

Routing Area Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 13, 2014

A. Atlas, Ed.
R. Kebler
Juniper Networks
IJ. Wijnands
Cisco Systems, Inc.
A. Csaszar
G. Enyedi
Ericsson
July 12, 2013

An Architecture for Multicast Protection Using Maximally Redundant Trees
draft-atlas-rtgw-mrt-mc-arch-02

Abstract

Failure protection is desirable for multicast traffic, whether signaled via PIM or mLDP. Different mechanisms are suitable for different use-cases and deployment scenarios. This document describes the architecture for global protection (aka multicast live-live) and for local protection (aka fast-reroute).

The general methods for global protection and local protection using alternate-trees are dependent upon the use of Maximally Redundant Trees. Local protection can also tunnel traffic in unicast tunnels to take advantage of the routing and fast-reroute mechanisms available for IP/LDP unicast destinations.

The failures protected against are single link or node failures. While the basic architecture might support protection against shared risk group failures, algorithms to dynamically compute MRTs supporting this are for future study.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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."

This Internet-Draft will expire on January 13, 2014.

Copyright Notice

Copyright (c) 2013 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 Simplified BSD License.

Table of Contents

| | |
|--|----|
| 1. Introduction | 4 |
| 1.1. Maximally Redundant Trees (MRTs) | 4 |
| 1.2. MRTs and Multicast | 6 |
| 2. Terminology | 6 |
| 3. Use-Cases and Applicability | 8 |
| 4. Global Protection: Multicast Live-Live | 9 |
| 4.1. Creation of MRMTs | 10 |
| 4.2. Traffic Self-Identification | 11 |
| 4.2.1. Merging MRMTs for PIM if Traffic Doesn't Self-Identify | 12 |
| 4.3. Convergence Behavior | 13 |
| 4.4. Inter-area/level Behavior | 14 |
| 4.4.1. Inter-area Node Protection with 2 border routers . . . | 15 |
| 4.4.2. Inter-area Node Protection with > 2 Border Routers . . | 16 |
| 4.5. PIM | 17 |
| 4.5.1. Traffic Handling: RPF Checks | 17 |
| 4.6. mLDP | 17 |
| 5. Local Repair: Fast-Reroute | 17 |
| 5.1. PLR-driven Unicast Tunnels | 18 |
| 5.1.1. Learning the MPs | 19 |
| 5.1.2. Using Unicast Tunnels and Indirection | 19 |
| 5.1.3. MP Alternate Traffic Handling | 20 |
| 5.1.4. Merge Point Reconvergence | 21 |
| 5.1.5. PLR termination of alternate traffic | 21 |
| 5.2. MP-driven Unicast Tunnels | 21 |
| 5.3. MP-driven Alternate Trees | 22 |
| 6. Acknowledgements | 23 |
| 7. IANA Considerations | 23 |
| 8. Security Considerations | 23 |
| 9. Appendix A | 23 |
| 9.1. MP-driven Alternate Trees | 23 |
| 9.1.1. PIM details for Alternate-Trees | 26 |
| 9.1.2. mLDP details for Alternate-Trees | 26 |
| 9.1.3. Traffic Handling by PLR | 26 |
| 9.2. Methods Compared for PIM | 27 |
| 9.3. Methods Compared for mLDP | 27 |
| 10. References | 27 |
| 10.1. Normative References | 27 |
| 10.2. Informative References | 28 |
| Authors' Addresses | 29 |

1. Introduction

This document describes how the algorithms in [I-D.enyedi-rtgwg-mrt-frr-algorithm], which are used in [I-D.ietf-rtgwg-mrt-frr-architecture] for unicast IP/LDP fast-reroute, can be used to provide protection for multicast traffic. It specifically applies to multicast state signaled by PIM[RFC4601] or mLDP[RFC6388]. There are additional protocols that depend upon these (e.g. VPLS, mVPN, etc.) and consideration of the applicability to such traffic will be in a future version.

In this document, global protection is used to refer to the method of having two maximally disjoint multicast trees where traffic may be sent on both and resolved by the receiver. This is similar to the ability with RSVP-TE LSPs to have a primary and a hot standby, except that it can operate in 1+1 mode. This capability is also referred to as multicast live-live and is a generalized form of that discussed in [I-D.ietf-rtgwg-mofrr]. In this document, local protection refers to the method of having alternate ways of reaching the pre-identified merge points upon detection of a local failure. This capability is also referred to as fast-reroute.

This document describes the general architecture, framework, and trade-offs of the different approaches to solving these general problems. It will recommend how to generally provide global protection and local protection for mLDP and PIM traffic. Where protocol extensions are necessary, they will be defined in separate documents as follows.

- o Global 1+1 Protection Using PIM
- o Global 1+1 Protection Using mLDP
- o Local Protection Using mLDP:
[I-D.wijnands-mpls-mldp-node-protection] This document describes how to provide node-protection and the necessary extensions using targeted LDP session.
- o Local Protection Using PIM

1.1. Maximally Redundant Trees (MRTs)

Maximally Redundant Trees (MRTs) are described in [I-D.enyedi-rtgwg-mrt-frr-algorithm]; here we only give a brief description about the concept. A pair of MRTs is a pair of directed spanning trees (red and blue tree) with a common root, directed so that each node can be reached from the root on both trees. Moreover, these trees are redundant, since they are constructed so that no

single link or single node failure can separate any node from the root on both trees, unless that failed link or node is splitting the network into completely separated components (e.g. the link or node was a cut-edge or cut-vertex).

Although for multicast, the arcs (directed links) are directed away from the root instead of towards the root, the same MRT computations are used and apply. This is similar to how multicast uses unicast routing's next-hops as the upstream-hops. Thus this definition slightly differs from the one presented in [I-D.enyedi-rtgwg-mrt-frr-algorithm], since the arcs are directed away and not towards the root. When we need two paths towards a given destination and not two away from it (e.g. for unicast detours for local repair solutions), we only need to reverse the arcs from how they are used for the unicast routing case; thus constructing MRTs towards or away from the root is the same problem. A pair of MRTs is depicted in Figure 1.

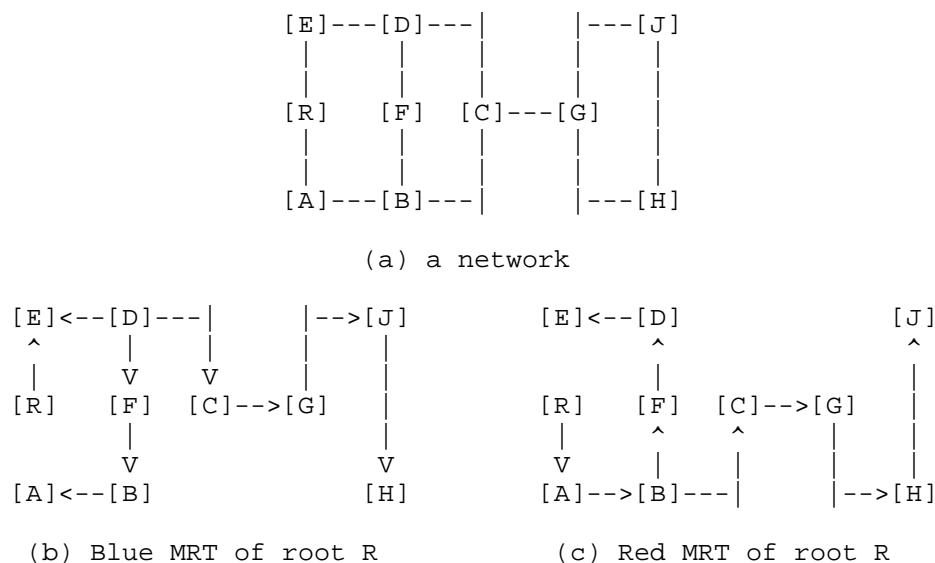


Figure 1: A network and two MRTs found in it

It is important to realize that this redundancy criterion does not imply that, after a failure, either of the MRTs remains intact, since a node failure must affect any spanning tree. Redundancy here means that there will be a set of nodes, which can be reached along the blue MRT, and there will be another set, which remains reachable along the red MRT. As an example, suppose that node F goes down; that would separate B and A on the blue MRT and D and E on the red

MRT. Naturally, it is possible that the intersection of these two sets is not empty, e.g. C, G, H and J will remain reachable on both MRTs. Additionally, observe that a single link can be used in both of the trees in different directions, so even a link failure can cut both trees. In this example, the failure of link $F \leftrightarrow B$ leads to the same reachability sets.

Finally, it is critical to recall that a pair of MRTs is always constructed together and they are not SPTs. While it would be useful to have an algorithm that could find a redundant pair for a given tree (e.g. for the SPT), that is impossible in general. Moreover, if there is a failure and at least one of the trees change, the other tree may need to change as well. Therefore, even if a node still receives the traffic along the red tree, it cannot keep the old red tree, and construct a blue pair for it; there can be reconfiguration in cases when traditional shortest-path-based-thinking would not expect it. To converge to a new pair of disjoint MRTs, it is generally necessary to update both the blue MRT and the red MRT.

The two MRTs provide two separate forwarding topologies that can be used in addition to the default shortest-path-tree (SPT) forwarding topology (usually MT-ID 0). There is a Blue MRT forwarding topology represented by one MT-ID; similarly there is a Red MRT forwarding topology represented by a different MT-ID. Naturally, a multicast protocol is required to use the forwarding topologies information to build the desired multicast trees. The multicast protocol can simply request appropriate upstream interfaces, but include the MT-ID when needed.

1.2. MRTs and Multicast

Maximally Redundant Trees (MRT) provide two advantages for protecting multicast traffic. First, for global protection, MRTs are precisely what needs to be computed to have maximally redundant multicast distribution trees. Second, for local repair, MRTs ensure that there will protection to the merge points; the certainty of a path from any merge point to the PLR that avoids the failure node allows for the creation of alternate trees.

A known disadvantage of MRT, and redundant trees in general, is that the trees do not necessarily provide shortest detour paths. Modeling is underway to investigate and compare the MRT lengths for the different algorithm options [I-D.enyedi-rtgwg-mrt-frr-algorithm].

2. Terminology

2-connected: A graph that has no cut-vertices. This is a graph that requires two nodes to be removed before the network is partitioned.

2-connected cluster: A maximal set of nodes that are 2-connected.

2-edge-connected: A network graph where at least two links must be removed to partition the network.

ADAG: Almost Directed Acyclic Graph - a graph that, if all links incoming to the root were removed, would be a DAG.

block: Either a 2-connected cluster, a cut-edge, or an isolated vertex.

cut-link: A link whose removal partitions the network. A cut-link by definition must be connected between two cut-vertices. If there are multiple parallel links, then they are referred to as cut-links in this document if removing the set of parallel links would partition the network.

cut-vertex: A vertex whose removal partitions the network.

DAG: Directed Acyclic Graph - a graph where all links are directed and there are no cycles in it.

GADAG: Generalized ADAG - a graph that is the combination of the ADAGs of all blocks.

Maximally Redundant Trees (MRT): A pair of trees where the path from any node X to the root R along the first tree and the path from the same node X to the root along the second tree share the minimum number of nodes and the minimum number of links. Each such shared node is a cut-vertex. Any shared links are cut-links. Any RT is an MRT but many MRTs are not RTs.

Maximally Redundant Multicast Trees (MRMT): A pair of multicast trees built of the sub-set of MRTs that is needed to reach all interested receivers.

network graph: A graph that reflects the network topology where all links connect exactly two nodes and broadcast links have been transformed into the standard pseudo-node representation.

Redundant Trees (RT): A pair of trees where the path from any node X to the root R along the first tree is node-disjoint with the path from the same node X to the root along the second tree. These can be computed in 2-connected graphs.

Merge Point (MP): For local repair, a router at which the alternate traffic rejoins the primary multicast tree. For global protection, a router which receives traffic on multiple trees and must decide which stream to forward on.

Point of Local Repair (PLR): The router that detects a local failure and decides whether and when to forward traffic on appropriate alternates.

MT-ID: Multi-topology identifier. The default shortest-path-tree topology is MT-ID 0.

MultiCast Ingress (MCI): Multicast Ingress, the node where the multicast stream enters the current transport technology (MPLS-mLDP or IP-PIM) domain. This maybe the router attached to the multicast source, the PIM Rendezvous Point (RP) or the mLDP Root node address.

Upstream Multicast Hop (UMH): Upstream Multicast Hop, a candidate next-hop that can be used to reach the MCI of the tree.

Stream Selection: The process by which a router determines which of the multiple primary multicast streams to accept and forward. The router can decide on a packet-by-packet basis or simply per-stream. This is done for global protection 1+1 and described in [I-D.ietf-rtgwg-mofrr].

MultiCast Egress (MCE): Multicast Egress, a node where the multicast stream exists the current transport technology (MPLS-mLDP or IP-PIM) domain. This is usually a receiving router that may forward the multicast traffic on towards receivers based upon IGMP or other technology.

3. Use-Cases and Applicability

Protection of multicast streams has gained importance with the use of multicast to distribute video, including live video such as IP-TV. There are a number of different scenarios and uses of multicast that require protection. A few preliminary examples are described below.

- o When video is distributed via IP or MPLS for a cable application, it is desirable to have global protection 1+1 so that the customer-perceived impact is limited. A QAM can join two multicast groups and determine which stream to use based upon the stream quality. A network implementing this may be custom-engineered for this particular purpose.

- o In financial markets, stock ticker data is distributed via multicast. The loss of data can have a significant financial impact. Depending on the network, either global protection 1+1 or local protection can minimize the impact.
- o Several solutions exist for updating software or firmwares of a large number of end-user or operator-owned networking equipment that are based on IP multicast. Since IP multicast is based on datagram transport so taking care of lost data is cumbersome and decreases the advantages offered by multicast. Solutions may rely on sending the updates several times: a properly protected network may result in that less repetitions are required. Other solutions rely on the recipient asking for lost data segments explicitly on-demand. A network failure could cause data loss for a significant number of receivers, which in turn would start requesting the the lost data in a burst that could overload the server. Properly engineered multicast fast reroute would minimise such impacts.
- o Some providers offer multicast VPN services to their customers. SLAs between the customer and provider may set low packet loss requirements. In such cases interruptions longer than the outage timescales targeted by FRR could cause direct financial losses for the provider.

Global protection 1+1 uses maximally redundant multicast trees (MRMTs) to simultaneously distribute a multicast stream on both MRMTs. The disadvantage is the extra state and bandwidth requirements of always sending the traffic twice. The advantage is that the latency of each MRMT can be known and the receiver can select the best stream.

Local protection provides a patch around the fault while the multicast tree reconverges. When PLR replication is used, there is no extra multicast state in the network, but the bandwidth requirements vary based upon how many potential merge-points must be provided. When alternate-trees are used, there is extra multicast state but the bandwidth requirements on a link can be minimized to no more than once for the primary multicast tree traffic and once for the alternate-tree traffic.

4. Global Protection: Multicast Live-Live

In MoFRR [I-D.ietf-rtgwg-mofrr], the idea of joining both a primary and a secondary tree is introduced with the requirement that the primary and secondary trees be link and node disjoint. This works well for networks where there are dual-planes, as explained in [I-D.ietf-rtgwg-mofrr]. For other networks, it is still desirable to

have two disjoint multicast trees and allow a receiver to join both and make its own decision about which traffic to accept.

Using MRTs gives the ability to guarantee that the two trees are as disjoint as possible and dynamically recomputed whenever the topology changes. The MRTs used are rooted at the MultiCast Ingress (MCI). One multicast tree is created using the Blue MRT forwarding topology. The second multicast tree is created using the Red MRT forwarding topology. This can be accomplished by specifying the appropriate MT-ID associated with each forwarding topology.

There are four different aspects of using MRTs for 1+1 Global Protection that are necessary to consider. They are as follows.

1. Creation of the maximally redundant multicast trees (MRMTs) based upon the forwarding topologies.
2. Traffic Identification: How to handle traffic when the two MRMTs overlap due to a cut-vertex or cut-link.
3. Convergence: How to converge after a network change and get back to a protected state.
4. Inter-area/inter-level Behavior: How to compute and use MRMTs when the multicast source is outside the area/level and how to provide border-router protection.

4.1. Creation of MRMTs

The creation of the two maximally redundant multicast trees occurs as described below. This assumes that the next-hops to the MCI associated with the Blue and Red forwarding topologies have already been computed and stored.

1. A receiving router determines that it wants to join both the Blue tree and the Red tree. The details on how it does this decision are not covered in this document and could be based on configuration, additional protocols, etc.
2. The router selects among the Blue next-hops an Upstream Multicast Hop (UMH) to reach the MCI node. The router joins the tree towards the selected UMH including a multi-topology id (MT-ID) identifying the Blue MRT.
3. The router selects among the Red next-hops an Upstream Multicast Hop (UMH) to reach the MCI node. The router joins the tree towards the selected UMH including a multi-topology id (MT-ID) identifying the Red MRT.

4. When a router receives a tree setup request specifying a particular MT-ID (e.g. Color), then the router selects among the Color next-hops to the MCI a UMH node, creates the necessary multicast state, and joins the tree towards the UMH node.

4.2. Traffic Self-Identification

Two maximally redundant trees will share any cut-vertices and cut-links in the network. In the multicast global protection 1+1 case, this means that the potential single failures of the other nodes and links in the network are still protected against. If each cut-vertex cannot associate traffic to a particular MRMT, then the traffic would be incorrectly replicated to both MRMT resulting in complete duplication of traffic. An example of such MRTs is given earlier in Figure 1 and repeated below in Figure 2, where there are two cut-vertices C and G and a cut-link C<->G.

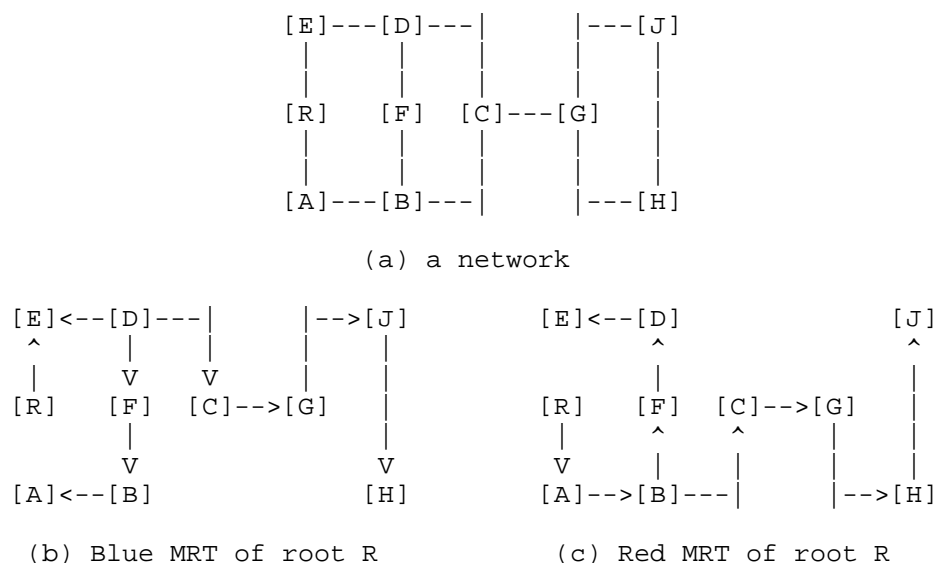


Figure 2: A network and two MRTs found in it

In this example, traffic from the multicast source R to a receiver G, J, or H will cross link C<->G on both the Blue and Red MRMTs. When this occurs, there are several different possibilities depending upon protocol.

mLDP: Different label bindings will be created for the Blue and Red MRMTs. As specified in [I-D.iwijnand-mpls-mldp-multi-topology], the P2MP FEC Element will use the MT IP Address Family to encode the Root node address and MRT T-ID. Each MRMT will therefore have a different P2MP FEC Element and be assigned an independent label.

PIM: There are three different ways to handle IP traffic forwarded based upon PIM when that traffic will overlap on a link.

- A. Different Groups: If different multicast groups are used for each MRMT, then the traffic clearly indicates which MRMT it belongs to. In this case, traffic on the Blue MRMT would use multicast group G-blue and traffic on the Red MRMT would use multicast group G-red.
- B. Different Source Loopbacks: Another option is to use different IP addresses for the source S, so S might announce S-red and S-blue. In this case, traffic on the Blue MRMT would have an IP source of S-blue and traffic on the Red MRMT would have an IP source of S-red.
- C. Stream Selection and Merging: The third option, described in Section 4.2.1, is to have a router that gets (S,G) Joins for both the Blue MT-ID and the Red MT-ID merge those into a single tree. The router may need to select which upstream stream to use, just as if it were a receiving router.

There are three options presented for PIM. The most appropriate will depend upon deployment scenario as well as router capabilities.

4.2.1. Merging MRMTs for PIM if Traffic Doesn't Self-Identify

When traffic doesn't self-identify, the cut-vertices must follow specific rules to avoid traffic duplication. This section describes that behavior which allows the same (S,G) to be used for both the Blue MT-ID and Red MT-ID (e.g. when the traffic doesn't self-identify as to its MT-ID).

The behavior described in this section differs from the conflict resolution described in [RFC6420] because these rules apply to the Global Protection 1+1 case. Specifically, it is not sufficient for a upstream router to pick only one of the two MT-IDs to join because that does not maximize the protection provided.

As described in [RFC6420], a router that receives (S,G) Joins for both the Blue MT-ID and the Red MT-ID can merge the set of downstream interfaces in its forwarding entry. Unlike the procedures defined in [RFC6420], the router must send a Join upstream for each MT-ID. If a

router has different upstream interfaces for these MRMTs, then the router will need to do stream selection and forward the selected stream to its outgoing interfaces, just as if it were an MCE. The stream selection methods of detecting failures and handle traffic discarding are described in [I-D.ietf-rtgwg-mofrr].

This method does not work if the MRMTs merge on a common LAN with different upstream routers. In this case, the traffic cannot be distinguished on the LAN and will result in duplication on the LAN. The normal PIM Assert procedure would stop one of the upstream routers from transmitting duplicates onto the LAN once it is detected. This, in turn, may cause the duplicate stream to be pruned back to the source. Thus, end-to-end protection in this case of the MRMTs converging on a single LAN with different upstream interfaces can only be accomplished by the methods of traffic self-identification.

4.3. Convergence Behavior

It is necessary to handle topology changes and get back to having two MRMTs that provide global protection. To understand the requirements and what can be computed, recall the following facts.

- a. It is not generally possible to compute a single tree that is maximally redundant to an existing tree.
- b. The pair of MRTs must be computed simultaneously.
- c. After a single link or node failure, there is one set of nodes that can be reached from the root on the Blue MRMT and a second set of nodes that can be reached from the root on the Red MRMT. If the failure wasn't a cut-vertex or cut-edge, all nodes will be in at least one of these two sets.

To gracefully converge, it is necessary to never have a router where both its red MRMT and blue MRMT are broken. There are three different ways in which this could be done. These options are being more fully explored to see which is most practical and provides the best set of trade-offs.

Ordered Convergence When a single failure occurs, each receiver determines whether it was affected or unaffected. First, the affected receivers identify the broken MRMT color (e.g. blue) and join the MRMT via their new UMH for that MRT color. Once the affected receivers receive confirmation that the new MRMT has been successfully created back to the MCI, then the affected receivers switch to using that MRMT. The affected receivers tear down the old broken MRMT state and join the MRMT via their new UMH for the

other MRT color (e.g. red). Finally, once the affected receivers receive confirmation that the new MRMT has been successfully created back to the MCI, the affected receivers can tear down the old working MRMT state. Once the affected receivers have updated their state, the unaffected receivers need to also do the same staging - first joining the MRMT via their new UMH for the Blue MRT, waiting for confirmation, switching to using traffic from the Blue MRMT, tearing down the old Blue MRMT state, joining the MRMT via their new UMH for the Red MRT, waiting for confirmation, and tearing down the old Red MRMT state. There are complexities remaining, such as determining how an Unaffected Receiver decides that the Affected Receivers are done. When the topology change isn't a failure, all receivers are unaffected and the same process can apply.

Protocol Make-Before-Break In the control plane, a router joins the tree on the new Blue topology but does not stop receiving traffic on the old Blue topology. Once traffic is observed from the new Blue UMH, then the router accepts traffic on the new Blue UMH and removes the old Blue UMH. This behavior can happen simultaneously with both Blue and Red forwarding topologies. An advantage is that it works regardless of the type of topology change and existing traffic streams aren't broken. Another advantage is that the complexity is limited and this method is well understood. The disadvantage is that the number of traffic-affecting events depends upon the number of hops to the MCI.

Multicast Source Make-Before-Break On a topology change, routers would create new MRMTs using new MRT forwarding state and leaving the old MRMTs as they are. After the new MRMTs are complete, the multicast source could switch from sending on the old MRMTs to sending on the new MRMTs. After a time, the old MRMTs could be torn down. There are a number of details to still investigate.

4.4. Inter-area/level Behavior

A source outside of the IGP area/level can be treated as a proxy node. When the join request reaches a border router (whether ABR for OSPF or LBR for ISIS), that border router needs to determine whether to use the Blue or Red forwarding topology in the next selected area/level.

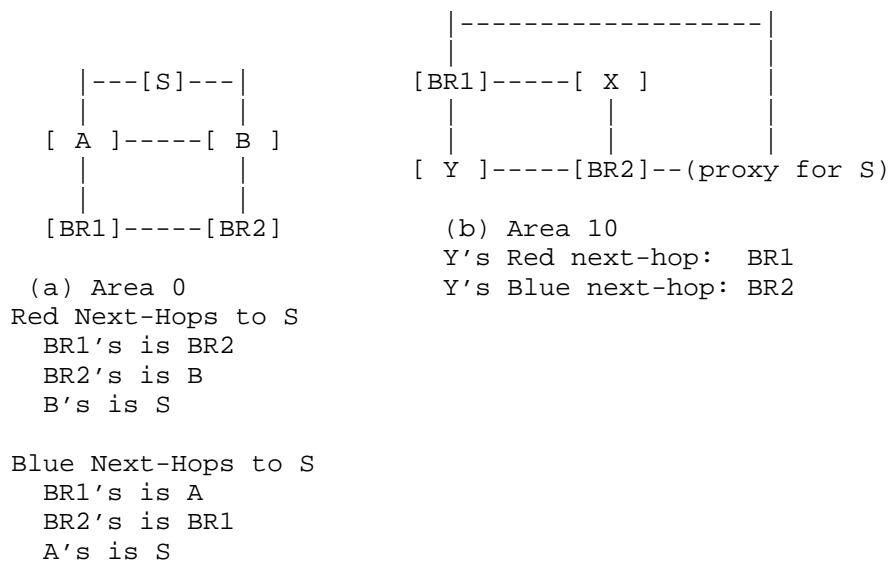


Figure 3: Inter-area Selection - next-hops towards S

Achieving maximally node-disjoint trees across multiple areas is hard due to the information-hiding and abstraction. If there is only one border router, it is trivial but protection of the border router is not possible. With exactly 2 border routers, inter-area/level node protection is reasonably straightforward but can require that the BR rewrite the (S,G) for PIM. With more than 2 border routers, inter-area node protection is possible at the cost of additional bandwidth and border router complexity. These two solutions are described in the following sub-sections.

4.4.1. Inter-area Node Protection with 2 border routers

If there are exactly two border routers between the areas, then the solution and necessary computation is straightforward. In that specific case, each BR knows that only the other BR must specifically be avoided in the second area when a forwarding topology is selected. As described in [I-D.enyedi-rtgwg-mrt-frr-algorithm], it is possible for a node X to determine whether the Red or Blue forwarding topology should be used to reach a node D while avoiding another node Y.

The results of this computation and the resulting changes in MT-ID from Red to Blue or Blue to Red are illustrated in Figure 3. It shows an example where BR1 must modify joins received from Area 10 for the Red MT-ID to use the Blue MT-ID in Area 0. Similarly, BR2 must modify joins received from Area 10 for the Blue MT-ID to use the

Red MT-ID in Area 0.

For mLDP, modifying the MT-ID in the control-plane is all that is needed. For PIM, if the same (S,G) is used for both the Blue MT-ID and the Red MT-ID, then only control-plane changes are needed. However, for PIM, if different group IDs (e.g. G-red and G-blue) or different source loopback addresses (S-red and S-blue) are used, it is necessary to modify the traffic to reflect the MT-ID included in the join message received on that interface. An alternative could be to use an MPLS label that indicates the MT-ID instead of different group IDs or source loopback addresses.

To summarize the necessary logic, when a BR1 receives a join from a neighbor in area N to a destination D in area M on the Color MT-ID, the BR1:

- a. Identifies the BR2 at the other end of the proxy node in area N.
- b. Determines which forwarding topology may avoid BR2 to reach D in area M. Refer to that as Color-2 MT-ID.
- c. Uses Color-2 MT-ID to determine the next-hops to S. When a join is sent upstream, the MT-ID used is that for Color-2.

4.4.2. Inter-area Node Protection with > 2 Border Routers

If there are more than two BRs between areas, then the problem of ensuring inter-area node-disjointness is not solved. Instead, once a request to join the multicast tree has been received by a BR from an area that isn't closest to the multicast source, the BR must join both the Red MT-ID and the Blue MT-ID in the area closest to the multicast source. Regardless of what single link or node failure happens, each BR will receive the multicast stream. Then, the BR can use the stream-selection techniques specified in [I-D.ietf-rtgwg-mofrr] to pick either the Blue or Red stream and forward it to downstream routers in the other area. Each of the BRs for the other area should be attached to a proxy-node representing the other area.

This approach ensures that a BR will receive the multicast stream in the closest area as long as the single link or node failure isn't a single point of failure. Thus, each area or level is independently protected. The BR is required to be able to select among the multicast streams and, if necessary for PIM, translate the traffic to contain the correct (S,G) for forwarding.

4.5. PIM

Capabilities need to be exchanged to determine that a neighbor supports using MRT forwarding topologies with PIM. Additional signaling extensions are not necessary to PIM to support Global Protection. [RFC6420] already defines how to specify an MT-ID as a Join Attribute.

4.5.1. Traffic Handling: RPF Checks

For PIM, RPF checks would still be enabled by the control plane. The control plane can program different forwarding entries on the G-blue incoming interface and on the G-red incoming interface. The other interfaces would still discard both G-blue and G-red traffic.

The receiver would still need to detect failures and handle traffic discarding as is specified in [I-D.ietf-rtgwg-mofrr].

4.6. mLDP

Capabilities need to be exchanged to determine that a neighbor supports using MRT forwarding topologies with mLDP. The basic mechanisms for mLDP to support multi-topology are already described in [I-D.iwijnand-mppls-mldp-multi-topology]. It may be desirable to extend the capability defined in this draft to indicate that MRT is or is not supported.

5. Local Repair: Fast-Reroute

Local repair for multicast traffic is different from unicast in several important ways.

- o There is more than a single final destination. The full set of receiving routers may not be known by the PLR and may be extremely large. Therefore, it makes sense to repair to the immediate next-hops for link-repair and the next-next-hops for node-repair. These are the potential merge points (MPs).
- o If a failure cannot be positively identified as a node-failure, then it is important to repair to the immediate next-hops since they may have receivers attached.
- o If a failure cannot be positively identified as a link-failure and node protection is desired, then it is important to repair to the next-next-hops since they may not receive traffic from the immediate next-hops.

- o Updating multicast forwarding state may take significantly longer than updating unicast state, since the multicast state is updated tree by tree based on control-plane signaling.
- o For tunnel-based IP/LDP approaches, neither the PLR nor the MP may be able to specify which interface the alternate traffic will arrive at the MP on. The simplest reason is the unicast forwarding includes the use of ECMP and the path selection is based upon internal router behavior for all paths between the PLR and the MP.

For multicast fast-reroute, there are three different mechanisms that can be used. As long as the necessary signaling is available, these methods can be combined in the same network and even for the same PLR and failure point.

PLR-driven Unicast Tunnels: The PLR learns the set of MPs that need protection. On a failure, the PLR replicates the traffic and tunnels it to each MP using the unicast route. If desired, an RSVP-TE tunnel could be used instead of relying upon unicast routing.

MP-driven Unicast Tunnels: Each MP learns the identity of the PLR. Before failure, each MP independently signals to the PLR the desire for protection and other information to use. On a failure, the PLR replicates the traffic and tunnels it to each MP using the unicast route. If desired, an RSVP-TE tunnel could be used instead of relying upon unicast routing.

MP-driven Alternate Trees: Each MP learns the identity of the PLR and the failure point (node and interface) to be protected against. Each MP selects an upstream interface and forwarding topology where the path will avoid the failure point; each MP signals a join towards that upstream interface to create that state.

Each of these options is described in more detail in their respective sections. Then the methods are compared and contrasted for PIM and for mLDP.

5.1. PLR-driven Unicast Tunnels

With PLR-driven unicast tunnels, the PLR learns the set of merge points (MPs) and, on a locally detected failure, uses the existing unicast routing to tunnel the multicast traffic to those merge points. The failure being protected against may be link or node failure. If unicast forwarding can provide an SRLG-protecting alternate, then SRLG-protection is also possible.

There are five aspects to making this work.

1. PLR needs to learn the MPs and their associated MPLS labels to create protection state.
2. Unicast routing has to offer alternates or have dedicated tunnels to reach the MPs. The PLR encapsulates the multicast traffic and directs it to be forwarded via unicast routing.
3. The MP must identify alternate traffic and decide when to accept and forward it or drop it.
4. When the MP reconverges, it must move to its new UMH using make-before-break so that traffic loss is minimized.
5. The PLR must know when to stop sending traffic on the alternates.

5.1.1. Learning the MPs

If link-protection is all that is desired, then the PLR already knows the identities of the MPs. For node-protection, this is not sufficient. In the PLR-driven case, there is no direct communication possible between the PLR and the next-next-hops on the multicast tree. (For mLDP, when targeted LDP sessions are used, this is considered to be MP-driven and is covered in Section 5.2.)

In addition to learning the identities of the MPs, the PLR must also learn the MPLS label, if any, associated with each MP. For mLDP, a different label should be supplied for the alternate traffic; this allows the MP to distinguish between the primary and alternate traffic. For PIM, an MPLS label is used to identify that traffic is the alternate. The unicast tunnel used to send traffic to the MP may have penultimate-hop-popping done; thus without an explicit MPLS label, there is no certainty that a packet could be conclusively identified as primary traffic or as alternate traffic.

A router must tell its UMH the identity of all downstream multicast routers, and their associated alternate labels, on the particular multicast tree. This clearly requires protocol extensions. The extensions for PIM are given in [I-D.kebler-pim-mrt-protection].

5.1.2. Using Unicast Tunnels and Indirection

The PLR must encapsulate the multicast traffic and tunnel it towards each MP. The key point is how that traffic then reaches the MP. There are basically two possibilities. It is possible that a dedicated RSVP-TE tunnel exists and can be used to reach the MP for just this traffic; such an RSVP-TE tunnel would be explicitly routed

to avoid the failure point. The second possibility is that the packet is tunneled via LDP and uses unicast routing. The second case is explored here.

It is necessary to assume that unicast LDP fast-reroute [I-D.ietf-rtgwg-mrt-frr-architecture][RFC5714][RFC5286] is supported by the PLR. Since multicast convergence takes longer than unicast convergence, the PLR may have two different routes to the MP over time. When the failure happens, the PLR will have an alternate, whether LFA or MRT, to reach the MP. Then the unicast routing converges and the PLR will have a new primary route to the MP. Once the routing has converged, it is important that alternate traffic is no longer carried on the MRT forwarding topologies. This rule allows the MRT forwarding topologies to reconverge and be available for the next failure. Therefore, it is also necessary for the tunneled multicast traffic to move from the alternate route to the new primary route when the PLR reconverges. Therefore, the tunneled multicast traffic should use indirection to obtain the unicast routing's current next-hops to the MP. If physical indirection is not feasible, then when the unicast LIB is updated, the associated multicast alternate tunnel state should be as well.

When the PLR detects a local failure, the PLR replicates each multicast packet, swaps or adds the alternate MPLS label needed by the MP, and finally pushes the appropriate label for the MP based upon the outgoing interface selected by the unicast routing.

For PIM, if no alternate labels are supplied by the MPs, then the multicast traffic could be tunneled in IP. This would require unicast IP fast-reroute.

5.1.3. MP Alternate Traffic Handling

A potential Merge Point must determine when and if to accept alternate traffic. There are two critical components to this decision. First, the MP must know the state of all links to its UMH. This allows the MP to determine whether the multicast stream could be received from the UMH. Second, the MP must be able to distinguish between a normal multicast tree packet and an alternate packet.

The logic is similar for PIM and mLDP, but in PIM there is only one RPF-interface or interface of interest to the UMH. In mLDP, all the directly connected interfaces to the UMH are of interest. When the MP detects a local failure, if that interface was the last connected to the UMH and used for the multicast group, then the MP must rapidly switch from accepting the normal multicast tree traffic to accepting the alternate traffic. This rapid change must happen within the same approximately 50 milliseconds that the PLR switching to send traffic

on the alternate takes and for the same reasons. It does no good for the PLR to send alternate traffic if the MP doesn't accept it when it is needed.

The MP can identify alternate traffic based upon the MPLS label. This will be the alternate label that the MP supplied to its UMH for this purpose.

5.1.4. Merge Point Reconvergence

After a failure, the MP will want to join the multicast tree according to the new topology. It is critical that the MP does this in a way that minimizes the traffic disruption. Whenever paths change, there is also the possibility for a traffic-affecting event due to different latencies. However, traffic impact above that should be avoided.

The MP must do make-before-break. Until the MP knows that its new UMH is fully connected to the MCI, the MP should continue to accept its old alternate traffic. The MP could learn that the new UMH is sufficient either via control-plane mechanisms or data-driven. In the latter case, the reception of traffic from the new UMH can trigger the change-over. If the data-driven approach is used, a time-out to force the switch should apply to handle multicast trees that have long quiet periods.

5.1.5. PLR termination of alternate traffic

The PLR sends traffic on the alternates for a configurable time-out. There is no clean way for the next-hop routers and/or next-next-hop routers to indicate that the traffic is no longer needed.

If better control were desired, each MP could tell its UMH what the desired time-out is. The UMH could forward this to the PLR as well. Then the PLR could send alternate traffic to different MPs based upon the MP's individual timer. This would only be an advantage if some of the MPs were expected to have a longer multicast reconvergence time than others - either due to load or router capabilities.

5.2. MP-driven Unicast Tunnels

MP-driven unicast tunnels are only relevant for mLDP where targeted LDP sessions are feasible. For PIM, there is no mechanism to communicate beyond a router's immediate neighbors; these techniques could work for link-protection, but even then there would not be a way of requesting that the PLR should stop sending traffic.

There are three differences for MP-driven unicast tunnels from PLR-

driven unicast tunnels.

1. The MPs learn the identity of the PLR from their UMH. The PLR does not learn the identities of the MPs.
2. The MPs create direct connections to the PLR and communicate their alternate labels.
3. When the MPs have converged, each explicitly tells the PLR to stop sending alternate traffic.

The first means that a router communicates its UMH to all its downstream multicast hops. Then each MP communicates to the PLR(s) (1 for link-protection and 1 for node-protection) and indicates the multicast tree that protection is desired for and the associated alternate label.

When the PLR learns about a new MP, it adds that MP and associated information to the set of MPs to be protected. On a failure, the PLR does the same behavior as for the PLR-driven unicast tunnels.

After the failure, the MP reconverges using make-before-break. Then the MP explicitly communicates to the PLR(s) that alternate traffic is no longer needed for that multicast tree. When the node-protecting PLR hasn't changed for a MP, it may be necessary to withdraw the old alternate label, which tells the PLR to stop transmitting alternate traffic, and then provide a new alternate label.

5.3. MP-driven Alternate Trees

In this document we have defined different solutions to achieve fast convergence for multicast link and node protection based on MRTs. At a high level these solutions can be separated in Local and Global protections. Alternate Trees, which is a Local protection schema, initially looked like an attractive solution for Multicast node protection since it avoids replicating the packet by the PLR to each of the receivers of the protected node and wasting bandwidth. However, this comes at the expense of extra multicast state and complexity. In order to mitigate the extra multicast state its possible to aggregate the Alternate Trees by creating an Alternate Tree per protected node and reuse it for all the multicast trees going through this node. This further complicates the procedures and upstream assigned labels are required to de-aggregate the trees. With aggregation we are also introducing an unwanted side effect. The receiver population of the aggregated trees will very likely not be the same. That means multicast packets will be forwarded on the Alternate Tree to node(s) that may not have a receiver(s) for the

protected tree. The more protected trees are aggregated, the higher the risk of forwarding unwanted multicast packets, this leads again to waisting bandwidth.

Considering the complexity of this solution and the unwanted side-effects, the authors of this document believe its better to solve Multicast node protection using a Global protection schema, as documented in Section 4. The solution previously defined in this section has been move to Appendix A (Section 9).

6. Acknowledgements

The authors would like to thank Kishore Tiruveedhula, Santosh Esale, and Maciek Konstantynowicz for their suggestions and review.

7. IANA Considerations

This doument includes no request to IANA.

8. Security Considerations

This architecture is not currently believed to introduce new security concerns.

9. Appendix A

9.1. MP-driven Alternate Trees

For some networks, it is highly desirable not to have the PLR perform replication to each MP. PLR replication can cause substantial congestion on links used by alternates to different MPs. At the same time, it is also desirable to have minimal extra state created in the network. This can be resolved by creating alternate-trees that can protect multiple multicast groups as a bypass-alternate-tree. An alternate-tree can also be created per multicast group, PLR and failure point.

It is not possible to merge alternate-trees for different PLRs or for different neighbors. This is shown in Figure 4 where G can't select an acceptable upstream node on the alternate tree that doesn't violate either the need to avoid C (for PLR A) or D (for PLR B).

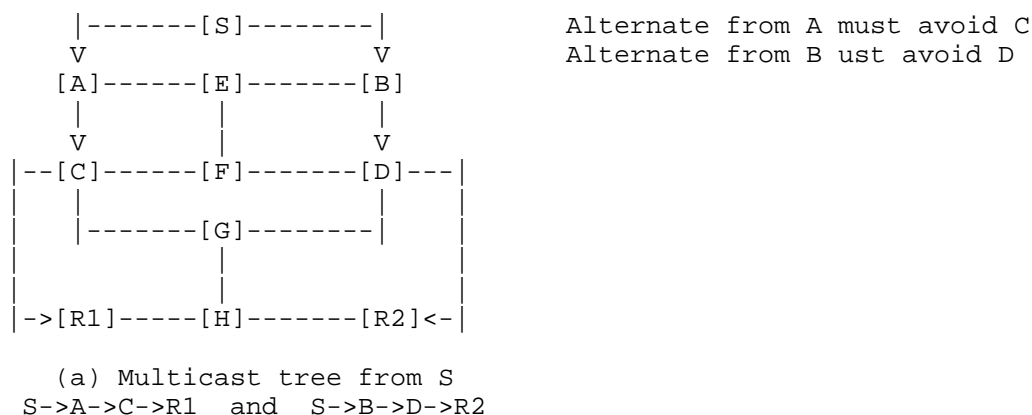


Figure 4: Alternate Trees from PLR A and B can't be merged

A MP that joins an alternate-tree for a particular multicast stream should not expect or request PLR-replicated tunneled alternate traffic for that same multicast stream.

