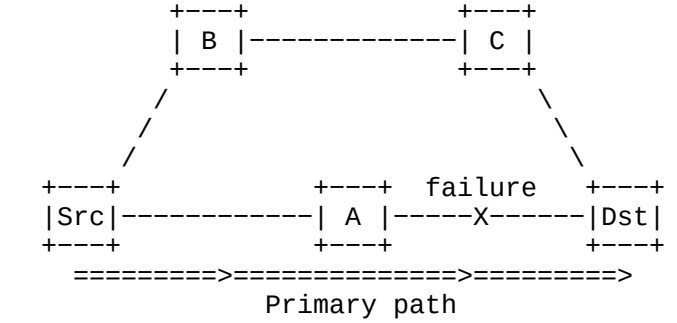


Figure 1 Forwarding inconsistency in case of local repair: The path of Src to Dst leads through A

The basic problem that previous IPFRR solutions struggle to solve is, therefore, to provide consistent routing hop-by-hop without explicit signaling of the failure.

To provide protection for all single failure cases in arbitrary topologies, the information about the failure must be given in *some* way to other nodes. That is, IPFRR solutions targeting full failure coverage need to signal the fact and to some extent the identity of the failure within the data packet as no explicit signaling is allowed. Such solutions have turned out to be considerably complex and hard or impossible to implement practically. The Loop Free Alternates (LFA) solution [RFC5286] does not give the failure information in any way to other routers, and so it cannot repair all failure cases such as the one in Figure 1.

As discussed in Section 2. , solutions that address full failure coverage and rely on local repair, i.e. carrying some failure information within the data packets, fail to present a practical alternative to LFA. This draft, therefore, suggests that relaxing the

local re-routing principle with carefully engineered explicit failure signaling is an effective approach.

The idea of using explicit failure notification for IPFRR has been proposed before for Remote LFA Paths [RLFAP]. RLFAP limits the radius in which the notification is propagated. This draft attempts to work out in more detail what kind of failure dissemination mechanism is required to facilitate remote repair efficiently. Requirements for explicit signaling are given in Section 3. This draft does not limit the failure advertisement radius as opposed to RLFAP. As a result, the detour paths remain stable in most cases, since they are identical to those that the IGP will calculation after IGP convergence. Hence, micro-loop will not occur after IGP convergence.

Note that the current -00 version of the draft only targets protection of single link and single node failures. SRLG protection is left for a future revision.

2. Overview of current IPFRR Proposals based on Local Repair

The only practically feasible solution, Loop Free Alternates [RFC5286], offers the simplest resolution of the consistency problem: a node performing fail-over may only use a next-hop as backup if it is guaranteed that it does not send the packets back. These neighbors are called Loop-Free Alternates (LFA). LFAs, however, do not always exist, as shown in Figure 1 above, i.e., node A has no LFAs with respect to Dst. while it is true that tweaking the network configuration may boost LFA failure case coverage considerably [Ret2011], LFAs cannot protect all failure cases in arbitrary network topologies.

The exact way of adding the information to data packets and its usage for forwarding is the most important property that differentiates most existing IPFRR proposals.

Packets can be marked "implicitly", when they are not altered in any way, but some extra information owned by the router helps deciding the correct way of forwarding. Such extra information can be for instance the direction of the packet, e.g., the interface, which the packet arrived through, e.g. as in [FIFR]. Such solutions require what is called interface-based or interface-specific forwarding.

Interface-based forwarding significantly changes the well-established nature of IP's destination-based forwarding principle, where the IP destination address alone describes the next hop. One embodiment would need to download different FIBs for each physical or virtual IP interface - not a very compelling idea. Another embodiment would

alter the next-hop selection process by adding the incoming interface id also to the lookup fields, which would impact forwarding performance considerably.

Other solutions mark data packets explicitly. Some proposals suggest using free bits in the IP header [MRC], which unfortunately do not exist in the IPv4 header. Other proposals resort to encapsulating re-routed packets with an additional IP header as in e.g. [NotVia] or [Eny2009b]. Encapsulation raises the problem of fragmentation and reassembly, which could be a performance bottleneck, if many packets are sent at MTU size. Another significant problem is the additional management complexity of the encapsulation addresses, which have their own semantics and need to be calculated in a failure specific manner.