Each alternate-tree is identified by the PLR which sources the traffic and the failure point (node and link) (FP) to be avoided. Different multicast groups with the same PLR and FP may have different sets of MPs - but they are all at most going to include the FP (for link protection) and the neighbors of FP except for the PLR. For a bypass-alternate-tree to work, it must be acceptable to temporarily send a multicast group's traffic to FP's neighbors that do not need it. This is the trade-off required to reduce alternate-tree state and use bypass-alternate-trees. As discussed in Section 5.1.3, a potential MP can determine whether to accept alternate traffic based upon the state of its normal upstream links. Alternate traffic for a group the MP hasn't joined can just be discarded.

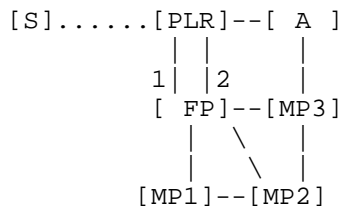


Figure 5: Alternate Tree Scenario

For any router, knowing the PLR and the FP to avoid will force selection of either the Blue MRT or the Red MRT. It is possible that the FP doesn't actually appear in either MRT path, but the FP will always be in either the set of nodes that might be used for the Blue MRT path or the set of nodes that might be used for the Red MRT path. The FP's membership in one of the sets is a function of the partial ordering and topological ordering created by the MRT algorithm and is consistent between routers in the network graph.

To create an alternate-tree, the following must happen:

1. For node-protection, the MP learns from its upstream (the FP) the node-id of its upstream (the PLR) and, optionally, a link identifier for the link used to the PLR. The link-id is only needed for traffic handling in PIM, since mLDp can have targeted sessions between the MP and the PLR.
2. For link-protection, the MP needs to know the node-id of its upstream (the PLR) and, optionally, its identifier for the link used to the PLR.
3. The MP determines whether to use the Blue or Red forwarding topology to reach the PLR while avoiding the FP and associated interface. This gives the MP its alternate-tree upstream interface.
4. The MP signals a backup-join to its alternate-tree upstream interface. The backup-join specifies the PLR, FP and, for PIM, the FP-PLR link identifier. If the alternate-tree is not to be used as a bypass-alternate-tree, then the multicast group (e.g. (S,G) or Opaque-Value) must be specified.
5. A router that receives a backup-join and is not the PLR needs to create multicast state and send a backup-join towards the PLR on the appropriate Blue or Red forwarding topology as is locally determined to avoid the FP and FP-PLR link.
6. Backup-joins for the same (PLR, FP, PLR-FP link-id) that reference the same multicast group can be merged into a single alternate-tree. Similarly, backup-joins for the same (PLR, FP, PLR-FP link-id) that reference no multicast group can be merged into a single alternate-tree.
7. When the PLR receives the backup-join, it associates either the specified multicast group with that alternate-tree, if such is given, or associates all multicast groups that go to the FP via the specified FP-PLR link with the alternate-tree.

For an example, look at Figure 5. FP would send a backup-join to MP3 indicating (PLR, FP, PLR-FP link-1). MP3 sends a backup-join to A. MP1 sends a backup-join to MP2 and MP2 sends a backup-join to MP3.

It is necessary that traffic on each alternate-tree self-identify as to which alternate-tree it is part of. This is because an alternate-tree for a multicast-group and a particular (PLR, FP, PLR-FP link-id) can easily overlap with an alternate-tree for the same multicast group and a different (PLR, FP, PLR-FP link-id). The best way of doing this depends upon whether PIM or mLDP is being used.

9.1.1. PIM details for Alternate-Trees

For PIM, the (S,G) of the IP packet is a globally unique identifier and is understood. To identify the alternate-tree, the most straightforward way is to use MPLS labels distributed in the PIM backup-join messages. A MP can use the incoming label to indicate the set of RPF-interfaces for which the traffic may be an alternate. If the alternate-tree isn't a bypass-alternate-tree, then only one RPF interface is referenced. If the alternate-tree is a bypass-alternate-tree, then multiple RPF-interfaces (parallel links to FP) might be intended. Alternate-tree traffic may cross an interface multiple times - either because the interface is a broadcast interface and different downstream-assigned labels are provided and/or because a MP may provide different labels.

9.1.2. mLDP details for Alternate-Trees

For mLDP, if bypass-alternate-trees are used, then the PLR must provide upstream-assigned labels to each multicast stream. The MP provides the label for the alternate-tree; if the alternate-tree is not a bypass-alternate-tree, this label also describes the multicast stream. If the alternate-tree is a bypass-alternate-tree, then it provides the context for the PLR-assigned labels for each multicast stream. If there are targeted LDP sessions between the PLR and the MPs, then the PLR could provide the necessary upstream-assigned labels.

9.1.3. Traffic Handling by PLR

An issue with traffic is how long should the PLR continue to send alternate traffic out. With an alternate-tree, the PLR can know to stop forwarding alternate traffic on the alternate-tree when that alternate-tree's state is torn down. This provides a clear signal that alternate traffic is no longer needed.

9.2. Methods Compared for PIM

The two approaches that are feasible for PIM are PLR-driven Unicast Tunnels and MP-driven Alternate-Trees.

| Aspect | PLR-driven Unicast Tunnels | MP-driven Alternate-Trees |
|--|-------------------------------|-----------------------------------|
| Worst-case Traffic Replication Per Link | 1 + number of MPs | 2 |
| PLR alternate-traffic | timer-based | control-plane terminated |
| Extra multicast state | none | per (PLR,FP,S) for bypass mode |

Which approach is preferred may be network-dependent. It should also be possible to use both in the same network.

9.3. Methods Compared for mLDP

All three approaches are feasible for mLDP. Below is a brief comparison of various aspects of each.

| Aspect | MP-driven Unicast Tunnels | PLR-driven Unicast Tunnels | MP-driven Alternate-Trees |
|--|---------------------------------|----------------------------------|-----------------------------------|
| Worst-case Traffic Replication Per Link | 1 + number of MPs | 1 + number of MPs | 2 |
| PLR alternate-traffic | control-plane terminated | timer-based | control-plane terminated |
| Extra multicast state | none | none | per (PLR,FP,S) for bypass mode |

10. References

10.1. Normative References

[I-D.enyedi-rtgwg-mrt-frr-algorithm]
 Atlas, A., Envedi, G., Csaszar, A., and A. Gopalan,
 "Algorithms for computing Maximally Redundant Trees for

IP/LDP Fast- Reroute",
draft-enyedi-rtgwg-mrt-frr-algorithm-03 (work in
progress), July 2013.

- [I-D.ietf-rtgwg-mrt-frr-architecture]
Atlas, A., Kebler, R., Envedi, G., Csaszar, A., Tantsura, J., Konstantynowicz, M., White, R., and M. Shand, "An Architecture for IP/LDP Fast-Reroute Using Maximally Redundant Trees", draft-ietf-rtgwg-mrt-frr-architecture-03 (work in progress), July 2013.
- [RFC4601] Fenner, B., Handley, M., Holbrook, H., and I. Kouvelas, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", RFC 4601, August 2006.
- [RFC6388] Wijnands, IJ., Minei, I., Kompella, K., and B. Thomas, "Label Distribution Protocol Extensions for Point-to-Multipoint and Multipoint-to-Multipoint Label Switched Paths", RFC 6388, November 2011.
- [RFC6420] Cai, Y. and H. Ou, "PIM Multi-Topology ID (MT-ID) Join Attribute", RFC 6420, November 2011.

10.2. Informative References

- [I-D.ietf-rtgwg-mofrr]
Karan, A., Filsfils, C., Farinacci, D., Wijnands, I., Decraene, B., Joerde, U., and W. Henderickx, "Multicast only Fast Re-Route", draft-ietf-rtgwg-mofrr-02 (work in progress), June 2013.
- [I-D.iwijnand-mpls-mldp-multi-topology]
Wijnands, I. and K. Raza, "mLDP Extensions for Multi Topology Routing",
draft-iwijnand-mpls-mldp-multi-topology-03 (work in progress), June 2013.
- [I-D.kebler-pim-mrt-protection]
Kebler, R., Atlas, A., Wijnands, IJ., and G. Enyedi, "PIM Extensions for Protection Using Maximally Redundant Trees", draft-kebler-pim-mrt-protection-00 (work in progress), March 2012.
- [I-D.wijnands-mpls-mldp-node-protection]
Wijnands, I., Rosen, E., Raza, K., Tantsura, J., Atlas, A., and Q. Zhao, "mLDP Node Protection",
draft-wijnands-mpls-mldp-node-protection-04 (work in progress), June 2013.

- [RFC5286] Atlas, A. and A. Zinin, "Basic Specification for IP Fast Reroute: Loop-Free Alternates", RFC 5286, September 2008.
- [RFC5714] Shand, M. and S. Bryant, "IP Fast Reroute Framework", RFC 5714, January 2010.

Authors' Addresses

Alia Atlas (editor)
Juniper Networks
10 Technology Park Drive
Westford, MA 01886
USA

Email: akatlas@juniper.net

Robert Kebler
Juniper Networks
10 Technology Park Drive
Westford, MA 01886
USA

Email: rkebler@juniper.net

IJsbrand Wijnands
Cisco Systems, Inc.

Email: ice@cisco.com

Andras Csaszar
Ericsson
Konyves Kalman krt 11
Budapest 1097
Hungary

Email: Andras.Csaszar@ericsson.com

Gabor Sandor Enyedi
Ericsson
Konyves Kalman krt 11.
Budapest 1097
Hungary

Email: Gabor.Sandor.Enyedi@ericsson.com

Routing Area Working Group
Internet-Draft
Intended status: Informational
Expires: April 24, 2014

G. Enyedi, Ed.
A. Csaszar
Ericsson
A. Atlas, Ed.
C. Bowers
Juniper Networks
A. Gopalan
University of Arizona
October 21, 2013

Algorithms for computing Maximally Redundant Trees for IP/LDP Fast-
Reroute
draft-enyedi-rtgwg-mrt-frr-algorithm-04

Abstract

A complete solution for IP and LDP Fast-Reroute using Maximally Redundant Trees is presented in [I-D.ietf-rtgwg-mrt-frr-architecture]. This document defines the associated MRT Lowpoint algorithm that is used in the default MRT profile to compute both the necessary Maximally Redundant Trees with their associated next-hops and the alternates to select for MRT-FRR.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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."

This Internet-Draft will expire on April 24, 2014.

Copyright Notice

Copyright (c) 2013 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 Simplified BSD License.

Table of Contents

| | |
|--|----|
| 1. Introduction | 3 |
| 2. Terminology and Definitions | 4 |
| 3. Algorithm Key Concepts | 6 |
| 3.1. Partial Ordering for Disjoint Paths | 6 |
| 3.2. Finding an Ear and the Correct Direction | 8 |
| 3.3. Low-Point Values and Their Uses | 11 |
| 3.4. Blocks in a Graph | 13 |
| 3.5. Determining Local-Root and Assigning Block-ID | 15 |
| 4. Algorithm Sections | 16 |
| 4.1. MRT Island Identification | 17 |
| 4.2. Root Selection | 18 |
| 4.3. Initialization | 18 |
| 4.4. MRT Lowpoint Algorithm: Computing GADAG using lowpoint inheritance | 19 |
| 4.5. Augmenting the GADAG by directing all links | 21 |
| 4.6. Compute MRT next-hops | 23 |
| 4.6.1. MRT next-hops to all nodes partially ordered with respect to the computing node | 24 |
| 4.6.2. MRT next-hops to all nodes not partially ordered with respect to the computing node | 24 |
| 4.6.3. Computing Redundant Tree next-hops in a 2-connected Graph | 25 |
| 4.6.4. Generalizing for graph that isn't 2-connected | 27 |
| 4.6.5. Complete Algorithm to Compute MRT Next-Hops | 28 |
| 4.7. Identify MRT alternates | 30 |
| 4.8. Finding FRR Next-Hops for Proxy-Nodes | 34 |
| 5. MRT Lowpoint Algorithm: Complete Specification | 36 |
| 6. Algorithm Alternatives and Evaluation | 37 |
| 6.1. Algorithm Evaluation | 37 |
| 7. Algorithm Work to Be Done | 47 |
| 8. IANA Considerations | 47 |
| 9. Security Considerations | 47 |
| 10. References | 47 |
| 10.1. Normative References | 47 |
| 10.2. Informative References | 47 |
| Appendix A. Option 2: Computing GADAG using SPF's | 49 |
| Appendix B. Option 3: Computing GADAG using a hybrid method | 53 |
| Authors' Addresses | 55 |

1. Introduction

MRT Fast-Reroute requires that packets can be forwarded not only on the shortest-path tree, but also on two Maximally Redundant Trees (MRTs), referred to as the MRT-Blue and the MRT-Red. A router which experiences a local failure must also have pre-determined which alternate to use. This document defines how to compute these three things for use in MRT-FRR and describes the algorithm design decisions and rationale. The algorithm is based on those presented in [MRTLinear] and expanded in [EnyediThesis].

Just as packets routed on a hop-by-hop basis require that each router compute a shortest-path tree which is consistent, it is necessary for each router to compute the MRT-Blue next-hops and MRT-Red next-hops in a consistent fashion. This document defines the MRT Lowpoint algorithm to be used as a standard in the default MRT profile for MRT-FRR.

As now, a router's FIB will contain primary next-hops for the current shortest-path tree for forwarding traffic. In addition, a router's FIB will contain primary next-hops for the MRT-Blue for forwarding received traffic on the MRT-Blue and primary next-hops for the MRT-Red for forwarding received traffic on the MRT-Red.

What alternate next-hops a point-of-local-repair (PLR) selects need not be consistent - but loops must be prevented. To reduce congestion, it is possible for multiple alternate next-hops to be selected; in the context of MRT alternates, each of those alternate next-hops would be equal-cost paths.

This document defines an algorithm for selecting an appropriate MRT alternate for consideration. Other alternates, e.g. LFAs that are downstream paths, may be preferred when available and that policy-based alternate selection process[I-D.ietf-rtgwg-lfa-manageability] is not captured in this document.

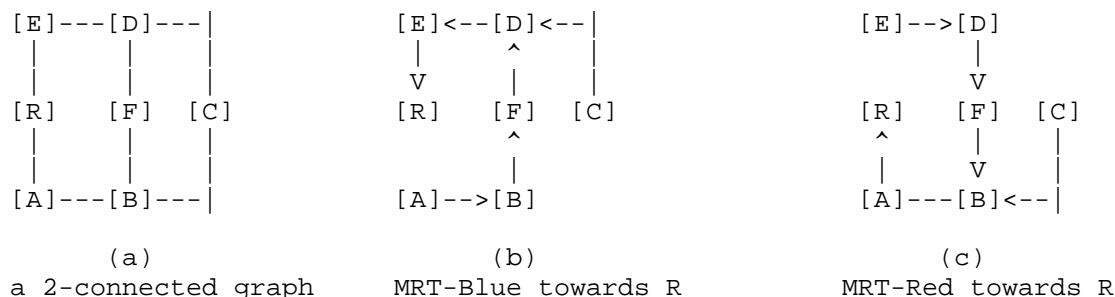
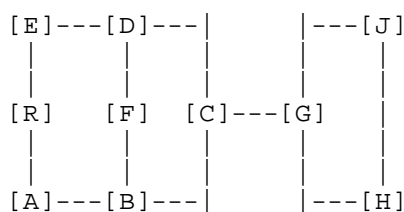
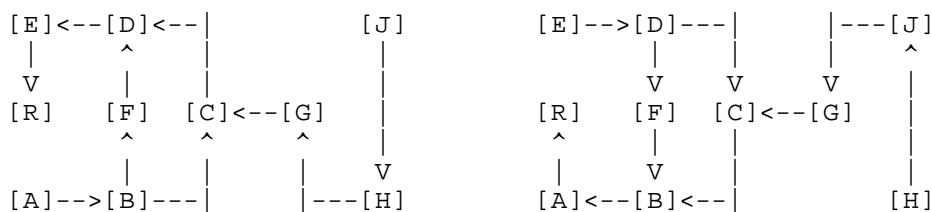


Figure 1

Algorithms for computing MRTs can handle arbitrary network topologies where the whole network graph is not 2-connected, as in Figure 2, as well as the easier case where the network graph is 2-connected (Figure 1). Each MRT is a spanning tree. The pair of MRTs provide two paths from every node X to the root of the MRTs. Those paths share the minimum number of nodes and the minimum number of links. Each such shared node is a cut-vertex. Any shared links are cut-links.



(a) a graph that isn't 2-connected



(b) MRT-Blue towards R

(c) MRT-Red towards R

Figure 2

2. Terminology and Definitions

network graph: A graph that reflects the network topology where all links connect exactly two nodes and broadcast links have been transformed into the standard pseudo-node representation.

Redundant Trees (RT): A pair of trees where the path from any node X to the root R on the first tree is node-disjoint with the path from the same node X to the root along the second tree. These can be computed in 2-connected graphs.

Maximally Redundant Trees (MRT): A pair of trees where the path from any node X to the root R along the first tree and the path from the same node X to the root along the second tree share the minimum number of nodes and the minimum number of links. Each such shared node is a cut-vertex. Any shared links are cut-links. Any RT is an MRT but many MRTs are not RTs.

MRT-Red: MRT-Red is used to describe one of the two MRTs; it is used to describe the associated forwarding topology and MT-ID. Specifically, MRT-Red is the decreasing MRT where links in the GADAG are taken in the direction from a higher topologically ordered node to a lower one.

MRT-Blue: MRT-Blue is used to describe one of the two MRTs; it is used to describe the associated forwarding topology and MT-ID. Specifically, MRT-Blue is the increasing MRT where links in the GADAG are taken in the direction from a lower topologically ordered node to a higher one.

cut-vertex: A vertex whose removal partitions the network.

cut-link: A link whose removal partitions the network. A cut-link by definition must be connected between two cut-vertices. If there are multiple parallel links, then they are referred to as cut-links in this document if removing the set of parallel links would partition the network.

2-connected: A graph that has no cut-vertices. This is a graph that requires two nodes to be removed before the network is partitioned.

spanning tree: A tree containing links that connects all nodes in the network graph.

back-edge: In the context of a spanning tree computed via a depth-first search, a back-edge is a link that connects a descendant of a node *x* with an ancestor of *x*.

2-connected cluster: A maximal set of nodes that are 2-connected. In a network graph with at least one cut-vertex, there will be multiple 2-connected clusters.

block: Either a 2-connected cluster, a cut-edge, or an isolated vertex.

DAG: Directed Acyclic Graph - a digraph containing no directed cycle.

ADAG: Almost Directed Acyclic Graph - a digraph that can be transformed into a DAG with removing a single node (the root node).

GADAG: Generalized ADAG - a digraph, which has only ADAGs as all of its blocks. The root of such a block is the node closest to the global root (e.g. with uniform link costs).

DFS: Depth-First Search

DFS ancestor: A node n is a DFS ancestor of x if n is on the DFS-tree path from the DFS root to x .

DFS descendant: A node n is a DFS descendant of x if x is on the DFS-tree path from the DFS root to n .

ear: A path along not-yet-included-in-the-GADAG nodes that starts at a node that is already-included-in-the-GADAG and that ends at a node that is already-included-in-the-GADAG. The starting and ending nodes may be the same node if it is a cut-vertex.

$X \gg Y$ or $Y \ll X$: Indicates the relationship between X and Y in a partial order, such as found in a GADAG. $X \gg Y$ means that X is higher in the partial order than Y . $Y \ll X$ means that Y is lower in the partial order than X .

$X > Y$ or $Y < X$: Indicates the relationship between X and Y in the total order, such as found via a topological sort. $X > Y$ means that X is higher in the total order than Y . $Y < X$ means that Y is lower in the total order than X .

proxy-node: A node added to the network graph to represent a multi-homed prefix or routers outside the local MRT-fast-reroute-supporting island of routers. The key property of proxy-nodes is that traffic cannot transit them.

3. Algorithm Key Concepts

There are five key concepts that are critical for understanding the MRT Lowpoint algorithm and other algorithms for computing MRTs. The first is the idea of partially ordering the nodes in a network graph with regard to each other and to the GADAG root. The second is the idea of finding an ear of nodes and adding them in the correct direction. The third is the idea of a Low-Point value and how it can be used to identify cut-vertices and to find a second path towards the root. The fourth is the idea that a non-2-connected graph is made up of blocks, where a block is a 2-connected cluster, a cut-edge or an isolated node. The fifth is the idea of a local-root for each node; this is used to compute ADAGs in each block.

3.1. Partial Ordering for Disjoint Paths

Given any two nodes X and Y in a graph, a particular total order means that either $X < Y$ or $X > Y$ in that total order. An example would be a graph where the nodes are ranked based upon their unique IP loopback addresses. In a partial order, there may be some nodes

for which it can't be determined whether $X \ll Y$ or $X \gg Y$. A partial order can be captured in a directed graph, as shown in Figure 3. In a graphical representation, a link directed from X to Y indicates that X is a neighbor of Y in the network graph and $X \ll Y$.

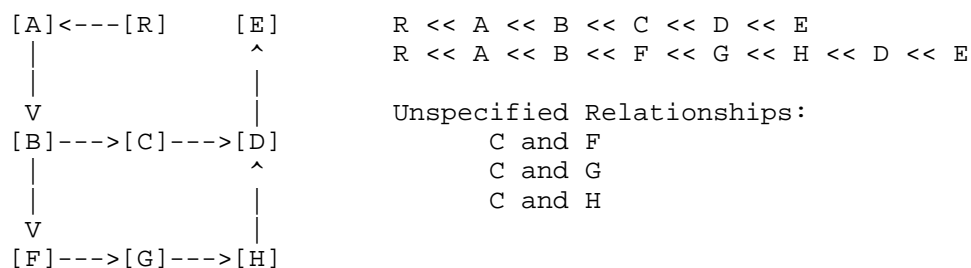


Figure 3: Directed Graph showing a Partial Order

To compute MRTs, the root of the MRTs is at both the very bottom and the very top of the partial ordering. This means that from any node X, one can pick nodes higher in the order until the root is reached. Similarly, from any node X, one can pick nodes lower in the order until the root is reached. For instance, in Figure 4, from G the higher nodes picked can be traced by following the directed links and are H, D, E and R. Similarly, from G the lower nodes picked can be traced by reversing the directed links and are F, B, A, and R. A graph that represents this modified partial order is no longer a DAG; it is termed an Almost DAG (ADAG) because if the links directed to the root were removed, it would be a DAG.

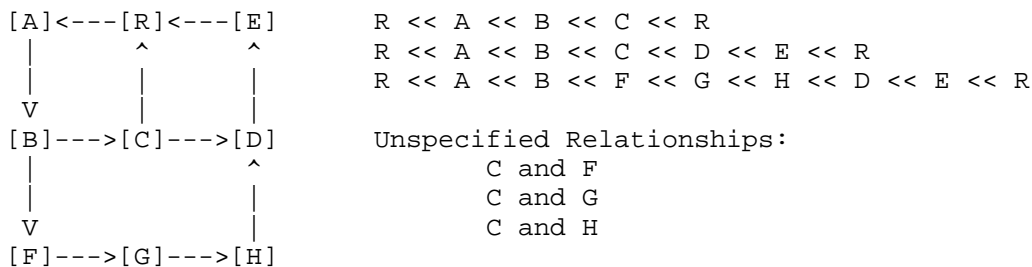


Figure 4: ADAG showing a Partial Order with R lowest and highest

Most importantly, if a node $Y \gg X$, then Y can only appear on the increasing path from X to the root and never on the decreasing path. Similarly, if a node $Z \ll X$, then Z can only appear on the decreasing path from X to the root and never on the increasing path.

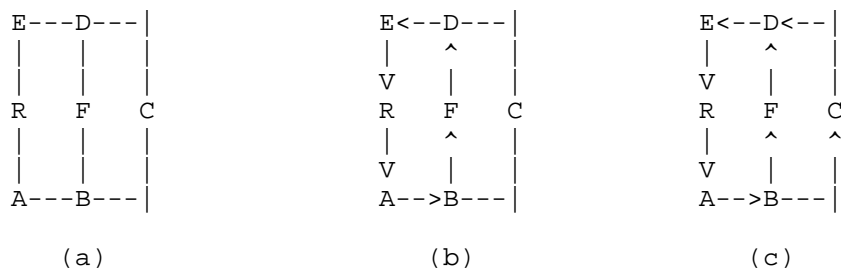
When following the increasing paths, it is possible to pick multiple higher nodes and still have the certainty that those paths will be disjoint from the decreasing paths. E.g. in the previous example node B has multiple possibilities to forward packets along an increasing path: it can either forward packets to C or F .

3.2. Finding an Ear and the Correct Direction

For simplicity, the basic idea of creating a GADAG by adding ears is described assuming that the network graph is a single 2-connected cluster so that an ADAG is sufficient. Generalizing to multiple blocks is done by considering the block-roots instead of the GADAG root - and the actual algorithm is given in Section 4.4.

In order to understand the basic idea of finding an ADAG, first suppose that we have already a partial ADAG, which doesn't contain all the nodes in the block yet, and we want to extend it to cover all the nodes. Suppose that we find a path from a node X to Y such that X and Y are already contained by our partial ADAG, but all the remaining nodes along the path are not added to the ADAG yet. We refer to such a path as an ear.

Recall that our ADAG is closely related to a partial order, more precisely, if we remove root R , the remaining DAG describes a partial order of the nodes. If we suppose that neither X nor Y is the root, we may be able to compare them. If one of them is definitely lesser with respect to our partial order (say $X \ll Y$), we can add the new path to the ADAG in a direction from X to Y . As an example consider Figure 5.



(a) A 2-connected graph
 (b) Partial ADAG (C is not included)
 (c) Resulting ADAG after adding path (or ear) B-C-D

Figure 5

In this partial ADAG, node C is not yet included. However, we can find path B-C-D, where both endpoints are contained by this partial ADAG (we say those nodes are **ready** in the sequel), and the remaining node (node C) is not contained yet. If we remove R, the remaining DAG defines a partial order, and with respect to this partial order we can say that $B \ll D$, so we can add the path to the ADAG in the direction from B to D (arcs B→C and C→D are added). If B were strictly greater than D, we would add the same path in reverse direction.

If in the partial order where an ear's two ends are X and Y, $X \ll Y$, then there must already be a directed path from X to Y already in the ADAG. The ear must be added in a direction such that it doesn't create a cycle; therefore the ear must go from X to Y.

In the case, when X and Y are not ordered with each other, we can select either direction for the ear. We have no restriction since neither of the directions can result in a cycle. In the corner case when one of the endpoints of an ear, say X, is the root (recall that the two endpoints must be different), we could use both directions again for the ear because the root can be considered both as smaller and as greater than Y. However, we strictly pick that direction in which the root is lower than Y. The logic for this decision is explained in Section 4.6

A partial ADAG is started by finding a cycle from the root R back to itself. This can be done by selecting a non-ready neighbor N of R and then finding a path from N to R that doesn't use any links between R and N. The direction of the cycle can be assigned either way since it is starting the ordering.

Once a partial ADAG is already present, we can always add ears to it: just select a non-ready neighbor N of a ready node Q, such that Q is not the root, find a path from N to the root in the graph with Q removed. This path is an ear where the first node of the ear is Q, the next is N, then the path until the first ready node the path reached (that second ready node is the other endpoint of the path). Since the graph is 2-connected, there must be a path from N to R without Q.

It is always possible to select a non-ready neighbor N of a ready node Q so that Q is not the root R . Because the network is 2-connected, N must be connected to two different nodes and only one can be R . Because the initial cycle has already been added to the ADAG, there are ready nodes that are not R . Since the graph is 2-connected, while there are non-ready nodes, there must be a non-ready neighbor N of a ready node that is not R .

```
Generic_Find_Ears_ADAG(root)
  Create an empty ADAG. Add root to the ADAG.
  Mark root as IN_GADAG.
  Select an arbitrary cycle containing root.
  Add the arbitrary cycle to the ADAG.
  Mark cycle's nodes as IN_GADAG.
  Add cycle's non-root nodes to process_list.
  while there exists connected nodes in graph that are not IN_GADAG
    Select a new ear. Let its endpoints be  $X$  and  $Y$ .
    if  $Y$  is root or  $(Y \ll X)$ 
      add the ear towards  $X$  to the ADAG
    else // (a)  $X$  is root or (b)  $X \ll Y$  or (c)  $X, Y$  not ordered
      Add the ear towards  $Y$  to the ADAG
```

Figure 6: Generic Algorithm to find ears and their direction in 2-connected graph

Algorithm Figure 6 merely requires that a cycle or ear be selected without specifying how. Regardless of the way of selecting the path, we will get an ADAG. The method used for finding and selecting the ears is important; shorter ears result in shorter paths along the MRTs. The MRT Lowpoint algorithm's method using Low-Point Inheritance is defined in Section 4.4. Other methods are described in the Appendices (Appendix A and Appendix B).

As an example, consider Figure 5 again. First, we select the shortest cycle containing R , which can be $R-A-B-F-D-E$ (uniform link costs were assumed), so we get to the situation depicted in Figure 5 (b). Finally, we find a node next to a ready node; that must be node C and assume we reached it from ready node B . We search a path from C to R without B in the original graph. The first ready node along this is node D , so the open ear is $B-C-D$. Since $B \ll D$, we add arc $B \rightarrow C$ and $C \rightarrow D$ to the ADAG. Since all the nodes are ready, we stop at this point.

3.3. Low-Point Values and Their Uses

A basic way of computing a spanning tree on a network graph is to run a depth-first-search, such as given in Figure 7. This tree has the important property that if there is a link (x, n) , then either n is a DFS ancestor of x or n is a DFS descendant of x . In other words, either n is on the path from the root to x or x is on the path from the root to n .

```

global_variable: dfs_number

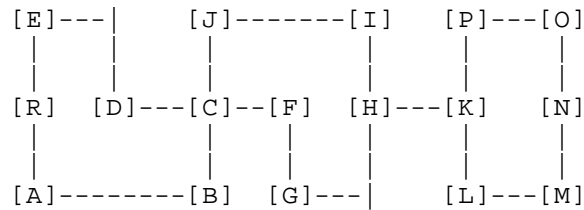
DFS_Visit(node x, node parent)
    D(x) = dfs_number
    dfs_number += 1
    x.dfs_parent = parent
    for each link (x, w)
        if D(w) is not set
            DFS_Visit(w, x)

Run_DFS(node root)
    dfs_number = 0
    DFS_Visit(root, NONE)

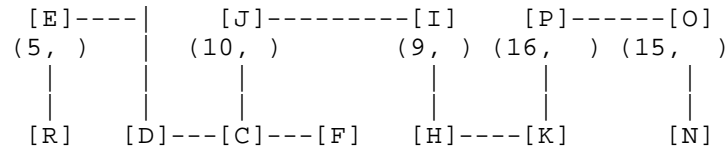
```

Figure 7: Basic Depth-First Search algorithm

Given a node x , one can compute the minimal DFS number of the neighbours of x , i.e. $\min(D(w) \text{ if } (x,w) \text{ is a link})$. This gives the highest attachment point neighbouring x . What is interesting, though, is what is the highest attachment point from x and x 's descendants. This is what is determined by computing the Low-Point value, as given in Algorithm Figure 9 and illustrated on a graph in Figure 8.



(a) a non-2-connected graph



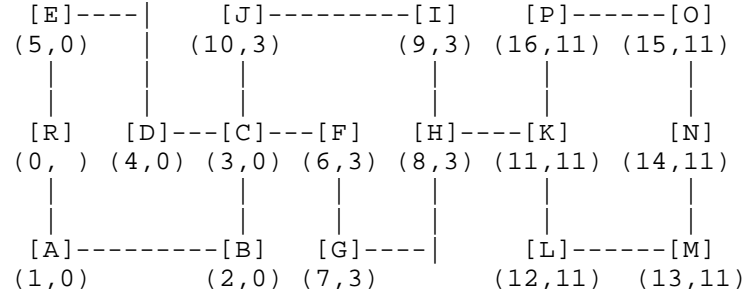
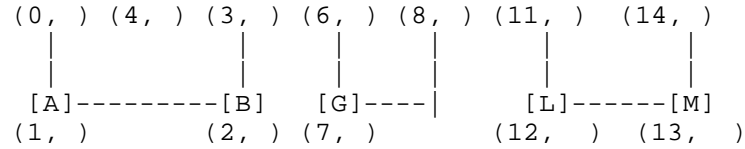


Figure 8

```

global_variable: dfs_number

Lowpoint_Visit(node x, node parent, interface p_to_x)
    D(x) = dfs_number
    L(x) = D(x)
    dfs_number += 1
    x.dfs_parent = parent
    x.dfs_parent_intf = p_to_x
    x.lowpoint_parent = NONE
    for each interface intf of x:
        if D(intf.remote_node) is not set
            Lowpoint_Visit(intf.remote_node, x, intf)
        if L(intf.remote_node) < L(x)
            L(x) = L(intf.remote_node)
            x.lowpoint_parent = intf.remote_node
            x.lowpoint_parent_intf = intf
        else if intf.remote_node is not parent
            if D(intf.remote_node) < L(x)
                L(x) = D(intf.remote)
                x.lowpoint_parent = intf.remote_node
                x.lowpoint_parent_intf = intf

Run_Lowpoint(node root)
    dfs_number = 0

```

```
Lowpoint_Visit(root, NONE, NONE)
```

Figure 9: Computing Low-Point value

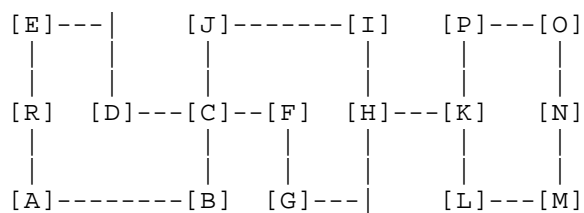
From the low-point value and lowpoint parent, there are two very useful things which motivate our computation.

First, if there is a child c of x such that $L(c) \geq D(x)$, then there are no paths in the network graph that go from c or its descendants to an ancestor of x - and therefore x is a cut-vertex. This is useful because it allows identification of the cut-vertices and thus the blocks. As seen in Figure 8, even if $L(x) < D(x)$, there may be a block that contains both the root and a DFS-child of a node while other DFS-children might be in different blocks. In this example, C 's child D is in the same block as R while F is not.

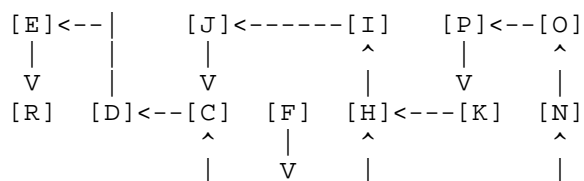
Second, by repeatedly following the path given by `lowpoint_parent`, there is a path from x back to an ancestor of x that does not use the link $[x, x.\text{dfs_parent}]$ in either direction. The full path need not be taken, but this gives a way of finding an initial cycle and then ears.

3.4. Blocks in a Graph

A key idea for an MRT algorithm is that any non-2-connected graph is made up by blocks (e.g. 2-connected clusters, cut-links, and/or isolated nodes). To compute GADAGs and thus MRTs, computation is done in each block to compute ADAGs or Redundant Trees and then those ADAGs or Redundant Trees are combined into a GADAG or MRT.



(a) A graph with four blocks that are:
3 2-connected clusters and a cut-link



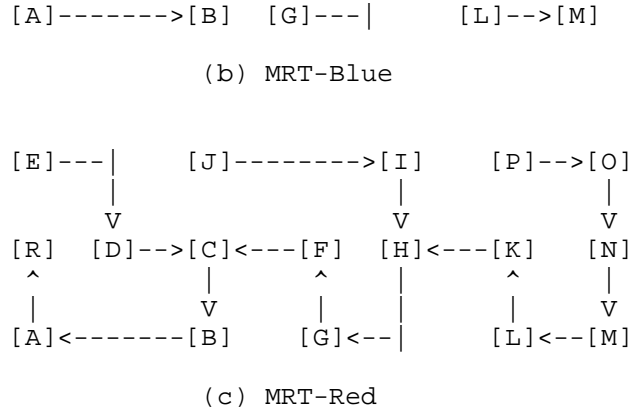


Figure 10

Consider the example depicted in Figure 10 (a). In this figure, a special graph is presented, showing us all the ways 2-connected clusters can be connected. It has four blocks: block 1 contains R, A, B, C, D, E, block 2 contains C, F, G, H, I, J, block 3 contains K, L, M, N, O, P, and block 4 is a cut-edge containing H and K. As can be observed, the first two blocks have one common node (node C) and blocks 2 and 3 do not have any common node, but they are connected through a cut-edge that is block 4. No two blocks can have more than one common node, since two blocks with at least 2 common nodes would qualify as a single 2-connected cluster.

Moreover, observe that if we want to get from one block to another, we must use a cut-vertex (the cut-vertices in this graph are C, H, K), regardless of the path selected, so we can say that all the paths from block 3 along the MRTs rooted at R will cross K first. This observation means that if we want to find a pair of MRTs rooted at R, then we need to build up a pair of RTs in block 3 with K as a root. Similarly, we need to find another one in block 2 with C as a root, and finally, we need the last one in block 1 with R as a root. When all the trees are selected, we can simply combine them; when a block is a cut-edge (as in block 4), that cut-edge is added in the same direction to both of the trees. The resulting trees are depicted in Figure 10 (b) and (c).

Similarly, to create a GADAG it is sufficient to compute ADAGs in each block and connect them.

It is necessary, therefore, to identify the cut-vertices, the blocks and identify the appropriate local-root to use for each block.

3.5. Determining Local-Root and Assigning Block-ID

Each node in a network graph has a local-root, which is the cut-vertex (or root) in the same block that is closest to the root. The local-root is used to determine whether two nodes share a common block.

```

Compute_Localroot(node x, node localroot)
  x.localroot = localroot
  for each DFS child c
    if L(c) < D(x)    //x is not a cut-vertex
      Compute_Localroot(c, x.localroot)
    else
      mark x as cut-vertex
      Compute_Localroot(c, x)

Compute_Localroot(root, root)

```

Figure 11: A method for computing local-roots

There are two different ways of computing the local-root for each node. The stand-alone method is given in Figure 11 and better illustrates the concept; it is used by the MRT algorithms given in the Appendices Appendix A and Appendix B. The method for local-root computation is used in the MRT Lowpoint algorithm for computing a GADAG using Low-Point inheritance and the essence of it is given in Figure 12.

```

Get the current node, s.
Compute an ear(either through lowpoint inheritance
or by following dfs parents) from s to a ready node e.
(Thus, s is not e, if there is such ear.)
if s is e
  for each node x in the ear that is not s
    x.localroot = s
else
  for each node x in the ear that is not s or e
    x.localroot = e.localroot

```

Figure 12: Ear-based method for computing local-roots

Once the local-roots are known, two nodes X and Y are in a common block if and only if one of the following three conditions apply.

- o Y's local-root is X's local-root : They are in the same block and neither is the cut-vertex closest to the root.

- o Y's local-root is X: X is the cut-vertex closest to the root for Y's block
- o Y is X's local-root: Y is the cut-vertex closest to the root for X's block

Once we have computed the local-root for each node in the network graph, we can assign for each node, a block id that represents the block in which the node is present. This computation is shown in Figure 13.

```

global_var: max_block_id

Assign_Block_ID(x, cur_block_id)
  x.block_id = cur_block_id
  foreach DFS child c of x
    if (c.local_root is x)
      max_block_id += 1
      Assign_Block_ID(c, max_block_id)
    else
      Assign_Block_ID(c, cur_block_id)

max_block_id = 0
Assign_Block_ID(root, max_block_id)

```

Figure 13: Assigning block id to identify blocks

4. Algorithm Sections

This algorithm computes one GADAG that is then used by a router to determine its MRT-Blue and MRT-Red next-hops to all destinations. Finally, based upon that information, alternates are selected for each next-hop to each destination. The different parts of this algorithm are described below. These work on a network graph after, for instance, its interfaces are ordered as per Figure 14.

1. Compute the local MRT Island for the particular MRT Profile. [See Section 4.1.]
2. Select the root to use for the GADAG. [See Section 4.2.]
3. Initialize all interfaces to UNDIRECTED. [See Section 4.3.]
4. Compute the DFS value, e.g. $D(x)$, and lowpoint value, $L(x)$. [See Figure 9.]
5. Construct the GADAG. [See Section 4.4]

6. Assign directions to all interfaces that are still `UNDIRECTED`. [See Section 4.5.]
7. From the computing router `x`, compute the next-hops for the MRT-Blue and MRT-Red. [See Section 4.6.]
8. Identify alternates for each next-hop to each destination by determining which one of the blue MRT and the red MRT the computing router `x` should select. [See Section 4.7.]

To ensure consistency in computation, all routers **MUST** order interfaces identically. This is necessary for the DFS, where the selection order of the interfaces to explore results in different trees, and for computing the GADAG, where the selection order of the interfaces to use to form ears can result in different GADAGs. The required ordering between two interfaces from the same router `x` is given in Figure 14.

```
Interface_Compare(interface a, interface b)
  if a.metric < b.metric
    return A_LESS_THAN_B
  if b.metric < a.metric
    return B_LESS_THAN_A
  if a.neighbor.loopback_addr < b.neighbor.loopback_addr
    return A_LESS_THAN_B
  if b.neighbor.loopback_addr < a.neighbor.loopback_addr
    return B_LESS_THAN_A
  // Same metric to same node, so the order doesn't matter anymore.
  // To have a unique, consistent total order,
  // tie-break based on, for example, the link's linkData as
  // distributed in an OSPF Router-LSA
  if a.link_data < b.link_data
    return A_LESS_THAN_B
  return B_LESS_THAN_A
```

Figure 14: Rules for ranking multiple interfaces. Order is from low to high.

4.1. MRT Island Identification

The local MRT Island for a particular MRT profile can be determined by starting from the computing router in the network graph and doing a breadth-first-search (BFS), exploring only links that aren't MRT-ineligible.

```
MRT_Island_Identification(topology, computing_rtr, profile_id)
  for all routers in topology
    rtr.IN_MRT_ISLAND = FALSE
```



```
computing_rtr.IN_MRT_ISLAND = TRUE
explore_list = { computing_rtr }
while (explore_list is not empty)
  next_rtr = remove_head(explore_list)
  for each interface in next_rtr
    if interface is not MRT-ineligible
      if ((interface.remote_node supports profile_id) and
          (interface.remote_node.IN_MRT_ISLAND is FALSE))
        interface.remote_node.IN_MRT_ISLAND = TRUE
        add_to_tail(explore_list, interface.remote_node)
```

Figure 15: MRT Island Identification

4.2. Root Selection

In [I-D.atlas-ospf-mrt], a mechanism is given for routers to advertise the GADAG Root Selection Priority and consistently select a GADAG Root inside the local MRT Island. Before beginning computation, the network graph is reduced to contain only the set of routers that support the specific MRT profile whose MRTs are being computed.

Off-line analysis that considers the centrality of a router may help determine how good a choice a particular router is for the role of GADAG root.

4.3. Initialization

Before running the algorithm, there is the standard type of initialization to be done, such as clearing any computed DFS-values, lowpoint-values, DFS-parents, lowpoint-parents, any MRT-computed next-hops, and flags associated with algorithm.

It is assumed that a regular SPF computation has been run so that the primary next-hops from the computing router to each destination are known. This is required for determining alternates at the last step.

Initially, all interfaces must be initialized to UNDIRECTED. Whether they are OUTGOING, INCOMING or both is determined when the GADAG is constructed and augmented.

It is possible that some links and nodes will be marked as unusable, whether because of configuration, IGP flooding (e.g. MRT-ineligible links in [I-D.atlas-ospf-mrt]), overload, or due to a transient cause such as [RFC3137]. In the algorithm description, it is assumed that such links and nodes will not be explored or used and no more discussion is given of this restriction.

4.4. MRT Lowpoint Algorithm: Computing GADAG using lowpoint inheritance

As discussed in Section 3.2, it is necessary to find ears from a node *x* that is already in the GADAG (known as IN_GADAG). There are two methods to find ears; both are required. The first is by going to a not IN_GADAG DFS-child and then following the chain of low-point parents until an IN_GADAG node is found. The second is by going to a not IN_GADAG neighbor and then following the chain of DFS parents until an IN_GADAG node is found. As an ear is found, the associated interfaces are marked based on the direction taken. The nodes in the ear are marked as IN_GADAG. In the algorithm, first the ears via DFS-children are found and then the ears via DFS-neighbors are found.

By adding both types of ears when an IN_GADAG node is processed, all ears that connect to that node are found. The order in which the IN_GADAG nodes is processed is, of course, key to the algorithm. The order is a stack of ears so the most recent ear is found at the top of the stack. Of course, the stack stores nodes and not ears, so an ordered list of nodes, from the first node in the ear to the last node in the ear, is created as the ear is explored and then that list is pushed onto the stack.

Each ear represents a partial order (see Figure 4) and processing the nodes in order along each ear ensures that all ears connecting to a node are found before a node higher in the partial order has its ears explored. This means that the direction of the links in the ear is always from the node *x* being processed towards the other end of the ear. Additionally, by using a stack of ears, this means that any unprocessed nodes in previous ears can only be ordered higher than nodes in the ears below it on the stack.

In this algorithm that depends upon Low-Point inheritance, it is necessary that every node have a low-point parent that is not itself. If a node is a cut-vertex, that may not yet be the case. Therefore, any nodes without a low-point parent will have their low-point parent set to their DFS parent and their low-point value set to the DFS-value of their parent. This assignment also properly allows an ear between two cut-vertices.

Finally, the algorithm simultaneously computes each node's local-root, as described in Figure 12. This is further elaborated as follows. The local-root can be inherited from the node at the end of the ear unless the end of the ear is *x* itself, in which case the local-root for all the nodes in the ear would be *x*. This is because whenever the first cycle is found in a block, or an ear involving a bridge is computed, the cut-vertex closest to the root would be *x* itself. In all other scenarios, the properties of lowpoint/dfs parents ensure that the end of the ear will be in the same block, and

thus inheriting its local-root would be the correct local-root for all newly added nodes.

The pseudo-code for the GADAG algorithm (assuming that the adjustment of lowpoint for cut-vertices has been made) is shown in Figure 16.

```
Construct_Ear(x, Stack, intf, type)
    ear_list = empty
    cur_node = intf.remote_node
    cur_intf = intf
    not_done = true

    while not_done
        cur_intf.UNDIRECTED = false
        cur_intf.OUTGOING = true
        cur_intf.remote_intf.UNDIRECTED = false
        cur_intf.remote_intf.INCOMING = true

        if cur_node.IN_GADAG is false
            cur_node.IN_GADAG = true
            add_to_list_end(ear_list, cur_node)
            if type is CHILD
                cur_intf = cur_node.lowpoint_parent_intf
                cur_node = cur_node.lowpoint_parent
            else type must be NEIGHBOR
                cur_intf = cur_node.dfs_parent_intf
                cur_node = cur_node.dfs_parent
        else
            not_done = false

    if (type is CHILD) and (cur_node is x)
        //x is a cut-vertex and the local root for
        //the block in which the ear is computed
        localroot = x
    else
        // Inherit local-root from the end of the ear
        localroot = cur_node.localroot
    while ear_list is not empty
        y = remove_end_item_from_list(ear_list)
        y.localroot = localroot
        push(Stack, y)

Construct_GADAG_via_Lowpoint(topology, root)
    root.IN_GADAG = true
    root.localroot = root
    Initialize Stack to empty
    push root onto Stack
    while (Stack is not empty)
```

```

x = pop(Stack)
foreach interface intf of x
  if ((intf.remote_node.IN_GADAG == false) and
      (intf.remote_node.dfs_parent is x))
    Construct_Ear(x, Stack, intf, CHILD)
foreach interface intf of x
  if ((intf.remote_node.IN_GADAG == false) and
      (intf.remote_node.dfs_parent is not x))
    Construct_Ear(x, Stack, intf, NEIGHBOR)

Construct_GADAG_via_Lowpoint(topology, root)

```

Figure 16: Low-point Inheritance GADAG algorithm

4.5. Augmenting the GADAG by directing all links

The GADAG, regardless of the algorithm used to construct it, at this point could be used to find MRTs but the topology does not include all links in the network graph. That has two impacts. First, there might be shorter paths that respect the GADAG partial ordering and so the alternate paths would not be as short as possible. Second, there may be additional paths between a router *x* and the root that are not included in the GADAG. Including those provides potentially more bandwidth to traffic flowing on the alternates and may reduce congestion compared to just using the GADAG as currently constructed.

The goal is thus to assign direction to every remaining link marked as `UNDIRECTED` to improve the paths and number of paths found when the MRTs are computed.

To do this, we need to establish a total order that respects the partial order described by the GADAG. This can be done using Kahn's topological sort [[Kahn_1962_topo_sort](#)] which essentially assigns a number to a node *x* only after all nodes before it (e.g. with a link incoming to *x*) have had their numbers assigned. The only issue with the topological sort is that it works on DAGs and not ADAGs or GADAGs.

To convert a GADAG to a DAG, it is necessary to remove all links that point to a root of block from within that block. That provides the necessary conversion to a DAG and then a topological sort can be done. Finally, all `UNDIRECTED` links are assigned a direction based upon the partial ordering. Any `UNDIRECTED` links that connect to a root of a block from within that block are assigned a direction `INCOMING` to that root. The exact details of this whole process are captured in Figure 17

```

Set_Block_Root_Incoming_Links(topo, root, mark_or_clear)
  foreach node x in topo
    if node x is a cut-vertex or root
      foreach interface i of x
        if (i.remote_node.localroot is x)
          if i.UNDIRECTED
            i.OUTGOING = true
            i.remote_intf.INCOMING = true
            i.UNDIRECTED = false
            i.remote_intf.UNDIRECTED = false
          if i.INCOMING
            if mark_or_clear is mark
              if i.OUTGOING // a cut-edge
                i.STORE_INCOMING = true
                i.INCOMING = false
                i.remote_intf.STORE_OUTGOING = true
                i.remote_intf.OUTGOING = false
                i.TEMP_UNUSABLE = true
                i.remote_intf.TEMP_UNUSABLE = true
              else
                i.TEMP_UNUSABLE = false
                i.remote_intf.TEMP_UNUSABLE = false
            if i.STORE_INCOMING and (mark_or_clear is clear)
              i.INCOMING = true
              i.STORE_INCOMING = false
              i.remote_intf.OUTGOING = true
              i.remote_intf.STORE_OUTGOING = false

Run_Topological_Sort_GADAG(topo, root)
  Set_Block_Root_Incoming_Links(topo, root, MARK)
  foreach node x
    set x.unvisited to the count of x's incoming interfaces
    that aren't marked TEMP_UNUSABLE
  Initialize working_list to empty
  Initialize topo_order_list to empty
  add_to_list_end(working_list, root)
  while working_list is not empty
    y = remove_start_item_from_list(working_list)
    add_to_list_end(topo_order_list, y)
    foreach interface i of y
      if (i.OUTGOING) and (not i.TEMP_UNUSABLE)
        i.remote_node.unvisited -= 1
        if i.remote_node.unvisited is 0
          add_to_list_end(working_list, i.remote_node)
  next_topo_order = 1
  while topo_order_list is not empty
    y = remove_start_item_from_list(topo_order_list)
    y.topo_order = next_topo_order

```

```

        next_topo_order += 1
    Set_Block_Root_Incoming_Links(topo, root, CLEAR)

Add_Undirected_Links(topo, root)
Run_Topological_Sort_GADAG(topo, root)
foreach node x in topo
    foreach interface i of x
        if i.UNDIRECTED
            if x.topo_order < i.remote_node.topo_order
                i.OUTGOING = true
                i.UNDIRECTED = false
                i.remote_intf.INCOMING = true
                i.remote_intf.UNDIRECTED = false
            else
                i.INCOMING = true
                i.UNDIRECTED = false
                i.remote_intf.OUTGOING = true
                i.remote_intf.UNDIRECTED = false

Add_Undirected_Links(topo, root)

```

Figure 17: Assigning direction to UNDIRECTED links

Proxy-nodes do not need to be added to the network graph. They cannot be transited and do not affect the MRTs that are computed. The details of how the MRT-Blue and MRT-Red next-hops are computed and how the appropriate alternate next-hops are selected is given in Section 4.8.

4.6. Compute MRT next-hops

As was discussed in Section 3.1, once a ADAG is found, it is straightforward to find the next-hops from any node X to the ADAG root. However, in this algorithm, we want to reuse the common GADAG and find not only the one pair of MRTs rooted at the GADAG root with it, but find a pair rooted at each node. This is useful since it is significantly faster to compute. It may also provide easier troubleshooting of the MRT-Red and MRT-Blue.

The method for computing differently rooted MRTs from the common GADAG is based on two ideas. First, if two nodes X and Y are ordered with respect to each other in the partial order, then an SPF along OUTGOING links (an increasing-SPF) and an SPF along INCOMING links (a decreasing-SPF) can be used to find the increasing and decreasing paths. Second, if two nodes X and Y aren't ordered with respect to each other in the partial order, then intermediary nodes can be used to create the paths by increasing/decreasing to the intermediary and then decreasing/increasing to reach Y.

As usual, the two basic ideas will be discussed assuming the network is two-connected. The generalization to multiple blocks is discussed in Section 4.6.4. The full algorithm is given in Section 4.6.5.

4.6.1. MRT next-hops to all nodes partially ordered with respect to the computing node

To find two node-disjoint paths from the computing router X to any node Y, depends upon whether $Y \gg X$ or $Y \ll X$. As shown in Figure 18, if $Y \gg X$, then there is an increasing path that goes from X to Y without crossing R; this contains nodes in the interval $[X, Y]$. There is also a decreasing path that decreases towards R and then decreases from R to Y; this contains nodes in the interval $[X, R\text{-small}]$ or $[R\text{-great}, Y]$. The two paths cannot have common nodes other than X and Y.

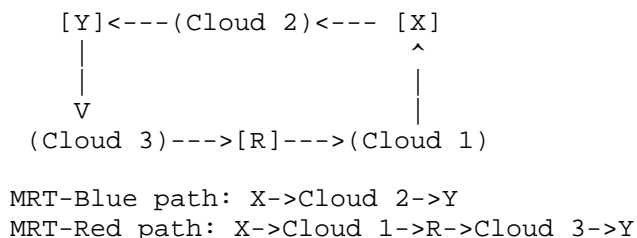


Figure 18: $Y \gg X$

Similar logic applies if $Y \ll X$, as shown in Figure 19. In this case, the increasing path from X increases to R and then increases from R to Y to use nodes in the intervals $[X, R\text{-great}]$ and $[R\text{-small}, Y]$. The decreasing path from X reaches Y without crossing R and uses nodes in the interval $[Y, X]$.

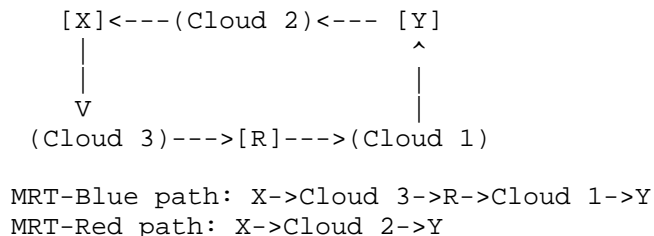


Figure 19: $Y \ll X$

4.6.2. MRT next-hops to all nodes not partially ordered with respect to the computing node

When X and Y are not ordered, the first path should increase until we get to a node G , where $G \gg Y$. At G , we need to decrease to Y . The other path should be just the opposite: we must decrease until we get to a node H , where $H \ll Y$, and then increase. Since R is smaller and greater than Y , such G and H must exist. It is also easy to see that these two paths must be node disjoint: the first path contains nodes in interval $[X, G]$ and $[Y, G]$, while the second path contains nodes in interval $[H, X]$ and $[H, Y]$. This is illustrated in Figure 20. It is necessary to decrease and then increase for the MRT-Blue and increase and then decrease for the MRT-Red; if one simply increased for one and decreased for the other, then both paths would go through the root R .

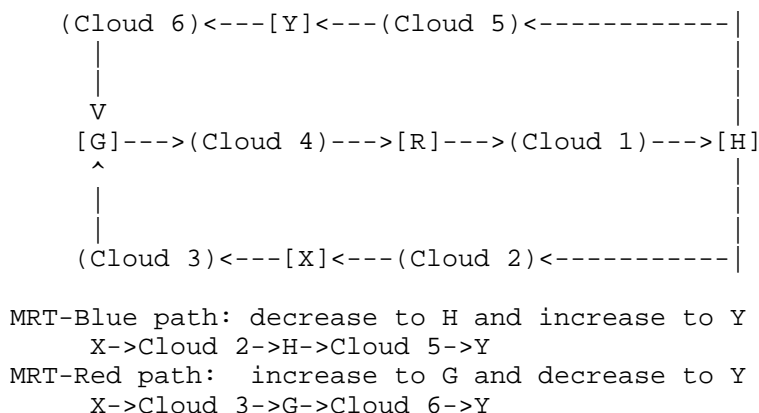


Figure 20: X and Y unordered

This gives disjoint paths as long as G and H are not the same node. Since $G \gg Y$ and $H \ll Y$, if G and H could be the same node, that would have to be the root R . This is not possible because there is only one incoming interface to the root R which is created when the initial cycle is found. Recall from Figure 6 that whenever an ear was found to have an end that was the root R , the ear was directed from R so that the associated interface on R is outgoing and not incoming. Therefore, there must be exactly one node M which is the largest one before R , so the MRT-Red path will never reach R ; it will turn at M and decrease to Y .

4.6.3. Computing Redundant Tree next-hops in a 2-connected Graph

The basic ideas for computing RT next-hops in a 2-connected graph were given in Section 4.6.1 and Section 4.6.2. Given these two ideas, how can we find the trees?

If some node X only wants to find the next-hops (which is usually the case for IP networks), it is enough to find which nodes are greater and less than X, and which are not ordered; this can be done by running an increasing-SPF and a decreasing-SPF rooted at X and not exploring any links from the ADAG root. (Traversal algorithms other than SPF could safely be used instead where one traversal takes the links in their given directions and the other reverses the links' directions.)

An increasing-SPF rooted at X and not exploring links from the root will find the increasing next-hops to all $Y \gg X$. Those increasing next-hops are X's next-hops on the MRT-Blue to reach Y. An decreasing-SPF rooted at X and not exploring links from the root will find the decreasing next-hops to all $Z \ll X$. Those decreasing next-hops are X's next-hops on the MRT-Red to reach Z. Since the root R is both greater than and less than X, after this increasing-SPF and decreasing-SPF, X's next-hops on the MRT-Blue and on the MRT-Red to reach R are known. For every node $Y \gg X$, X's next-hops on the MRT-Red to reach Y are set to those on the MRT-Red to reach R. For every node $Z \ll X$, X's next-hops on the MRT-Blue to reach Z are set to those on the MRT-Blue to reach R.

For those nodes, which were not reached, we have the next-hops as well. The increasing MRT-Blue next-hop for a node, which is not ordered, is the next-hop along the decreasing MRT-Red towards R and the decreasing MRT-Red next-hop is the next-hop along the increasing MRT-Blue towards R. Naturally, since R is ordered with respect to all the nodes, there will always be an increasing and a decreasing path towards it. This algorithm does not provide the complete specific path taken but just the appropriate next-hops to use. The identity of G and H is not determined.

The final case to considered is when the root R computes its own next-hops. Since the root R is \ll all other nodes, running an increasing-SPF rooted at R will reach all other nodes; the MRT-Blue next-hops are those found with this increasing-SPF. Similarly, since the root R is \gg all other nodes, running a decreasing-SPF rooted at R will reach all other nodes; the MRT-Red next-hops are those found with this decreasing-SPF.



(a) (b)
A 2-connected graph A spanning ADAG rooted at R

Figure 21

As an example consider the situation depicted in Figure 21. There node C runs an increasing-SPF and a decreasing-SPF. The increasing-SPF reaches D, E and R and the decreasing-SPF reaches B, A and R. So towards E the increasing next-hop is D (it was reached through D), and the decreasing next-hop is B (since R was reached through B). Since both D and B, A and R will compute the next hops similarly, the packets will reach E.

We have the next-hops towards F as well: since F is not ordered with respect to C, the MRT-Blue next-hop is the decreasing one towards R (which is B) and the MRT-Red next-hop is the increasing one towards R (which is D). Since B is ordered with F, it will find, for its MRT-Blue, a real increasing next-hop, so packet forwarded to B will get to F on path C-B-F. Similarly, D will have, for its MRT-Red, a real decreasing next-hop, and the packet will use path C-D-F.

4.6.4. Generalizing for graph that isn't 2-connected

If a graph isn't 2-connected, then the basic approach given in Section 4.6.3 needs some extensions to determine the appropriate MRT next-hops to use for destinations outside the computing router X's blocks. In order to find a pair of maximally redundant trees in that graph we need to find a pair of RTs in each of the blocks (the root of these trees will be discussed later), and combine them.

When computing the MRT next-hops from a router X, there are three basic differences:

1. Only nodes in a common block with X should be explored in the increasing-SPF and decreasing-SPF.
2. Instead of using the GADAG root, X's local-root should be used. This has the following implications:
 - a. The links from X's local-root should not be explored.
 - b. If a node is explored in the outgoing SPF so $Y \gg X$, then X's MRT-Red next-hops to reach Y uses X's MRT-Red next-hops to reach X's local-root and if $Z \ll X$, then X's MRT-Blue next-hops to reach Z uses X's MRT-Blue next-hops to reach X's local-root.

- c. If a node W in a common block with X was not reached in the increasing-SPF or decreasing-SPF, then W is unordered with respect to X. X's MRT-Blue next-hops to W are X's decreasing aka MRT-Red next-hops to X's local-root. X's MRT-Red next-hops to W are X's increasing aka Blue MRT next-hops to X's local-root.
- 3. For nodes in different blocks, the next-hops must be inherited via the relevant cut-vertex.

These are all captured in the detailed algorithm given in Section 4.6.5.

4.6.5. Complete Algorithm to Compute MRT Next-Hops

The complete algorithm to compute MRT Next-Hops for a particular router X is given in Figure 22. In addition to computing the MRT-Blue next-hops and MRT-Red next-hops used by X to reach each node Y, the algorithm also stores an "order_proxy", which is the proper cut-vertex to reach Y if it is outside the block, and which is used later in deciding whether the MRT-Blue or the MRT-Red can provide an acceptable alternate for a particular primary next-hop.

```

In_Common_Block(x, y)
    if (((x.localroot is y.localroot) and (x.block_id is y.block_id))
        or (x is y.localroot) or (y is x.localroot))
        return true
    return false

Store_Results(y, direction, spf_root, store_nhs)
    if direction is FORWARD
        y.higher = true
        if store_nhs
            y.blue_next_hops = y.next_hops
    if direction is REVERSE
        y.lower = true
        if store_nhs
            y.red_next_hops = y.next_hops

SPF_No_Traverse_Root(spf_root, block_root, direction, store_nhs)
    Initialize spf_heap to empty
    Initialize nodes' spf_metric to infinity and next_hops to empty
    spf_root.spf_metric = 0
    insert(spf_heap, spf_root)
    while (spf_heap is not empty)
        min_node = remove_lowest(spf_heap)
        Store_Results(min_node, direction, spf_root, store_nhs)
        if ((min_node is spf_root) or (min_node is not block_root))

```

```

    foreach interface intf of min_node
        if (((direction is FORWARD) and intf.OUTGOING) or
            ((direction is REVERSE) and intf.INCOMING) and
            In_Common_Block(spf_root, intf.remote_node))
            path_metric = min_node.spf_metric + intf.metric
            if path_metric < intf.remote_node.spf_metric
                intf.remote_node.spf_metric = path_metric
                if min_node is spf_root
                    intf.remote_node.next_hops = make_list(intf)
                else
                    intf.remote_node.next_hops = min_node.next_hops
                    insert_or_update(spf_heap, intf.remote_node)
            else if path_metric is intf.remote_node.spf_metric
                if min_node is spf_root
                    add_to_list(intf.remote_node.next_hops, intf)
                else
                    add_list_to_list(intf.remote_node.next_hops,
                                    min_node.next_hops)

SetEdge(y)
    if y.blue_next_hops is empty and y.red_next_hops is empty
        if (y.local_root != y) {
            SetEdge(y.localroot)
        }
        y.blue_next_hops = y.localroot.blue_next_hops
        y.red_next_hops = y.localroot.red_next_hops
        y.order_proxy = y.localroot.order_proxy

Compute_MRT_NextHops(x, root)
    foreach node y
        y.higher = y.lower = false
        clear y.red_next_hops and y.blue_next_hops
        y.order_proxy = y
        SPF_No_Traverse_Root(x, x.localroot, FORWARD, TRUE)
        SPF_No_Traverse_Root(x, x.localroot, REVERSE, TRUE)

    // red and blue next-hops are stored to x.localroot as different
    // paths are found via the SPF and reverse-SPF.
    // Similarly any nodes whose local-root is x will have their
    // red_next_hops and blue_next_hops already set.

    // Handle nodes in the same block that aren't the local-root
    foreach node y
        if (y.IN_MRT_ISLAND and (y is not x) and
            (y.localroot is x.localroot) and
            ((y is x.localroot) or (x is y.localroot) or
             (y.block_id is x.block_id)))
            if y.higher

```

```

        y.red_next_hops = x.localroot.red_next_hops
    else if y.lower
        y.blue_next_hops = x.localroot.blue_next_hops
    else
        y.blue_next_hops = x.localroot.red_next_hops
        y.red_next_hops = x.localroot.blue_next_hops

    // Inherit next-hops and order_proxies to other components
    if x is not root
        root.blue_next_hops = x.localroot.blue_next_hops
        root.red_next_hops = x.localroot.red_next_hops
        root.order_proxy = x.localroot
    foreach node y
        if (y is not root) and (y is not x) and y.IN_MRT_ISLAND
            SetEdge(y)

max_block_id = 0
Assign_Block_ID(root, max_block_id)
Compute_MRT_NextHops(x, root)

```

Figure 22

4.7. Identify MRT alternates

At this point, a computing router S knows its MRT-Blue next-hops and MRT-Red next-hops for each destination in the MRT Island. The primary next-hops along the SPT are also known. It remains to determine for each primary next-hop to a destination D, which of the MRTs avoids the primary next-hop node F. This computation depends upon data set in Compute_MRT_NextHops such as each node y's y.blue_next_hops, y.red_next_hops, y.order_proxy, y.higher, y.lower and topo_orders. Recall that any router knows only which are the nodes greater and lesser than itself, but it cannot decide the relation between any two given nodes easily; that is why we need topological ordering.

For each primary next-hop node F to each destination D, S can call Select_Alternates(S, D, F, primary_intf) to determine whether to use the MRT-Blue next-hops as the alternate next-hop(s) for that primary next hop or to use the MRT-Red next-hops. The algorithm is given in Figure 23 and discussed afterwards.

```

Select_Alternates_Internal(S, D, F, primary_intf,
                          D_lower, D_higher, D_topo_order)

    //When D==F, we can do only link protection
    if ((D is F) or (D.order_proxy is F))

```

```
    if an MRT doesn't use primary_intf
        indicate alternate is not node-protecting
        return that MRT color
    else // parallel links are cut-edge
        return AVOID_LINK_ON_BLUE

if (D_lower and D_higher and F_lower and F_higher)
    if F_topo_order < D_topo_order
        return USE_RED
    else
        return USE_BLUE

if (D_lower and D_higher)
    if F_higher
        return USE_RED
    else
        return USE_BLUE

if (F_lower and F_higher)
    if D_lower
        return USE_RED
    else if D_higher
        return USE_BLUE
    else
        if primary_intf.OUTGOING and primary_intf.INCOMING
            return AVOID_LINK_ON_BLUE
        if primary_intf.OUTGOING is true
            return USE_BLUE
        if primary_intf.INCOMING is true
            return USE_RED

if D_higher
    if F_higher
        if F_topo_order < D_topo_order
            return USE_RED
        else
            return USE_BLUE
    else if F_lower
        return USE_BLUE
    else
        // F and S are neighbors so either F << S or F >> S
else if D_lower
    if F_higher
        return USE_RED
    else if F_lower
        if F_topo_order < D_topo_order
            return USE_RED
        else
```

```

        return USE_BLUE
    else
        // F and S are neighbors so either F << S or F >> S
    else // D and S not ordered
        if F.lower
            return USE_RED
        else if F.higher
            return USE_BLUE
        else
            // F and S are neighbors so either F << S or F >> S

Select_Alternates(S, D, F, primary_intf)
    if D.order_proxy is not D
        D_lower = D.order_proxy.lower
        D_higher = D.order_proxy.higher
        D_topo_order = D.order_proxy.topo_order
    else
        D_lower = D.lower
        D_higher = D.higher
        D_topo_order = D.topo_order
    return Select_Alternates_Internal(S, D, F, primary_intf,
                                     D_lower, D_higher, D_topo_order)

```

Figure 23

If either $D \gg S \gg F$ or $D \ll S \ll F$ holds true, the situation is simple: in the first case we should choose the increasing Blue next-hop, in the second case, the decreasing Red next-hop is the right choice.

However, when both D and F are greater than S the situation is not so simple, there can be three possibilities: (i) $F \gg D$ (ii) $F \ll D$ or (iii) F and D are not ordered. In the first case, we should choose the path towards D along the Blue tree. In contrast, in case (ii) the Red path towards the root and then to D would be the solution. Finally, in case (iii) both paths would be acceptable. However, observe that if e.g. $F.topo_order > D.topo_order$, either case (i) or case (iii) holds true, which means that selecting the Blue next-hop is safe. Similarly, if $F.topo_order < D.topo_order$, we should select the Red next-hop. The situation is almost the same if both F and D are less than S.

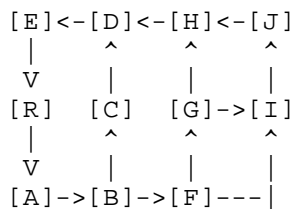
Recall that we have added each link to the GADAG in some direction, so it is impossible that S and F are not ordered. But it is possible that S and D are not ordered, so we need to deal with this case as well. If $F < S$, we can use the Red next-hop, because that path is first increasing until a node definitely greater than D is reached, then decreasing; this path must avoid using F. Similarly, if $F > S$, we should use the Blue next-hop.

Additionally, the cases where either F or D is ordered both higher and lower must be considered; this can happen when one is a block-root or its order_proxy is. If D is both higher and lower than S, then the MRT to use is the one that avoids F so if F is higher, then the MRT-Red should be used and if F is lower, then the MRT-Blue should be used; F and S must be ordered because they are neighbors. If F is both higher and lower, then if D is lower, using the MRT-Red to decrease reaches D and if D is higher, using the Blue MRT to increase reaches D; if D is unordered compared to S, then the situation is a bit more complicated.

In the case where $F < S < F$ and D and S are unordered, the direction of the link in the GADAG between S and F should be examined. If the link is directed $S \rightarrow F$, then use the MRT-Blue (decrease to avoid that link and then increase). If the link is directed $S \leftarrow F$, then use the MRT-Red (increase to avoid that link and then decrease). If the link is $S \leftrightarrow F$, then the link must be a cut-link and there is no node-protecting alternate. If there are multiple links between S and F, then they can protect against each other; of course, in this situation, they are probably already ECMP.

Finally, there is the case where D is also F. In this case, only link protection is possible. The MRT that doesn't use the indicated primary next-hop is used. If both MRTs use the primary next-hop, then the primary next-hop must be a cut-edge so either MRT could be used but the set of MRT next-hops must be pruned to avoid that primary next-hop. To indicate this case, `Select_Alternates` returns `AVOID_LINK_ON_BLUE`.

As an example, consider the ADAG depicted in Figure 24 and first suppose that G is the source, D is the destination and H is the failed next-hop. Since $D > G$, we need to compare `H.topo_order` and `D.topo_order`. Since $D.topo_order > H.topo_order$, D must be not smaller than H, so we should select the decreasing path towards the root. If, however, the destination were instead J, we must find that $H.topo_order > J.topo_order$, so we must choose the increasing Blue next-hop to J, which is I. In the case, when instead the destination is C, we find that we need to first decrease to avoid using H, so the Blue, first decreasing then increasing, path is selected.



(a)

a 2-connected graph

Figure 24

4.8. Finding FRR Next-Hops for Proxy-Nodes

As discussed in Section 10.2 of [I-D.ietf-rtgwg-mrt-frr-architecture], it is necessary to find MRT-Blue and MRT-Red next-hops and MRT-FRR alternates for a named proxy-nodes. An example case is for a router that is not part of that local MRT Island, when there is only partial MRT support in the domain.

A first incorrect and naive approach to handling proxy-nodes, which cannot be transited, is to simply add these proxy-nodes to the graph of the network and connect it to the routers through which the new proxy-node can be reached. Unfortunately, this can introduce some new ordering between the border routers connected to the new node which could result in routing MRT paths through the proxy-node. Thus, this naive approach would need to recompute GADAGs and redo SPTs for each proxy-node.

Instead of adding the proxy-node to the original network graph, each individual proxy-node can be individually added to the GADAG. The proxy-node is connected to at most two nodes in the GADAG. Section 10.2 of [I-D.ietf-rtgwg-mrt-frr-architecture] defines how the proxy-node attachments MUST be determined. The degenerate case where the proxy-node is attached to only one node in the GADAG is trivial as all needed information can be derived from that attachment node; if there are different interfaces, then some can be assigned to MRT-Red and others to MRT-Blue.

Now, consider the proxy-node that is attached to exactly two nodes in the GADAG. Let the `order_proxies` of these nodes be A and B. Let the current node, where next-hop is just being calculated, be S. If one of these two nodes A and B is the local root of S, let `A=S.local_root` and the other one be B. Otherwise, let `A.topo_order < B.topo_order`.

A valid GADAG was constructed. Instead doing an increasing-SPF and a decreasing-SPF to find ordering for the proxy-nodes, the following simple rules, providing the same result, can be used independently for each different proxy-node. For the following rules, let $X=A.local_root$, and if A is the local root, let that be strictly lower than any other node. Always take the first rule that matches.

| Rule | Condition | Blue NH | Red NH | Notes |
|------|-----------|-----------|-----------|----------------------|
| 1 | $S=X$ | Blue to A | Red to B | |
| 2 | $S<<A$ | Blue to A | Red to R | |
| 3 | $S>>B$ | Blue to R | Red to B | |
| 4 | $A<<S<<B$ | Red to A | Blue to B | |
| 5 | $A<<S$ | Red to A | Blue to R | S not ordered w/ B |
| 6 | $S<<B$ | Red to R | Blue to B | S not ordered w/ A |
| 7 | Otherwise | Red to R | Blue to R | S not ordered w/ A+B |

These rules are realized in the following pseudocode where P is the proxy-node, X and Y are the nodes that P is attached to, and S is the computing router:

```

Select_Proxy_Node_NHs(P, S, X, Y)
  if (X.order_proxy.topo_order < Y.order_proxy.topo_order)
    //This fits even if X.order_proxy=S.local_root
    A=X.order_proxy
    B=Y.order_proxy
  else
    A=Y.order_proxy
    B=X.order_proxy

  if (S==A.local_root)
    P.blue_next_hops = A.blue_next_hops
    P.red_next_hops  = B.red_next_hops
    return
  if (A.higher)
    P.blue_next_hops = A.blue_next_hops
    P.red_next_hops  = R.red_next_hops
    return
  if (B.lower)
    P.blue_next_hops = R.blue_next_hops
    P.red_next_hops  = B.red_next_hops
    return
  if (A.lower && B.higher)
    P.blue_next_hops = A.red_next_hops
    P.red_next_hops  = B.blue_next_hops
    return
  if (A.lower)

```

```

        P.blue_next_hops = R.red_next_hops
        P.red_next_hops  = B.blue_next_hops
        return
    if (B.higher)
        P.blue_next_hops = A.red_next_hops
        P.red_next_hops  = R.blue_next_hops
        return
    P.blue_next_hops = R.red_next_hops
    P.red_next_hops  = R.blue_next_hops
    return

```

After finding the the red and the blue next-hops, it is necessary to know which one of these to use in the case of failure. This can be done by `Select_Alternates_Inner()`. In order to use `Select_Alternates_Internal()`, we need to know if P is greater, less or unordered with S, and P.topo_order. P.lower = B.lower, P.higher = A.higher, and any value is OK for P.topo_order, until A.topo_order<=P.topo_order<=B.topo_order and P.topo_order is not equal to the topo_order of the failed node. So for simplicity let P.topo_order=A.topo_order when the next-hop is not A, and P.topo_order=B.topo_order otherwise. This gives the following pseudo-code:

```

Select_Alternates_Proxy_Node(S, P, F, primary_intf)
    if (F is not P.neighbor_A)
        return Select_Alternates_Internal(S, P, F, primary_intf,
                                           P.neighbor_B.lower,
                                           P.neighbor_A.higher,
                                           P.neighbor_A.topo_order)
    else
        return Select_Alternates_Internal(S, P, F, primary_intf,
                                           P.neighbor_B.lower,
                                           P.neighbor_A.higher,
                                           P.neighbor_B.topo_order)

```

Figure 25

5. MRT Lowpoint Algorithm: Complete Specification

This specification defines the MRT Lowpoint Algorithm, which include the construction of a common GADAG and the computation of MRT-Red and MRT-Blue next-hops to each node in the graph. An implementation MAY select any subset of next-hops for MRT-Red and MRT-Blue that respect the available nodes that are described in Section 4.6 for each of the MRT-Red and MRT-Blue and the selected next-hops are further along in the interval of allowed nodes towards the destination.

For example, the MRT-Blue next-hops used when the destination $Y \gg S$, the computing router, MUST be one or more nodes, T , whose `topo_order` is in the interval $[X.topo_order, Y.topo_order]$ and where $Y \gg T$ or Y is T . Similarly, the MRT-Red next-hops MUST be have a `topo_order` in the interval $[R-small.topo_order, X.topo_order]$ or $[Y.topo_order, R-big.topo_order]$.

Implementations SHOULD implement the `Select_Alternates()` function to pick an MRT-FRR alternate.

In a future version, this section will include pseudo-code describing the full code path through the pseudo-code given earlier in the draft.

6. Algorithm Alternatives and Evaluation

This specification defines the MRT Lowpoint Algorithm, which is one option among several possible MRT algorithms. Other alternatives are described in the appendices.

In addition, it is possible to calculate Destination-Rooted GADAG, where for each destination, a GADAG rooted at that destination is computed. Then a router can compute the blue MRT and red MRT next-hops to that destination. Building GADAGs per destination is computationally more expensive, but may give somewhat shorter alternate paths. It may be useful for live-live multicast along MRTs.

6.1. Algorithm Evaluation

This section compares MRT and remote LFA for IP Fast Reroute in 19 service provider network topologies, focusing on coverage and alternate path length. Figure 26 shows the node-protecting coverage provided by local LFA (LLFA), remote LFA (RLFA), and MRT against different failure scenarios in these topologies. The coverage values are calculated as the percentage of source-destination pairs protected by the given IPFRR method relative to those protectable by optimal routing, against the same failure modes. More details on alternate selection policies used for this analysis are described later in this section.

| Topology | percentage of failure scenarios covered by IPFRR method | | |
|----------|---|---------|-----|
| | NP_LLFA | NP_RLFA | MRT |
| T201 | 37 | 90 | 100 |
| T202 | 73 | 83 | 100 |
| T203 | 51 | 80 | 100 |
| T204 | 55 | 81 | 100 |
| T205 | 92 | 93 | 100 |
| T206 | 71 | 74 | 100 |
| T207 | 57 | 74 | 100 |
| T208 | 66 | 81 | 100 |
| T209 | 79 | 79 | 100 |
| T210 | 95 | 98 | 100 |
| T211 | 68 | 71 | 100 |
| T212 | 59 | 63 | 100 |
| T213 | 84 | 84 | 100 |
| T214 | 68 | 78 | 100 |
| T215 | 84 | 88 | 100 |
| T216 | 43 | 59 | 100 |
| T217 | 78 | 88 | 100 |
| T218 | 72 | 75 | 100 |
| T219 | 78 | 84 | 100 |

Figure 26

For the topologies analyzed here, LLFA is able to provide node-protecting coverage ranging from 37% to 95% of the source-destination pairs, as seen in the column labeled NP_LLFA. The use of RLFA in addition to LLFA is generally able to increase the node-protecting coverage. The percentage of node-protecting coverage with RLFA is provided in the column labeled NP_RLFA, ranges from 59% to 98% for these topologies. The node-protecting coverage provided by MRT is 100% since MRT is able to provide protection for any source-destination pair for which a path still exists after the failure.

We would also like to measure the quality of the alternate paths produced by these different IPFRR methods. An obvious approach is to take an average of the alternate path costs over all source-destination pairs and failure modes. However, this presents a problem, which we will illustrate by presenting an example of results for one topology using this approach (Figure 27). In this table, the average relative path length is the alternate path length for the IPFRR method divided by the optimal alternate path length, averaged

over all source-destination pairs and failure modes. The first three columns of data in the table give the path length calculated from the sum of IGP metrics of the links in the path. The results for topology T208 show that the metric-based path lengths for NP_LLFA and NP_RLFA alternates are on average 78 and 66 times longer than the path lengths for optimal alternates. The metric-based path lengths for MRT alternates are on average 14 times longer than for optimal alternates.

| Topology | average relative alternate path length | | | | | |
|----------|--|---------|------|----------|---------|------|
| | IGP metric | | | hopcount | | |
| | NP_LLFA | NP_RLFA | MRT | NP_LLFA | NP_RLFA | MRT |
| T208 | 78.2 | 66.0 | 13.6 | 0.99 | 1.01 | 1.32 |

Figure 27

The network topology represented by T208 uses values of 10, 100, and 1000 as IGP costs, so small deviations from the optimal alternate path can result in large differences in relative path length. LLFA, RLFA, and MRT all allow for at least one hop in the alternate path to be chosen independent of the cost of the link. This can easily result in an alternate using a link with cost 1000, which introduces noise into the path length measurement. In the case of T208, the adverse effects of using metric-based path lengths is obvious. However, we have observed that the metric-based path length introduces noise into alternate path length measurements in several other topologies as well. For this reason, we have opted to measure the alternate path length using hopcount. While IGP metrics may be adjusted by the network operator for a number of reasons (e.g. traffic engineering), the hopcount is a fairly stable measurement of path length. As shown in the last three columns of Figure 27, the hopcount-based alternate path lengths for topology T208 are fairly well-behaved.

Figure 28, Figure 29, Figure 30, and Figure 31 present the hopcount-based path length results for the 19 topologies examined. The topologies in the four tables are grouped based on the size of the topologies, as measured by the number of nodes, with Figure 28 having the smallest topologies and Figure 31 having the largest topologies. Instead of trying to represent the path lengths of a large set of alternates with a single number, we have chosen to present a histogram of the path lengths for each IPFRR method and alternate selection policy studied. The first eight columns of data represent

the percentage of failure scenarios protected by an alternate N hops longer than the primary path, with the first column representing an alternate 0 or 1 hops longer than the primary path, all the way up through the eighth column representing an alternate 14 or 15 hops longer than the primary path. The last column in the table gives the percentage of failure scenarios for which there is no alternate less than 16 hops longer than the primary path. In the case of LLFA and RLFA, this category includes failure scenarios for which no alternate was found.

For each topology, the first row (labeled OPTIMAL) is the distribution of the number of hops in excess of the primary path hopcount for optimally routed alternates. (The optimal routing was done with respect to IGP metrics, as opposed to hopcount.) The second row (labeled NP_LLFA) is the distribution of the extra hops for node-protecting LLFA. The third row (labeled NP_LLFA_THEN_NP_RLFA) is the hopcount distribution when one adds node-protecting RLFA to increase the coverage. The alternate selection policy used here first tries to find a node-protecting LLFA. If that does not exist, then it tries to find an RLFA, and checks if it is node-protecting. Comparing the hopcount distribution for RLFA and LLFA across these topologies, one can see how the coverage is increased at the expense of using longer alternates. It is also worth noting that while superficially LLFA and RLFA appear to have better hopcount distributions than OPTIMAL, the presence of entries in the last column (no alternate < 16) mainly represent failure scenarios that are not protected, for which the hopcount is effectively infinite.