3. Requirements of an Explicit Failure Signaling Mechanism

Any signaling mechanism which should be used to advertise failure notifications and so to facilitate extremely quick remote repair should have the following properties.

1. The signaling mechanism should be reliable. The mechanism needs to propagate the failure information to all interested nodes even in a network where a single link or a node is down.

2. The mechanism should be fast in the sense that getting the notification packet to remote nodes through possible multiple hops should not require (considerably) more processing at each hop than plain fast path packet forwarding.

3. The mechanism should involve simple and efficient processing to be feasible for implementation in the dataplane. This goal manifests itself in three ways: Origination of notification should very easy, e.g. creating a simple IP packet, the payload of which can be filled easily. When receiving the packet, it should be easy to recognize by dataplane linecards so that processing can commence after forwarding. No complex operations should be required in order to extract the information from the packet needed to activate the correct backup routes.

4. The mechanism should be trustable; that is, it should provide means to verify the authenticity of the notifications without significant increase of the processing burden in the dataplane.

5. Duplication of notification packets should be either strictly bounded or handled without significant dataplane processing burden.

These requirements present a trade-off. A proper balance needs to be found that offers good enough authentication and reliability while keeping processing complexity sufficiently low to be feasible for data plane implementation. One such solution is proposed in [fn-transport], which is the assumed notification protocol in the following.

4. Conceptual Operation of IPFRR relying on Fast Notification

This section outlines the operation of an IPFRR mechanism relying on Fast Notification.

4.1. Preparation Phase

Like each IPFRR solution, here it is also required to have means for quick failure detection in place, such as lower layer notifications or BFD.
The FN service needs to be activated and configured. The FN service should be bound to failure detection in such a way that FN can disseminate the information identifying the failure to the area.

Failure specific alternative path computation should typically be executed at lower priority than other routing processing.

Pre-computing the next hops on the new shortest paths for all the possible single failures may seem complex, however, it is not so difficult to realize: First, it can be done "offline", while the network is intact and the CP has few things to do. Second, for a single node, it is not needed to compute all the shortest paths with respect to any possible failures; only those link failures are needed to be taken into consideration, which are in the shortest path tree starting from the node.

After having calculated the failure specific alternative next-hops, those which represent a change to the primary next-hop, should be pre-installed to the linecards together with the identifier of the failure, which triggers the switch-over. (The resource needs of an example implementation are briefly discussed in Appendix A.)

4.2. Failure Reaction Phase

The main steps to be taken after a failure are the following:

1. Quick dataplane failure detection

2. Send information about failure using FN service right from dataplane.

3. Forward the received notification as defined by the actually used FN protocol such as the one in [fn-transport]

4. After learning about a local or remote failure, identify failure and activate failure specific backups, if needed, directly within dataplane

After a node detects the loss of connectivity to another node, it should make a decision whether the failure can be handled locally. If local repair is not possible or not configured, for example because LFA is not configured or there are destinations for which no LFA exists, it should trigger the FN service to disseminate the failure description. For instance, if BFD detects a dataplane failure it normally invokes routines to notify the control plane. For the purpose of IPFRR, BFD (or any other lower layer failure detection method) should first trigger FPN before notifying the CP.

After receiving the trigger, without any DP-CP communication involved, FN constructs a packet and adds the description of the failure (described in Section 5.1.) to the payload. The description shall enable recipient nodes to decode that, e.g., node X lost connectivity to node Z. The encoding of the IPFRR-FN packet is described in Section 5.1.

The packet is then disseminated by the FN service in the routing area. Note the synergy of the relation between BFD and IGP Hellos and between FN and IGP link state advertisements. BFD makes a dataplane optimized implementation of the routing protocol's Hello mechanism, Fast Notification makes a dataplane optimized implementation of the link state advertisement flooding mechanism of IGPs.

In each hop, the recipient node needs to perform a "punt and forward". That is, the FN packet not only needs to be forwarded to the FN neighbors as the specific FN mechanism dictates, but a replica needs to be detached and, after forwarding, started to be processed by the dataplane card.