The fourth and fifth rows of each topology show the hopcount distributions for two alternate selection policies using MRT alternates. The policy represented by the label NP_LLFA_THEN_MRT_LOWPOINT will first use a node-protecting LLFA. If a node-protecting LLFA does not exist, then it will use an MRT alternate. The policy represented by the label MRT_LOWPOINT instead will use the MRT alternate even if a node-protecting LLFA exists. One can see from the data that combining node-protecting LLFA with MRT results in a significant shortening of the alternate hopcount distribution.

| Topology name and alternate selection policy evaluated | percentage of failure scenarios protected by an alternate N hops longer than the primary path | | | | | | | | |
|--|---|----------------------------|---------------------------|----------------------|---------------------|-------------------|------------------|------------------|------------------|
| | 0-1 | 2-3 | 4-5 | 6-7 | 8-9 | 10-11 | 12-13 | 14-15 | no alt <16 |
| T201(avg primary hops=3.5) OPTIMAL NP_LLFA NP_LLFA_THEN_NP_RLFA NP_LLFA_THEN_MRT_LOWPOINT MRT_LOWPOINT | 37 37 37 37 33 | 37 34 33 36 | 20 19 21 23 | 3 6 6 | 3 3 3 | | | | 63 10 |
| T202(avg primary hops=4.8) OPTIMAL NP_LLFA NP_LLFA_THEN_NP_RLFA NP_LLFA_THEN_MRT_LOWPOINT MRT_LOWPOINT_ONLY | 90 71 78 80 48 | 9 2 5 12 29 | 5 13 | 2 7 | 1 2 | 1 | | | 27 17 |
| T203(avg primary hops=4.1) OPTIMAL NP_LLFA NP_LLFA_THEN_NP_RLFA NP_LLFA_THEN_MRT_LOWPOINT MRT_LOWPOINT_ONLY | 36 34 35 36 31 | 37 15 19 35 35 | 21 3 22 22 26 | 4 4 5 7 | 2 2 2 | | | | 49 20 |
| T204(avg primary hops=3.7) OPTIMAL NP_LLFA NP_LLFA_THEN_NP_RLFA NP_LLFA_THEN_MRT_LOWPOINT MRT_LOWPOINT_ONLY | 76 54 67 70 58 | 20 1 10 18 27 | 3 4 8 11 | 1 3 3 | 1 1 | | | | 45 19 |
| T205(avg primary hops=3.4) OPTIMAL NP_LLFA NP_LLFA_THEN_NP_RLFA NP_LLFA_THEN_MRT_LOWPOINT MRT_LOWPOINT_ONLY | 92 89 90 91 62 | 8 3 4 9 33 | 5 | 1 | | | | | 8 7 |

Figure 28

| Topology name and alternate selection policy evaluated | percentage of failure scenarios protected by an alternate N hops longer than the primary path | | | | | | | | |
|---|---|-----|-----|-----|-----|-------|-------|-------|------------------|
| | 0-1 | 2-3 | 4-5 | 6-7 | 8-9 | 10-11 | 12-13 | 14-15 | no alt <16 |
| T206(avg primary hops=3.7) | | | | | | | | | |
| OPTIMAL | 63 | 30 | 7 | | | | | | |
| NP_LLFA | 60 | 9 | 1 | | | | | | 29 |
| NP_LLFA_THEN_NP_RLFA | 60 | 13 | 1 | | | | | | 26 |
| NP_LLFA_THEN_MRT_LOWPOINT | 64 | 29 | 7 | | | | | | |
| MRT_LOWPOINT | 55 | 32 | 13 | | | | | | |
| T207(avg primary hops=3.9) | | | | | | | | | |
| OPTIMAL | 71 | 24 | 5 | 1 | | | | | |
| NP_LLFA | 55 | 2 | | | | | | | 43 |
| NP_LLFA_THEN_NP_RLFA | 63 | 10 | | | | | | | 26 |
| NP_LLFA_THEN_MRT_LOWPOINT | 70 | 20 | 7 | 2 | 1 | | | | |
| MRT_LOWPOINT_ONLY | 57 | 29 | 11 | 3 | 1 | | | | |
| T208(avg primary hops=4.6) | | | | | | | | | |
| OPTIMAL | 58 | 28 | 12 | 2 | 1 | | | | |
| NP_LLFA | 53 | 11 | 3 | | | | | | 34 |
| NP_LLFA_THEN_NP_RLFA | 56 | 17 | 7 | 1 | | | | | 19 |
| NP_LLFA_THEN_MRT_LOWPOINT | 58 | 19 | 10 | 7 | 3 | 1 | | | |
| MRT_LOWPOINT_ONLY | 34 | 24 | 21 | 13 | 6 | 2 | 1 | | |
| T209(avg primary hops=3.6) | | | | | | | | | |
| OPTIMAL | 85 | 14 | 1 | | | | | | |
| NP_LLFA | 79 | | | | | | | | 21 |
| NP_LLFA_THEN_NP_RLFA | 79 | | | | | | | | 21 |
| NP_LLFA_THEN_MRT_LOWPOINT | 82 | 15 | 2 | | | | | | |
| MRT_LOWPOINT_ONLY | 63 | 29 | 8 | | | | | | |
| T210(avg primary hops=2.5) | | | | | | | | | |
| OPTIMAL | 95 | 4 | 1 | | | | | | |
| NP_LLFA | 94 | 1 | | | | | | | 5 |
| NP_LLFA_THEN_NP_RLFA | 94 | 3 | 1 | | | | | | 2 |
| NP_LLFA_THEN_MRT_LOWPOINT | 95 | 4 | 1 | | | | | | |
| MRT_LOWPOINT_ONLY | 91 | 6 | 2 | | | | | | |

Figure 29

| Topology name and alternate selection policy evaluated | percentage of failure scenarios protected by an alternate N hops longer than the primary path | | | | | | | | | |
|---|---|-----|-----|-----|-----|-------|-------|-------|--------|-----|
| | 0-1 | 2-3 | 4-5 | 6-7 | 8-9 | 10-11 | 12-13 | 14-15 | no alt | <16 |
| T211(avg primary hops=3.3) | | | | | | | | | | |
| OPTIMAL | 88 | 11 | | | | | | | | |
| NP_LLFA | 66 | 1 | | | | | | | | 32 |
| NP_LLFA_THEN_NP_RLFA | 68 | 3 | | | | | | | | 29 |
| NP_LLFA_THEN_MRT_LOWPOINT | 88 | 12 | | | | | | | | |
| MRT_LOWPOINT | 85 | 15 | 1 | | | | | | | |
| T212(avg primary hops=3.5) | | | | | | | | | | |
| OPTIMAL | 76 | 23 | 1 | | | | | | | |
| NP_LLFA | 59 | | | | | | | | | 41 |
| NP_LLFA_THEN_NP_RLFA | 61 | 1 | 1 | | | | | | | 37 |
| NP_LLFA_THEN_MRT_LOWPOINT | 75 | 24 | 1 | | | | | | | |
| MRT_LOWPOINT_ONLY | 66 | 31 | 3 | | | | | | | |
| T213(avg primary hops=4.3) | | | | | | | | | | |
| OPTIMAL | 91 | 9 | | | | | | | | |
| NP_LLFA | 84 | | | | | | | | | 16 |
| NP_LLFA_THEN_NP_RLFA | 84 | | | | | | | | | 16 |
| NP_LLFA_THEN_MRT_LOWPOINT | 89 | 10 | 1 | | | | | | | |
| MRT_LOWPOINT_ONLY | 75 | 24 | 1 | | | | | | | |
| T214(avg primary hops=5.8) | | | | | | | | | | |
| OPTIMAL | 71 | 22 | 5 | 2 | | | | | | |
| NP_LLFA | 58 | 8 | 1 | 1 | | | | | | 32 |
| NP_LLFA_THEN_NP_RLFA | 61 | 13 | 3 | 1 | | | | | | 22 |
| NP_LLFA_THEN_MRT_LOWPOINT | 66 | 14 | 7 | 5 | 3 | 2 | 1 | 1 | 1 | |
| MRT_LOWPOINT_ONLY | 30 | 20 | 18 | 12 | 8 | 4 | 3 | 2 | 3 | |
| T215(avg primary hops=4.8) | | | | | | | | | | |
| OPTIMAL | 73 | 27 | | | | | | | | |
| NP_LLFA | 73 | 11 | | | | | | | | 16 |
| NP_LLFA_THEN_NP_RLFA | 73 | 13 | 2 | | | | | | | 12 |
| NP_LLFA_THEN_MRT_LOWPOINT | 74 | 19 | 3 | 2 | 1 | 1 | 1 | | | |
| MRT_LOWPOINT_ONLY | 32 | 31 | 16 | 12 | 4 | 3 | 1 | | | |

Figure 30

| Topology name and alternate selection policy evaluated | percentage of failure scenarios protected by an alternate N hops longer than the primary path | | | | | | | | |
|---|---|-----|-----|-----|-----|-------|-------|-------|------------|
| | 0-1 | 2-3 | 4-5 | 6-7 | 8-9 | 10-11 | 12-13 | 14-15 | no alt <16 |
| T216(avg primary hops=5.2) | | | | | | | | | |
| OPTIMAL | 60 | 32 | 7 | 1 | | | | | |
| NP_LLFA | 39 | 4 | | | | | | | 57 |
| NP_LLFA_THEN_NP_RLFA | 46 | 12 | 2 | | | | | | 41 |
| NP_LLFA_THEN_MRT_LOWPOINT | 48 | 20 | 12 | 7 | 5 | 4 | 2 | 1 | 1 |
| MRT_LOWPOINT | 28 | 25 | 18 | 11 | 7 | 6 | 3 | 2 | 1 |
| T217(avg primary hops=8.0) | | | | | | | | | |
| OPTIMAL | 81 | 13 | 5 | 1 | | | | | |
| NP_LLFA | 74 | 3 | 1 | | | | | | 22 |
| NP_LLFA_THEN_NP_RLFA | 76 | 8 | 3 | 1 | | | | | 12 |
| NP_LLFA_THEN_MRT_LOWPOINT | 77 | 7 | 5 | 4 | 3 | 2 | 1 | 1 | |
| MRT_LOWPOINT_ONLY | 25 | 18 | 18 | 16 | 12 | 6 | 3 | 1 | |
| T218(avg primary hops=5.5) | | | | | | | | | |
| OPTIMAL | 85 | 14 | 1 | | | | | | |
| NP_LLFA | 68 | 3 | | | | | | | 28 |
| NP_LLFA_THEN_NP_RLFA | 71 | 4 | | | | | | | 25 |
| NP_LLFA_THEN_MRT_LOWPOINT | 77 | 12 | 7 | 4 | 1 | | | | |
| MRT_LOWPOINT_ONLY | 37 | 29 | 21 | 10 | 3 | 1 | | | |
| T219(avg primary hops=7.7) | | | | | | | | | |
| OPTIMAL | 77 | 15 | 5 | 1 | 1 | | | | |
| NP_LLFA | 72 | 5 | | | | | | | 22 |
| NP_LLFA_THEN_NP_RLFA | 73 | 8 | 2 | | | | | | 16 |
| NP_LLFA_THEN_MRT_LOWPOINT | 74 | 8 | 3 | 3 | 2 | 2 | 2 | 2 | 4 |
| MRT_LOWPOINT_ONLY | 19 | 14 | 15 | 12 | 10 | 8 | 7 | 6 | 10 |

Figure 31

In the preceding analysis, the following procedure for selecting an RLFA was used. Nodes were ordered with respect to distance from the source and checked for membership in Q and P-space. The first node to satisfy this condition was selected as the RLFA. More sophisticated methods to select node-protecting RLFAs is an area of active research.

The analysis presented above uses the MRT Lowpoint Algorithm defined in this specification with a common GADAG root. The particular choice of a common GADAG root is expected to affect the quality of the MRT alternate paths, with a more central common GADAG root resulting in shorter MRT alternate path lengths. For the analysis above, the GADAG root was chosen for each topology by calculating node centrality as the sum of costs of all shortest paths to and from a given node. The node with the lowest sum was chosen as the common GADAG root. In actual deployments, the common GADAG root would be chosen based on the GADAG Root Selection Priority advertised by each router, the values of which would be determined off-line.

In order to measure how sensitive the MRT alternate path lengths are to the choice of common GADAG root, we performed the same analysis using different choices of GADAG root. All of the nodes in the network were ordered with respect to the node centrality as computed above. Nodes were chosen at the 0th, 25th, and 50th percentile with respect to the centrality ordering, with 0th percentile being the most central node. The distribution of alternate path lengths for those three choices of GADAG root are shown in Figure 32 for a subset of the 19 topologies (chosen arbitrarily). The third row for each topology (labeled MRT_LOWPOINT (0 percentile)) reproduces the results presented above for MRT_LOWPOINT_ONLY. The fourth and fifth rows show the alternate path length distribution for the 25th and 50th percentile choice for GADAG root. One can see some impact on the path length distribution with the less central choice of GADAG root resulting in longer path lengths.

We also looked at the impact of MRT algorithm variant on the alternate path lengths. The first two rows for each topology present results of the same alternate path length distribution analysis for the SPF and Hybrid methods for computing the GADAG. These two methods are described in Appendix A and Appendix B. For three of the topologies in this subset (T201, T206, and T211), the use of SPF or Hybrid methods does not appear to provide a significant advantage over the Lowpoint method with respect to path length. Instead, the choice of GADAG root appears to have more impact on the path length. However, for two of the topologies in this subset (T216 and T219) and for this particular choice of GADAG root, the use of the SPF method results in noticeably shorter alternate path lengths than the use of the Lowpoint or Hybrid methods. It remains to be determined if this effect applies generally across more topologies or is sensitive to choice of GADAG root.

| Topology name | percentage of failure scenarios protected by an alternate N hops longer than the primary path | | | | | | | | | |
|---------------------------------------|---|-----|-----|-----|-----|-------|-------|-------|-----------|-----|
| MRT algorithm variant | | | | | | | | | | |
| (GADAG root centrality percentile) | 0-1 | 2-3 | 4-5 | 6-7 | 8-9 | 10-11 | 12-13 | 14-15 | no alt | <16 |
| T201(avg primary hops=3.5) | | | | | | | | | | |
| MRT_HYBRID (0 percentile) | 33 | 26 | 23 | 6 | 3 | | | | | |
| MRT_SPF (0 percentile) | 33 | 36 | 23 | 6 | 3 | | | | | |
| MRT_LOWPOINT (0 percentile) | 33 | 36 | 23 | 6 | 3 | | | | | |
| MRT_LOWPOINT (25 percentile) | 27 | 29 | 23 | 11 | 10 | | | | | |
| MRT_LOWPOINT (50 percentile) | 27 | 29 | 23 | 11 | 10 | | | | | |
| T206(avg primary hops=3.7) | | | | | | | | | | |
| MRT_HYBRID (0 percentile) | 50 | 35 | 13 | 2 | | | | | | |
| MRT_SPF (0 percentile) | 50 | 35 | 13 | 2 | | | | | | |
| MRT_LOWPOINT (0 percentile) | 55 | 32 | 13 | | | | | | | |
| MRT_LOWPOINT (25 percentile) | 47 | 25 | 22 | 6 | | | | | | |
| MRT_LOWPOINT (50 percentile) | 38 | 38 | 14 | 11 | | | | | | |
| T211(avg primary hops=3.3) | | | | | | | | | | |
| MRT_HYBRID (0 percentile) | 86 | 14 | | | | | | | | |
| MRT_SPF (0 percentile) | 86 | 14 | | | | | | | | |
| MRT_LOWPOINT (0 percentile) | 85 | 15 | 1 | | | | | | | |
| MRT_LOWPOINT (25 percentile) | 70 | 25 | 5 | 1 | | | | | | |
| MRT_LOWPOINT (50 percentile) | 80 | 18 | 2 | | | | | | | |
| T216(avg primary hops=5.2) | | | | | | | | | | |
| MRT_HYBRID (0 percentile) | 23 | 22 | 18 | 13 | 10 | 7 | 4 | 2 | 2 | |
| MRT_SPF (0 percentile) | 35 | 32 | 19 | 9 | 3 | 1 | | | | |
| MRT_LOWPOINT (0 percentile) | 28 | 25 | 18 | 11 | 7 | 6 | 3 | 2 | 1 | |
| MRT_LOWPOINT (25 percentile) | 24 | 20 | 19 | 16 | 10 | 6 | 3 | 1 | | |
| MRT_LOWPOINT (50 percentile) | 19 | 14 | 13 | 10 | 8 | 6 | 5 | 5 | 10 | |
| T219(avg primary hops=7.7) | | | | | | | | | | |
| MRT_HYBRID (0 percentile) | 20 | 16 | 13 | 10 | 7 | 5 | 5 | 5 | 3 | |
| MRT_SPF (0 percentile) | 31 | 23 | 19 | 12 | 7 | 4 | 2 | 1 | | |
| MRT_LOWPOINT (0 percentile) | 19 | 14 | 15 | 12 | 10 | 8 | 7 | 6 | 10 | |
| MRT_LOWPOINT (25 percentile) | 19 | 14 | 15 | 13 | 12 | 10 | 6 | 5 | 7 | |
| MRT_LOWPOINT (50 percentile) | 19 | 14 | 14 | 12 | 11 | 8 | 6 | 6 | 10 | |

Figure 32

7. Algorithm Work to Be Done

Broadcast Interfaces: The algorithm assumes that broadcast interfaces are already represented as pseudo-nodes in the network graph. Given maximal redundancy, one of the MRT will try to avoid both the pseudo-node and the next hop. The exact rules need to be fully specified.

8. IANA Considerations

This document includes no request to IANA.

9. Security Considerations

This architecture is not currently believed to introduce new security concerns.

10. References

10.1. Normative References

[I-D.ietf-rtgwg-mrt-frr-architecture]
Atlas, A., Kebler, R., Envedi, G., Csaszar, A., Tantsura, J., Konstantynowicz, M., and R. White, "An Architecture for IP/LDP Fast-Reroute Using Maximally Redundant Trees", draft-ietf-rtgwg-mrt-frr-architecture-03 (work in progress), July 2013.

10.2. Informative References

[EnyediThesis]
Enyedi, G., "Novel Algorithms for IP Fast Reroute", Department of Telecommunications and Media Informatics, Budapest University of Technology and Economics Ph.D. Thesis, February 2011, <http://www.omikk.bme.hu/collections/phd/Villamosmernoki_es_Informatikai_Kar/2011/Enyedi_Gabor/ertekezes.pdf>.

[I-D.atlas-ospf-mrt]
Atlas, A., Hegde, S., Chris, C., and J. Tantsura, "OSPF Extensions to Support Maximally Redundant Trees", draft-atlas-ospf-mrt-00 (work in progress), July 2013.

[I-D.ietf-rtgwg-ipfrr-notvia-addresses]
Bryant, S., Previdi, S., and M. Shand, "A Framework for IP and MPLS Fast Reroute Using Not-via Addresses", draft-ietf-rtgwg-ipfrr-notvia-addresses-11 (work in progress), May 2013.

- [I-D.ietf-rtgwg-lfa-manageability]
Litkowski, S., Decraene, B., Filsfils, C., and K. Raza,
"Operational management of Loop Free Alternates", draft-
ietf-rtgwg-lfa-manageability-00 (work in progress), May
2013.
- [I-D.ietf-rtgwg-remote-lfa]
Bryant, S., Filsfils, C., Previdi, S., Shand, M., and S.
Ning, "Remote LFA FRR", draft-ietf-rtgwg-remote-lfa-02
(work in progress), May 2013.
- [Kahn_1962_topo_sort]
Kahn, A., "Topological sorting of large networks",
Communications of the ACM, Volume 5, Issue 11 , Nov 1962,
<<http://dl.acm.org/citation.cfm?doid=368996.369025>>.
- [LFARevisited]
Retvari, G., Tapolcai, J., Enyedi, G., and A. Csaszar, "IP
Fast ReRoute: Loop Free Alternates Revisited", Proceedings
of IEEE INFOCOM , 2011, <http://opti.tmit.bme.hu/~tapolcai/papers/retvari2011lfa_infocom.pdf>.
- [LightweightNotVia]
Enyedi, G., Retvari, G., Szilagyi, P., and A. Csaszar, "IP
Fast ReRoute: Lightweight Not-Via without Additional
Addresses", Proceedings of IEEE INFOCOM , 2009,
<<http://mycite.omikk.bme.hu/doc/71691.pdf>>.
- [MRTLlinear]
Enyedi, G., Retvari, G., and A. Csaszar, "On Finding
Maximally Redundant Trees in Strictly Linear Time", IEEE
Symposium on Computers and Communications (ISCC) , 2009,
<<http://opti.tmit.bme.hu/~enyedi/ipfrr/distMaxRedTree.pdf>>.
- [RFC3137] Retana, A., Nguyen, L., White, R., Zinin, A., and D.
McPherson, "OSPF Stub Router Advertisement", RFC 3137,
June 2001.
- [RFC5286] Atlas, A. and A. Zinin, "Basic Specification for IP Fast
Reroute: Loop-Free Alternates", RFC 5286, September 2008.
- [RFC5714] Shand, M. and S. Bryant, "IP Fast Reroute Framework", RFC
5714, January 2010.

[RFC6571] Filsfils, C., Francois, P., Shand, M., Decraene, B., Uttaro, J., Leymann, N., and M. Horneffer, "Loop-Free Alternate (LFA) Applicability in Service Provider (SP) Networks", RFC 6571, June 2012.

Appendix A. Option 2: Computing GADAG using SPF

The basic idea in this option is to use slightly-modified SPF computations to find ears. In every block, an SPF computation is first done to find a cycle from the local root and then SPF computations in that block find ears until there are no more interfaces to be explored. The used result from the SPF computation is the path of interfaces indicated by following the previous hops from the minimized IN_GADAG node back to the SPF root.

To do this, first all cut-vertices must be identified and local-roots assigned as specified in Figure 12.

The slight modifications to the SPF are as follows. The root of the block is referred to as the block-root; it is either the GADAG root or a cut-vertex.

- a. The SPF is rooted at a neighbor x of an IN_GADAG node y. All links between y and x are marked as TEMP_UNUSABLE. They should not be used during the SPF computation.
- b. If y is not the block-root, then it is marked TEMP_UNUSABLE. It should not be used during the SPF computation. This prevents ears from starting and ending at the same node and avoids cycles; the exception is because cycles to/from the block-root are acceptable and expected.
- c. Do not explore links to nodes whose local-root is not the block-root. This keeps the SPF confined to the particular block.
- d. Terminate when the first IN_GADAG node z is minimized.
- e. Respect the existing directions (e.g. INCOMING, OUTGOING, UNDIRECTED) already specified for each interface.

```
Mod_SPF(spf_root, block_root)
  Initialize spf_heap to empty
  Initialize nodes' spf_metric to infinity
  spf_root.spf_metric = 0
  insert(spf_heap, spf_root)
  found_in_gadag = false
  while (spf_heap is not empty) and (found_in_gadag is false)
```



```

    min_node = remove_lowest(spf_heap)
    if min_node.IN_GADAG is true
        found_in_gadag = true
    else
        foreach interface intf of min_node
            if ((intf.OUTGOING or intf.UNDIRECTED) and
                ((intf.remote_node.localroot is block_root) or
                 (intf.remote_node is block_root)) and
                (intf.remote_node is not TEMP_UNUSABLE) and
                (intf is not TEMP_UNUSABLE))
                path_metric = min_node.spf_metric + intf.metric
                if path_metric < intf.remote_node.spf_metric
                    intf.remote_node.spf_metric = path_metric
                    intf.remote_node.spf_prev_intf = intf
                    insert_or_update(spf_heap, intf.remote_node)
        return min_node

SPF_for_Ear(cand_intf.local_node, cand_intf.remote_node, block_root,
            method)
    Mark all interfaces between cand_intf.remote_node
        and cand_intf.local_node as TEMP_UNUSABLE
    if cand_intf.local_node is not block_root
        Mark cand_intf.local_node as TEMP_UNUSABLE
    Initialize ear_list to empty
    end_ear = Mod_SPF(spf_root, block_root)
    y = end_ear.spf_prev_hop
    while y.local_node is not spf_root
        add_to_list_start(ear_list, y)
        y.local_node.IN_GADAG = true
        y = y.local_node.spf_prev_intf
    if(method is not hybrid)
        Set_Ear_Direction(ear_list, cand_intf.local_node,
                           end_ear, block_root)
    Clear TEMP_UNUSABLE from all interfaces between
        cand_intf.remote_node and cand_intf.local_node
    Clear TEMP_UNUSABLE from cand_intf.local_node
    return end_ear

```

Figure 33: Modified SPF for GADAG computation

Assume that an ear is found by going from y to x and then running an SPF that terminates by minimizing z (e.g. $y \leftrightarrow x \dots q \leftrightarrow z$). Now it is necessary to determine the direction of the ear; if $y \ll z$, then the path should be $y \rightarrow x \dots q \rightarrow z$ but if $y \gg z$, then the path should be $y \leftarrow x \dots q \leftarrow z$. In Section 4.4, the same problem was handled by finding

all ears that started at a node before looking at ears starting at nodes higher in the partial order. In this algorithm, using that approach could mean that new ears aren't added in order of their total cost since all ears connected to a node would need to be found before additional nodes could be found.

The alternative is to track the order relationship of each node with respect to every other node. This can be accomplished by maintaining two sets of nodes at each node. The first set, `Higher_Nodes`, contains all nodes that are known to be ordered above the node. The second set, `Lower_Nodes`, contains all nodes that are known to be ordered below the node. This is the approach used in this algorithm.

```
Set_Ear_Direction(ear_list, end_a, end_b, block_root)
// Default of A_TO_B for the following cases:
// (a) end_a and end_b are the same (root)
// or (b) end_a is in end_b's Lower_Nodes
// or (c) end_a and end_b were unordered with respect to each
// other
direction = A_TO_B
if (end_b is block_root) and (end_a is not end_b)
    direction = B_TO_A
else if end_a is in end_b.Higher_Nodes
    direction = B_TO_A
if direction is B_TO_A
    foreach interface i in ear_list
        i.UNDIRECTED = false
        i.INCOMING = true
        i.remote_intf.UNDIRECTED = false
        i.remote_intf.OUTGOING = true
else
    foreach interface i in ear_list
        i.UNDIRECTED = false
        i.OUTGOING = true
        i.remote_intf.UNDIRECTED = false
        i.remote_intf.INCOMING = true
if end_a is end_b
    return
// Next, update all nodes' Lower_Nodes and Higher_Nodes
if (end_a is in end_b.Higher_Nodes)
    foreach node x where x.localroot is block_root
        if end_a is in x.Lower_Nodes
            foreach interface i in ear_list
                add i.remote_node to x.Lower_Nodes
        if end_b is in x.Higher_Nodes
            foreach interface i in ear_list
                add i.local_node to x.Higher_Nodes
```

```

else
  foreach node x where x.localroot is block_root
    if end_b is in x.Lower_Nodes
      foreach interface i in ear_list
        add i.local_node to x.Lower_Nodes
    if end_a is in x.Higher_Nodes
      foreach interface i in ear_list
        add i.remote_node to x.Higher_Nodes

```

Figure 34: Algorithm to assign links of an ear direction

A goal of the algorithm is to find the shortest cycles and ears. An ear is started by going to a neighbor x of an IN_GADAG node y. The path from x to an IN_GADAG node is minimal, since it is computed via SPF. Since a shortest path is made of shortest paths, to find the shortest ears requires reaching from the set of IN_GADAG nodes to the closest node that isn't IN_GADAG. Therefore, an ordered tree is maintained of interfaces that could be explored from the IN_GADAG nodes. The interfaces are ordered by their characteristics of metric, local loopback address, remote loopback address, and ifindex, as in the algorithm previously described in Figure 14.

The algorithm ignores interfaces picked from the ordered tree that belong to the block root if the block in which the interface is present already has an ear that has been computed. This is necessary since we allow at most one incoming interface to a block root in each block. This requirement stems from the way next-hops are computed as will be seen in Section 4.6. After any ear gets computed, we traverse the newly added nodes to the GADAG and insert interfaces whose far end is not yet on the GADAG to the ordered tree for later processing.

Finally, cut-edges are a special case because there is no point in doing an SPF on a block of 2 nodes. The algorithm identifies cut-edges simply as links where both ends of the link are cut-vertices. Cut-edges can simply be added to the GADAG with both OUTGOING and INCOMING specified on their interfaces.

```

add_eligible_interfaces_of_node(ordered_intfs_tree,node)
  for each interface of node
    if intf.remote_node.IN_GADAG is false
      insert(intf,ordered_intfs_tree)

check_if_block_has_ear(x,block_id)
  block_has_ear = false
  for all interfaces of x
    if (intf.remote_node.block_id == block_id) &&
      (intf.remote_node.IN_GADAG is true)

```

```

        block_has_ear = true
    return block_has_ear

Construct_GADAG_via_SPF(topology, root)
    Compute_Localroot (root,root)
    Assign_Block_ID(root,0)
    root.IN_GADAG = true
    add_eligible_interfaces_of_node(ordered_intfs_tree,root)
    while ordered_intfs_tree is not empty
        cand_intf = remove_lowest(ordered_intfs_tree)
        if cand_intf.remote_node.IN_GADAG is false
            if L(cand_intf.remote_node) == D(cand_intf.remote_node)
                // Special case for cut-edges
                cand_intf.UNDIRECTED = false
                cand_intf.remote_intf.UNDIRECTED = false
                cand_intf.OUTGOING = true
                cand_intf.INCOMING = true
                cand_intf.remote_intf.OUTGOING = true
                cand_intf.remote_intf.INCOMING = true
                cand_intf.remote_node.IN_GADAG = true
            add_eligible_interfaces_of_node(
                ordered_intfs_tree,cand_intf.remote_node)
        else
            if (cand_intf.remote_node.local_root ==
                cand_intf.local_node) &&
                check_if_block_has_ear
                    (cand_intf.local_node,
                     cand_intf.remote_node.block_id))
                /* Skip the interface since the block root
                   already has an incoming interface in the
                   block */
            else
                ear_end = SPF_for_Ear(cand_intf.local_node,
                                       cand_intf.remote_node,
                                       cand_intf.remote_node.localroot,
                                       SPF method)
                y = ear_end.spf_prev_hop
                while y.local_node is not cand_intf.local_node
                    add_eligible_interfaces_of_node(
                        ordered_intfs_tree,
                        y.local_node)
                    y = y.local_node.spf_prev_intf

```

Figure 35: SPF-based GADAG algorithm

Appendix B. Option 3: Computing GADAG using a hybrid method

In this option, the idea is to combine the salient features of the above two options. To this end, we process nodes as they get added to the GADAG just like in the lowpoint inheritance by maintaining a stack of nodes. This ensures that we do not need to maintain lower and higher sets at each node to ascertain ear directions since the ears will always be directed from the node being processed towards the end of the ear. To compute the ear however, we resort to an SPF to have the possibility of better ears (path lengths) thus giving more flexibility than the restricted use of lowpoint/dfs parents.

Regarding ears involving a block root, unlike the SPF method which ignored interfaces of the block root after the first ear, in the hybrid method we would have to process all interfaces of the block root before moving on to other nodes in the block since the direction of an ear is pre-determined. Thus, whenever the block already has an ear computed, and we are processing an interface of the block root, we mark the block root as unusable before the SPF run that computes the ear. This ensures that the SPF terminates at some node other than the block-root. This in turn guarantees that the block-root has only one incoming interface in each block, which is necessary for correctly computing the next-hops on the GADAG.

As in the SPF gadag, bridge ears are handled as a special case.

The entire algorithm is shown below in Figure 36

```

find_spf_stack_ear(stack, x, y, xy_intf, block_root)
  if L(y) == D(y)
    // Special case for cut-edges
    xy_intf.UNDIRECTED = false
    xy_intf.remote_intf.UNDIRECTED = false
    xy_intf.OUTGOING = true
    xy_intf.INCOMING = true
    xy_intf.remote_intf.OUTGOING = true
    xy_intf.remote_intf.INCOMING = true
    xy_intf.remote_node.IN_GADAG = true
    push y onto stack
    return
  else
    if (y.local_root == x) &&
      check_if_block_has_ear(x,y.block_id)
      //Avoid the block root during the SPF
      Mark x as TEMP_UNUSABLE
    end_ear = SPF_for_Ear(x,y,block_root,hybrid)
    If x was set as TEMP_UNUSABLE, clear it
    cur = end_ear
    while (cur != y)
      intf = cur.spf_prev_hop

```

```
        prev = intf.local_node
        intf.UNDIRECTED = false
        intf.remote_intf.UNDIRECTED = false
        intf.OUTGOING = true
        intf.remote_intf.INCOMING = true
        push prev onto stack
    cur = prev
    xy_intf.UNDIRECTED = false
    xy_intf.remote_intf.UNDIRECTED = false
    xy_intf.OUTGOING = true
    xy_intf.remote_intf.INCOMING = true
    return

Construct_GADAG_via_hybrid(topology,root)
    Compute_Localroot (root,root)
    Assign_Block_ID(root,0)
    root.IN_GADAG = true
    Initialize Stack to empty
    push root onto Stack
    while (Stack is not empty)
        x = pop(Stack)
        for each interface intf of x
            y = intf.remote_node
            if y.IN_GADAG is false
                find_spf_stack_ear(stack, x, y, intf, y.block_root)
```

Figure 36: Hybrid GADAG algorithm

Authors' Addresses

Gabor Sandor Enyedi (editor)
Ericsson
Konyves Kalman krt 11
Budapest 1097
Hungary

Email: Gabor.Sandor.Enyedi@ericsson.com

Andras Csaszar
Ericsson
Konyves Kalman krt 11
Budapest 1097
Hungary

Email: Andras.Csaszar@ericsson.com

Alia Atlas (editor)
Juniper Networks
10 Technology Park Drive
Westford, MA 01886
USA

Email: akatlas@juniper.net

Chris Bowers
Juniper Networks
1194 N. Mathilda Ave.
Sunnyvale, CA 94089
USA

Email: cbowers@juniper.net

Abishek Gopalan
University of Arizona
1230 E Speedway Blvd.
Tucson, AZ 85721
USA

Email: abishek@ece.arizona.edu

Routing Area Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 8, 2016

A. Atlas
C. Bowers
Juniper Networks
G. Enyedi
Ericsson
February 5, 2016

An Architecture for IP/LDP Fast-Reroute Using Maximally Redundant Trees
draft-ietf-rtgwg-mrt-frr-architecture-10

Abstract

This document defines the architecture for IP and LDP Fast-Reroute using Maximally Redundant Trees (MRT-FRR). MRT-FRR is a technology that gives link-protection and node-protection with 100% coverage in any network topology that is still connected after the failure.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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."

This Internet-Draft will expire on August 8, 2016.

Copyright Notice

Copyright (c) 2016 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 Simplified BSD License.

Table of Contents

| | |
|---|----|
| 1. Introduction | 3 |
| 1.1. Importance of 100% Coverage | 4 |
| 1.2. Partial Deployment and Backwards Compatibility | 5 |
| 2. Requirements Language | 5 |
| 3. Terminology | 5 |
| 4. Maximally Redundant Trees (MRT) | 7 |
| 5. Maximally Redundant Trees (MRT) and Fast-Reroute | 9 |
| 6. Unicast Forwarding with MRT Fast-Reroute | 9 |
| 6.1. Introduction to MRT Forwarding Options | 10 |
| 6.1.1. MRT LDP labels | 10 |
| 6.1.1.1. Topology-scoped FEC encoded using a single label (Option 1A) | 10 |
| 6.1.1.2. Topology and FEC encoded using a two label stack (Option 1B) | 11 |
| 6.1.1.3. Compatibility of MRT LDP Label Options 1A and 1B | 12 |
| 6.1.1.4. Required support for MRT LDP Label options | 12 |
| 6.1.2. MRT IP tunnels (Options 2A and 2B) | 12 |
| 6.2. Forwarding LDP Unicast Traffic over MRT Paths | 13 |
| 6.2.1. Forwarding LDP traffic using MRT LDP Label Option 1A | 13 |
| 6.2.2. Forwarding LDP traffic using MRT LDP Label Option 1B | 14 |
| 6.2.3. Other considerations for forwarding LDP traffic using MRT LDP Labels | 14 |
| 6.2.4. Required support for LDP traffic | 14 |
| 6.3. Forwarding IP Unicast Traffic over MRT Paths | 14 |
| 6.3.1. Tunneling IP traffic using MRT LDP Labels | 15 |
| 6.3.1.1. Tunneling IP traffic using MRT LDP Label Option 1A | 15 |
| 6.3.1.2. Tunneling IP traffic using MRT LDP Label Option 1B | 15 |
| 6.3.2. Tunneling IP traffic using MRT IP Tunnels | 16 |
| 6.3.3. Required support for IP traffic | 16 |
| 7. MRT Island Formation | 16 |
| 7.1. IGP Area or Level | 17 |
| 7.2. Support for a specific MRT profile | 17 |
| 7.3. Excluding additional routers and interfaces from the MRT Island | 18 |
| 7.3.1. Existing IGP exclusion mechanisms | 18 |
| 7.3.2. MRT-specific exclusion mechanism | 19 |
| 7.4. Connectivity | 19 |
| 7.5. Algorithm for MRT Island Identification | 19 |
| 8. MRT Profile | 19 |
| 8.1. MRT Profile Options | 19 |
| 8.2. Router-specific MRT paramaters | 21 |

| | |
|--|----|
| 8.3. Default MRT profile | 21 |
| 9. LDP signaling extensions and considerations | 22 |
| 10. Inter-area Forwarding Behavior | 22 |
| 10.1. ABR Forwarding Behavior with MRT LDP Label Option 1A . . | 23 |
| 10.1.1. Motivation for Creating the Rainbow-FEC | 24 |
| 10.2. ABR Forwarding Behavior with IP Tunneling (option 2) . . | 24 |
| 10.3. ABR Forwarding Behavior with MRT LDP Label option 1B . . | 25 |
| 11. Prefixes Multiply Attached to the MRT Island | 26 |
| 11.1. Protecting Multi-Homed Prefixes using Tunnel Endpoint Selection | 28 |
| 11.2. Protecting Multi-Homed Prefixes using Named Proxy-Nodes | 29 |
| 11.3. MRT Alternates for Destinations Outside the MRT Island . | 31 |
| 12. Network Convergence and Preparing for the Next Failure . . . | 31 |
| 12.1. Micro-loop prevention and MRTs | 32 |
| 12.2. MRT Recalculation for the Default MRT Profile | 33 |
| 13. Implementation Status | 34 |
| 14. Operational Considerations | 35 |
| 14.1. Verifying Forwarding on MRT Paths | 35 |
| 14.2. Traffic Capacity on Backup Paths | 36 |
| 14.3. MRT IP Tunnel Loopback Address Management | 38 |
| 14.4. MRT-FRR in a Network with Degraded Connectivity | 38 |
| 14.5. Partial Deployment of MRT-FRR in a Network | 38 |
| 15. Acknowledgements | 39 |
| 16. IANA Considerations | 39 |
| 17. Security Considerations | 40 |
| 18. Contributors | 40 |
| 19. References | 41 |
| 19.1. Normative References | 41 |
| 19.2. Informative References | 42 |
| Appendix A. Inter-level Forwarding Behavior for IS-IS | 43 |
| Appendix B. General Issues with Area Abstraction | 44 |
| Authors' Addresses | 45 |

1. Introduction

This document describes a solution for IP/LDP fast-reroute [RFC5714]. MRT-FRR creates two alternate forwarding trees which are distinct from the primary next-hop forwarding used during stable operation. These two trees are maximally diverse from each other, providing link and node protection for 100% of paths and failures as long as the failure does not cut the network into multiple pieces. This document defines the architecture for IP/LDP fast-reroute with MRT.

[I-D.ietf-rtgwg-mrt-frr-algorithm] describes how to compute maximally redundant trees using a specific algorithm, the MRT Lowpoint algorithm. The MRT Lowpoint algorithm is used by a router that supports the Default MRT Profile, as specified in this document.

IP/LDP Fast-Reroute with MRT (MRT-FRR) uses two maximally diverse forwarding topologies to provide alternates. A primary next-hop should be on only one of the diverse forwarding topologies; thus, the other can be used to provide an alternate. Once traffic has been moved to one of the MRTs by one point of local repair (PLR), that traffic is not subject to further repair actions by another PLR, even in the event of multiple simultaneous failures. Therefore, traffic repaired by MRT-FRR will not loop between different PLRs responding to different simultaneous failures.

While MRT provides 100% protection for a single link or node failure, it may not protect traffic in the event of multiple simultaneous failures, nor does take into account Shared Risk Link Groups (SRLGs). Also, while the MRT Lowpoint algorithm is computationally efficient, it is also new. In order for MRT-FRR to function properly, all of the other nodes in the network that support MRT must correctly compute next-hops based on the same algorithm, and install the corresponding forwarding state. This is in contrast to other FRR methods where the calculation of backup paths generally involves repeated application of the simpler and widely-deployed shortest path first (SPF) algorithm, and backup paths themselves re-use the forwarding state used for shortest path forwarding of normal traffic. Section 14 provides operational guidance related to verification of MRT forwarding paths.

In addition to supporting IP and LDP unicast fast-reroute, the diverse forwarding topologies and guarantee of 100% coverage permit fast-reroute technology to be applied to multicast traffic as described in [I-D.atlas-rtgwg-mrt-mc-arch]. However, the current document does not address the multicast applications of MRTs.

1.1. Importance of 100% Coverage

Fast-reroute is based upon the single failure assumption - that the time between single failures is long enough for a network to reconverge and start forwarding on the new shortest paths. That does not imply that the network will only experience one failure or change.

It is straightforward to analyze a particular network topology for coverage. However, a real network does not always have the same topology. For instance, maintenance events will take links or nodes out of use. Simply costing out a link can have a significant effect on what loop-free alternates (LFAs) are available. Similarly, after a single failure has happened, the topology is changed and its associated coverage. Finally, many networks have new routers or links added and removed; each of those changes can have an effect on the coverage for topology-sensitive methods such as LFA and Remote

LFA. If fast-reroute is important for the network services provided, then a method that guarantees 100% coverage is important to accommodate natural network topology changes.

When a network needs to use Ordered FIB[RFC6976] or Nearside Tunneling[RFC5715] as a micro-loop prevention mechanism [RFC5715], then the whole IGP area needs to have alternates available. This allows the micro-loop prevention mechanism, which requires slower network convergence, to take the necessary time without adversely impacting traffic. Without complete coverage, traffic to the unprotected destinations will be dropped for significantly longer than with current convergence - where routers individually converge as fast as possible. See Section 12.1 for more discussion of micro-loop prevention and MRTs.

1.2. Partial Deployment and Backwards Compatibility

MRT-FRR supports partial deployment. Routers advertise their ability to support MRT. Inside the MRT-capable connected group of routers (referred to as an MRT Island), the MRTs are computed. Alternates to destinations outside the MRT Island are computed and depend upon the existence of a loop-free neighbor of the MRT Island for that destination. MRT Islands are discussed in detail in Section 7, and partial deployment is discussed in more detail in Section 14.5.

2. 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].

3. Terminology

network graph: A graph that reflects the network topology where all links connect exactly two nodes and broadcast links have been transformed into the standard pseudo-node representation.

cut-link: A link whose removal partitions the network. A cut-link by definition must be connected between two cut-vertices. If there are multiple parallel links, then they are referred to as cut-links in this document if removing the set of parallel links would partition the network graph.

cut-vertex: A vertex whose removal partitions the network graph.

2-connected: A graph that has no cut-vertices. This is a graph that requires two nodes to be removed before the network is partitioned.

2-connected cluster: A maximal set of nodes that are 2-connected.

block: Either a 2-connected cluster, a cut-edge, or an isolated vertex.

Redundant Trees (RT): A pair of trees where the path from any node X to the root R along the first tree is node-disjoint with the path from the same node X to the root along the second tree. Redundant trees can always be computed in 2-connected graphs.

Maximally Redundant Trees (MRT): A pair of trees where the path from any node X to the root R along the first tree and the path from the same node X to the root along the second tree share the minimum number of nodes and the minimum number of links. Each such shared node is a cut-vertex. Any shared links are cut-links. In graphs that are not 2-connected, it is not possible to compute RTs. However, it is possible to compute MRTs. MRTs are maximally redundant in the sense that they are as redundant as possible given the constraints of the network graph.

Directed Acyclic Graph (DAG): A graph where all links are directed and there are no cycles in it.

Almost Directed Acyclic Graph (ADAG): A graph with one node designated as the root. The graph has the property that if all links incoming to the root were removed, then resulting graph would be a DAG.

Generalized ADAG (GADAG): A graph that is the combination of the ADAGs of all blocks.

MRT-Red: MRT-Red is used to describe one of the two MRTs; it is used to describe the associated forwarding topology and MPLS multi-topology identifier (MT-ID). Specifically, MRT-Red is the decreasing MRT where links in the GADAG are taken in the direction from a higher topologically ordered node to a lower one.

MRT-Blue: MRT-Blue is used to describe one of the two MRTs; it is used to describe the associated forwarding topology and MPLS MT-ID. Specifically, MRT-Blue is the increasing MRT where links in the GADAG are taken in the direction from a lower topologically ordered node to a higher one.

Rainbow MRT: It is useful to have an MPLS MT-ID that refers to the multiple MRT forwarding topologies and to the default forwarding topology. This is referred to as the Rainbow MRT MPLS MT-ID and is used by LDP to reduce signaling and permit the same label to always be advertised to all peers for the same (MT-ID, Prefix).

MRT Island: The set of routers that support a particular MRT profile and the links connecting them that support MRT.

Island Border Router (IBR): A router in the MRT Island that is connected to a router not in the MRT Island and both routers are in a common area or level.

Island Neighbor (IN): A router that is not in the MRT Island but is adjacent to an IBR and in the same area/level as the IBR.

named proxy-node: A proxy-node can represent a destination prefix that can be attached to the MRT Island via at least two routers. It is named if there is a way that traffic can be encapsulated to reach specifically that proxy node; this could be because there is an LDP FEC (Forwarding Equivalence Class) for the associated prefix or because MRT-Red and MRT-Blue IP addresses are advertised in an undefined fashion for that proxy-node.

4. Maximally Redundant Trees (MRT)

A pair of Maximally Redundant Trees is a pair of directed spanning trees that provides maximally disjoint paths towards their common root. Only links or nodes whose failure would partition the network (i.e. cut-links and cut-vertices) are shared between the trees. The MRT Lowpoint algorithm is given in [I-D.ietf-rtgwg-mrt-frr-algorithm]. This algorithm can be computed in $O(e + n \log n)$; it is less than three SPF's. This document describes how the MRTs can be used and not how to compute them.

MRT provides destination-based trees for each destination. Each router stores its normal primary next-hop(s) as well as MRT-Blue next-hop(s) and MRT-Red next-hop(s) toward each destination. The alternate will be selected between the MRT-Blue and MRT-Red.

The most important thing to understand about MRTs is that for each pair of destination-routed MRTs, there is a path from every node X to the destination D on the Blue MRT that is as disjoint as possible from the path on the Red MRT.

For example, in Figure 1, there is a network graph that is 2-connected in (a) and associated MRTs in (b) and (c). One can consider the paths from B to R; on the Blue MRT, the paths are B->F->D->E->R or B->C->D->E->R. On the Red MRT, the path is B->A->R. These are clearly link and node-disjoint. These MRTs are redundant trees because the paths are disjoint.

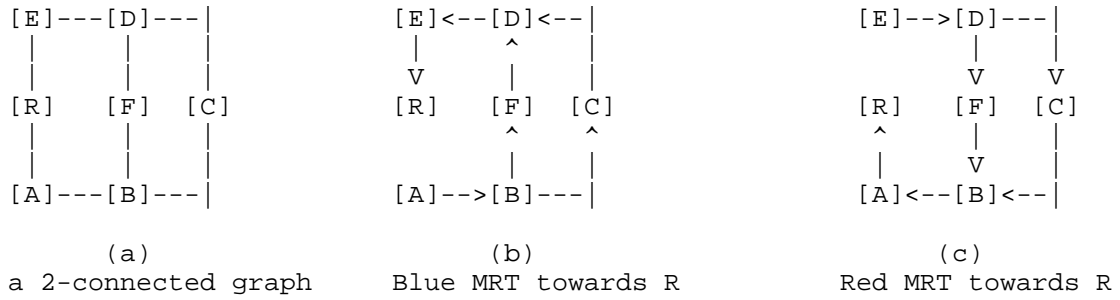


Figure 1: A 2-connected Network

By contrast, in Figure 2, the network in (a) is not 2-connected. If F, G or the link F<->G failed, then the network would be partitioned. It is clearly impossible to have two link-disjoint or node-disjoint paths from G, I or J to R. The MRTs given in (b) and (c) offer paths that are as disjoint as possible. For instance, the paths from B to R are the same as in Figure 1 and the path from G to R on the Blue MRT is G->F->D->E->R and on the Red MRT is G->F->B->A->R.

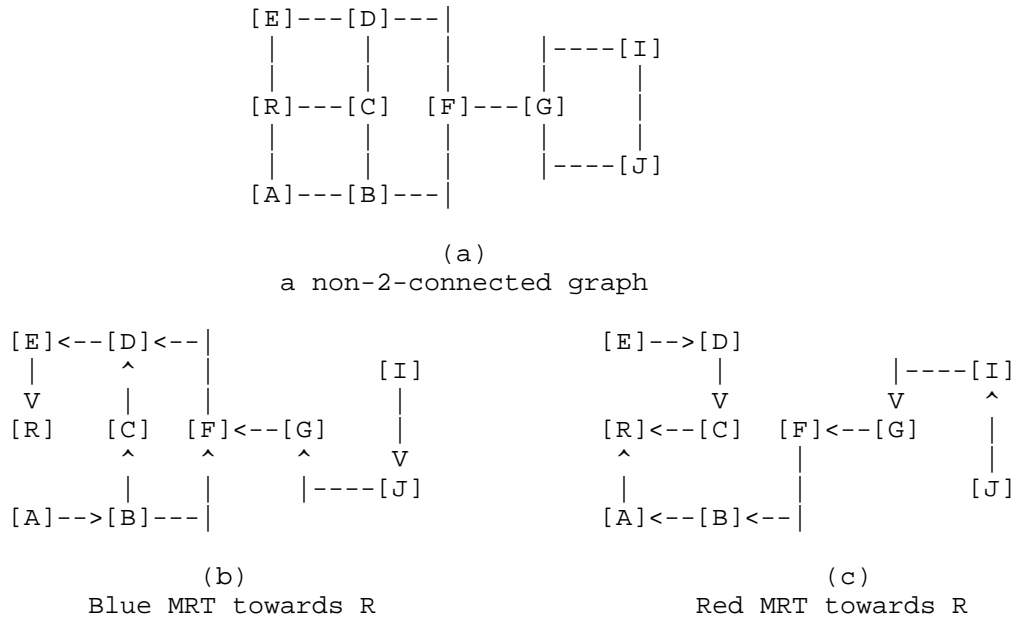


Figure 2: A non-2-connected network

5. Maximally Redundant Trees (MRT) and Fast-Reroute

In normal IGP routing, each router has its shortest path tree (SPT) to all destinations. From the perspective of a particular destination, D, this looks like a reverse SPT. To use maximally redundant trees, in addition, each destination D has two MRTs associated with it; by convention these will be called the MRT-Blue and MRT-Red. MRT-FRR is realized by using multi-topology forwarding. There is a MRT-Blue forwarding topology and a MRT-Red forwarding topology.

Any IP/LDP fast-reroute technique beyond LFA requires an additional dataplane procedure, such as an additional forwarding mechanism. The well-known options are multi-topology forwarding (used by MRT-FRR), tunneling (e.g. [RFC6981] or [RFC7490]), and per-interface forwarding (e.g. Loop-Free Failure Insensitive Routing in [EnyediThesis]).

When there is a link or node failure affecting, but not partitioning, the network, each node will still have at least one path via one of the MRTs to reach the destination D. For example, in Figure 2, C would normally forward traffic to R across the C->R link. If that C->R link fails, then C could use the Blue MRT path C->D->E->R.

As is always the case with fast-reroute technologies, forwarding does not change until a local failure is detected. Packets are forwarded along the shortest path. The appropriate alternate to use is pre-computed. [I-D.ietf-rtgwg-mrt-frr-algorithm] describes exactly how to determine whether the MRT-Blue next-hops or the MRT-Red next-hops should be the MRT alternate next-hops for a particular primary next-hop to a particular destination.

MRT alternates are always available to use. It is a local decision whether to use an MRT alternate, a Loop-Free Alternate or some other type of alternate.

As described in [RFC5286], when a worse failure than is anticipated happens, using LFAs that are not downstream neighbors can cause looping among alternates. Section 1.1 of [RFC5286] gives an example of link-protecting alternates causing a loop on node failure. Even if a worse failure than anticipated happens, the use of MRT alternates will not cause looping.

6. Unicast Forwarding with MRT Fast-Reroute

There are three possible types of routers involved in forwarding a packet along an MRT path. At the MRT ingress router, the packet leaves the shortest path to the destination and follows an MRT path

to the destination. In an FRR application, the MRT ingress router is the PLR. An MRT transit router takes a packet that arrives already associated with the particular MRT, and forwards it on that same MRT. In some situations (to be discussed later), the packet will need to leave the MRT path and return to the shortest path. This takes place at the MRT egress router. The MRT ingress and egress functionality may depend on the underlying type of packet being forwarded (LDP or IP). The MRT transit functionality is independent of the type of packet being forwarded. We first consider several MRT transit forwarding mechanisms. Then we look at how these forwarding mechanisms can be applied to carrying LDP and IP traffic.

6.1. Introduction to MRT Forwarding Options

The following options for MRT forwarding mechanisms are considered.

1. MRT LDP Labels

- A. Topology-scoped FEC encoded using a single label
- B. Topology and FEC encoded using a two label stack

2. MRT IP Tunnels

- A. MRT IPv4 Tunnels
- B. MRT IPv6 Tunnels

6.1.1. MRT LDP labels

We consider two options for the MRT forwarding mechanisms using MRT LDP labels.

6.1.1.1. Topology-scoped FEC encoded using a single label (Option 1A)

[RFC7307] provides a mechanism to distribute FEC-Label bindings scoped to a given MPLS topology (represented by MPLS MT-ID). To use multi-topology LDP to create MRT forwarding topologies, we associate two MPLS MT-IDs with the MRT-Red and MRT-Blue forwarding topologies, in addition to the default shortest path forwarding topology with MT-ID=0.

With this forwarding mechanism, a single label is distributed for each topology-scoped FEC. For a given FEC in the default topology (call it default-FEC-A), two additional topology-scoped FECs would be created, corresponding to the Red and Blue MRT forwarding topologies (call them red-FEC-A and blue-FEC-A). A router supporting this MRT transit forwarding mechanism advertises a different FEC-label binding

for each of the three topology-scoped FECs. When a packet is received with a label corresponding to red-FEC-A (for example), an MRT transit router will determine the next-hop for the MRT-Red forwarding topology for that FEC, swap the incoming label with the outgoing label corresponding to red-FEC-A learned from the MRT-Red next-hop router, and forward the packet.

This forwarding mechanism has the useful property that the FEC associated with the packet is maintained in the labels at each hop along the MRT. We will take advantage of this property when specifying how to carry LDP traffic on MRT paths using multi-topology LDP labels.

This approach is very simple for hardware to support. However, it reduces the label space for other uses, and it increases the memory needed to store the labels and the communication required by LDP to distribute FEC-label bindings. In general, this approach will also increase the time needed to install the FRR entries in the Forwarding Information Base (FIB) and hence the time needed before the next failure can be protected.

This forwarding option uses the LDP signaling extensions described in [RFC7307]. The MRT-specific LDP extensions required to support this option will be described elsewhere.

6.1.1.2. Topology and FEC encoded using a two label stack (Option 1B)

With this forwarding mechanism, a two label stack is used to encode the topology and the FEC of the packet. The top label (topology-id label) identifies the MRT forwarding topology, while the second label (FEC label) identifies the FEC. The top label would be a new FEC type with two values corresponding to MRT Red and Blue topologies.

When an MRT transit router receives a packet with a topology-id label, the router pops the top label and uses that it to guide the next-hop selection in combination with the next label in the stack (the FEC label). The router then swaps the FEC label, using the FEC-label bindings learned through normal LDP mechanisms. The router then pushes the topology-id label for the next-hop.

As with Option 1A, this forwarding mechanism also has the useful property that the FEC associated with the packet is maintained in the labels at each hop along the MRT.

This forwarding mechanism has minimal usage of additional labels, memory and LDP communication. It does increase the size of packets and the complexity of the required label operations and look-ups.

This forwarding option is consistent with context-specific label spaces, as described in [RFC5331]. However, the precise LDP behavior required to support this option for MRT has not been specified.

6.1.1.3. Compatibility of MRT LDP Label Options 1A and 1B

MRT transit forwarding based on MRT LDP Label options 1A and 1B can coexist in the same network, with a packet being forwarded along a single MRT path using the single label of option 1A for some hops and the two label stack of option 1B for other hops. However, to simplify the process of MRT Island formation we require that all routers in the MRT Island support at least one common forwarding mechanism. As an example, the Default MRT Profile requires support for the MRT LDP Label Option 1A forwarding mechanism. This ensures that the routers in an MRT island supporting the Default MRT Profile will be able to establish MRT forwarding paths based on MRT LDP Label Option 1A. However, an implementation supporting Option 1A may also support Option 1B. If the scaling or performance characteristics for the two options differ in this implementation, then it may be desirable for a pair of adjacent routers to use Option 1B labels instead of the Option 1A labels. If those routers successfully negotiate the use of Option 1B labels, they are free to use them. This can occur without any of the other routers in the MRT Island being made aware of it.

Note that this document only defines the Default MRT Profile which requires support for the MRT LDP Label Option 1A forwarding mechanism.

6.1.1.4. Required support for MRT LDP Label options

If a router supports a profile that includes the MRT LDP Label Option 1A for the MRT transit forwarding mechanism, then it **MUST** support option 1A, which encodes topology-scoped FECs using a single label. The router **MAY** also support option 1B.

If a router supports a profile that includes the MRT LDP Label Option 1B for the MRT transit forwarding mechanism, then it **MUST** support option 1B, which encodes the topology and FEC using a two label stack. The router **MAY** also support option 1A.

6.1.2. MRT IP tunnels (Options 2A and 2B)

IP tunneling can also be used as an MRT transit forwarding mechanism. Each router supporting this MRT transit forwarding mechanism announces two additional loopback addresses and their associated MRT color. Those addresses are used as destination addresses for MRT-blue and MRT-red IP tunnels respectively. The special loopback

addresses allow the transit nodes to identify the traffic as being forwarded along either the MRT-blue or MRT-red topology to reach the tunnel destination. For example, an MRT ingress router can cause a packet to be tunneled along the MRT-red path to router X by encapsulating the packet using the MRT-red loopback address advertised by router X. Upon receiving the packet, router X would remove the encapsulation header and forward the packet based on the original destination address.

Either IPv4 (option 2A) or IPv6 (option 2B) can be used as the tunneling mechanism.

Note that the two forwarding mechanisms using LDP Label options do not require additional loopbacks per router, as is required by the IP tunneling mechanism. This is because LDP labels are used on a hop-by-hop basis to identify MRT-blue and MRT-red forwarding topologies.

6.2. Forwarding LDP Unicast Traffic over MRT Paths

In the previous section, we examined several options for providing MRT transit forwarding functionality, which is independent of the type of traffic being carried. We now look at the MRT ingress functionality, which will depend on the type of traffic being carried (IP or LDP). We start by considering LDP traffic.

We also simplify the initial discussion by assuming that the network consists of a single IGP area, and that all routers in the network participate in MRT. Other deployment scenarios that require MRT egress functionality are considered later in this document.

In principle, it is possible to carry LDP traffic in MRT IP tunnels. However, for LDP traffic, it is desirable to avoid tunneling. Tunneling LDP traffic to a remote node requires knowledge of remote FEC-label bindings so that the LDP traffic can continue to be forwarded properly when it leaves the tunnel. This requires targeted LDP sessions which can add management complexity. As described below, the two MRT forwarding mechanisms that use LDP labels do not require targeted LDP sessions.

6.2.1. Forwarding LDP traffic using MRT LDP Label Option 1A

The MRT LDP Label option 1A forwarding mechanism uses topology-scoped FECs encoded using a single label as described in section Section 6.1.1.1. When a PLR receives an LDP packet that needs to be forwarded on the Red MRT (for example), it does a label swap operation, replacing the usual LDP label for the FEC with the Red MRT label for that FEC received from the next-hop router in the Red MRT computed by the PLR. When the next-hop router in the Red MRT

receives the packet with the Red MRT label for the FEC, the MRT transit forwarding functionality continues as described in Section 6.1.1.1. In this way the original FEC associated with the packet is maintained at each hop along the MRT.

6.2.2. Forwarding LDP traffic using MRT LDP Label Option 1B

The MRT LDP Label option 1B forwarding mechanism encodes the topology and the FEC using a two label stack as described in Section 6.1.1.2. When a PLR receives an LDP packet that needs to be forwarded on the Red MRT, it first does a normal LDP label swap operation, replacing the incoming normal LDP label associated with a given FEC with the outgoing normal LDP label for that FEC learned from the next-hop on the Red MRT. In addition, the PLR pushes the topology-identification label associated with the Red MRT, and forward the packet to the appropriate next-hop on the Red MRT. When the next-hop router in the Red MRT receives the packet with the Red MRT label for the FEC, the MRT transit forwarding functionality continues as described in Section 6.1.1.2. As with option 1A, the original FEC associated with the packet is maintained at each hop along the MRT.

6.2.3. Other considerations for forwarding LDP traffic using MRT LDP Labels

Note that forwarding LDP traffic using MRT LDP Labels can be done without the use of targeted LDP sessions when an MRT path to the destination FEC is used. The alternates selected in [I-D.ietf-rtgwg-mrt-frr-algorithm] use the MRT path to the destination FEC, so targeted LDP sessions are not needed. If instead one found it desirable to have the PLR use an MRT to reach the primary next-next-hop for the FEC, and then continue forwarding the LDP packet along the shortest path tree from the primary next-next-hop, this would require tunneling to the primary next-next-hop and a targeted LDP session for the PLR to learn the FEC-label binding for primary next-next-hop to correctly forward the packet.

6.2.4. Required support for LDP traffic

For greatest hardware compatibility, routers implementing MRT fast-reroute of LDP traffic MUST support Option 1A of encoding the MT-ID in the labels (See Section 9).

6.3. Forwarding IP Unicast Traffic over MRT Paths

For IPv4 traffic, there is no currently practical alternative except tunneling to gain the bits needed to indicate the MRT-Blue or MRT-Red forwarding topology. For IPv6 traffic, in principle one could define bits in the IPv6 options header to indicate the MRT-Blue or MRT-Red

forwarding topology. However, in this document, we have chosen not to define a solution that would work for IPv6 traffic but not for IPv4 traffic.

The choice of tunnel egress is flexible since any router closer to the destination than the next-hop can work. This architecture assumes that the original destination in the area is selected (see Section 11 for handling of multi-homed prefixes); another possible choice is the next-next-hop towards the destination. As discussed in the previous section, for LDP traffic, using the MRT to the original destination simplifies MRT-FRR by avoiding the need for targeted LDP sessions to the next-next-hop. For IP, that consideration doesn't apply.

Some situations require tunneling IP traffic along an MRT to a tunnel endpoint that is not the destination of the IP traffic. These situations will be discussed in detail later. We note here that an IP packet with a destination in a different IGP area/level from the PLR should be tunneled on the MRT to the Area Border Router (ABR) or Level Border Router (LBR) on the shortest path to the destination. For a destination outside of the PLR's MRT Island, the packet should be tunneled on the MRT to a non-proxy-node immediately before the named proxy-node on that particular color MRT.

6.3.1. Tunneling IP traffic using MRT LDP Labels

An IP packet can be tunneled along an MRT path by pushing the appropriate MRT LDP label(s). Tunneling using LDP labels, as opposed to IP headers, has the advantage that more installed routers can do line-rate encapsulation and decapsulation using LDP than using IP. Also, no additional IP addresses would need to be allocated or signaled.

6.3.1.1. Tunneling IP traffic using MRT LDP Label Option 1A

The MRT LDP Label option 1A forwarding mechanism uses topology-scoped FECs encoded using a single label as described in section Section 6.1.1.1. When a PLR receives an IP packet that needs to be forwarded on the Red MRT to a particular tunnel endpoint, it does a label push operation. The label pushed is the Red MRT label for a FEC originated by the tunnel endpoint, learned from the next-hop on the Red MRT.

6.3.1.2. Tunneling IP traffic using MRT LDP Label Option 1B

The MRT LDP Label option 1B forwarding mechanism encodes the topology and the FEC using a two label stack as described in Section 6.1.1.2. When a PLR receives an IP packet that needs to be forwarded on the

Red MRT to a particular tunnel endpoint, the PLR pushes two labels on the IP packet. The first (inner) label is the normal LDP label learned from the next-hop on the Red MRT, associated with a FEC originated by the tunnel endpoint. The second (outer) label is the topology-identification label associated with the Red MRT.

For completeness, we note here a potential variation that uses a single label as opposed to two labels. In order to tunnel an IP packet over an MRT to the destination of the IP packet (as opposed to an arbitrary tunnel endpoint), then we could just push a topology-identification label directly onto the packet. An MRT transit router would need to pop the topology-id label, do an IP route lookup in the context of that topology-id, and push the topology-id label.

6.3.2. Tunneling IP traffic using MRT IP Tunnels

In order to tunnel over the MRT to a particular tunnel endpoint, the PLR encapsulates the original IP packet with an additional IP header using the MRT-Blue or MRT-Red loopback address of the tunnel endpoint.

6.3.3. Required support for IP traffic

For greatest hardware compatibility and ease in removing the MRT-topology marking at area/level boundaries, routers that support MPLS and implement IP MRT fast-reroute MUST support tunneling of IP traffic using MRT LDP Label Option 1A (topology-scoped FEC encoded using a single label).

7. MRT Island Formation

The purpose of communicating support for MRT is to indicate that the MRT-Blue and MRT-Red forwarding topologies are created for transit traffic. The MRT architecture allows for different, potentially incompatible options. In order to create consistent MRT forwarding topologies, the routers participating in a particular MRT Island need to use the same set of options. These options are grouped into MRT profiles. In addition, the routers in an MRT Island all need to use the same set of nodes and links within the Island when computing the MRT forwarding topologies. This section describes the information used by a router to determine the nodes and links to include in a particular MRT Island. Some information already exists in the IGP and can be used by MRT in Island formation, subject to the interpretation defined here.

Other information needs to be communicated between routers for which there do not currently exist protocol extensions. This new information needs to be shared among all routers in an IGP area, so

defining extensions to existing IGPs to carry this information makes sense. These new protocol extensions will be defined elsewhere.

Deployment scenarios using multi-topology OSPF or IS-IS, or running both IS-IS and OSPF on the same routers is out of scope for this specification. As with LFA, it is expected that OSPF Virtual Links will not be supported.

At a high level, an MRT Island is defined as the set of routers supporting the same MRT profile, in the same IGP area/level and the bi-directional links interconnecting those routers. More detailed descriptions of these criteria are given below.

7.1. IGP Area or Level

All links in an MRT Island are bidirectional and belong to the same IGP area or level. For IS-IS, a link belonging to both level 1 and level 2 would qualify to be in multiple MRT Islands. A given ABR or LBR can belong to multiple MRT Islands, corresponding to the areas or levels in which it participates. Inter-area forwarding behavior is discussed in Section 10.

7.2. Support for a specific MRT profile

All routers in an MRT Island support the same MRT profile. A router advertises support for a given MRT profile using an 8-bit MRT Profile ID value. The registry for the MRT Profile ID is defined in this document. The protocol extensions for advertising the MRT Profile ID value will be defined elsewhere. A given router can support multiple MRT profiles and participate in multiple MRT Islands. The options that make up an MRT profile, as well as the default MRT profile, are defined in Section 8.

The process of MRT Island formation takes place independently for each MRT profile advertised by a given router. For example, consider a network with 40 connected routers in the same area advertising support for MRT Profile A and MRT Profile B. Two distinct MRT Islands will be formed corresponding to Profile A and Profile B, with each island containing all 40 routers. A complete set of maximally redundant trees will be computed for each island following the rules defined for each profile. If we add a third MRT Profile to this example, with Profile C being advertised by a connected subset of 30 routers, there will be a third MRT Island formed corresponding to those 30 routers, and a third set of maximally redundant trees will be computed. In this example, 40 routers would compute and install two sets of MRT transit forwarding entries corresponding to Profiles A and B, while 30 routers would compute and install three sets of MRT transit forwarding entries corresponding to Profiles A, B, and C.

7.3. Excluding additional routers and interfaces from the MRT Island

MRT takes into account existing IGP mechanisms for discouraging traffic from using particular links and routers, and it introduces an MRT-specific exclusion mechanism for links.

7.3.1. Existing IGP exclusion mechanisms

Mechanisms for discouraging traffic from using particular links already exist in IS-IS and OSPF. In IS-IS, an interface configured with a metric of $2^{24}-2$ (0xFFFFFE) will only be used as a last resort. (An interface configured with a metric of $2^{24}-1$ (0xFFFFF) will not be advertised into the topology.) In OSPF, an interface configured with a metric of $2^{16}-1$ (0xFFFF) will only be used as a last resort. These metrics can be configured manually to enforce administrative policy, or they can be set in an automated manner as with LDP IGP synchronization [RFC5443].

Mechanisms also already exist in IS-IS and OSPF to discourage or prevent transit traffic from using a particular router. In IS-IS, the overload bit is prevents transit traffic from using a router.

For OSPFv2 and OSPFv3, [RFC6987] specifies setting all outgoing interface metrics to 0xFFFF to discourage transit traffic from using a router. ([RFC6987] defines the metric value 0xFFFF as MaxLinkMetric, a fixed architectural value for OSPF.) For OSPFv3, [RFC5340] specifies that a router be excluded from the intra-area shortest path tree computation if the V6-bit or R-bit of the LSA options is not set in the Router LSA.

The following rules for MRT Island formation ensure that MRT FRR protection traffic does not use a link or router that is discouraged or prevented from carrying traffic by existing IGP mechanisms.

1. A bidirectional link MUST be excluded from an MRT Island if either the forward or reverse cost on the link is 0xFFFFFE (for IS-IS) or 0xFFFF for OSPF.
2. A router MUST be excluded from an MRT Island if it is advertised with the overload bit set (for IS-IS), or it is advertised with metric values of 0xFFFF on all of its outgoing interfaces (for OSPFv2 and OSPFv3).
3. A router MUST be excluded from an MRT Island if it is advertised with either the V6-bit or R-bit of the LSA options not set in the Router LSA.

7.3.2. MRT-specific exclusion mechanism

This architecture also defines a means of excluding an otherwise usable link from MRT Islands. The protocol extensions for advertising that a link is MRT-Ineligible will be defined elsewhere. A link with either interface advertised as MRT-Ineligible MUST be excluded from an MRT Island. Note that an interface advertised as MRT-Ineligible by a router is ineligible with respect to all profiles advertised by that router.

7.4. Connectivity

All of the routers in an MRT Island MUST be connected by bidirectional links with other routers in the MRT Island. Disconnected MRT Islands will operate independently of one another.

7.5. Algorithm for MRT Island Identification

An algorithm that allows a computing router to identify the routers and links in the local MRT Island satisfying the above rules is given in section 5.2 of [I-D.ietf-rtgwg-mrt-frr-algorithm].

8. MRT Profile

An MRT Profile is a set of values and options related to MRT behavior. The complete set of options is designated by the corresponding 8-bit Profile ID value.

This document specifies the values and options that correspond to the Default MRT Profile (Profile ID = 0). Future documents may define other MRT Profiles by specifying the MRT Profile Options below.

8.1. MRT Profile Options

Below is a description of the values and options that define an MRT Profile.

MRT Algorithm: This identifies the particular algorithm for computing maximally redundant trees used by the router for this profile.

MRT-Red MT-ID: This specifies the MPLS MT-ID to be associated with the MRT-Red forwarding topology. It is allocated from the MPLS Multi-Topology Identifiers Registry.

MRT-Blue MT-ID: This specifies the MPLS MT-ID to be associated with the MRT-Blue forwarding topology. It is allocated from the MPLS Multi-Topology Identifiers Registry.

GADAG Root Selection Policy: This specifies the manner in which the GADAG root is selected. All routers in the MRT island need to use the same GADAG root in the calculations used to construct the MRTs. A valid GADAG Root Selection Policy **MUST** be such that each router in the MRT island chooses the same GADAG root based on information available to all routers in the MRT island. GADAG Root Selection Priority values, advertised as router-specific MRT parameters, **MAY** be used in a GADAG Root Selection Policy.

MRT Forwarding Mechanism: This specifies which forwarding mechanism the router uses to carry transit traffic along MRT paths. A router which supports a specific MRT forwarding mechanism must program appropriate next-hops into the forwarding plane. The current options are MRT LDP Label Option 1A, MRT LDP Label Option 1B, IPv4 Tunneling, IPv6 Tunneling, and None. If IPv4 is supported, then both MRT-Red and MRT-Blue IPv4 Loopback Addresses **SHOULD** be specified. If IPv6 is supported, both MRT-Red and MRT-Blue IPv6 Loopback Addresses **SHOULD** be specified.

Recalculation: Recalculation specifies the process and timing by which new MRTs are computed after the topology has been modified.

Area/Level Border Behavior: This specifies how traffic traveling on the MRT-Blue or MRT-Red in one area should be treated when it passes into another area.

Other Profile-Specific Behavior: Depending upon the use-case for the profile, there may be additional profile-specific behavior.

When a new MRT Profile is defined, new and unique values should be allocated from the MPLS Multi-Topology Identifiers Registry, corresponding to the MRT-Red and MRT-Blue MT-ID values for the new MRT Profile .

If a router advertises support for multiple MRT profiles, then it **MUST** create the transit forwarding topologies for each of those, unless the profile specifies the None option for MRT Forwarding Mechanism.

The ability of MRT-FRR to support transit forwarding entries for multiple profiles can be used to facilitate a smooth transition from an existing deployed MRT Profile to a new MRT Profile. The new profile can be activated in parallel with the existing profile, installing the transit forwarding entries for the new profile without affecting the transit forwarding entries for the existing profile. Once the new transit forwarding state has been verified, the router can be configured to use the alternates computed by the new profile in the event of a failure.

8.2. Router-specific MRT parameters

For some profiles, additional router-specific MRT parameters may need to be advertised. While the set of options indicated by the MRT Profile ID must be identical for all routers in an MRT Island, these router-specific MRT parameters may differ between routers in the same MRT island. Several such parameters are described below.

GADAG Root Selection Priority: A GADAG Root Selection Policy MAY rely on the GADAG Root Selection Priority values advertised by each router in the MRT island. A GADAG Root Selection Policy may use the GADAG Root Selection Priority to allow network operators to configure a parameter to ensure that the GADAG root is selected from a particular subset of routers. An example of this use of the GADAG Root Selection Priority value by the GADAG Root Selection Policy is given in the Default MRT profile below.

MRT-Red Loopback Address: This provides the router's loopback address to reach the router via the MRT-Red forwarding topology. It can be specified for either IPv4 or IPv6. Note that this parameter is not needed to support the Default MRT profile.

MRT-Blue Loopback Address: This provides the router's loopback address to reach the router via the MRT-Blue forwarding topology. It can be specified for either IPv4 and IPv6. Note that this parameter is not needed to support the Default MRT profile.

Protocol extensions for advertising a router's GADAG Root Selection Priority value will be defined in other documents. Protocol extensions for the advertising a router's MRT-Red and MRT-Blue Loopback Addresses will be defined elsewhere.

8.3. Default MRT profile

The following set of options defines the default MRT Profile. The default MRT profile is indicated by the MRT Profile ID value of 0.

MRT Algorithm: MRT Lowpoint algorithm defined in [I-D.ietf-rtgwg-mrt-frr-algorithm].

MRT-Red MPLS MT-ID: This value will be allocated from the MPLS Multi-Topology Identifiers Registry. The IANA request for this allocation will be in another document.

MRT-Blue MPLS MT-ID: This value will be allocated from the MPLS Multi-Topology Identifiers Registry. The IANA request for this allocation will be in another document.

GADAG Root Selection Policy: Among the routers in the MRT Island with the lowest numerical value advertised for GADAG Root Selection Priority, an implementation MUST pick the router with the highest Router ID to be the GADAG root. Note that a lower numerical value for GADAG Root Selection Priority indicates a higher preference for selection.

Forwarding Mechanisms: MRT LDP Label Option 1A

Recalculation: Recalculation of MRTs SHOULD occur as described in Section 12.2. This allows the MRT forwarding topologies to support IP/LDP fast-reroute traffic.

Area/Level Border Behavior: As described in Section 10, ABRs/LBRs SHOULD ensure that traffic leaving the area also exits the MRT-Red or MRT-Blue forwarding topology.

9. LDP signaling extensions and considerations

The protocol extensions for LDP will be defined in another document. A router must indicate that it has the ability to support MRT; having this explicit allows the use of MRT-specific processing, such as special handling of FECs sent with the Rainbow MRT MT-ID.

A FEC sent with the Rainbow MRT MT-ID indicates that the FEC applies to all the MRT-Blue and MRT-Red MT-IDs in supported MRT profiles. The FEC-label bindings for the default shortest-path based MT-ID 0 MUST still be sent (even though it could be inferred from the Rainbow FEC-label bindings) to ensure continuous operation of normal LDP forwarding. The Rainbow MRT MT-ID is defined to provide an easy way to handle the special signaling that is needed at ABRs or LBRs. It avoids the problem of needing to signal different MPLS labels to different LDP neighbors for the same FEC. Because the Rainbow MRT MT-ID is used only by ABRs/LBRs or an LDP egress router, it is not MRT profile specific.

The value of the Rainbow MRT MPLS MT-ID will be allocated from the MPLS Multi-Topology Identifiers Registry. The IANA request for this allocation will be in another document.

10. Inter-area Forwarding Behavior

An ABR/LBR has two forwarding roles. First, it forwards traffic within areas. Second, it forwards traffic from one area into another. These same two roles apply for MRT transit traffic. Traffic on MRT-Red or MRT-Blue destined inside the area needs to stay on MRT-Red or MRT-Blue in that area. However, it is desirable for

traffic leaving the area to also exit MRT-Red or MRT-Blue and return to shortest path forwarding.

For unicast MRT-FRR, the need to stay on an MRT forwarding topology terminates at the ABR/LBR whose best route is via a different area/level. It is highly desirable to go back to the default forwarding topology when leaving an area/level. There are three basic reasons for this. First, the default topology uses shortest paths; the packet will thus take the shortest possible route to the destination. Second, this allows a single router failure that manifests itself in multiple areas (as would be the case with an ABR/LBR failure) to be separately identified and repaired around. Third, the packet can be fast-rerouted again, if necessary, due to a second distinct failure in a different area.

In OSPF, an ABR that receives a packet on MRT-Red or MRT-Blue towards destination Z should continue to forward the packet along MRT-Red or MRT-Blue only if the best route to Z is in the same OSPF area as the interface that the packet was received on. Otherwise, the packet should be removed from MRT-Red or MRT-Blue and forwarded on the shortest-path default forwarding topology.

The above description applies to OSPF. The same essential behavior also applies to IS-IS if one substitutes IS-IS level for OSPF area. However, the analogy with OSPF is not exact. An interface in OSPF can only be in one area, whereas an interface in IS-IS can be in both Level-1 and Level-2. Therefore, to avoid confusion and address this difference, we explicitly describe the behavior for IS-IS in Appendix A. In the following sections only the OSPF terminology is used.

10.1. ABR Forwarding Behavior with MRT LDP Label Option 1A

For LDP forwarding where a single label specifies (MT-ID, FEC), the ABR is responsible for advertising the proper label to each neighbor. Assume that an ABR has allocated three labels for a particular destination; those labels are *L_primary*, *L_blue*, and *L_red*. To those routers in the same area as the best route to the destination, the ABR advertises the following FEC-label bindings: *L_primary* for the default topology, *L_blue* for the MRT-Blue MT-ID and *L_red* for the MRT-Red MT-ID, as expected. However, to routers in other areas, the ABR advertises the following FEC-label bindings: *L_primary* for the default topology, and *L_primary* for the Rainbow MRT MT-ID. Associating *L_primary* with the Rainbow MRT MT-ID causes the receiving routers to use *L_primary* for the MRT-Blue MT-ID and for the MRT-Red MT-ID.

The ABR installs all next-hops for the best area: primary next-hops for `L_primary`, MRT-Blue next-hops for `L_blue`, and MRT-Red next-hops for `L_red`. Because the ABR advertised (Rainbow MRT MT-ID, FEC) with `L_primary` to neighbors not in the best area, packets from those neighbors will arrive at the ABR with a label `L_primary` and will be forwarded into the best area along the default topology. By controlling what labels are advertised, the ABR can thus enforce that packets exiting the area do so on the shortest-path default topology.

10.1.1. Motivation for Creating the Rainbow-FEC

The desired forwarding behavior could be achieved in the above example without using the Rainbow-FEC. This could be done by having the ABR advertise the following FEC-label bindings to neighbors not in the best area: `L1_primary` for the default topology, `L1_primary` for the MRT-Blue MT-ID, and `L1_primary` for the MRT-Red MT-ID. Doing this would require machinery to spoof the labels used in FEC-label binding advertisements on a per-neighbor basis. Such label-spoofing machinery does not currently exist in most LDP implementations and doesn't have other obvious uses.

Many existing LDP implementations do however have the ability to filter FEC-label binding advertisements on a per-neighbor basis. The Rainbow-FEC allows us to re-use the existing per-neighbor FEC filtering machinery to achieve the desired result. By introducing the Rainbow FEC, we can use per-neighbor FEC-filtering machinery to advertise the FEC-label binding for the Rainbow-FEC (and filter those for MRT-Blue and MRT-Red) to non-best-area neighbors of the ABR.

An ABR may choose to either advertise the Rainbow-FEC or advertise separate MRT-Blue and MRT-Red advertisements. This is a local choice. A router that supports the MRT LDP Label Option 1A Forwarding Mechanism MUST be able to receive and correctly interpret the Rainbow-FEC.

10.2. ABR Forwarding Behavior with IP Tunneling (option 2)

If IP tunneling is used, then the ABR behavior is dependent upon the outermost IP address. If the outermost IP address is an MRT loopback address of the ABR, then the packet is decapsulated and forwarded based upon the inner IP address, which should go on the default SPT topology. If the outermost IP address is not an MRT loopback address of the ABR, then the packet is simply forwarded along the associated forwarding topology. A PLR sending traffic to a destination outside its local area/level will pick the MRT and use the associated MRT loopback address of the selected ABR advertising the lowest cost to the external destination.

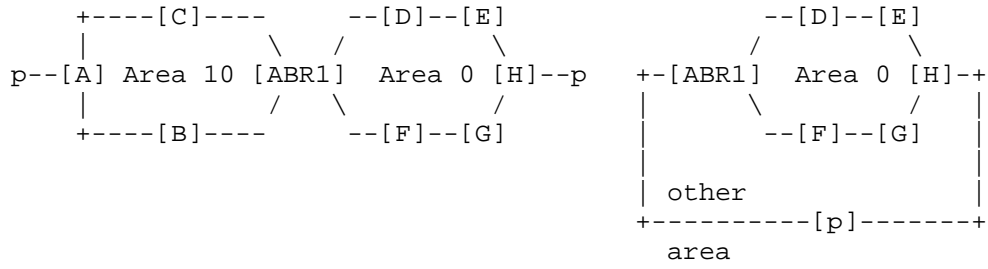
Thus, for these two MRT Forwarding Mechanisms (MRT LDP Label option 1A and IP tunneling option 2), there is no need for additional computation or per-area forwarding state.

10.3. ABR Forwarding Behavior with MRT LDP Label option 1B

The other MRT forwarding mechanism described in Section 6 uses two labels, a topology-id label, and a FEC-label. This mechanism would require that any router whose MRT-Red or MRT-Blue next-hop is an ABR would need to determine whether the ABR would forward the packet out of the area/level. If so, then that router should pop off the topology-identification label before forwarding the packet to the ABR.

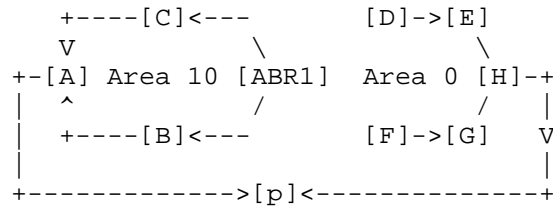
For example, in Figure 3, if node H fails, node E has to put traffic towards prefix p onto MRT-Red. But since node D knows that ABR1 will use a best route from another area, it is safe for D to pop the Topology-Identification Label and just forward the packet to ABR1 along the MRT-Red next-hop. ABR1 will use the shortest path in Area 10.