4.2.1. Activating Failure Specific Backups

After the forwarding element extracted the contents of the notification packet, it knows that a node X has lost connectivity to a node Z via a link L. The recipient now needs to decide whether the failure was a link or a node failure. Two approaches can be thought

of. Both options are based on the property that notifications advance in the network as fast as possible.

In the first option, the router does not immediately make the decision, but instead starts a timer set to fire after a couple of milliseconds. If, the failure was a node failure, the node will receive further notifications saying that another node Y has lost connectivity to node Z through another link M. That is, if node Z is common in the notifications, the recipient can conclude that it is a node failure and already knows which node it is (Z). If link L is common in the notifications, then the recipient can decide for link failure (L). If further inconclusive notifications arrive, then it means multiple failures which case is not in scope for IPFRR, and is left for regular IGP convergence.

After concluding about the exact failure, the data plane element needs to check in its pre-installed IPFRR database whether this particular failure results in any route changes. If yes, the linecard replaces the next-hops impacted by that failure with their failure specific backups which were pre-installed in the preparation phase.

In the second option, the first received notification is handled immediately as a link failure, hence the router may start replacing its next-hops. In many cases this is a good decision. If, however, another notification arrives a couple of milliseconds later that points to a node failure, the router then needs to start replacing its next-hops again. This may cause a route flap but due to the quick dissemination mechanism the routing inconsistency is very short lived and likely takes only a couple of milliseconds.

This draft recommends that out of the several FN delivery options defined in [fn-transport], the Redundant Tree transport option is preferred, which ensures that any event can reach each node from any source with any single link or node failure present in the network area as long as theoretically possible. This also means that any node, when activating failure specific backup entries in its FIB, may assume that other nodes have been notified as well and have changed their FIBs to present consistent routing. The exception is the case of legacy nodes, see Section 5.2. for details.

5. Operation Details

5.1. Message Handling and Encoding

A failure identifier is needed that unambiguously describes the failed resource consistently among the nodes in the area. The schemantics of the identifiers are defined by the IGP used to pre-

calculate and pre-install the backup forwarding entries, e.g. OSPF or ISIS.

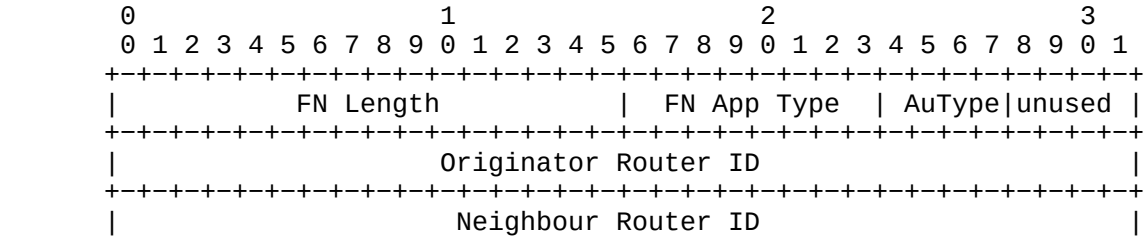
This draft defines a Failure Identification message class. Members of this class represent a routing protocol specific Failure Identification message to be carried with the Fast Notification transport protocol. Each message within the Failure Identification message class shall contain the following fields, the lengths of which are routing protocol specific. The exact values shall be aligned with the WG of the routing protocol:

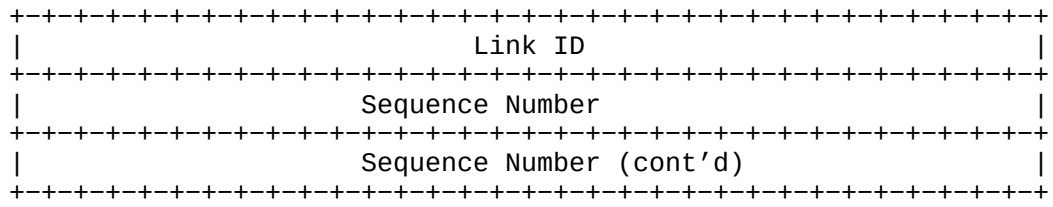
- o Originator Router ID: the identifier of the router advertising the failure;
- o Neighbour Router ID: the identifier of the neighbour node to which the originator lost connectivity.
- o Link ID: the identifier of the link, through which connectivity was lost to the neighbour. The routing protocol should assign the same Link ID for bidirectional, broadcast or multi access links from each access point, consistently.
- o Sequence Number: [fn-transport] expects the applications of the FN service that require replay attack protection to create and verify a sequence number in FN messages.