In all cases for IS-IS and most cases for OSPF, the penultimate router can determine what decision the adjacent ABR will make. The one case where it can't be determined is when two ASBRs are in different non-backbone areas attached to the same ABR, then the ASBR's Area ID may be needed for tie-breaking (prefer the route with the largest OPSF area ID) and the Area ID isn't announced as part of the ASBR link-state advertisement (LSA). In this one case, suboptimal forwarding along the MRT in the other area would happen. If that becomes a realistic deployment scenario, protocol extensions could be developed to address this issue.

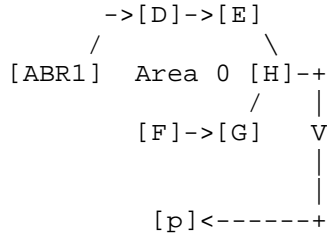


(a) Example topology

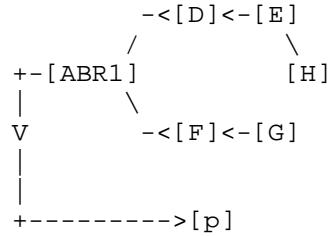
(b) Proxy node view in Area 0 nodes



(c) rSPT towards destination p



(d) Blue MRT in Area 0



(e) Red MRT in Area 0

Figure 3: ABR Forwarding Behavior and MRTs

11. Prefixes Multiply Attached to the MRT Island

How a computing router *S* determines its local MRT Island for each supported MRT profile is already discussed in Section 7.

There are two types of prefixes or FECs that may be multiply attached to an MRT Island. The first type are multi-homed prefixes that usually connect at a domain or protocol boundary. The second type represent routers that do not support the profile for the MRT Island.

The key difference is whether the traffic, once out of the MRT Island, might re-enter the MRT Island if a loop-free exit point is not selected.

FRR using LFA has the useful property that it is able to protect multi-homed prefixes against ABR failure. For instance, if a prefix from the backbone is available via both ABR A and ABR B, if A fails, then the traffic should be redirected to B. This can be accomplished with MRT FRR as well.

If ASBR protection is desired, this has additional complexities if the ASBRs are in different areas. Similarly, protecting labeled BGP traffic in the event of an ASBR failure has additional complexities due to the per-ASBR label spaces involved.

As discussed in [RFC5286], a multi-homed prefix could be:

- o An out-of-area prefix announced by more than one ABR,
- o An AS-External route announced by 2 or more ASBRs,
- o A prefix with iBGP multipath to different ASBRs,
- o etc.

See Appendix B for a discussion of a general issue with multi-homed prefixes connected in two different areas.

There are also two different approaches to protection. The first is tunnel endpoint selection where the PLR picks a router to tunnel to where that router is loop-free with respect to the failure-point. Conceptually, the set of candidate routers to provide LFAs expands to all routers that can be reached via an MRT alternate, attached to the prefix.

The second is to use a proxy-node, that can be named via MPLS label or IP address, and pick the appropriate label or IP address to reach it on either MRT-Blue or MRT-Red as appropriate to avoid the failure point. A proxy-node can represent a destination prefix that can be attached to the MRT Island via at least two routers. It is termed a named proxy-node if there is a way that traffic can be encapsulated to reach specifically that proxy-node; this could be because there is an LDP FEC for the associated prefix or because MRT-Red and MRT-Blue IP addresses are advertised (in an as-yet undefined fashion) for that proxy-node. Traffic to a named proxy-node may take a different path than traffic to the attaching router; traffic is also explicitly forwarded from the attaching router along a predetermined interface towards the relevant prefixes.

For IP traffic, multi-homed prefixes can use tunnel endpoint selection. For IP traffic that is destined to a router outside the MRT Island, if that router is the egress for a FEC advertised into the MRT Island, then the named proxy-node approach can be used.

For LDP traffic, there is always a FEC advertised into the MRT Island. The named proxy-node approach should be used, unless the computing router S knows the label for the FEC at the selected tunnel endpoint.

If a FEC is advertised from outside the MRT Island into the MRT Island and the forwarding mechanism specified in the profile includes LDP, then the routers learning that FEC MUST also advertise labels for (MRT-Red, FEC) and (MRT-Blue, FEC) to neighbors inside the MRT Island. Any router receiving a FEC corresponding to a router outside the MRT Island or to a multi-homed prefix MUST compute and install the transit MRT-Blue and MRT-Red next-hops for that FEC. The FEC-label bindings for the topology-scoped FECs ((MT-ID 0, FEC), (MRT-Red, FEC), and (MRT-Blue, FEC)) MUST also be provided via LDP to neighbors inside the MRT Island.

11.1. Protecting Multi-Homed Prefixes using Tunnel Endpoint Selection

Tunnel endpoint selection is a local matter for a router in the MRT Island since it pertains to selecting and using an alternate and does not affect the transit MRT-Red and MRT-Blue forwarding topologies.

Let the computing router be S and the next-hop F be the node whose failure is to be avoided. Let the destination be prefix p. Have A be the router to which the prefix p is attached for S's shortest path to p.

The candidates for tunnel endpoint selection are those to which the destination prefix is attached in the area/level. For a particular candidate B, it is necessary to determine if B is loop-free to reach p with respect to S and F for node-protection or at least with respect to S and the link (S, F) for link-protection. If B will always prefer to send traffic to p via a different area/level, then this is definitional. Otherwise, distance-based computations are necessary and an SPF from B's perspective may be necessary. The following equations give the checks needed; the rationale is similar to that given in [RFC5286]. In the inequalities below, $D_{opt}(X,Y)$ means the shortest distance from node X to node Y, and $D_{opt}(X,p)$ means the shortest distance from node X to prefix p.

Loop-Free for S: $D_{opt}(B, p) < D_{opt}(B, S) + D_{opt}(S, p)$

Loop-Free for F: $D_{opt}(B, p) < D_{opt}(B, F) + D_{opt}(F, p)$

The latter is equivalent to the following, which avoids the need to compute the shortest path from F to p.

Loop-Free for F: $D_{\text{opt}}(B, p) < D_{\text{opt}}(B, F) + D_{\text{opt}}(S, p) - D_{\text{opt}}(S, F)$

Finally, the rules for Endpoint selection are given below. The basic idea is to repair to the prefix-advertising router selected for the shortest-path and only to select and tunnel to a different endpoint if necessary (e.g. $A=F$ or F is a cut-vertex or the link (S,F) is a cut-link).

1. Does S have a node-protecting alternate to A? If so, select that. Tunnel the packet to A along that alternate. For example, if LDP is the forwarding mechanism, then push the label (MRT-Red, A) or (MRT-Blue, A) onto the packet.
2. If not, then is there a router B that is loop-free to reach p while avoiding both F and S? If so, select B as the end-point. Determine the MRT alternate to reach B while avoiding F. Tunnel the packet to B along that alternate. For example, with LDP, push the label (MRT-Red, B) or (MRT-Blue, B) onto the packet.
3. If not, then does S have a link-protecting alternate to A? If so, select that.
4. If not, then is there a router B that is loop-free to reach p while avoiding S and the link from S to F? If so, select B as the endpoint and the MRT alternate for reaching B from S that avoid the link (S,F).

The tunnel endpoint selected will receive a packet destined to itself and, being the egress, will pop that MPLS label (or have signaled Implicit Null) and forward based on what is underneath. This suffices for IP traffic since the tunnel endpoint can use the IP header of the original packet to continue forwarding the packet. However, tunnelling of LDP traffic requires targeted LDP sessions for learning the FEC-label binding at the tunnel endpoint.

11.2. Protecting Multi-Homed Prefixes using Named Proxy-Nodes

Instead, the named proxy-node method works with LDP traffic without the need for targeted LDP sessions. It also has a clear advantage over tunnel endpoint selection, in that it is possible to explicitly forward from the MRT Island along an interface to a loop-free island neighbor when that interface may not be a primary next-hop.

A named proxy-node represents one or more destinations and, for LDP forwarding, has a FEC associated with it that is signalled into the MRT Island. Therefore, it is possible to explicitly label packets to go to (MRT-Red, FEC) or (MRT-Blue, FEC); at the border of the MRT Island, the label will swap to meaning (MT-ID 0, FEC). It would be possible to have named proxy-nodes for IP forwarding, but this would require extensions to signal two IP addresses to be associated with MRT-Red and MRT-Blue for the proxy-node. A named proxy-node can be uniquely represented by the two routers in the MRT Island to which it is connected. The extensions to signal such IP addresses will be defined elsewhere. The details of what label-bindings must be originated will be described in another document.

Computing the MRT next-hops to a named proxy-node and the MRT alternate for the computing router S to avoid a particular failure node F is straightforward. The details of the simple constant-time functions, `Select_Proxy_Node_NHs()` and `Select_Alternates_Proxy_Node()`, are given in [I-D.ietf-rtgwg-mrt-frr-algorithm]. A key point is that computing these MRT next-hops and alternates can be done as new named proxy-nodes are added or removed without requiring a new MRT computation or impacting other existing MRT paths. This maps very well to, for example, how OSPFv2 (see [RFC2328] Section 16.5) does incremental updates for new summary-LSAs.

The remaining question is how to attach the named proxy-node to the MRT Island; all the routers in the MRT Island MUST do this consistently. No more than 2 routers in the MRT Island can be selected; one should only be selected if there are no others that meet the necessary criteria. The named proxy-node is logically part of the area/level.

There are two sources for candidate routers in the MRT Island to connect to the named proxy-node. The first set are those routers in the MRT Island that are advertising the prefix; the named-proxy-cost assigned to each prefix-advertising router is the announced cost to the prefix. The second set are those routers in the MRT Island that are connected to routers not in the MRT Island but in the same area/level; such routers will be defined as Island Border Routers (IBRs). The routers connected to the IBRs that are not in the MRT Island and are in the same area/level as the MRT island are Island Neighbors (INs).

Since packets sent to the named proxy-node along MRT-Red or MRT-Blue may come from any router inside the MRT Island, it is necessary that whatever router to which an IBR forwards the packet be loop-free with respect to the whole MRT Island for the destination. Thus, an IBR is a candidate router only if it possesses at least one IN whose

shortest path to the prefix does not enter the MRT Island. A method for identifying loop-free Island Neighbors(LFINs) is given in [I-D.ietf-rtgwg-mrt-frr-algorithm]. The named-proxy-cost assigned to each (IBR, IN) pair is $\text{cost}(\text{IBR}, \text{IN}) + D_{\text{opt}}(\text{IN}, \text{prefix})$.

From the set of prefix-advertising routers and the set of IBRs with at least one LFIN, the two routers with the lowest named-proxy-cost are selected. Ties are broken based upon the lowest Router ID. For ease of discussion, the two selected routers will be referred to as proxy-node attachment routers.

A proxy-node attachment router has a special forwarding role. When a packet is received destined to (MRT-Red, prefix) or (MRT-Blue, prefix), if the proxy-node attachment router is an IBR, it MUST swap to the shortest path forwarding topology (e.g. swap to the label for (MT-ID 0, prefix) or remove the outer IP encapsulation) and forward the packet to the IN whose cost was used in the selection. If the proxy-node attachment router is not an IBR, then the packet MUST be removed from the MRT forwarding topology and sent along the interface(s) that caused the router to advertise the prefix; this interface might be out of the area/level/AS.

11.3. MRT Alternates for Destinations Outside the MRT Island

A natural concern with new functionality is how to have it be useful when it is not deployed across an entire IGP area. In the case of MRT FRR, where it provides alternates when appropriate LFAs aren't available, there are also deployment scenarios where it may make sense to only enable some routers in an area with MRT FRR. A simple example of such a scenario would be a ring of 6 or more routers that is connected via two routers to the rest of the area.

Destinations inside the local island can obviously use MRT alternates. Destinations outside the local island can be treated like a multi-homed prefix and either Endpoint Selection or Named Proxy-Nodes can be used. Named Proxy-Nodes MUST be supported when LDP forwarding is supported and a label-binding for the destination is sent to an IBR.

Naturally, there are more complicated options to improve coverage, such as connecting multiple MRT islands across tunnels, but the need for the additional complexity has not been justified.

12. Network Convergence and Preparing for the Next Failure

After a failure, MRT detours ensure that packets reach their intended destination while the IGP has not reconverged onto the new topology. As link-state updates reach the routers, the IGP process calculates

the new shortest paths. Two things need attention: micro-loop prevention and MRT re-calculation.

12.1. Micro-loop prevention and MRTs

A micro-loop is a transient packet forwarding loop among two or more routers that can occur during convergence of IGP forwarding state. [RFC5715] discusses several techniques for preventing micro-loops. This section discusses how MRT-FRR relates to two of the micro-loop prevention techniques discussed in [RFC5715], Nearside Tunneling and Farside Tunneling.

In Nearside Tunneling, a router (PLR) adjacent to a failure perform local repair and inform remote routers of the failure. The remote routers initially tunnel affected traffic to the nearest PLR, using tunnels which are unaffected by the failure. Once the forwarding state for normal shortest path routing has converged, the remote routers return the traffic to shortest path forwarding. MRT-FRR is relevant for Nearside Tunneling for the following reason. The process of tunneling traffic to the PLRs and waiting a sufficient amount of time for IGP forwarding state convergence with Nearside Tunneling means that traffic will generally be relying on the local repair at the PLR for longer than it would in the absence of Nearside Tunneling. Since MRT-FRR provides 100% coverage for single link and node failure, it may be an attractive option to provide the local repair paths when Nearside Tunneling is deployed.

MRT-FRR is also relevant for the Farside Tunneling micro-loop prevention technique. In Farside Tunneling, remote routers tunnel traffic affected by a failure to a node downstream of the failure with respect to traffic destination. This node can be viewed as being on the farside of the failure with respect to the node initiating the tunnel. Note that the discussion of Farside Tunneling in [RFC5715] focuses on the case where the farside node is immediately adjacent to a failed link or node. However, the farside node may be any node downstream of the failure with respect to traffic destination, including the destination itself. The tunneling mechanism used to reach the farside node must be unaffected by the failure. The alternative forwarding paths created by MRT-FRR have the potential to be used to forward traffic from the remote routers upstream of the failure all the way to the destination. In the event of failure, either the MRT-Red or MRT-Blue path from the remote upstream router to the destination is guaranteed to avoid a link failure or inferred node failure. The MRT forwarding paths are also guaranteed to not be subject to micro-loops because they are locked to the topology before the failure.

We note that the computations in [I-D.ietf-rtgwg-mrt-frr-algorithm] address the case of a PLR adjacent to a failure determining which choice of MRT-Red or MRT-Blue will avoid a failed link or node. More computation may be required for an arbitrary remote upstream router to determine whether to choose MRT-Red or MRT-Blue for a given destination and failure.

12.2. MRT Recalculation for the Default MRT Profile

This section describes how the MRT recalculation SHOULD be performed for the Default MRT Profile. This is intended to support FRR applications. Other approaches are possible, but they are not specified in this document.

When a failure event happens, traffic is put by the PLRs onto the MRT topologies. After that, each router recomputes its shortest path tree (SPT) and moves traffic over to that. Only after all the PLRs have switched to using their SPTs and traffic has drained from the MRT topologies should each router install the recomputed MRTs into the FIBs.

At each router, therefore, the sequence is as follows:

1. Receive failure notification
2. Recompute SPT.
3. Install the new SPT in the FIB.
4. If the network was stable before the failure occurred, wait a configured (or advertised) period for all routers to be using their SPTs and traffic to drain from the MRTs.
5. Recompute MRTs.
6. Install new MRTs in the FIB.

While the recomputed MRTs are not installed in the FIB, protection coverage is lowered. Therefore, it is important to recalculate the MRTs and install them quickly.

New protocol extensions for advertising the time needed to recompute shortest path routes and install them in the FIB will be defined elsewhere.

13. Implementation Status

[RFC Editor: please remove this section prior to publication.]

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in [RFC6982]. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to [RFC6982], "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

Juniper Networks Implementation

- o Organization responsible for the implementation: Juniper Networks
- o Implementation name: MRT-FRR
- o Implementation description: MRT-FRR using OSPF as the IGP has been implemented and verified.
- o The implementation's level of maturity: prototype
- o Protocol coverage: This implementation of the MRT-FRR includes Island identification, GADAG root selection, MRT Lowpoint algorithm, augmentation of GADAG with additional links, and calculation of MRT transit next-hops alternate next-hops based on draft "draft-ietf-rtgwg-mrt-frr-algorithm-00". This implementation also includes the M-bit in OSPF based on "draft-atlas-ospf-mrt-01" as well as LDP MRT Capability based on "draft-atlas-mpls-ldp-mrt-00".
- o Licensing: proprietary

- o Implementation experience: Implementation was useful for verifying functionality and lack of gaps. It has also been useful for improving aspects of the algorithm.
- o Contact information: akatlas@juniper.net, shraddha@juniper.net, kishoret@juniper.net

Huawei Technology Implementation

- o Organization responsible for the implementation: Huawei Technology Co., Ltd.
- o Implementation name: MRT-FRR and IS-IS extensions for MRT.
- o Implementation description: The MRT-FRR using IS-IS extensions for MRT and LDP multi-topology have been implemented and verified.
- o The implementation's level of maturity: prototype
- o Protocol coverage: This implementation of the MRT algorithm includes Island identification, GADAG root selection, MRT Lowpoint algorithm, augmentation of GADAG with additional links, and calculation of MRT transit next-hops alternate next-hops based on draft "draft-enyedi-rtgwg-mrt-frr-algorithm-03". This implementation also includes IS-IS extension for MRT based on "draft-li-mrt-00".
- o Licensing: proprietary
- o Implementation experience: It is important produce a second implementation to verify the algorithm is implemented correctly without looping. It is important to verify the IS-IS extensions work for MRT-FRR.
- o Contact information: lizhenbin@huawei.com, eric.wu@huawei.com

14. Operational Considerations

The following aspects of MRT-FRR are useful to consider when deploying the technology in different operational environments and network topologies.

14.1. Verifying Forwarding on MRT Paths

The forwarding paths created by MRT-FRR are not used by normal (non-FRR) traffic. They are only used to carry FRR traffic for a short period of time after a failure has been detected. It is RECOMMENDED that an operator proactively monitor the MRT forwarding paths in

order to be certain that the paths will be able to carry FRR traffic when needed. Therefore, an implementation SHOULD provide an operator with the ability to test MRT paths with Operations, Administration, and Maintenance (OAM) traffic. For example, when MRT paths are realized using LDP labels distributed for topology-scoped FECs, an implementation can use the MPLS ping and traceroute as defined in [RFC4379] and extended in [RFC7307] for topology-scoped FECs.

14.2. Traffic Capacity on Backup Paths

During a fast-reroute event initiated by a PLR in response to a network failure, the flow of traffic in the network will generally not be identical to the flow of traffic after the IGP forwarding state has converged, taking the failure into account. Therefore, even if a network has been engineered to have enough capacity on the appropriate links to carry all traffic after the IGP has converged after the failure, the network may still not have enough capacity on the appropriate links to carry the flow of traffic during a fast-reroute event. This can result in more traffic loss during the fast-reroute event than might otherwise be expected.

Note that there are two somewhat distinct aspects to this phenomenon. The first is that the path from the PLR to the destination during the fast-reroute event may be different from the path after the IGP converges. In this case, any traffic for the destination that reaches the PLR during the fast-reroute event will follow a different path from the PLR to the destination than will be followed after IGP convergence.

The second aspect is that the amount of traffic arriving at the PLR for affected destinations during the fast-reroute event may be larger than the amount of traffic arriving at the PLR for affected destinations after IGP convergence. Immediately after a failure, any non-PLR routers that were sending traffic to the PLR before the failure will continue sending traffic to the PLR, and that traffic will be carried over backup paths from the PLR to the destinations. After IGP convergence, upstream non-PLR routers may direct some traffic away from the PLR.

In order to reduce or eliminate the potential for transient traffic loss due to inadequate capacity during fast-reroute events, an operator can model the amount of traffic taking different paths during a fast-reroute event. If it is determined that there is not enough capacity to support a given fast-reroute event, the operator can address the issue either by augmenting capacity on certain links or modifying the backup paths themselves.

The MRT Lowpoint algorithm produces a pair of diverse paths to each destination. These paths are generated by following the directed links on a common GADAG. The decision process for constructing the GADAG in the MRT Lowpoint algorithm takes into account individual IGP link metrics. At any given node, links are explored in order from lowest IGP metric to highest IGP metric. Additionally, the process for constructing the MRT-Red and Blue trees uses SPF traversals of the GADAG. Therefore, the IGP link metric values affect the computed backup paths. However, adjusting the IGP link metrics is not a generally applicable tool for modifying the MRT backup paths. Achieving a desired set of MRT backup paths by adjusting IGP metrics while at the same time maintaining the desired flow of traffic along the shortest paths is not possible in general.

MRT-FRR allows an operator to exclude a link from the MRT Island, and thus the GADAG, by advertising it as MRT-Ineligible. Such a link will not be used on the MRT forwarding path for any destination. Advertising links as MRT-Ineligible is the main tool provided by MRT-FRR for keeping backup traffic off of lower bandwidth links during fast-reroute events.

Note that all of the backup paths produced by the MRT Lowpoint algorithm are closely tied to the common GADAG computed as part of that algorithm. Therefore, it is generally not possible to modify a subset of paths without affecting other paths. This precludes more fine-grained modification of individual backup paths when using only paths computed by the MRT Lowpoint algorithm.

However, it may be desirable to allow an operator to use MRT-FRR alternates together with alternates provided by other FRR technologies. A policy-based alternate selection process can allow an operator to select the best alternate from those provided by MRT and other FRR technologies. As a concrete example, it may be desirable to implement a policy where a downstream LFA (if it exists for a given failure mode and destination) is preferred over a given MRT alternate. This combination gives the operator the ability to affect where traffic flows during a fast-reroute event, while still producing backup paths that use no additional labels for LDP traffic and will not loop under multiple failures. This and other choices of alternate selection policy can be evaluated in the context of their effect on fast-reroute traffic flow and available capacity, as well as other deployment considerations.

Note that future documents may define MRT profiles in addition to the default profile defined here. Different MRT profiles will generally produce alternate paths with different properties. An implementation may allow an operator to use different MRT profiles instead of or in addition to the default profile.

14.3. MRT IP Tunnel Loopback Address Management

As described in Section 6.1.2, if an implementation uses IP tunneling as the mechanism to realize MRT forwarding paths, each node must advertise an MRT-Red and an MRT-Blue loopback address. These IP addresses must be unique within the routing domain to the extent that they do not overlap with each other or with any other routing table entries. It is expected that operators will use existing tools and processes for managing infrastructure IP addresses to manage these additional MRT-related loopback addresses.

14.4. MRT-FRR in a Network with Degraded Connectivity

Ideally, routers in a service provider network using MRT-FRR will be initially deployed in a 2-connected topology, allowing MRT-FRR to find completely diverse paths to all destinations. However, a network can differ from an ideal 2-connected topology for many possible reasons, including network failures and planned maintenance events.

MRT-FRR is designed to continue to function properly when network connectivity is degraded. When a network contains cut-vertices or cut-links dividing the network into different 2-connected blocks, MRT-FRR will continue to provide completely diverse paths for destinations within the same block as the PLR. For a destination in a different block from the PLR, the redundant paths created by MRT-FRR will be link and node diverse within each block, and the paths will only share links and nodes that are cut-links or cut-vertices in the topology.

If a network becomes partitioned with one set of routers having no connectivity to another set of routers, MRT-FRR will function independently in each set of connected routers, providing redundant paths to destinations in same set of connected routers as a given PLR.

14.5. Partial Deployment of MRT-FRR in a Network

A network operator may choose to deploy MRT-FRR only on a subset of routers in an IGP area. MRT-FRR is designed to accommodate this partial deployment scenario. Only routers that advertise support for a given MRT profile will be included in a given MRT Island. For a PLR within the MRT Island, MRT-FRR will create redundant forwarding paths to all destinations with the MRT Island using maximally redundant trees all the way to those destinations. For destinations outside of the MRT Island, MRT-FRR creates paths to the destination which use forwarding state created by MRT-FRR within the MRT Island and shortest path forwarding state outside of the MRT Island. The

paths created by MRT-FRR to non-Island destinations are guaranteed to be diverse within the MRT Island (if topologically possible).

However, the part of the paths outside of the MRT Island may not be diverse.

15. Acknowledgements

The authors would like to thank Mike Shand for his valuable review and contributions.

The authors would like to thank Joel Halpern, Hannes Gredler, Ted Qian, Kishore Tiruveedhula, Shraddha Hegde, Santosh Esale, Nitin Bahadur, Harish Sitaraman, Raveendra Torvi, Anil Kumar SN, Bruno Decraene, Eric Wu, Janos Farkas, Rob Shakir, Stewart Bryant, and Alvaro Retana for their suggestions and review.

16. IANA Considerations

IANA is requested to create a registry entitled "MRT Profile Identifier Registry". The range is 0 to 255. The Default MRT Profile defined in this document has value 0. Values 1-200 are allocated by Standards Action. Values 201-220 are for Experimental Use. Values 221-254 are for Private Use. Value 255 is reserved for future registry extension. (The allocation and use policies are described in [RFC5226].)

The initial registry is shown below.

| Value | Description | Reference |
|---------|--|--------------|
| ----- | ----- | ----- |
| 0 | Default MRT Profile | [This draft] |
| 1-200 | Unassigned | |
| 201-220 | Experimental Use | |
| 221-254 | Private Use | |
| 255 | Reserved (for future registry extension) | |

The MRT Profile Identifier Registry is a new registry in the IANA Matrix. Following existing conventions, <http://www.iana.org/protocols> should display a new header entitled "Maximally Redundant Tree (MRT) Parameters". Under that header, there should be an entry for "MRT Profile Identifier Registry" with a link to the registry itself at <http://www.iana.org/assignments/mrt-parameters/mrt-parameters.xhtml#mrt-profile-registry>.

17. Security Considerations

In general, MRT forwarding paths do not follow shortest paths. The transit forwarding state corresponding to the MRT paths is created during normal operations (before a failure occurs). Therefore, a malicious packet with an appropriate header injected into the network from a compromised location would be forwarded to a destination along a non-shortest path. When this technology is deployed, a network security design should not rely on assumptions about potentially malicious traffic only following shortest paths.

It should be noted that the creation of non-shortest forwarding paths is not unique to MRT.

MRT-FRR requires that routers advertise information used in the formation of MRT backup paths. While this document does not specify the protocol extensions used to advertise this information, we discuss security considerations related to the information itself. Injecting false MRT-related information could be used to direct some MRT backup paths over compromised transmission links. Combined with the ability to generate network failures, this could be used to send traffic over compromised transmission links during a fast-reroute event. In order to prevent this potential exploit, a receiving router needs to be able to authenticate MRT-related information that claims to have been advertised by another router.

18. Contributors

Robert Kebler
Juniper Networks
10 Technology Park Drive
Westford, MA 01886
USA
Email: rkebler@juniper.net

Andras Csaszar
Ericsson
Konyves Kalman krt 11
Budapest 1097
Hungary
Email: Andras.Csaszar@ericsson.com

Jeff Tantsura
Ericsson
300 Holger Way
San Jose, CA 95134
USA
Email: jeff.tantsura@ericsson.com

Russ White
VCE
Email: russw@riw.us

19. References

19.1. Normative References

- [I-D.ietf-rtgwg-mrt-frr-algorithm]
Envedi, G., Csaszar, A., Atlas, A., Bowers, C., and A. Gopalan, "Algorithms for computing Maximally Redundant Trees for IP/LDP Fast- Reroute", draft-ietf-rtgwg-mrt-frr-algorithm-06 (work in progress), October 2015.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, DOI 10.17487/RFC5226, May 2008, <<http://www.rfc-editor.org/info/rfc5226>>.

- [RFC7307] Zhao, Q., Raza, K., Zhou, C., Fang, L., Li, L., and D. King, "LDP Extensions for Multi-Topology", RFC 7307, DOI 10.17487/RFC7307, July 2014, <<http://www.rfc-editor.org/info/rfc7307>>.

19.2. Informative References

- [EnyediThesis] Enyedi, G., "Novel Algorithms for IP Fast Reroute", Department of Telecommunications and Media Informatics, Budapest University of Technology and Economics Ph.D. Thesis, February 2011, <http://timon.tmit.bme.hu/theses/thesis_book.pdf>.
- [I-D.atlas-rtgwg-mrt-mc-arch] Atlas, A., Kebler, R., Wijnands, I., Csaszar, A., and G. Envedi, "An Architecture for Multicast Protection Using Maximally Redundant Trees", draft-atlas-rtgwg-mrt-mc-arch-02 (work in progress), July 2013.
- [RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, DOI 10.17487/RFC2328, April 1998, <<http://www.rfc-editor.org/info/rfc2328>>.
- [RFC4379] Kompella, K. and G. Swallow, "Detecting Multi-Protocol Label Switched (MPLS) Data Plane Failures", RFC 4379, DOI 10.17487/RFC4379, February 2006, <<http://www.rfc-editor.org/info/rfc4379>>.
- [RFC5286] Atlas, A., Ed. and A. Zinin, Ed., "Basic Specification for IP Fast Reroute: Loop-Free Alternates", RFC 5286, DOI 10.17487/RFC5286, September 2008, <<http://www.rfc-editor.org/info/rfc5286>>.
- [RFC5331] Aggarwal, R., Rekhter, Y., and E. Rosen, "MPLS Upstream Label Assignment and Context-Specific Label Space", RFC 5331, DOI 10.17487/RFC5331, August 2008, <<http://www.rfc-editor.org/info/rfc5331>>.
- [RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, DOI 10.17487/RFC5340, July 2008, <<http://www.rfc-editor.org/info/rfc5340>>.
- [RFC5443] Jork, M., Atlas, A., and L. Fang, "LDP IGP Synchronization", RFC 5443, DOI 10.17487/RFC5443, March 2009, <<http://www.rfc-editor.org/info/rfc5443>>.

- [RFC5714] Shand, M. and S. Bryant, "IP Fast Reroute Framework", RFC 5714, DOI 10.17487/RFC5714, January 2010, <<http://www.rfc-editor.org/info/rfc5714>>.
- [RFC5715] Shand, M. and S. Bryant, "A Framework for Loop-Free Convergence", RFC 5715, DOI 10.17487/RFC5715, January 2010, <<http://www.rfc-editor.org/info/rfc5715>>.
- [RFC6976] Shand, M., Bryant, S., Previdi, S., Filsfils, C., Francois, P., and O. Bonaventure, "Framework for Loop-Free Convergence Using the Ordered Forwarding Information Base (oFIB) Approach", RFC 6976, DOI 10.17487/RFC6976, July 2013, <<http://www.rfc-editor.org/info/rfc6976>>.
- [RFC6981] Bryant, S., Previdi, S., and M. Shand, "A Framework for IP and MPLS Fast Reroute Using Not-Via Addresses", RFC 6981, DOI 10.17487/RFC6981, August 2013, <<http://www.rfc-editor.org/info/rfc6981>>.
- [RFC6982] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", RFC 6982, DOI 10.17487/RFC6982, July 2013, <<http://www.rfc-editor.org/info/rfc6982>>.
- [RFC6987] Retana, A., Nguyen, L., Zinin, A., White, R., and D. McPherson, "OSPF Stub Router Advertisement", RFC 6987, DOI 10.17487/RFC6987, September 2013, <<http://www.rfc-editor.org/info/rfc6987>>.
- [RFC7490] Bryant, S., Filsfils, C., Previdi, S., Shand, M., and N. So, "Remote Loop-Free Alternate (LFA) Fast Reroute (FRR)", RFC 7490, DOI 10.17487/RFC7490, April 2015, <<http://www.rfc-editor.org/info/rfc7490>>.

Appendix A. Inter-level Forwarding Behavior for IS-IS

In the description below, we use the terms "Level-1-only interface", "Level-2-only interface", and "Level-1-and-Level-2 interface" to mean in interface which has formed only a Level-1 adjacency, only a Level-2 adjacency, or both Level-1 and Level-2 adjacencies. Note that IS-IS also defines the concept of areas. A router is configured with an IS-IS area identifier, and a given router may be configured with multiple IS-IS area identifiers. For an IS-IS Level-1 adjacency to form between two routers, at least one IS-IS area identifier must match. IS-IS Level-2 adjacencies do not require any area identifiers to match. The behavior described below does not explicitly refer to IS-IS area identifiers. However, IS-IS area identifiers will

indirectly affect the behavior by affecting the formation of Level-1 adjacencies.

First consider a packet destined to Z on MRT-Red or MRT-Blue received on a Level-1-only interface. If the best shortest path route to Z was learned from a Level-1 advertisement, then the packet should continue to be forwarded along MRT-Red or MRT-Blue. If instead the best route was learned from a Level-2 advertisement, then the packet should be removed from MRT-Red or MRT-Blue and forwarded on the shortest-path default forwarding topology.

Now consider a packet destined to Z on MRT-Red or MRT-Blue received on a Level-2-only interface. If the best route to Z was learned from a Level-2 advertisement, then the packet should continue to be forwarded along MRT-Red or MRT-Blue. If instead the best route was learned from a Level-1 advertisement, then the packet should be removed from MRT-Red or MRT-Blue and forwarded on the shortest-path default forwarding topology.

Finally, consider a packet destined to Z on MRT-Red or MRT-Blue received on a Level-1-and-Level-2 interface. This packet should continue to be forwarded along MRT-Red or MRT-Blue, regardless of which level the route was learned from.

An implementation may simplify the decision-making process above by using the interface of the next-hop for the route to Z to determine the level that the best route to Z was learned from. If the next-hop points out a Level-1-only interface, then the route was learned from a Level-1 advertisement. If the next-hop points out a Level-2-only interface, then the route was learned from a Level-2 advertisement. A next-hop that points out a Level-1-and-Level-2 interface does not provide enough information to determine the source of the best route. With this simplification, an implementation would need to continue forwarding along MRT-Red or MRT-Blue when the next-hop points out a Level-1-and-Level-2 interface. Therefore, a packet on MRT-Red or MRT-Blue going from Level-1 to Level-2 (or vice versa) that traverses a Level-1-and-Level-2 interface in the process will remain on MRT-Red or MRT-Blue. This simplification may not always produce the optimal forwarding behavior, but it does not introduce interoperability problems. The packet will stay on an MRT backup path longer than necessary, but it will still reach its destination.

Appendix B. General Issues with Area Abstraction

When a multi-homed prefix is connected in two different areas, it may be impractical to protect them without adding the complexity of explicit tunneling. This is also a problem for LFA and Remote-LFA.

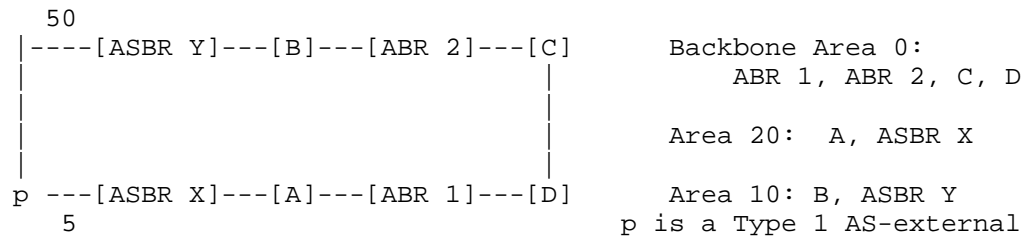


Figure 4: AS external prefixes in different areas

Consider the network in Figure 4 and assume there is a richer connective topology that isn't shown, where the same prefix is announced by ASBR X and ASBR Y which are in different non-backbone areas. If the link from A to ASBR X fails, then an MRT alternate could forward the packet to ABR 1 and ABR 1 could forward it to D, but then D would find the shortest route is back via ABR 1 to Area 20. This problem occurs because the routers, including the ABR, in one area are not yet aware of the failure in a different area.

The only way to get it from A to ASBR Y is to explicitly tunnel it to ASBR Y. If the traffic is unlabeled or the appropriate MPLS labels are known, then explicit tunneling MAY be used as long as the shortest-path of the tunnel avoids the failure point. In that case, A must determine that it should use an explicit tunnel instead of an MRT alternate.

Authors' Addresses

Alia Atlas
Juniper Networks
10 Technology Park Drive
Westford, MA 01886
USA

Email: akatlas@juniper.net

Chris Bowers
Juniper Networks
1194 N. Mathilda Ave.
Sunnyvale, CA 94089
USA

Email: cbowers@juniper.net

Gabor Sandor Enyedi
Ericsson
Konyves Kalman krt 11.
Budapest 1097
Hungary

Email: Gabor.Sandor.Enyedi@ericsson.com

Routing Area Working Group
Internet-Draft
Intended status: Informational
Expires: April 27, 2015

A. Retana
Cisco Systems, Inc.
R. White
Ericsson
M. Paul
Deutsche Telekom AG
October 24, 2014

A Framework and Requirements for Energy Aware Control Planes
draft-retana-rtgwg-eacp-03

Abstract

There has been, for several years, a rising concern over the energy usage of large scale networks. This concern is strongly focused on campus, data center, and other highly concentrated deployments of network infrastructure. Given the steadily increasing demand for higher network speeds, always-on service models, and ubiquitous network coverage, it is also of growing importance for telecommunication networks both local and wide area in scope. One of the issues in moving forward to reduce energy usage is to ensure that the network can still meet the performance specifications required to support the applications running over it.

This document provides an overview of the various areas of concern in the interaction between network performance and efforts at energy aware control planes, as a guide for those working on modifying current control planes or designing new control planes to improve the energy efficiency of high density, highly complex, network deployments.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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."

This Internet-Draft will expire on April 27, 2015.

Copyright Notice

Copyright (c) 2014 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 Simplified BSD License.

Table of Contents

| | |
|---|----|
| 1. Introduction | 4 |
| 2. Requirements Language | 4 |
| 3. Background | 5 |
| 3.1. Scope | 5 |
| 3.2. Business Drivers | 6 |
| 3.3. Application Drivers | 6 |
| 4. Framework | 7 |
| 4.1. Modes of Reducing Energy Usage | 7 |
| 4.1.1. Example Network | 8 |
| 4.1.2. Examples of Energy Reduction | 8 |
| 4.2. Global Verses Local Decisions | 9 |
| 5. Considerations and Requirements | 9 |
| 5.1. Energy Efficiency and Bandwidth Reduction | 9 |
| 5.1.1. An Example of Lowered Bandwidth | 10 |
| 5.1.2. Requirements | 10 |
| 5.2. Energy Efficiency and Stretch | 10 |
| 5.2.1. An Example of Stretch | 11 |
| 5.2.2. Requirements | 11 |
| 5.3. Energy Efficiency and Fast Recovery | 12 |
| 5.3.1. An Example of Impact on Fast Recovery | 12 |
| 5.3.2. Requirements | 12 |
| 5.4. Introducing Jitter Through Microsleeps | 13 |
| 5.4.1. An Example of Microsleeps to Reduce Energy Usage | 13 |
| 5.4.2. Requirements | 14 |
| 5.5. Other Operational Aspects | 14 |
| 5.5.1. An Example of Operational Impact | 14 |
| 5.5.2. Requirements | 14 |
| 6. Security Considerations | 14 |
| 7. Acknowledgements | 15 |
| 8. References | 15 |
| 8.1. Normative References | 15 |
| 8.2. Informative References | 15 |
| Appendix A. Change Log | 15 |
| A.1. Changes between the -00 and -01 versions. | 15 |
| A.2. Changes between the -01 and -02 versions. | 16 |
| A.3. Changes between the -02 and -03 versions. | 16 |
| Authors' Addresses | 16 |

1. Introduction

As energy prices continue to increase, and energy awareness becomes a watchword for most large companies, places where the network infrastructure used a good deal of power have come under increased scrutiny for savings. There is a concern, however, in saving energy at the cost of network operations --to reduce performance along with energy consumption, negatively impacting the operation of a network and the applications reliant on that network. This concern is primarily focused on the network control plane, but will necessarily apply to network performance and energy usage overall.

This document provides a background, a framework for understanding and managing the tradeoffs between modifications made to network protocols to conserve energy and network performance metrics and requirements, and a set of requirements for protocol designers to consider in proposals for new control plane protocols or modifications to existing control plane protocols. It is intended to encourage work on mechanisms that will reduce network energy usage while providing perspective on balancing energy usage against performance. The ultimate goal is to provide the tools and knowledge necessary for protocol designers to modify network protocols to best balance efficiency against performance, and to provide the background information network operators will need to intelligently deploy and use protocol modifications to network protocols.

The document is organized as follows. Section 3 provides material the reader needs to understand to appreciate the challenges inherent in balancing energy reduction with effective network performance. This section includes subsections considering the application and business requirements that are the basis of the rest of the document. Section 4 provides a framework for understanding mechanisms common to all energy management schemes proposed to date in general terms. Section 5 provides an analysis of the areas highlighted, including an explanation of how the specific area interacts with energy management, and example of the interaction, and, finally, a set of requirements protocol designers should consider when proposing either new protocols or modifications to existing protocols to reduce energy usage.

2. 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].

3. Background

The background covered here describes the underlying business and application drivers for the consideration and requirements sections below. This section also contains a small example network used throughout the remainder of this document for explaining various mechanisms and technical points.

3.1. Scope

The reader should differentiate between radio based and wireline (or rather, "plugged in"), networks. Radio based networks designed for rapid deployment for highly mobile users (often called Mobile Ad Hoc Networks, or MANETs [MANET]), and sensor networks designed for low power, processing, and memory (such as those described in [ROLL]), are not the target of this document. Readers should refer to the groups working within those areas for energy management requirements based on those specialized environment. While protocol developers for those environments may draw useful information from this document, this work is not intended to address those specialized networks specifically. Mobile cellular networks however are similarly affected by excess energy consumption as wireline networks and seek to save energy by methods as described in the following (see e.g. [3GPP]).

The reader should also differentiate between intradomain and interdomain applications. Interdomain applications require more work in policy than in technical and business considerations, and therefore fall outside the scope of this document. Intradomain control planes are (intuitively) where most energy savings will be attained, at any rate. Most high concentrations of routers, such as data centers and campus networks, are under a single administrative domain. Therefore, placing interdomain control planes outside the scope of this document does not limit its usefulness in any meaningful way.

The reader should further differentiate between the components of an energy management system, namely energy monitoring and energy control. Energy monitoring deals with the collection of information related to energy utilization and characteristics, as described in [EMAN]. Energy control relates to directly influencing the optimization and/or efficiency of devices in the network. The focus of this document is on understanding the tradeoffs between modifications made to network protocols to conserve energy and network performance metrics and requirements, not on the functions, steps or procedures required for energy monitoring.