Routers forwarding the FN packets should ensure that Failure Identification messages are not lost, e.g. due to congestion. FN packets can be put a high precedence traffic class (e.g. Network Control). If the network environment is known to be lossy, the FN sender should repeat the same notification a couple of times, like a salvo fire.

After the forwarding element processed the FN packet and extracted the Failure Identification message, it should decide what backups need to be activated if at all - as described in Section 4.2.1.

5.1.1. Failure Identification Message for OSPF





FN Header fields:

FN Length
The length of the Failure Identification message for OSPF is 16 bytes.

FN App Type
The exact values are to be assigned by IANA for the Failure Identification message class. For example, FN App Type values between 0x0008 and 0x000F could represent Failure Identification messages, from which 0x0008 could mean OSPF, 0x0009 could be ISIS.

AuType
IPFRR-FN relies on the authentication options offered the FN transport service. Cryptographic authentication is recommended.

Originator Router ID
If the routing protocol is OSPF, then the value can take the OSPF Router ID of the advertising router.

Neighbour Router ID
The OSPF Router ID of the neighbour router to which connectivity was lost.

Link ID
If the link is a LAN, the Link ID takes the LSAID of its representing Network LSA.
If the link is a point-to-point link, the Link ID can take the minimum or the maximum of the two interface IDs. The requirement is that it is performed consistently.

Sequence Number
This field stores a digest of the LSDB of the routing protocol, as described in Section 6.

5.1.2. Failure Identification TLV for ISIS

TBA.

5.2. Bypassing Legacy Nodes

Legacy nodes, while cannot originate fast notifications and cannot process them either, can be assumed to be able to forward the notifications. As [fn-transport] discusses, FN forwarding is based on multicast. It is safe to assume that legacy routers' multicast configuration can be set up statically so as to be able to propagate fast notifications as needed.

When calculating failure specific alternative routes, IPFRR-FN capable nodes must consider legacy nodes as being fixed directed links since legacy nodes do not change packet forwarding in the case of failure. There are situations when an FN-IPFRR capable node can, exceptionally, bypass a non-IPFRR-FN capable node in order to handle a remote failure.

As an example consider the topology depicted in Figure 2, where the link between C and D fails. C cannot locally repair the failure.

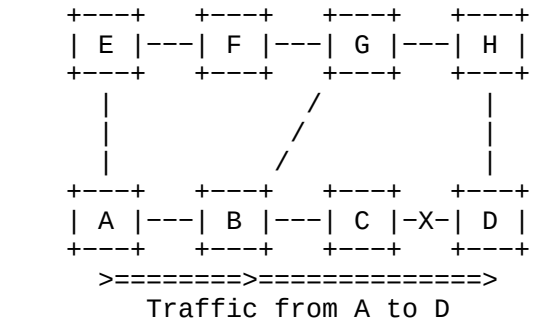


Figure 2 Example for bypassing legacy nodes

First, let us assume that each node is IPFRR-FN capable. C would advertise the failure information using FN. Each node learns that the link between C and D fails, as a result of which C changes its forwarding table to send any traffic destined to D via B. B also makes a change, replacing its default next-hop (C) with G. Note that other nodes do not need to modify their forwarding at all.

Now, let us assume that B is a legacy router not supporting IPFRR-FN but it is statically configured to multicast fast notifications as needed. As such, A will receive the notification. A's pre-

calculations have been done knowing that B is unable to correct the failure. Node A, therefore, has pre-calculated E as the failure specific next-hop. Traffic entering at A and heading to D can thus be repaired.