3.2. Business Drivers

Networks are primarily built to support both broad and narrow business requirements. Broad business requirements might include general communication requirements, such as providing email service between internal and external personnel, or providing general access to the World Wide Web for research and business support. Narrow requirements would relate to specific applications, such as supporting a particular financial application in the case of a bank or other financial enterprise, or supporting customer traffic in the case of a service provider. Application requirements will be considered in greater detail in the next section.

Another class of requirements business place on networks can be called operational requirements. These include (but are not limited to), capital expense, operational expense, and the restrictions the network architecture places on the growth and operation of the business itself. These, in turn, drive requirements such as change management, total uptime (availability), and the ability of the network to be easily and quickly modified to meet new business demands, or to shed old business demands. Operational expense is the primary area this document covers in relation to business requirements, because this is where energy management most obviously overlaps with network performance.

3.3. Application Drivers

Applications drivers provide the background for each of the technical sections below. When approaching a specific application, there are only a small number of questions network and protocol designers need to fully understand to shape networks and protocols so a specific application can be supported. The first two questions revolve around bandwidth; how much bandwidth will the application consume, and is this bandwidth consumption fairly steady, or highly variable? For instance, applications such as streaming video tend to have long lasting flows with high bandwidth requirements, file transfers tend to produce shorter flows requiring high bandwidth, and HTML traffic tends to be bursty, with much lower bandwidth requirements.

The next question a protocol or network designer might ask about a specific application is it's tolerance to jitter. Real time applications, such as voice and video conferencing, have a very low toleration for jitter. File transfers and streaming video, on the other hand, can often handle large variations in packet arrival times. If packets are delayed long enough, the application may actually time out, shutting down sessions. Users will often "hang up" after a short period of time, as well, causing loss of revenue and productivity.

Delay is another crucial factor in the performance of many applications. Many server virtualization protocols, for instance, have very low tolerance for delay, having been written with a short wire local broadcast segment in mind. Applications such as stock and commodity trading, remote medical, and collaborative video editing also exhibit very little tolerance for delay.

These last two application drivers, jitter and delay, are normally the result of two underlying causes within a network's control plane: stretch and convergence. Stretch (defined more fully in the section considering stretch below) causes longer paths to be taken through the network. Each hop in the network path adds serialization into and out of a set of queues in device memory, along with the delays of various queuing mechanisms implemented on that device. Each hop in the network increases delay directly, and has the potential to increase jitter as packets pass into and out of the additional devices.

Network convergence will also show up as jitter in an application's stream; if packets are held up or looped for hundreds of milliseconds during a network convergence event, applications running over the converging topology will see this convergence time as a massive jitter event, or a short term delay in the delivery of packets.

Jitter and delay can also be introduced directly into the packet stream by reducing the throughput of individual links, or putting devices and/or links into energy reduced modes for very short periods of time (microsleeps). If a link is asleep when the first and third packets from a flow arrive at the head end of the link, and not when the second packet from that same flow arrives, each packet is going to be processed differently, and hence will have a different delay across the path.

The specific technical problems addressed in the following sections, then, are bandwidth reduction, increasing stretch, network convergence, and introducing jitter through microsleeps.

4. Framework

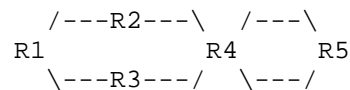
4.1. Modes of Reducing Energy Usage

Regardless of whether the control plane is centralized (such as some form of centrally computed traffic engineering or software defined network), or distributed (traditional routing protocols), there are four primary ways in which energy usage can be reduced:

- o Removing redundant links from the network topology
- o Removing redundant network equipment from the network topology
- o Reducing the amount of time equipment or links are operational
- o Reducing the link speed or processing rate of equipment

4.1.1. Example Network

To illustrate the impacts of link and device removal throughout the rest of this document, the following network is used.



This network is overly simplistic so the impact of removing various links and devices from the topology can be more clearly illustrated. More complex topologies will often exhibit these same impacts without being so obvious.

4.1.2. Examples of Energy Reduction

In the example network above, several different modes of energy reduction might be:

- o Shutting down one of the two links between R4 and R5
- o Shutting down one of the two links between R4 and R5, and shutting down any line cards (or part of the nodes themselves) associated with the removal of these links
- o Shutting down R2 or R3, since these represent alternate paths to reach the same set of destinations
- o Shutting down the link between R2 and R4, since similar connectivity is provided through R1->R3->R4
- o Shutting down all links and devices for fractions of time in a coordinated fashion
- o Shutting down individual links as traffic or the control plane permits for fractions of time (here the momentary shutdown of various links is not coordinated, but undertaken hop by hop)
- o Reducing the speed of all links and devices for fractions of time in a coordinated fashion

- o Reducing the speed of individual links as traffic or the control plane permits for fractions of time (here the momentary slowdown of various links is not coordinated, but undertaken hop by hop)

4.2. Global Verses Local Decisions

Independent of whether the control plane is centralized or distributed, the scope considered when making a decision about energy efficiency may affect the result and effectiveness of the system. There are clearly two extreme options when looking at the scope of the information used to make decisions. The first extreme is that of every device in the network considering only local conditions, and determining the proper local state from that information. An example of this mode of operation might be a local link where the devices on either side of that link measure the link utilization, and independently decide to automatically shut the link down when utilization reaches a specific threshold. An example of the other end of the spectrum might be a network control plane in which all the nodes involved agree before taking a specific action; in the case of two parallel links, the devices on each end not only would have similar configured policies, but would coordinate if one of the links was to be turned off. It is outside the scope of this document to determine which of these two options may be optimal or "best."

There are some considerations and tradeoffs which need to be outlined in considering the global versus local decisions in relation to energy efficiency. System designers should take note of the difficulties with preventing pathological conditions when purely localized decisions are made. For instance, in the example network, assume R1 determines to put the R1->R2 link into an energy saving mode, while R4 determines to put the R4->R3 link into an energy saving mode. In this case, no path will remain available through the network. It is also possible for the opposite to occur, that is for no links or devices to be placed into a reduced energy state because R1 and R4 don't agree through the control plane which links and devices should be removed from the topology.

Protocol designers should consider these tradeoffs in proposals for energy aware control planes.

5. Considerations and Requirements

5.1. Energy Efficiency and Bandwidth Reduction

Bandwidth is an important consideration in high density networks; most data centers are designed to provide a specific amount of bandwidth into and out of each server and to facilitate virtual

server movement among physical devices. In campus and core networks bandwidth is finely coupled with quality of service guarantees for applications and services. It should be obvious that removing links or devices from a network topology will adversely affect the amount of available bandwidth, which could, in turn, cause well thought out quality of service mechanisms to degrade or fail.

What might not be so obvious is the relationship between available bandwidth and jitter, or other network quality of service measures. If higher speed links are removed from the topology in order to continue using lower speed (and therefore presumably lower power) links, then serialization delays will have a larger impact on traffic flow. Longer serialization delays can cause input queues to back up, which impacts not only delay but jitter, and possibly even traffic delivery.

5.1.1. An Example of Lowered Bandwidth

In the network illustrated above, one of the two links between R4 and R5 could be an obvious candidate for removal from the network. Especially if the network load can easily be transferred to the remaining link without failure, and without serious consequences for delay or jitter in the network, there is a strong case to be made for doing so --particularly if the accompanying line cards could also be shut down to add to the energy savings.

5.1.2. Requirements

Modifications to control plane protocols to achieve network energy efficiency SHOULD provide the ability to set the minimal bandwidth, jitter, and delay through the network, and not shut down links or devices that would violate those minimal requirements.

5.2. Energy Efficiency and Stretch

In any given network, there is a shortest path between any source and any destination. Network protocols discover these paths from the destination's perspective --routing draws traffic along a path, rather than driving along a path. Along with the shortest path, there are a number of paths that can also carry traffic from a given source to a given destination without the packets passing along the same logical link, or through the same logical device, more than once. These are considered loop-free alternate [RFC5714] paths.

The primary difference between the shortest path and the loop-free alternate paths is the total cost of using the path. In simple terms, this difference can be calculated as the number of links and devices a packet must pass through when being carried from the source

to the destination --the hop count. While most networks use much more sophisticated metrics based on bandwidth, congestion, and other factors, the hop count will stand in as the only metric used throughout this document.

When the control plane causes traffic to pass from the source to the destination along a path which is longer than the shortest path, the network is said to have stretch (see [Krioukov] for a more in depth explanation of network stretch). To measure stretch, simply subtract the metric of the shortest path from the metric of the longer path. For example, in hop count terms, if the best path is three hops, and the current path is four hops, the network exhibits a stretch of 1.

5.2.1. An Example of Stretch

In the network illustrated above, if a modification is made to the control plane in order to remove the link between R1 and R4 in order to save energy, all the destinations shown in the diagram remain reachable. However, from the perspective of R1, the best path available to reach R2 has increased in length by two hops. The original path is R1->R2, the new path is R1->R3->R4->R2. This represents a stretch of 2.

Along with this increased stretch will most likely also come increased delay through the network; each hop in the network represents a measurable amount of delay. This increased stretch might also represent an increased amount of jitter, as there are more queues and more serialization events in the path of each packet carried. There will also be the modifications in jitter as the network switches between the optimal performance configuration and an energy efficient configuration.

5.2.2. Requirements

Designers who propose modifications to control plane protocols to achieve network energy efficiency SHOULD analyze the impact of their mechanisms on the stretch in typical network topologies, and SHOULD include such analysis when explaining the applicability of their proposals. This analysis may include an examination of the absolute, or maximum, stretch caused by the modifications to the control plane as well as analysis at the 95th percentile, the average stretch increase in a given set of topologies, and/or the mean increase in stretch.

Mechanisms that could impact the stretch of a network SHOULD provide the ability for the network administrator to limit the amount of stretch the network will encounter when moving into a more energy efficient mode.

5.3. Energy Efficiency and Fast Recovery

A final area where modifications to the control plane for energy efficiency is fast convergence or fast recovery. Many networks are now designed to recover from failures quickly enough to only cause a handful of traffic to be lost; recovery on the order of half a second is not an uncommon goal. It should be obvious that removing redundant links and devices from the network to reduce energy consumption could adversely affect these goals.

5.3.1. An Example of Impact on Fast Recovery

In the network shown, assume R2 and its associated links are removed from the topology in order to save energy. Rather than this second path being available for immediate recovery on the failure of the R1->R3 link, some process must be followed to bring R2 and its associated links back up, reinject them into the topology, and finally begin routing traffic across this path.

In many situations, only links and devices which are a "third point of failure" may be acceptable as removal candidates in order to conserve energy.

5.3.2. Requirements

Modifications to the control plane in order to remove links or nodes to conserve energy SHOULD entail the ability to choose the level of redundancy available after the network topology has been trimmed. For instance, it might be acceptable in some situations to move to single points of failure throughout the network, or in specific sections of the network, for certain periods of time. In other situations, it may only be acceptable to reduce the network to a double point of failure, and never to a single point of failure.

The complete removal of nodes or links from the network topology has several impacts on the control plane which must be considered. In these cases, the control plane must:

- o Modify the network topology so removed links or devices are not used to forward traffic
- o Remember that such links exist, possibly including the neighbors and destinations reachable through those links or devices

5.4. Introducing Jitter Through Microsleeps

One proposed mechanism to reduce energy usage in a network is to sleep links or devices for very short periods of time, called microsleeps. For instance, if a particular link is only used at 50% of the actual available bandwidth, it should be possible to place the link in some lower power state for 50% of the time, thus reducing energy usage by something percentage.

Such schemes introduce delay and jitter into the network path directly; if a packet arrives while the link to the next hop, or the next hop itself, is in a reduced energy state, the packet must wait until the link or next hop device enter a normal operational mode before it can be forwarded. Most of the time the proposed sleep states are so small as to be presumably inconsequential on overall packet delay, but multiple packets crossing a series of links, each encountering different links in different states, could take very different amounts of time to pass along the path.

One possible way to resolve this somewhat random accrual of delays on a per packet basis is to coordinate these sleep states such that packets accepted at the entry of the network are consistently passed through the network when all links and devices are in a normal operating mode, and simply delaying all packets at the entry point into the network while the devices in the network are in some energy reduced state. This solution still introduces some amount of jitter; some packets will be delayed by the sleep state at the edge of the network, while others will not. This solution also requires coordinated timers at the speed of forwarding itself to effectively control the sleep and wake cycles of the network.

5.4.1. An Example of Microsleeps to Reduce Energy Usage

In the example network, assume the bandwidth utilization along the path R1->R2->R4->R5 is 50% of the actual available bandwidth along this path. It is possible to consider a scheme where R1->R2, R2->R4, and R4->R5 are all put into some energy reduced operational mode 50% of the time, since packets are only available to send 50% of the time. A packet entering at R1 may encounter a short delay at R1->R2, at R2->R4, and at R4->R5, or it might not. Even if these delays are very small, say 200ms at each hop, the accumulated delay through the network due to sleep states may be 0ms (all links and devices awake) or 600ms (all links and devices asleep) as the packet passes through the network.

As network paths lengthen to more realistic path lengths in real deployments, the jitter introduced varies more widely, which could cause problems for the operation of a number of applications.

5.4.2. Requirements

Protocol designers SHOULD analyze the impact of accumulated jitter when proposing mechanisms that rely on microsleeves in either equipment or links. This analysis SHOULD include both worst case and best case scenarios, as well as an analysis of how coordinated clocks are to be handled in the case of coordinated sleep states.

5.5. Other Operational Aspects

Modification of the network topology in order to save energy needs to consider the operational needs of the network as well as application requirements. Change management, operational downtime, and business usage of the network need to be considered when determining which links and nodes should be placed into a low energy state. Energy provisions have to be assigned and changed for nodes and links, optimally according to network usage profiles over the time of day.

Control plane protocol operation, in terms of operational efficiency on the wire, also needs to be considered when modifying protocol parameters. Any changes that negatively impact the operation of the protocol, in terms of the amount of traffic, the size of routing information transmitted over the network, and interaction with network management operations need to be carefully analyzed for scaling and operational implications.

5.5.1. An Example of Operational Impact

Time of day is an important consideration in business operations. During normal operational hours, the network needs to be fully available, including all available redundancy and bandwidth. During holidays, night hours, and other times when a campus might not be used, or when there are lower traffic and resiliency demands on the network, network elements can be removed to reduce energy usage.

5.5.2. Requirements

Protocol designers SHOULD analyze operational requirements, such as time of day and network traffic load considerations, and explain how proposed protocols or modifications to protocols will interact with these types of requirements. Protocol designers SHOULD analyze increases in network traffic and the operational efficiency impact of proposed changes or protocols.

6. Security Considerations

None.

7. Acknowledgements

The authors of this document would like to acknowledge the suggestions and ideas provided by Sujata Banerjee, Puneet Sharma and Dirk Von Hugo.

8. References

8.1. Normative References

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

8.2. Informative References

- [3GPP] 3GPP, "3GPP TR 25-927 Solutions for energy saving within UTRA Node B", 2011,
<<http://3gpp.org/ftp/Specs/html-info/25927.htm>>.
- [EMAN] IETF, "Energy Management Working Group Charter", 2012,
<<http://datatracker.ietf.org/wg/eman/charter/>>.
- [Krioukov] Krioukov, D., "On Compact Routing for the Internet", 2007,
<http://www.caida.org/publications/papers/2007/compact_routing/>.
- [MANET] IETF, "Mobile Ad Hoc Networks Charter", 2012,
<<http://datatracker.ietf.org/wg/manet/charter/>>.
- [RFC5714] Shand, M. and S. Bryant, "IP Fast Reroute Framework", RFC 5714, January 2010.
- [ROLL] IETF, "Routing Over Low power and Lossy networks Charter", 2012,
<<http://datatracker.ietf.org/wg/roll/charter/>>.

Appendix A. Change Log

A.1. Changes between the -00 and -01 versions.

- o Updated authors' contact information.
- o Modified some of the rfc2119 keywords.

A.2. Changes between the -01 and -02 versions.

- o Updated authors' contact information.

A.3. Changes between the -02 and -03 versions.

- o Updated authors' contact information.

Authors' Addresses

Alvaro Retana
Cisco Systems, Inc.
7025 Kit Creek Rd.
Raleigh, NC 27709
USA

Email: aretana@cisco.com

Russ White
Ericsson

Email: russw@riw.us

Manuel Paul
Deutsche Telekom AG
Winterfeldtstr. 21-27
Berlin 10781
Germany

Email: Manuel.Paul@telekom.de

RTGWG
Internet-Draft
Intended status: Informational
Expires: December 31, 2012

S. Ning
Tata Communications
D. McDysan
Verizon
E. Osborne
Cisco
L. Yong
Huawei USA
C. Villamizar
Outer Cape Cod Network
Consulting
June 29, 2012

Composite Link Framework in Multi Protocol Label Switching (MPLS)
draft-so-yong-rtgwg-cl-framework-06

Abstract

This document specifies a framework for support of composite link in MPLS networks. A composite link consists of a group of homogenous or non-homogenous links that have the same forward adjacency and can be considered as a single TE link or an IP link in routing. A composite link relies on its component links to carry the traffic over the composite link. Applicability is described for a single pair of MPLS-capable nodes, a sequence of MPLS-capable nodes, or a set of layer networks connecting MPLS-capable nodes.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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."

This Internet-Draft will expire on December 31, 2012.

Copyright Notice

Copyright (c) 2012 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 Simplified BSD License.

Table of Contents

| | |
|--|----|
| 1. Introduction | 4 |
| 1.1. Architecture Summary | 4 |
| 1.2. Conventions used in this document | 5 |
| 1.2.1. Terminology | 5 |
| 2. Composite Link Key Characteristics | 5 |
| 2.1. Flow Identification | 6 |
| 2.2. Composite Link in Control Plane | 8 |
| 2.3. Composite Link in Data Plane | 11 |
| 3. Architecture Tradeoffs | 11 |
| 3.1. Scalability Motivations | 12 |
| 3.2. Reducing Routing Information and Exchange | 12 |
| 3.3. Reducing Signaling Load | 13 |
| 3.3.1. Reducing Signaling Load using LDP | 14 |
| 3.3.2. Reducing Signaling Load using Hierarchy | 14 |
| 3.3.3. Using Both LDP and RSVP-TE Hierarchy | 14 |
| 3.4. Reducing Forwarding State | 14 |
| 3.5. Avoiding Route Oscillation | 15 |
| 4. New Challenges | 16 |
| 4.1. Control Plane Challenges | 16 |
| 4.1.1. Delay and Jitter Sensitive Routing | 17 |
| 4.1.2. Local Control of Traffic Distribution | 17 |
| 4.1.3. Path Symmetry Requirements | 17 |
| 4.1.4. Requirements for Contained LSP | 18 |
| 4.1.5. Retaining Backwards Compatibility | 19 |
| 4.2. Data Plane Challenges | 19 |
| 4.2.1. Very Large LSP | 20 |
| 4.2.2. Very Large Microflows | 20 |
| 4.2.3. Traffic Ordering Constraints | 20 |
| 4.2.4. Accounting for IP and LDP Traffic | 21 |
| 4.2.5. IP and LDP Limitations | 21 |
| 5. Existing Mechanisms | 22 |
| 5.1. Link Bundling | 22 |
| 5.2. Classic Multipath | 24 |

| | |
|---|----|
| 6. Mechanisms Proposed in Other Documents | 24 |
| 6.1. Loss and Delay Measurement | 24 |
| 6.2. Link Bundle Extensions | 25 |
| 6.3. Fat PW and Entropy Labels | 26 |
| 6.4. Multipath Extensions | 26 |
| 7. Required Protocol Extensions and Mechanisms | 27 |
| 7.1. Brief Review of Requirements | 27 |
| 7.2. Required Document Coverage | 28 |
| 7.2.1. Component Link Grouping | 28 |
| 7.2.2. Delay and Jitter Extensions | 29 |
| 7.2.3. Path Selection and Admission Control | 29 |
| 7.2.4. Dynamic Multipath Balance | 30 |
| 7.2.5. Frequency of Load Balance | 30 |
| 7.2.6. Inter-Layer Communication | 30 |
| 7.2.7. Packet Ordering Requirements | 31 |
| 7.2.8. Minimally Disruption Load Balance | 31 |
| 7.2.9. Path Symmetry | 31 |
| 7.2.10. Performance, Scalability, and Stability | 32 |
| 7.2.11. IP and LDP Traffic | 32 |
| 7.2.12. LDP Extensions | 32 |
| 7.2.13. Pseudowire Extensions | 33 |
| 7.2.14. Multi-Domain Composite Link | 33 |
| 7.3. Open Issues Regarding Requirements | 34 |
| 7.4. Framework Requirement Coverage by Protocol | 34 |
| 7.4.1. OSPF-TE and ISIS-TE Protocol Extensions | 35 |
| 7.4.2. PW Protocol Extensions | 35 |
| 7.4.3. LDP Protocol Extensions | 35 |
| 7.4.4. RSVP-TE Protocol Extensions | 35 |
| 7.4.5. RSVP-TE Path Selection Changes | 35 |
| 7.4.6. RSVP-TE Admission Control and Preemption | 35 |
| 7.4.7. Flow Identification and Traffic Balance | 35 |
| 8. Security Considerations | 35 |
| 9. Acknowledgments | 36 |
| 10. References | 36 |
| 10.1. Normative References | 36 |
| 10.2. Informative References | 37 |
| Authors' Addresses | 40 |

1. Introduction

Composite Link functional requirements are specified in [I-D.ietf-rtgwg-cl-requirement]. Composite Link use cases are described in [I-D.symmvo-rtgwg-cl-use-cases]. This document specifies a framework to meet these requirements.

Classic multipath, including Ethernet Link Aggregation has been widely used in today's MPLS networks [RFC4385][RFC4928]. Classic multipath using non-Ethernet links are often advertised using MPLS Link bundling. A link bundle [RFC4201] bundles a group of homogeneous links as a TE link to make IGP-TE information exchange and RSVP-TE signaling more scalable. A composite link allows bundling non-homogenous links together as a single logical link. The motivations for using a composite link are described in [I-D.ietf-rtgwg-cl-requirement] and [I-D.symmvo-rtgwg-cl-use-cases].

This document describes a composite link framework in the context of MPLS networks using an IGP-TE and RSVP-TE MPLS control plane with GMPLS extensions [RFC3209][RFC3630][RFC3945][RFC5305].

A composite link is a single logical link in MPLS network that contains multiple parallel component links between two MPLS LSR. Unlike a link bundle [RFC4201], the component links in a composite link can have different properties such as cost or capacity.

Specific protocol solutions are outside the scope of this document, however a framework for the extension of existing protocols is provided. Backwards compatibility is best achieved by extending existing protocols where practical rather than inventing new protocols. The focus is on examining where existing protocol mechanisms fall short with respect to [I-D.ietf-rtgwg-cl-requirement] and on extensions that will be required to accommodate functionality that is called for in [I-D.ietf-rtgwg-cl-requirement].

1.1. Architecture Summary

Networks aggregate information, both in the control plane and in the data plane, as a means to achieve scalability. A tradeoff exists between the needs of scalability and the needs to identify differing path and link characteristics and differing requirements among flows contained within further aggregated traffic flows. These tradeoffs are discussed in detail in Section 3.

Some aspects of Composite Link requirements present challenges for which multiple solutions may exist. In Section 4 various challenges and potential approaches are discussed.

A subset of the functionality called for in [I-D.ietf-rtgwg-cl-requirement] is available through MPLS Link Bundling [RFC4201]. Link bundling and other existing standards applicable to Composite Link are covered in Section 5.

The most straightforward means of supporting Composite Link requirements is to extend MPLS protocols and protocol semantics and in particular to extend link bundling. Extensions which have already been proposed in other documents which are applicable to Composite Link are discussed in Section 6.

Goals of most new protocol work within IETF is to reuse existing protocol encapsulations and mechanisms where they meet requirements and extend existing mechanisms such that additional complexity is minimized while meeting requirements and such that backwards compatibility is preserved to the extent it is practical to do so. These goals are considered in proposing a framework for further protocol extensions and mechanisms in Section 7.

1.2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

1.2.1. Terminology

Terminology defined in [I-D.ietf-rtgwg-cl-requirement] is used in this document.

The abbreviation IGP-TE is used as a shorthand indicating either OSPF-TE [RFC3630] or ISIS-TE [RFC5305].

2. Composite Link Key Characteristics

[I-D.ietf-rtgwg-cl-requirement] defines external behavior of Composite Links. The overall framework approach involves extending existing protocols in a backwards compatible manner and reusing ongoing work elsewhere in IETF where applicable, defining new protocols or semantics only where necessary. Given the requirements, and this approach of extending MPLS, Composite Link key characteristics can be described in greater detail than given requirements alone.

2.1. Flow Identification

Traffic mapping to component links is a data plane operation. Control over how the mapping is done may be directly dictated or constrained by the control plane or by the management plane. When unconstrained by the control plane or management plane, distribution of traffic is entirely a local matter. Regardless of constraints or lack of constraints, the traffic distribution is required to keep packets belonging to individual flows in sequence and meet QoS criteria specified per LSP by either signaling or management [RFC2475][RFC3260]. A key objective of the traffic distribution is to not overload any component link, and be able to perform local recovery when one of component link fails.

The network operator may have other objectives such as placing a bidirectional flow or LSP on the same component link in both direction, load balance over component links, composite link energy saving, and etc. These new requirements are described in [I-D.ietf-rtgwg-cl-requirement].

Examples of means to identify a flow may in principle include:

1. an LSP identified by an MPLS label,
2. a sub-LSP [I-D.kompella-mpls-rsvp-ecmp] identified by an MPLS label,
3. a pseudowire (PW) [RFC3985] identified by an MPLS PW label,
4. a flow or group of flows within a pseudowire (PW) [RFC6391] identified by an MPLS flow label,
5. a flow or flow group in an LSP [I-D.ietf-mpls-entropy-label] identified by an MPLS entropy label,
6. all traffic between a pair of IP hosts, identified by an IP source and destination pair,
7. a specific connection between a pair of IP hosts, identified by an IP source and destination pair, protocol, and protocol port pair,
8. a layer-2 conversation within a pseudowire (PW), where the identification is PW payload type specific, such as Ethernet MAC addresses and VLAN tags within an Ethernet PW (RFC4448).

Although in principle a layer-2 conversation within a pseudowire (PW), may be identified by PW payload type specific information, in

practice this is impractical at LSP midpoints when PW are carried. The PW ingress may provide equivalent information in a PW flow label [RFC6391]. Therefore, in practice, item #8 above is covered by [RFC6391] and may be dropped from the list.

An LSR must at least be capable of identifying flows based on MPLS labels. Most MPLS LSP do not require that traffic carried by the LSP are carried in order. MPLS-TP is a recent exception. If it is assumed that no LSP require strict packet ordering of the LSP itself (only of flows within the LSP), then the entire label stack can be used as flow identification. If some LSP may require strict packet ordering but those LSP cannot be distinguished from others, then only the top label can be used as a flow identifier. If only the top label is used (for example, as specified by [RFC4201] when the "all-ones" component described in [RFC4201] is not used), then there may not be adequate flow granularity to accomplish well balanced traffic distribution and it will not be possible to carry LSP that are larger than any individual component link.

The number of flows can be extremely large. This may be the case when the entire label stack is used and is always the case when IP addresses are used in provider networks carrying Internet traffic. Current practice for native IP load balancing at the time of writing were documented in [RFC2991], [RFC2992]. These practices as described, make use of IP addresses. The common practices were extended to include the MPLS label stack and the common practice of looking at IP addresses within the MPLS payload. These extended practices are described in [RFC4385] and [RFC4928] due to their impact on pseudowires without a PWE3 Control Word. Additional detail on current multipath practices can be found in the appendices of [I-D.symmvo-rtgwg-cl-use-cases].

Using only the top label supports too coarse a traffic balance. Using the full label stack or IP addresses as flow identification provides a sufficiently fine traffic balance, but is capable of identifying such a high number of distinct flows, that a technique of grouping flows, such as hashing on the flow identification criteria, becomes essential to reduce the stored state, and is an essential scaling technique. Other means of grouping flows may be possible.

In summary:

1. Load balancing using only the MPLS label stack provides too coarse a granularity of load balance.
2. Tracking every flow is not scalable due to the extremely large number of flows in provider networks.

3. Existing techniques, IP source and destination hash in particular, have proven in over two decades of experience to be an excellent way of identifying groups of flows.
4. If a better way to identify groups of flows is discovered, then that method can be used.
5. IP address hashing is not required, but use of this technique is strongly encouraged given the technique's long history of successful deployment.

2.2. Composite Link in Control Plane

A composite Link is advertised as a single logical interface between two connected routers, which forms forwarding adjacency (FA) between the routers. The FA is advertised as a TE-link in a link state IGP, using either OSPF-TE or ISIS-TE. The IGP-TE advertised interface parameters for the composite link can be preconfigured by the network operator or be derived from its component links. Composite link advertisement requirements are specified in [I-D.ietf-rtgwg-cl-requirement].

In IGP-TE, a composite link is advertised as a single TE link between two connected routers. This is similar to a link bundle [RFC4201]. Link bundle applies to a set of homogenous component links. Composite link allows homogenous and non-homogenous component links. Due to the similarity, and for backwards compatibility, extending link bundling is viewed as both simple and as the best approach.

In order for a route computation engine to calculate a proper path for a LSP, it is necessary for composite link to advertise the summarized available bandwidth as well as the maximum bandwidth that can be made available for single flow (or single LSP where no finer flow identification is available). If a composite link contains some non-homogeneous component links, the composite link also should advertise the summarized bandwidth and the maximum bandwidth for single flow per each homogeneous component link group.

Both LDP [RFC5036] and RSVP-TE [RFC3209] can be used to signal a LSP over a composite link. LDP cannot be extended to support traffic engineering capabilities [RFC3468].

When an LSP is signaled using RSVP-TE, the LSP MUST be placed on the component link that meets the LSP criteria indicated in the signaling message.

When an LSP is signaled using LDP, the LSP MUST be placed on the component link that meets the LSP criteria, if such a component link

is available. LDP does not support traffic engineering capabilities, imposing restrictions on LDP use of Composite Link. See Section 4.2.5 for further details.

A composite link may contain non-homogeneous component links. The route computing engine may select one group of component links for a LSP. The routing protocol MUST make this grouping available in the TE-LSDB. The route computation used in RSVP-TE MUST be extended to include only the capacity of groups within a composite link which meet LSP criteria. The signaling protocol MUST be able to indicate either the criteria, or which groups may be used. A composite link MUST place the LSP on a component link or group which meets or exceeds the LSP criteria.

Composite link capacity is aggregated capacity. LSP capacity MAY be larger than individual component link capacity. Any aggregated LSP can determine a bounds on the largest microflow that could be carried and this constraint can be handled as follows.

1. If no information is available through signaling, management plane, or configuration, the largest microflow is bound by one of the following:
 - A. the largest single LSP if most traffic is RSVP-TE signaled and further aggregated,
 - B. the largest pseudowire if most traffic is carrying pseudowire payloads that are aggregated within RSVP-TE LSP,
 - C. or the largest source and sink interface if a large amount of IP or LDP traffic is contained within the aggregate.

If a very large amount of traffic being aggregated is IP or LDP, then the largest microflow is bound by the largest component link on which IP traffic can arrive. For example, if an LSR is acting as an LER and IP and LDP traffic is arriving on 10 Gb/s edge interfaces, then no microflow larger than 10 Gb/s will be present on the RSVP-TE LSP that aggregate traffic across the core, even if the core interfaces are 100 Gb/s interfaces.

2. The prior conditions provide a bound on the largest microflow when no signaling extensions indicate a bounds. If an LSP is aggregating smaller LSP for which the largest expected microflow carried by the smaller LSP is signaled, then the largest microflow expected in the containing LSP (the aggregate) is the maximum of the largest expected microflow for any contained LSP. For example, RSVP-TE LSP may be large but aggregate traffic for which the source or sink are all 1 Gb/s or smaller interfaces

(such as in mobile applications in which cell sites backhauls are no larger than 1 Gb/s). If this information is carried in the LSP originated at the cell sites, then further aggregates across a core may make use of this information.

3. The IGP must provide the bounds on the largest microflow that a composite link can accommodate, which is the maximum capacity on a component link that can be made available by moving other traffic. This information is needed by the ingress LER for path determination.
4. A means to signal an LSP whose capacity is larger than individual component link capacity is needed [I-D.ietf-rtgwg-cl-requirement] and also signal the largest microflow expected to be contained in the LSP. If a bounds on the largest microflow is not signaled there is no means to determine if an LSP which is larger than any component link can be subdivided into flows and therefore should be accepted by admission control.

When a bidirectional LSP request is signaled over a composite link, if the request indicates that the LSP must be placed on the same component link, the routers of the composite link MUST place the LSP traffic in both directions on a same component link. This is particularly challenging for aggregated capacity which makes use of the label stack for traffic distribution. The two requirements are mutually exclusive for any one LSP. No one LSP may be both larger than any individual component link and require symmetrical paths for every flow. Both requirements can be accommodated by the same composite link for different LSP, with any one LSP requiring no more than one of these two features.

Individual component link may fail independently. Upon component link failure, a composite link MUST support a minimally disruptive local repair, preempting any LSP which can no longer be supported. Available capacity in other component links MUST be used to carry impacted traffic. The available bandwidth after failure MUST be advertised immediately to avoid looped crankback.

When a composite link is not able to transport all flows, it preempts some flows based upon local management configuration and informs the control plane on these preempted flows. The composite link MUST support soft preemption [RFC5712]. This action ensures the remaining traffic is transported properly. FR#10 requires that the traffic be restored. FR#12 requires that any change be minimally disruptive. These two requirements are interpreted to include preemption among the types of changes that must be minimally disruptive.

2.3. Composite Link in Data Plane

The data plane must first identify groups of flows. Flow identification is covered in Section 2.1. Having identified groups of flows the groups must be placed on individual component links. This second step is called traffic distribution or traffic placement. The two steps together are known as traffic balancing or load balancing.

Traffic distribution may be determined by or constrained by control plane or management plane. Traffic distribution may be changed due to component link status change, subject to constraints imposed by either the management plane or control plane. The distribution function is local to the routers in which a composite link belongs to and is not specified here.

When performing traffic placement, a composite link does not differentiate multicast traffic vs. unicast traffic.

In order to maintain scalability, existing data plane forwarding retains state associated with the top label only. The use of flow group identification is in a second step in the forwarding process. Data plane forwarding makes use of the top label to select a composite link, or a group of components within a composite link or for the case where an LSP is pinned (see [RFC4201]), a specific component link. For those LSP for which the LSP selects only the composite link or a group of components within a composite link, the load balancing makes use of the flow group identification.

The most common traffic placement techniques uses the a flow group identification as an index into a table. The table provides an indirection. The number of bits of hash is constrained to keep table size small. While this is not the best technique, it is the most common. Better techniques exist but they are outside the scope of this document and some are considered proprietary.

Requirements to limit frequency of load balancing can be adhered to by keeping track of when a flow group was last moved and imposing a minimum period before that flow group can be moved again. This is straightforward for a table approach. For other approaches it may be less straightforward but is achievable.

3. Architecture Tradeoffs

Scalability and stability are critical considerations in protocol design where protocols may be used in a large network such as today's service provider networks. Composite Link is applicable to networks

which are large enough to require that traffic be split over multiple paths. Scalability is a major consideration for networks that reach a capacity large enough to require Composite Link.

Some of the requirements of Composite Link could potentially have a negative impact on scalability. For example, Composite Link requires additional information to be carried in situations where component links differ in some significant way.

3.1. Scalability Motivations

In the interest of scalability information is aggregated in situations where information about a large amount of network capacity or a large amount of network demand provides is adequate to meet requirements. Routing information is aggregated to reduce the amount of information exchange related to routing and to simplify route computation (see Section 3.2).

In an MPLS network large routing changes can occur when a single fault occurs. For example, a single fault may impact a very large number of LSP traversing a given link. As new LSP are signaled to avoid the fault, resources are consumed elsewhere, and routing protocol announcements must flood the resource changes. If protection is in place, there is less urgency to converging quickly. If multiple faults occur that are not covered by shared risk groups (SRG), then some protection may fail, adding urgency to converging quickly even where protection was deployed.

Reducing the amount of information allows the exchange of information during a large routing change to be accomplished more quickly and simplifies route computation. Simplifying route computation improves convergence time after very significant network faults which cannot be handled by preprovisioned or precomputed protection mechanisms. Aggregating smaller LSP into larger LSP is a means to reduce path computation load and reduce RSVP-TE signaling (see Section 3.3).

Neglecting scaling issues can result in performance issues, such as slow convergence. Neglecting scaling in some cases can result in networks which perform so poorly as to become unstable.

3.2. Reducing Routing Information and Exchange

Link bundling at the very least provides a means of aggregating control plane information. Even where the all-ones component link supported by link bundling is not used, the amount of control information is reduced by the average number of component links in a bundle.

Fully deaggregating link bundle information would negate this benefit. If there is a need to deaggregate, such as to distinguish between groups of links within specified ranges of delay, then no more deaggregation than is necessary should be done.

For example, in supporting the requirement for heterogeneous component links, it makes little sense to fully deaggregate link bundles when adding support for groups of component links with common attributes within a link bundle can maintain most of the benefit of aggregation while adequately supporting the requirement to support heterogeneous component links.

Routing information exchange is also reduced by making sensible choices regarding the amount of change to link parameters that require link readvertisement. For example, if delay measurements include queuing delay, then a much more coarse granularity of delay measurement would be called for than if the delay does not include queuing and is dominated by geographic delay (speed of light delay).

3.3. Reducing Signaling Load

Aggregating traffic into very large hierarchical LSP in the core very substantially reduces the number of LSP that need to be signaled and the number of path computations any given LSR will be required to perform when a major network fault occurs.

In the extreme, applying MPLS to a very large network without hierarchy could exceed the 20 bit label space. For example, in a network with 4,000 nodes, with 2,000 on either side of a cutset, would have 4,000,000 LSP crossing the cutset. Even in a degree four cutset, an uneven distribution of LSP across the cutset, or the loss of one link would result in a need to exceed the size of the label space. Among provider networks, 4,000 access nodes is not at all large.

In less extreme cases, having each node terminate hundreds of LSP to achieve a full mesh creates a very large computational load. The time complexity of one CSPF computation is $\text{order}(N \log N)$, where L is proportional to N , and N and L are the number of nodes and number of links, respectively. If each node must perform $\text{order}(N)$ computations when a fault occurs, then the computational load increases as $\text{order}(N^2 \log N)$ as the number of nodes increases. In practice at the time of writing, this imposes a limit of a few hundred nodes in a full mesh of MPLS LSP before the computational load is sufficient to result in unacceptable convergence times.

Two solutions are applied to reduce the amount of RSVP-TE signaling. Both involve subdividing the MPLS domain into a core and a set of

regions.

3.3.1. Reducing Signaling Load using LDP

LDP can be used for edge-to-edge LSP, using RSVP-TE to carry the LDP intra-core traffic and also optionally also using RSVP-TE to carry the LDP intra-region traffic within each region. LDP does not support traffic engineering, but does support multipoint-to-point (MPTP) LSP, which require less signaling than edge-to-edge RSVP-TE point-to-point (PTP) LSP. A drawback of this approach is the inability to use RSVP-TE protection (FRR or GMPLS protection) against failure of the border LSR sitting at a core/region boundary.

3.3.2. Reducing Signaling Load using Hierarchy

When the number of nodes grows too large, the amount of RSVP-TE signaling can be reduced using the MPLS PSC hierarchy [RFC4206]. A core within the hierarchy can divide the topology into M regions of on average N/M nodes. Within a region the computational load is reduced by more than M^2 . Within the core, the computational load generally becomes quite small since M is usually a fairly small number (a few tens of regions) and each region is generally attached to the core in typically only two or three places on average.

Using hierarchy improves scaling but has two consequences. First, hierarchy effectively forces the use of platform label space. When a containing LSP is rerouted, the labels assigned to the contained LSP cannot be changed but may arrive on a different interface. Second, hierarchy results in much larger LSP. These LSP today are larger than any single component link and therefore force the use of the all-ones component in link bundles.

3.3.3. Using Both LDP and RSVP-TE Hierarchy

It is also possible to use both LDP and RSVP-TE hierarchy. MPLS networks with a very large number of nodes may benefit from the use of both LDP and RSVP-TE hierarchy. The two techniques are certainly not mutually exclusive.

3.4. Reducing Forwarding State

Both LDP and MPLS hierarchy have the benefit of reducing the amount of forwarding state. Using the example from Section 3.3, and using MPLS hierarchy, the worst case generally occurs at borders with the core.

For example, consider a network with approximately 1,000 nodes divided into 10 regions. At the edges, each node requires 1,000 LSP

to other edge nodes. The edge nodes also require 100 intra-region LSP. Within the core, if the core has only 3 attachments to each region the core LSR have less than 100 intra-core LSP. At the border cutset between the core and a given region, in this example there are 100 edge nodes with inter-region LSP crossing that cutset, destined to 900 other edge nodes. That yields forwarding state for on the order of 90,000 LSP at the border cutset. These same routers need only reroute well under 200 LSP when a multiple fault occurs, as long as only links are affected and a border LSR does not go down.

In the core, the forwarding state is greatly reduced. If inter-region LSP have different characteristics, it makes sense to make use of aggregates with different characteristics. Rather than exchange information about every inter-region LSP within the intra-core LSP it makes more sense to use multiple intra-core LSP between pairs of core nodes, each aggregating sets of inter-region LSP with common characteristics or common requirements.

3.5. Avoiding Route Oscillation

Networks can become unstable when a feedback loop exists such that moving traffic to a link causes a metric such as delay to increase, which then causes traffic to move elsewhere. For example, the original ARPANET routing used a delay based cost metric and proved prone to route oscillations [DBP].

Delay may be used as a constraint in routing for high priority traffic, where the movement of traffic cannot impact the delay. The safest way to measure delay is to make measurements based on traffic which is prioritized such that it is queued ahead of the traffic which will be affected. This is a reasonable measure of delay for high priority traffic for which constraints have been set which allow this type of traffic to consume only a fraction of link capacities with the remaining capacity available to lower priority traffic.

Any measurement of jitter (delay variation) that is used in route decision is likely to cause oscillation. Jitter that is caused by queuing effects and cannot be measured using a very high priority measurement traffic flow.