5.3. Capability Advertisement

The solution requires nodes to know which other nodes in the area are capable of IPFRR-FN. The most straightforward way to achieve this is to rely on the Router Capability TLVs available both in OSPF [RFC4970] and in IS-IS [RFC4971].

6. Protection against Replay Attacks

To defend against replay attacks, recipients should be able to ignore a re-sent recording of a previously sent FN packet. This suggests that some sort of sequence number should be included in the FN packet, the verification of which should not need control plane involvement. Since the solution should be simple to implement in the dataplane, maintaining and verifying per-source sequence numbers is not the best option.

We propose, therefore, that messages should be stamped with the digest of the actual routing configuration, i.e., a digest of the link state database of the link state routing protocol. The digest has to be picked carefully, so that if two LSDBs describe the same connectivity information, their digest should be identical as well, and different LSDBs should result in different digest values with high probability.

The conceptual way of handling these digests could be the following:

- When the LSDB changes, the IGP re-calculates the digest and downloads the new value to the dataplane element(s), in a secure way.
- When a FN packet is originated, the digest is put into the FN message into the Sequence Number field.
- Network nodes distribute (forward) the FN packet.
- When processing, the dataplane element first performs an authentication check of the FN packet, as described in [fn-transport].

- Finally, before processing the failure notification, the dataplane element should check whether its own known LSDB digest is identical with the one in the message.

If due to a failure event a node disseminates a failure notification with FN, an attacker might capture the whole packet and re-send it later. If it resends the packet after the IGP re-converged on the new topology, the active LSDB digest is different, so the packet can be ignored. If the packet is replayed to a recipient who still has the same LSDB digest, then it means that the original failure notification was already processed but the IGP has not yet finished converging; the IPFRR detour is already active, the replica has no impact.

6.1. Calculating LSDB Digest

We propose to create an LSDB digest that is conceptually similar to [ISISDigest]. The operation is proposed to be the following:

- Create a hash from each LSA(OSPF)/LSP(ISIS) one by one
- XOR these hashes together
- When an LSA/LSP is removed, the new LSDB digest is received by computing the hash of the removed LSA, and then XOR to the existing digest
- When an LSA/LSP is added, the new LSDB digest is received by computing the hash of the new LSA, and then XOR to the existing digest

7. Security Considerations

The IPFRR application of Fast Notification does not raise further known security consideration in addition to those already present in Fast Notification itself. If an attacker could send false Failure Identification Messages or could hinder the transmission of legal messages, then the network would produce an undesired routing behavior. These issues should be solved, however, in [fn-transport].

IPFRR-FN relies on the authentication mechanism provided by the Fast Notification transport protocol [fn-transport]. The specification of the FN transport protocol requires applications to protect against replay attacks with application specific sequence numbers. This draft, therefore, describes its own proposed sequence number in Section 6.

8. IANA Considerations

The Failure Identification message types need to be allocated a value in the FN App Type field.

IPFRR-FN capability needs to be allocated within Router Capability TLVs both for OSPF [RFC4970] and in IS-IS [RFC4971].

9. References

9.1. Normative References

[RFC5286] A. Atlas, A. Zinin, "Basic specification for IP Fast-Reroute: Loop-Free Alternates", Internet Engineering Task Force: RFC 5286, 2008.

[fn-transport] W. Lu, S. Kini, A. Csaszar, G. Enyedi, J. Tantsura, A. Tian, "Transport of Fast Notifications Messages", draft-lu-fn-transport-00, 2011

[RFC4970] A. Lindem et al., Extensions to OSPF for Advertising Optional Router Capabilities, RFC 4970, 2007

[RFC4971] JP. Vasseur et al., Intermediate System to Intermediate System (IS-IS) Extensions for Advertising Router Information, RFC 4971, 2007

9.2. Informative References

[BFD] D. Katz, D. Ward, "Bidirectional forwarding detection", RFC 5880, IETF, 2010

[RFC5714] M. Shand, S. Bryant, "IP Fast Reroute Framework", RFC 5714, IETF, 2010.

[Eny2009a]Gabor Enyedi, Gabor Retvari, Andras Csaszar, "On Finding Maximally Redundant Trees in Strictly Linear Time", IEEE Symposium on Computers and Communications (ISCC), 2009.

[Eny2009b] Gabor Enyedi, Peter Szilagyi, Gabor Retvari, Andras Csaszar, "IP Fast ReRoute: Lightweight Not-Via without Additional Addresses", IEEE INFOCOM-MiniConference, Rio de Janeiro, Brazil, 2009.

[FIFR] J. Wand, S. Nelakuditi, "IP fast reroute with failure inferencing", In Proceedings of ACM SIGCOMM Workshop on Internet Network Management - The Five-Nines Workshop, 2007.

[MRC] T. Cicic, A. F. Hansen, A. Kvalbein, M. Hartmann, R. Martin, M. Menth, S. Gjessing, O. Lysne, "Relaxed multiple routing configurations IP fast reroute for single and correlated failures", IEEE Transactions on Network and Service Management, available online: http://www3.informatik.uni-wuerzburg.de/staff/menth/Publications/papers/Menth08-Sub-4.pdf, September 2010.

[NotVia] S. Bryant, M. Shand, S. Previdi, "IP fast reroute using Not-via addresses", Internet Draft, draft-ietf-rtgwg-ipfrr-notvia-addresses-06, 2010.

[RLFAP] I. Hokelek, M. Fecko, P. Gurung, S. Samtani, S. Cevher, J. Sucec, "Loop-Free IP Fast Reroute Using Local and Remote LFAPs", Internet Draft, draft-hokelek-rlfap-01 (expired), 2008.

[Ret2011] G. Retvari, J. Tapolcai, G. Enyedi, A. Csaszar, "IP Fast ReRoute: Loop Free Alternates Revisited", to appear at IEEE INFOCOM 2011

[ISISDigest] J. Chiabaut and D. Fedyk. IS-IS Multicast Synchronization Digest. Available online: http://www.ieee802.org/1/files/public/docs2008/aq-fedyk-ISIS-digest-1108-v1.pdf, Nov 2008.

10. Acknowledgments

The authors would like to thank Jeff Tantsura, Albert Tian, Wenhu Lu and Acee Lindem for the continuous discussions and comments on the topic, as well as Joel Halpern for his comments and review.

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

Appendix A. Memory needs of a Naive Implementation

Practical background might suggest that storing and maintaining backup next-hops for many potential remote failures could overwhelm the resources of router linecards. This section attempts to provide a calculation describing the approximate memory needs in reasonable sized networks with a possible implementation.

A.1. An Example Implementation

Let us suppose that the forwarding engine is optimized for forwarding performance in the sense that recursive lookups are not performed for external destinations but each IP lookup gives back an adjacency (a number describing the next hop for the router), even if the packet will be terminated outside the current area. From the aspect of storing backup next-hops per destination, this is worse than using recursive lookup, since in this case the update of a lot more destinations is needed. In case of recursive lookup, external prefixes are resolved to internal destinations, so we can simply choose not to deal with external prefixes.

This implementation uses an array for all the nodes in the area (node array in the sequel), made up by two pointers per record. Both of these pointers point to another array with a header describing its lengths. The first array (called alternative array) is basically an enumeration containing the IDs of those failures influencing a shortest path towards that node and an alternative neighbor, which can be used, when such a failure occurs. When a failure is detected, (either locally, or by FN), we can easily find the proper record in all the lists. Moreover, since these arrays can be sorted based on the failure ID, we can even use binary search to find the needed record.

Now, we only need to know, which records in the FIB should be updated. Therefore there is a second pointer in the node array pointing to another enumeration (called FIB array in the sequel) containing pointers to the corresponding FIB entries. Recall, that if the node is an egress router, FIB array contains more than one entry. Moreover, there can be some prefixes reachable through more than one egres routers, thus these entries may be in more than one FIB arrays.

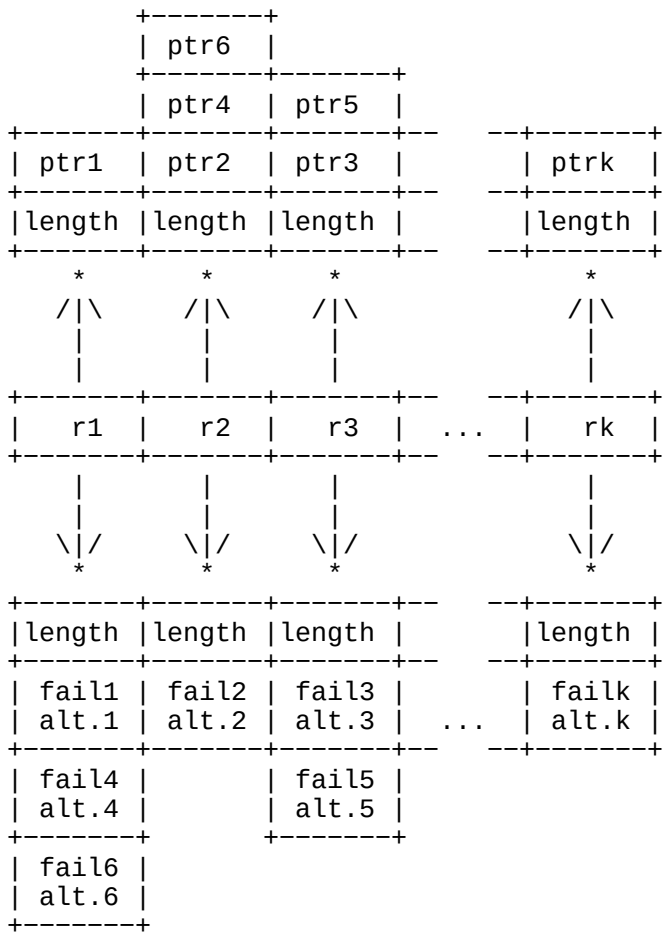


Figure 3 The way of storing alternatives

A.2. Estimation of Memory Requirements.

Now, suppose that there are D prefixes, the area containing the router has V routers, a prefix is connected to K egress routers in average, a neighbor descriptor takes X bytes, a failure ID takes Y bytes and a pointer takes Z bytes. In this way, if there is no ECMP, this data structure takes

$$2*Z*(V-1) + (2*(X+Y)*(V-1)+Y)*(V-1) + (K*D*Z+Y*(V-1))$$

bytes altogether. The first part is the memory consumption of the node array. The memory consumption of all the FIB arrays is described

by the last part (V-1 length fields and K*D pointers). The remaining part describes the maximum memory needed by an alternative arrays: any path can contain at most V-1 nodes and V-1 links, each record needs X+Y bytes plus we have a header for the array; there are records for all the other nodes in the area (V-1 nodes). Observe that this is a very rough overestimation, since most of the possible failures influencing the path will not change the next hop.

For computing memory consumption, suppose that neighbor descriptors, failure IDs and pointers take 4 bytes, there are 200 (500) nodes in the area and we have 500K prefixes installed, and a prefix is reachable through 2 egress routers in average. In this case, we get that the node array needs about 1.6KB (4KB), the alternative array needs about 620KB (4MB), and the FIB array needs about 4MB (4MB). That is altogether less than 5MB (8MB) in reality, if there is no ECMP.

If however, there are paths with equal costs, the size of the alternative array increases. Suppose that there are 10 equal paths between ANY two nodes in the network. This would cause that the alternative list gets 10 times bigger, and now it needs 6.2MB. Observe that now we need about 11MB (44MB) even in this extremely unrealistic case, which is likely acceptable for modern linecards with gigs of DRAM. Moreover, we need to stress here again that this is an extremely rough overestimation, so in reality much less memory will be enough.

Authors' Addresses

Andras Csaszar
Ericsson
Irinyi utca 4-10, Budapest, Hungary, 1117
Email: Andras.Csaszar@ericsson.com

Gabor Sandor Enyedi
Ericsson
Irinyi utca 4-10, Budapest, Hungary, 1117
Email: Gabor.Sandor.Enyedi@ericsson.com

Sriganesh Kini
Ericsson
300 Holger Way, San Jose, CA 95134
Email: sriganesh.kini@ericsson.com