It may be possible to find links with constrained queuing delay or jitter using a theoretical maximum or a probability based bound on queuing delay or jitter at a given priority based on the types and amounts of traffic accepted and combining that theoretical limit with a measured delay at very high priority.

Instability can occur due to poor performance and interaction with protocol timers. In this way a computational scaling problem can

become a stability problem when a network becomes sufficiently large. For this reason, [I-D.ietf-rtgwg-cl-requirement] has a number of requirements focusing on minimally impacting scalability.

4. New Challenges

New technical challenges are posed by [I-D.ietf-rtgwg-cl-requirement] in both the control plane and data plane.

Among the more difficult challenges are the following.

1. requirements related delay or jitter (see Section 4.1.1),
2. the combination of ingress control over LSP placement and retaining an ability to move traffic as demands dictate can pose challenges and such requirements can even be conflicting (see target="sect.local-control" />),
3. path symmetry requires extensions and is particularly challenging for very large LSP (see Section 4.1.3),
4. accommodating a very wide range of requirements among contained LSP can lead to inefficiency if the most stringent requirements are reflected in aggregates, or reduce scalability if a large number of aggregates are used to provide a too fine a reflection of the requirements in the contained LSP (see Section 4.1.4),
5. backwards compatibility is somewhat limited due to the need to accommodate legacy multipath interfaces which provide too little information regarding their configured default behavior, and legacy LSP which provide too little information regarding their requirements (see Section 4.1.5),
6. data plane challenges include those of accommodating very large LSP, large microflows, traffic ordering constraints imposed by a subset of LSP, and accounting for IP and LDP traffic (see Section 4.2).

4.1. Control Plane Challenges

Some of the control plane requirements are particularly challenging. Handling large flows which aggregate smaller flows must be accomplished with minimal impact on scalability. Potentially conflicting are requirements for jitter and requirements for stability. Potentially conflicting are the requirements for ingress control of a large number of parameters, and the requirements for local control needed to achieve traffic balance across a composite

link. These challenges and potential solutions are discussed in the following sections.

4.1.1. Delay and Jitter Sensitive Routing

Delay and jitter sensitive routing are called for in [I-D.ietf-rtgwg-cl-requirement] in requirements FR#2, FR#7, FR#8, FR#9, FR#15, FR#16, FR#17, FR#18. Requirement FR#17 is particularly problematic, calling for constraints on jitter.

A tradeoff exists between scaling benefits of aggregating information, and potential benefits of using a finer granularity in delay reporting. To maintain the scaling benefit, measured link delay for any given composite link SHOULD be aggregated into a small number of delay ranges. IGP-TE extensions MUST be provided which advertise the available capacities for each of the selected ranges.

For path selection of delay sensitive LSP, the ingress SHOULD bias link metrics based on available capacity and select a low cost path which meets LSP total path delay criteria. To communicate the requirements of an LSP, the ERO MUST be extended to indicate the per link constraints. To communicate the type of resource used, the RRO SHOULD be extended to carry an identification of the group that is used to carry the LSP at each link bundle hop.

4.1.2. Local Control of Traffic Distribution

Many requirements in [I-D.ietf-rtgwg-cl-requirement] suggest that a node immediately adjacent to a component link should have a high degree of control over how traffic is distributed, as long as network performance objectives are met. Particularly relevant are FR#18 and FR#19.

The requirements to allow local control are potentially in conflict with requirement FR#21 which gives full control of component link select to the LSP ingress. While supporting this capability is mandatory, use of this feature is optional per LSP.

A given network deployment will have to consider this pair of conflicting requirements and make appropriate use of local control of traffic placement and ingress control of traffic placement to best meet network requirements.

4.1.3. Path Symmetry Requirements

Requirement FR#21 in [I-D.ietf-rtgwg-cl-requirement] includes a provision to bind both directions of a bidirectional LSP to the same component. This is easily achieved if the LSP is directly signaled

across a composite link. This is not as easily achieved if a set of LSP with this requirement are signaled over a large hierarchical LSP which is in turn carried over a composite link. The basis for load distribution in such a case is the label stack. The labels in either direction are completely independent.

This could be accommodated if the ingress, egress, and all midpoints of the hierarchical LSP make use of an entropy label in the distribution, and use only that entropy label. A solution for this problem may add complexity with very little benefit. There is little or no true benefit of using symmetrical paths rather than component links of identical characteristics.

Traffic symmetry and large LSP capacity are a second pair of conflicting requirements. Any given LSP can meet one of these two requirements but not both. A given network deployment will have to make appropriate use of each of these features to best meet network requirements.

4.1.4. Requirements for Contained LSP

[I-D.ietf-rtgwg-cl-requirement] calls for new LSP constraints. These constraints include frequency of load balancing rearrangement, delay and jitter, packet ordering constraints, and path symmetry.

When LSP are contained within hierarchical LSP, there is no signaling available at midpoint LSR which identifies the contained LSP let alone providing the set of requirements unique to each contained LSP. Defining extensions to provide this information would severely impact scalability and defeat the purpose of aggregating control information and forwarding information into hierarchical LSP. For the same scalability reasons, not aggregating at all is not a viable option for large networks where scalability and stability problems may occur as a result.

As pointed out in Section 4.1.3, the benefits of supporting symmetric paths among LSP contained within hierarchical LSP may not be sufficient to justify the complexity of supporting this capability.

A scalable solution which accommodates multiple sets of LSP between given pairs of LSR is to provide multiple hierarchical LSP for each given pair of LSR, each hierarchical LSP aggregating LSP with common requirements and a common pair of endpoints. This is a network design technique available to the network operator rather than a protocol extension. This technique can accommodate multiple sets of delay and jitter parameters, multiple sets of frequency of load balancing parameters, multiple sets of packet ordering constraints, etc.

4.1.5. Retaining Backwards Compatibility

Backwards compatibility and support for incremental deployment requires considering the impact of legacy LSR in the role of LSP ingress, and considering the impact of legacy LSR advertising ordinary links, advertising Ethernet LAG as ordinary links, and advertising link bundles.

Legacy LSR in the role of LSP ingress cannot signal requirements which are not supported by their control plane software. The additional capabilities supported by other LSR has no impact on these LSR. These LSR however, being unaware of extensions, may try to make use of scarce resources which support specific requirements such as low delay. To a limited extent it may be possible for a network operator to avoid this issue using existing mechanisms such as link administrative attributes and attribute affinities [RFC3209].

Legacy LSR advertising ordinary links will not advertise attributes needed by some LSP. For example, there is no way to determine the delay or jitter characteristics of such a link. Legacy LSR advertising Ethernet LAG pose additional problems. There is no way to determine that packet ordering constraints would be violated for LSP with strict packet ordering constraints, or that frequency of load balancing rearrangement constraints might be violated.

Legacy LSR advertising link bundles have no way to advertise the configured default behavior of the link bundle. Some link bundles may be configured to place each LSP on a single component link and therefore may not be able to accommodate an LSP which requires bandwidth in excess of the size of a component link. Some link bundles may be configured to spread all LSP over the all-ones component. For LSR using the all-ones component link, there is no documented procedure for correctly setting the "Maximum LSP Bandwidth". There is currently no way to indicate the largest microflow that could be supported by a link bundle using the all-ones component link.

Having received the RRO, it is possible for an ingress to look for the all-ones component to identify such link bundles after having signaled at least one LSP. Whether any LSR collects this information on legacy LSR and makes use of it to set defaults, is an implementation choice.

4.2. Data Plane Challenges

Flow identification is briefly discussed in Section 2.1. Traffic distribution is briefly discussed in Section 2.3. This section discusses issues specific to particular requirements specified in

[I-D.ietf-rtgwg-cl-requirement].

4.2.1. Very Large LSP

Very large LSP may exceed the capacity of any single component of a composite link. In some cases contained LSP may exceed the capacity of any single component. These LSP may the use of the equivalent of the all-ones component of a link bundle, or may use a subset of components which meet the LSP requirements.

Very large LSP can be accommodated as long as they can be subdivided (see Section 4.2.2). A very large LSP cannot have a requirement for symmetric paths unless complex protocol extensions are proposed (see Section 2.2 and Section 4.1.3).

4.2.2. Very Large Microflows

Within a very large LSP there may be very large microflows. A very large microflow is a very large flows which cannot be further subdivided. Flows which cannot be subdivided must be no larger than the capacity of any single component.

Current signaling provides no way to specify the largest microflow that can be supported on a given link bundle in routing advertisements. Extensions which address this are discussed in Section 6.4. Absent extensions of this type, traffic containing microflows that are too large for a given composite link may be present. There is no data plane solution for this problem that would not require reordering traffic at the composite link egress.

Some techniques are susceptible to statistical collisions where an algorithm to distribute traffic is unable to disambiguate traffic among two or more very large microflow where their sum is in excess of the capacity of any single component. Hash based algorithms which use too small a hash space are particularly susceptible and require a change in hash seed in the event that this were to occur. A change in hash seed is highly disruptive, causing traffic reordering among all traffic flows over which the hash function is applied.

4.2.3. Traffic Ordering Constraints

Some LSP have strict traffic ordering constraints. Most notable among these are MPLS-TP LSP. In the absence of aggregation into hierarchical LSP, those LSP with strict traffic ordering constraints can be placed on individual component links if there is a means of identifying which LSP have such a constraint. If LSP with strict traffic ordering constraints are aggregated in hierarchical LSP, the hierarchical LSP capacity may exceed the capacity of any single

component link. In such a case the load balancing for the containing may be constrained to look only at the top label and the first contained label. This and related issues are discussed further in Section 6.4.

4.2.4. Accounting for IP and LDP Traffic

Networks which carry RSVP-TE signaled MPLS traffic generally carry low volumes of native IP traffic, often only carrying control traffic as native IP. There is no architectural guarantee of this, it is just how network operators have made use of the protocols.

[I-D.ietf-rtgwg-cl-requirement] requires that native IP and native LDP be accommodated. In some networks, a subset of services may be carried as native IP or carried as native LDP. Today this may be accommodated by the network operator estimating the contribution of IP and LDP and configuring a lower set of available bandwidth figures on the RSVP-TE advertisements.

The only improvement that Composite Link can offer is that of measuring the IP and LDP traffic levels and automatically reducing the available bandwidth figures on the RSVP-TE advertisements. The measurements would have to be significantly filtered. This is similar to a feature in existing LSR, commonly known as "autobandwidth" with a key difference. In the "autobandwidth" feature, the bandwidth request of an RSVP-TE signaled LSP is adjusted in response to traffic measurements. In this case the IP or LDP traffic measurements are used to reduce the link bandwidth directly, without first encapsulating in an RSVP-TE LSP.

This may be a subtle and perhaps even a meaningless distinction if Composite Link is used to form a Sub-Path Maintenance Element (SPME). A SPME is in practice essentially an unsignaled single hop LSP with PHP enabled [RFC5921]. A Composite Link SPME looks very much like classic multipath, where there is no signaling, only management plane configuration creating the multipath entity (of which Ethernet Link Aggregation is a subset).

4.2.5. IP and LDP Limitations

IP does not offer traffic engineering. LDP cannot be extended to offer traffic engineering [RFC3468]. Therefore there is no traffic engineered fallback to an alternate path for IP and LDP traffic if resources are not adequate for the IP and/or LDP traffic alone on a given link in the primary path. The only option for IP and LDP would be to declare the link down. Declaring a link down due to resource exhaustion would reduce traffic to zero and eliminate the resource exhaustion. This would cause oscillations and is therefore not a

viable solution.

Congestion caused by IP or LDP traffic loads is a pathologic case that can occur if IP and/or LDP are carried natively and there is a high volume of IP or LDP traffic. This situation can be avoided by carrying IP and LDP within RSVP-TE LSP.

It is also not possible to route LDP traffic differently for different FEC. LDP traffic engineering is specifically disallowed by [RFC3468]. It may be possible to support multi-topology IGP extensions to accommodate more than one set of criteria. If so, the additional IGP could be bound to the forwarding criteria, and the LDP FEC bound to a specific IGP instance, inheriting the forwarding criteria. Alternately, one IGP instance can be used and the LDP SPF can make use of the constraints, such as delay and jitter, for a given LDP FEC. [Note: WG needs to discuss this and decide first whether to solve this at all and then if so, how.]

5. Existing Mechanisms

In MPLS the one mechanisms which support explicit signaling of multiple parallel links is Link Bundling [RFC4201]. The set of techniques known as "classis multipath" support no explicit signaling, except in two cases. In Ethernet Link Aggregation the Link Aggregation Control Protocol (LACP) coordinates the addition or removal of members from an Ethernet Link Aggregation Group (LAG). The use of the "all-ones" component of a link bundle indicates use of classis multipath, however the ability to determine if a link bundle makes use of classis multipath is not yet supported.

5.1. Link Bundling

Link bundling supports advertisement of a set of homogenous links as a single route advertisement. Link bundling supports placement of an LSP on any single component link, or supports placement of an LSP on the all-ones component link. Not all link bundling implementations support the all-ones component link. There is no way for an ingress LSR to tell which potential midpoint LSR support this feature and use it by default and which do not. Based on [RFC4201] it is unclear how to advertise a link bundle for which the all-ones component link is available and used by default. Common practice is to violate the specification and set the Maximum LSP Bandwidth to the Available Bandwidth. There is no means to determine the largest microflow that could be supported by a link bundle that is using the all-ones component link.

[RFC6107] extends the procedures for hierarchical LSP but also

extends link bundles. An LSP can be explicitly signaled to indicate that it is an LSP to be used as a component of a link bundle. Prior to that the common practice was to simply not advertise the component link LSP into the IGP, since only the ingress and egress of the link bundle needed to be aware of their existence, which they would be aware of due to the RSVP-TE signaling used in setting up the component LSP.

While link bundling can be the basis for composite links, a significant number of small extension needs to be added.

1. To support link bundles of heterogeneous links, a means of advertising the capacity available within a group of homogeneous needs to be provided.
2. Attributes need to be defined to support the following parameters for the link bundle or for a group of homogeneous links.
 - A. delay range
 - B. jitter (delay variation) range
 - C. group metric
 - D. all-ones component capable
 - E. capable of dynamically balancing load
 - F. largest supportable microflow
 - G. abilities to support strict packet ordering requirements within contained LSP
3. For each of the prior extended attributes, the constraint based routing path selection needs to be extended to reflect new constraints based on the extended attributes.
4. For each of the prior extended attributes, LSP admission control needs to be extended to reflect new constraints based on the extended attributes.
5. Dynamic load balance must be provided for flows within a given set of links with common attributes such that NPO are not violated including frequency of load balance adjustment for any given flow.

5.2. Classic Multipath

Classic multipath is defined in [I-D.symmvo-rtgwg-cl-use-cases].

Classic multipath refers to the most common current practice in implementation and deployment of multipath. The most common current practice makes use of a hash on the MPLS label stack and if IPv4 or IPv6 are indicated under the label stack, makes use of the IP source and destination addresses [RFC4385] [RFC4928].

Classic multipath provides a highly scalable means of load balancing. Adaptive multipath has proven value in assuring an even loading on component link and an ability to adapt to change in offered load that occurs over periods of hundreds of milliseconds or more. Classic multipath scalability is due to the ability to effectively work with an extremely large number of flows (IP host pairs) using relatively little resources (a data structure accessed using a hash result as a key or using ranges of hash results).

Classic multipath meets a small subset of Composite Link requirements. Due to scalability of the approach, classic multipath seems to be an excellent candidate for extension to meet the full set of Composite Link forwarding requirements.

Additional detail can be found in [I-D.symmvo-rtgwg-cl-use-cases].

6. Mechanisms Proposed in Other Documents

A number of documents which at the time of writing are works in progress address parts of the requirements of Composite Link, or assist in making some of the goals achievable.

6.1. Loss and Delay Measurement

Procedures for measuring loss and delay are provided in [RFC6374]. These are OAM based measurements. This work could be the basis of delay measurements and delay variation measurement used for metrics called for in [I-D.ietf-rtgwg-cl-requirement].

Currently there are two additional Internet-Drafts that address delay and delay variation metrics.

draft-wang-ccamp-latency-te-metric

[I-D.wang-ccamp-latency-te-metric] is designed specifically to meet this requirement. OSPF-TE and ISIS-TE extensions are defined to indicate link delay and delay variance. The RSVP-TE ERO is extended to include service level requirements. A latency

accumulation object is defined to provide a means of verification of the service level requirements. This draft is intended to proceed in the CCAMP WG. It is currently an individual submission. The 03 version of this draft expired in September 2012.

draft-giacalone-ospf-te-express-path

This document proposes to extend OSPF-TE only. Extensions support delay, delay variance, loss, residual bandwidth, and available bandwidth. No extensions to RSVP-TE are proposed. This draft is intended to proceed in the CCAMP WG. It is currently an individual submission. The 02 version will expire in March 2012.

A possible course of action may be to combine these two drafts. The delay variance, loss, residual bandwidth, and available bandwidth extensions are particularly prone to network instability. The question as to whether queuing delay and delay variation should be considered, and if so for which diffserv Per-Hop Service Class (PSC) is not addressed.

Note to co-authors: The ccamp-latency-te-metric draft refers to [I-D.ietf-rtgwg-cl-requirement] and is well matched to those requirements, including stability. The ospf-te-express-path draft refers to the "Alto Protocol" (draft-ietf-alto-protocol) and therefore may not be intended for RSVP-TE use. The authors of the two drafts may be able to resolve this. It may be best to drop ospf-te-express-path from this framework document.

6.2. Link Bundle Extensions

A set of link bundling extensions are defined in [I-D.ietf-mpls-explicit-resource-control-bundle]. This document provides extensions to the ERO and RRO to explicitly control the labels and resources within a bundle used by an LSP.

The extensions in this document could be further extended to support indicating a group of component links in the ERO or RRO, where the group is given an interface identification like the bundle itself. The extensions could also be further extended to support specification of the all-ones component link in the ERO or RRO.

[I-D.ietf-mpls-explicit-resource-control-bundle] does not provide a means to advertise the link bundle components. It is not certain how the ingress LSR would determine the set of link bundle component links available for a given link bundle.

[I-D.ospf-cc-stlv] provides a baseline draft for extending link

bundling to advertise components. A new component TVL (C-TLV) is proposed, which must reference a Composite Link Link TLV. [I-D.ospf-cc-stlv] is intended for the OSPF WG and submitted for the "Experimental" track. The 00 version expired in February 2012.

6.3. Fat PW and Entropy Labels

Two documents provide a means to add entropy for the purpose of improving load balance. MPLS encapsulation can bury information that is needed to identify microflows. These two documents allow a pseudowire ingress and LSP ingress respectively to add a label solely for the purpose of providing a finer granularity of microflow groups.

[RFC6391] allows pseudowires which carry a large volume of traffic, where microflows can be identified to be load balanced across multiple members of an Ethernet LAG or an MPLS link bundle. This is accomplished by adding a flow label below the pseudowire label in the MPLS label stack. For this to be effective the link bundle load balance must make use of the label stack up to and including this flow label.

[I-D.ietf-mpls-entropy-label] provides a means for a LER to put an additional label known as an entropy label on the MPLS label stack. As defined, only the LER can add the entropy label.

Core LSR acting as LER for aggregated LSP can add entropy labels based on deep packet inspection and place an entropy label indicator (ELI) and entropy label (EL) just below the label being acted on. This would be helpful in situations where the label stack depth to which load distribution can operate is limited by implementation or is limited for other reasons such as carrying both MPLS-TP and MPLS with entropy labels within the same hierarchical LSP.

6.4. Multipath Extensions

The multipath extensions drafts address one aspect of Composite Link. These drafts deal with the issue of accommodating LSP which have strict packet ordering constraints in a network containing multipath. MPLS-TP has become the one important instance of LSP with strict packet ordering constraints and has driven this work.

[I-D.villamizar-mpls-tp-multipath] outlines requirements and gives a number of options for dealing with the apparent incompatibility of MPLS-TP and multipath. A preferred option is described.

[I-D.villamizar-mpls-tp-multipath-te-extn] provides protocol extensions needed to implement the preferred option described in [I-D.villamizar-mpls-tp-multipath].

Other issues pertaining to multipath are also addressed. Means to advertise the largest microflow supportable are defined. Means to indicate the largest expected microflow within an LSP are defined. Issues related to hierarchy are addressed.

7. Required Protocol Extensions and Mechanisms

Prior sections have reviewed key characteristics, architecture tradeoffs, new challenges, existing mechanisms, and relevant mechanisms proposed in existing new documents.

This section first summarizes and groups requirements. A set of documents coverage groupings are proposed with existing works-in-progress noted where applicable. The set of extensions are then grouped by protocol affected as a convenience to implementors.

7.1. Brief Review of Requirements

The following list provides a categorization of requirements specified in [I-D.ietf-rtgwg-cl-requirement] along with a short phrase indication what topic the requirement covers.

routing information aggregation

FR#1 (routing summarization), FR#20 (composite link may be a component of another composite link)

restoration speed

FR#2 (restoration speed meeting NPO), FR#12 (minimally disruptive load rebalance), DR#6 (fast convergence), DR#7 (fast worst case failure convergence)

load distribution, stability, minimal disruption

FR#3 (automatic load distribution), FR#5 (must not oscillate), FR#11 (dynamic placement of flows), FR#12 (minimally disruptive load rebalance), FR#13 (bounded rearrangement frequency), FR#18 (flow placement must satisfy NPO), FR#19 (flow identification finer than per top level LSP), MR#6 (operator initiated flow rebalance)

backward compatibility and migration

FR#4 (smooth incremental deployment), FR#6 (management and diagnostics must continue to function), DR#1 (extend existing protocols), DR#2 (extend LDP, no LDP TE)

delay and delay variation

FR#7 (expose lower layer measured delay), FR#8 (precision of latency reporting), FR#9 (limit latency on per LSP basis), FR#15 (minimum delay path), FR#16 (bounded delay path), FR#17 (bounded jitter path)

admission control, preemption, traffic engineering

FR#10 (admission control, preemption), FR#14 (packet ordering), FR#21 (ingress specification of path), FR#22 (path symmetry), DR#3 (IP and LDP traffic), MR#3 (management specification of path)

single vs multiple domain

DR#4 (IGP extensions allowed within single domain), DR#5 (IGP extensions disallowed in multiple domain case)

general network management

MR#1 (polling, configuration, and notification), MR#2 (activation and de-activation)

path determination, connectivity verification

MR#4 (path trace), MR#5 (connectivity verification)

The above list is not intended as a substitute for [I-D.ietf-rtgwg-cl-requirement], but rather as a concise grouping and reminder or requirements to serve as a means of more easily determining requirements coverage of a set of protocol documents.

7.2. Required Document Coverage

The primary areas where additional protocol extensions and mechanisms are required include the topics described in the following subsections.

There are candidate documents for a subset of the topics below. This grouping of topics does not require that each topic be addressed by a separate document. In some cases, a document may cover multiple topics, or a specific topic may be addressed as applicable in multiple documents.

7.2.1. Component Link Grouping

An extension to link bundling is needed to specify a group of components with common attributes. This can be a TLV defined within the link bundle that carries the same encapsulations as the link bundle. Two interface indices would be needed for each group.

- a. An index is needed that if included in an ERO would indicate the need to place the LSP on any one component within the group.
- b. A second index is needed that if included in an ERO would indicate the need to balance flows within the LSP across all components of the group. This is equivalent to the "all-ones" component for the entire bundle.

[I-D.ospf-cc-stlv] can be extended to include multipath treatment capabilities. An ISIS solution is also needed. An extension of RSVP-TE signaling is needed to indicate multipath treatment preferences.

If a component group is allowed to support all of the parameters of a link bundle, then a group TE metric would be accommodated. This can be supported with the component TLV (C-TLV) defined in [I-D.ospf-cc-stlv].

The primary focus of this document, among the sets of requirements listed in Section 7.1 is the "routing information aggregation" set of requirements. The "restoration speed", "backward compatibility and migration", and "general network management" requirements must also be considered.

7.2.2. Delay and Jitter Extensions

A extension is needed in the IGP-TE advertisement to support delay and delay variation for links, link bundles, and forwarding adjacencies. Whatever mechanism is described must take precautions that insure that route oscillations cannot occur. [I-D.wang-ccamp-latency-te-metric] may be a good starting point.

The primary focus of this document, among the sets of requirements listed in Section 7.1 is the "delay and delay variation" set of requirements. The "restoration speed", "backward compatibility and migration", and "general network management" requirements must also be considered.

7.2.3. Path Selection and Admission Control

Path selection and admission control changes must be documented in each document that proposes a protocol extension that advertises a new capability or parameter that must be supported by changes in path selection and admission control.

The primary focus of this document, among the sets of requirements listed in Section 7.1 are the "load distribution, stability, minimal disruption" and "admission control, preemption, traffic engineering"

sets of requirements. The "restoration speed" and "path determination, connectivity verification" requirements must also be considered. The "backward compatibility and migration", and "general network management" requirements must also be considered.

7.2.4. Dynamic Multipath Balance

FR#11 explicitly calls for dynamic load balancing similar to existing adaptive multipath. In implementations where flow identification uses a coarse granularity, the adjustments would have to be equally coarse, in the worst case moving entire LSP. The impact of flow identification granularity and potential adaptive multipath approaches may need to be documented in greater detail than provided here.

The primary focus of this document, among the sets of requirements listed in Section 7.1 are the "restoration speed" and the "load distribution, stability, minimal disruption" sets of requirements. The "path determination, connectivity verification" requirements must also be considered. The "backward compatibility and migration", and "general network management" requirements must also be considered.

7.2.5. Frequency of Load Balance

IGP-TE and RSVP-TE extensions are needed to support frequency of load balancing rearrangement called for in FR#13, and FR#15-FR#17. Constraints are not defined in RSVP-TE, but could be modeled after administrative attribute affinities in RFC3209 and elsewhere.

The primary focus of this document, among the sets of requirements listed in Section 7.1 is the "load distribution, stability, minimal disruption" set of requirements. The "path determination, connectivity verification" must also be considered. The "backward compatibility and migration" and "general network management" requirements must also be considered.

7.2.6. Inter-Layer Communication

Lower layer to upper layer communication called for in FR#7 and FR#20. This is addressed for a subset of parameters related to packet ordering in [I-D.villamizar-mpls-tp-multipath] where layers are MPLS. Remaining parameters, specifically delay and delay variation, need to be addressed. Passing information from a lower non-MPLS layer to an MPLS layer needs to be addressed, though this may largely be generic advice encouraging a coupling of MPLS to lower layer management plane or control plane interfaces. This topic can be addressed in each document proposing a protocol extension, where applicable.

The primary focus of this document, among the sets of requirements listed in Section 7.1 is the "restoration speed" set of requirements. The "backward compatibility and migration" and "general network management" requirements must also be considered.

7.2.7. Packet Ordering Requirements

A document is needed to define extensions supporting various packet ordering requirements, ranging from requirements to preservice microflow ordering only, to requirements to preservice full LSP ordering (as in MPLS-TP). This is covered by [I-D.villamizar-mpls-tp-multipath] and [I-D.villamizar-mpls-tp-multipath-te-extn].

The primary focus of this document, among the sets of requirements listed in Section 7.1 are the "admission control, preemption, traffic engineering" and the "path determination, connectivity verification" sets of requirements. The "backward compatibility and migration" and "general network management" requirements must also be considered.

7.2.8. Minimally Disruption Load Balance

The behavior of hash methods used in classic multipath needs to be described in terms of FR#12 which calls for minimally disruptive load adjustments. For example, reseeding the hash violates FR#12. Using modulo operations is significantly disruptive if a link comes or goes down, as pointed out in [RFC2992]. In addition, backwards compatibility with older hardware needs to be accommodated.

The primary focus of this document, among the sets of requirements listed in Section 7.1 is the "load distribution, stability, minimal disruption" set of requirements.

7.2.9. Path Symmetry

Protocol extensions are needed to support dynamic load balance as called for to meet FR#22 (path symmetry) and to meet FR#11 (dynamic placement of flows). Currently path symmetry can only be supported in link bundling if the path is pinned. When a flow is moved both ingress and egress must make the move as close to simultaneously as possible to satisfy FR#22 and FR#12 (minimally disruptive load rebalance). If a group of flows are identified using a hash, then the hash must be identical on the pair of LSR at the endpoint, using the same hash seed and with one side swapping source and destination. If the label stack is used, then either the entire label stack must be a special case flow identification, since the set of labels in either direction are not correlated, or the two LSR must conspire to use the same flow identifier. For example, using a common entropy

label value, and using only the entropy label in the flow identification would satisfy this requirement.

The primary focus of this document, among the sets of requirements listed in Section 7.1 are the "load distribution, stability, minimal disruption" and the "admission control, preemption, traffic engineering" sets of requirements. The "backward compatibility and migration" and "general network management" requirements must also be considered. Path symmetry simplifies support for the "path determination, connectivity verification" set of requirements, but with significant complexity added elsewhere.

7.2.10. Performance, Scalability, and Stability

A separate document providing analysis of performance, scalability, and stability impacts of changes may be needed. The topic of traffic adjustment oscillation must also be covered. If sufficient coverage is provided in each document covering a protocol extension, a separate document would not be needed.

The primary focus of this document, among the sets of requirements listed in Section 7.1 is the "restoration speed" set of requirements. This is not a simple topic and not a topic that is well served by scattering it over multiple documents, therefore it may be best to put this in a separate document and put citations in documents called for in Section 7.2.1, Section 7.2.2, Section 7.2.3, Section 7.2.9, Section 7.2.11, Section 7.2.12, Section 7.2.13, and Section 7.2.14. Citation may also be helpful in Section 7.2.4, and Section 7.2.5.

7.2.11. IP and LDP Traffic

A document is needed to define the use of measurements native IP and native LDP traffic levels to reduce link advertised bandwidth amounts.

The primary focus of this document, among the sets of requirements listed in Section 7.1 are the "load distribution, stability, minimal disruption" and the "admission control, preemption, traffic engineering" set of requirements. The "path determination, connectivity verification" must also be considered. The "backward compatibility and migration" and "general network management" requirements must also be considered.

7.2.12. LDP Extensions

Extending LDP is called for in DR#2. LDP can be extended to couple FEC admission control to local resource availability without providing LDP traffic engineering capability. Other LDP extensions

such as signaling a bound on microflow size and LDP LSP requirements would provide useful information without providing LDP traffic engineering capability.

The primary focus of this document, among the sets of requirements listed in Section 7.1 is the "admission control, preemption, traffic engineering" set of requirements. The "backward compatibility and migration" and "general network management" requirements must also be considered.

7.2.13. Pseudowire Extensions

PW extensions such as signaling a bound on microflow size and PW requirements would provide useful information.

The primary focus of this document, among the sets of requirements listed in Section 7.1 is the "admission control, preemption, traffic engineering" set of requirements. The "backward compatibility and migration" and "general network management" requirements must also be considered.

7.2.14. Multi-Domain Composite Link

DR#5 calls for Composite Link to span multiple network topologies. Component LSP may already span multiple network topologies, though most often in practice these are LDP signaled. Component LSP which are RSVP-TE signaled may also span multiple network topologies using at least three existing methods (per domain [RFC5152], BRPC [RFC5441], PCE [RFC4655]). When such component links are combined in a Composite Link, the Composite Link spans multiple network topologies. It is not clear in which document this needs to be described or whether this description in the framework is sufficient. The authors and/or the WG may need to discuss this. DR#5 mandates that IGP-TE extension cannot be used. This would disallow the use of [RFC5316] or [RFC5392] in conjunction with [RFC5151].

The primary focus of this document, among the sets of requirements listed in Section 7.1 are "single vs multiple domain" and "admission control, preemption, traffic engineering". The "routing information aggregation" and "load distribution, stability, minimal disruption" requirements need attention due to their use of the IGP in single domain Composite Link. Other requirements such as "delay and delay variation", can more easily be accommodated by carrying metrics within BGP. The "path determination, connectivity verification" requirements need attention due to requirements to restrict disclosure of topology information across domains in multi-domain deployments. The "backward compatibility and migration" and "general network management" requirements must also be considered.

7.3. Open Issues Regarding Requirements

Note to co-authors: This section needs to be reduced to an empty section and then removed.

The following topics in the requirements document are not addressed. Since they are explicitly mentioned in the requirements document some mention of how they are supported is needed, even if to say nother needed to be done. If we conclude any particular topic is irrelevant, maybe the topic should be removed from the requirement document. At that point we could add the management requirements that have come up and were missed.

1. L3VPN RFC 4364, RFC 4797, L2VPN RFC 4664, VPWS, VPLS RFC 4761, RFC 4762 and VPMS VPMS Framework (draft-ietf-l2vpn-vpms-frmwk-requirements). It is not clear what additional Composite Link requirements these references imply, if any. If no additional requirements are implied, then these references are considered to be informational only.
2. Migration may not be adequately covered in Section 4.1.5. It might also be necessary to say more here on performance, scalability, and stability as it related to migration. Comments on this from co-authors or the WG?
3. We may need a performance section in this document to specifically address #DR6 (fast convergence), and #DR7 (fast worst case failure convergence), though we do already have scalability discussion. The performance section would have to say "no worse than before, except were there was no alternative to make it very slightly worse" (in a bit more detail than that). It would also have to better define the nature of the performance criteria.

7.4. Framework Requirement Coverage by Protocol

As an aid to implementors, this section summarizes requirement coverage listed in Section 7.2 by protocol or LSR functionality affected.

Some documentation may be purely informational, proposing no changes and proposing usage at most. This includes Section 7.2.3, Section 7.2.8, Section 7.2.10, and Section 7.2.14.

Section 7.2.9 may require a new protocol.

7.4.1. OSPF-TE and ISIS-TE Protocol Extensions

Many of the changes listed in Section 7.2 require IGP-TE changes, though most are small extensions to provide additional information. This set includes Section 7.2.1, Section 7.2.2, Section 7.2.5, Section 7.2.6, and Section 7.2.7. An adjustment to existing advertised parameters is suggested in Section 7.2.11.

7.4.2. PW Protocol Extensions

The only suggestion of pseudowire (PW) extensions is in Section 7.2.13.

7.4.3. LDP Protocol Extensions

Potential LDP extensions are described in Section 7.2.12.

7.4.4. RSVP-TE Protocol Extensions

RSVP-TE protocol extensions are called for in Section 7.2.1, Section 7.2.5, Section 7.2.7, and Section 7.2.9.

7.4.5. RSVP-TE Path Selection Changes

Section 7.2.3 calls for path selection to be addressed in individual documents that require change. These changes would include those proposed in Section 7.2.1, Section 7.2.2, Section 7.2.5, and Section 7.2.7.

7.4.6. RSVP-TE Admission Control and Preemption

When a change is needed to path selection, a corresponding change is needed in admission control. The same set of sections applies: Section 7.2.1, Section 7.2.2, Section 7.2.5, and Section 7.2.7. Some resource changes such as a link delay change might trigger preemption. The rules of preemption remain unchanged, still based on holding priority.

7.4.7. Flow Identification and Traffic Balance

The following describe either the state of the art in flow identification and traffic balance or propose changes: Section 7.2.4, Section 7.2.5, Section 7.2.7, and Section 7.2.8.

8. Security Considerations

The security considerations for MPLS/GMPLS and for MPLS-TP are

documented in [RFC5920] and [I-D.ietf-mpls-tp-security-framework].

The types protocol extensions proposed in this framework document provide additional information about links, forwarding adjacencies, and LSP requirements. The protocol semantics changes described in this framework document propose additional LSP constraints applied at path computation time and at LSP admission at midpoints LSR. The additional information and constraints provide no additional security considerations beyond the security considerations already documented in [RFC5920] and [I-D.ietf-mpls-tp-security-framework].

9. Acknowledgments

Authors would like to thank Adrian Farrel, Fred Jounay, Yuji Kamite for his extensive comments and suggestions regarding early versions of this document, Ron Bonica, Nabil Bitar, Eric Gray, Lou Berger, and Kireeti Kompella for their reviews of early versions and great suggestions.

Authors would like to thank Iftekhhar Hussain for review and suggestions regarding recent versions of this document.

In the interest of full disclosure of affiliation and in the interest of acknowledging sponsorship, past affiliations of authors are noted. Much of the work done by Ning So occurred while Ning was at Verizon. Much of the work done by Curtis Villamizar occurred while at Infinera. Infinera continues to sponsor this work on a consulting basis.

10. References

10.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, December 2001.
- [RFC3630] Katz, D., Kompella, K., and D. Yeung, "Traffic Engineering (TE) Extensions to OSPF Version 2", RFC 3630, September 2003.
- [RFC4201] Kompella, K., Rekhter, Y., and L. Berger, "Link Bundling in MPLS Traffic Engineering (TE)", RFC 4201, October 2005.

- [RFC4206] Kompella, K. and Y. Rekhter, "Label Switched Paths (LSP) Hierarchy with Generalized Multi-Protocol Label Switching (GMPLS) Traffic Engineering (TE)", RFC 4206, October 2005.
- [RFC5036] Andersson, L., Minei, I., and B. Thomas, "LDP Specification", RFC 5036, October 2007.
- [RFC5305] Li, T. and H. Smit, "IS-IS Extensions for Traffic Engineering", RFC 5305, October 2008.
- [RFC5712] Meyer, M. and JP. Vasseur, "MPLS Traffic Engineering Soft Preemption", RFC 5712, January 2010.
- [RFC6107] Shiimoto, K. and A. Farrel, "Procedures for Dynamically Signaled Hierarchical Label Switched Paths", RFC 6107, February 2011.
- [RFC6374] Frost, D. and S. Bryant, "Packet Loss and Delay Measurement for MPLS Networks", RFC 6374, September 2011.
- [RFC6391] Bryant, S., Filsfils, C., Drafz, U., Kompella, V., Regan, J., and S. Amante, "Flow-Aware Transport of Pseudowires over an MPLS Packet Switched Network", RFC 6391, November 2011.

10.2. Informative References

- [DBP] Bertsekas, D., "Dynamic Behavior of Shortest Path Routing Algorithms for Communication Networks", IEEE Trans. Auto. Control 1982.
- [I-D.ietf-mpls-entropy-label]
Drake, J., Kompella, K., Yong, L., Amante, S., and W. Henderickx, "The Use of Entropy Labels in MPLS Forwarding", draft-ietf-mpls-entropy-label-01 (work in progress), October 2011.
- [I-D.ietf-mpls-explicit-resource-control-bundle]
Zamfir, A., Ali, Z., and P. Dimitri, "Component Link Recording and Resource Control for TE Links", draft-ietf-mpls-explicit-resource-control-bundle-10 (work in progress), April 2011.
- [I-D.ietf-mpls-tp-security-framework]
Niven-Jenkins, B., Fang, L., Graveman, R., and S. Mansfield, "MPLS-TP Security Framework", draft-ietf-mpls-tp-security-framework-02 (work in progress), October 2011.

- [I-D.ietf-rtgwg-cl-requirement]
Malis, A., Villamizar, C., McDysan, D., Yong, L., and N. So, "Requirements for MPLS Over a Composite Link", draft-ietf-rtgwg-cl-requirement-05 (work in progress), January 2012.
- [I-D.kompella-mpls-rsvp-ecmp]
Kompella, K., "Multi-path Label Switched Paths Signaled Using RSVP-TE", draft-kompella-mpls-rsvp-ecmp-01 (work in progress), October 2011.
- [I-D.ospf-cc-stlv]
Osborne, E., "Component and Composite Link Membership in OSPF", draft-ospf-cc-stlv-00 (work in progress), August 2011.
- [I-D.symmvo-rtgwg-cl-use-cases]
Malis, A., Villamizar, C., McDysan, D., Yong, L., and N. So, "Composite Link Use Cases and Design Considerations", draft-symmvo-rtgwg-cl-use-cases-00 (work in progress), February 2012.
- [I-D.villamizar-mpls-tp-multipath]
Villamizar, C., "Use of Multipath with MPLS-TP and MPLS", draft-villamizar-mpls-tp-multipath-01 (work in progress), March 2011.
- [I-D.villamizar-mpls-tp-multipath-te-extn]
Villamizar, C., "Multipath Extensions for MPLS Traffic Engineering", draft-villamizar-mpls-tp-multipath-te-extn-00 (work in progress), July 2011.
- [I-D.wang-ccamp-latency-te-metric]
Fu, X., Betts, M., Wang, Q., McDysan, D., and A. Malis, "GMPLS extensions to communicate latency as a traffic engineering performance metric", draft-wang-ccamp-latency-te-metric-03 (work in progress), March 2011.
- [RFC2475] Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., and W. Weiss, "An Architecture for Differentiated Services", RFC 2475, December 1998.
- [RFC2991] Thaler, D. and C. Hopps, "Multipath Issues in Unicast and Multicast Next-Hop Selection", RFC 2991, November 2000.
- [RFC2992] Hopps, C., "Analysis of an Equal-Cost Multi-Path

Algorithm", RFC 2992, November 2000.

- [RFC3260] Grossman, D., "New Terminology and Clarifications for Diffserv", RFC 3260, April 2002.
- [RFC3468] Andersson, L. and G. Swallow, "The Multiprotocol Label Switching (MPLS) Working Group decision on MPLS signaling protocols", RFC 3468, February 2003.
- [RFC3945] Mannie, E., "Generalized Multi-Protocol Label Switching (GMPLS) Architecture", RFC 3945, October 2004.
- [RFC3985] Bryant, S. and P. Pate, "Pseudo Wire Emulation Edge-to-Edge (PWE3) Architecture", RFC 3985, March 2005.
- [RFC4385] Bryant, S., Swallow, G., Martini, L., and D. McPherson, "Pseudowire Emulation Edge-to-Edge (PWE3) Control Word for Use over an MPLS PSN", RFC 4385, February 2006.
- [RFC4655] Farrel, A., Vasseur, J., and J. Ash, "A Path Computation Element (PCE)-Based Architecture", RFC 4655, August 2006.
- [RFC4928] Swallow, G., Bryant, S., and L. Andersson, "Avoiding Equal Cost Multipath Treatment in MPLS Networks", BCP 128, RFC 4928, June 2007.
- [RFC5151] Farrel, A., Ayyangar, A., and JP. Vasseur, "Inter-Domain MPLS and GMPLS Traffic Engineering -- Resource Reservation Protocol-Traffic Engineering (RSVP-TE) Extensions", RFC 5151, February 2008.
- [RFC5152] Vasseur, JP., Ayyangar, A., and R. Zhang, "A Per-Domain Path Computation Method for Establishing Inter-Domain Traffic Engineering (TE) Label Switched Paths (LSPs)", RFC 5152, February 2008.
- [RFC5316] Chen, M., Zhang, R., and X. Duan, "ISIS Extensions in Support of Inter-Autonomous System (AS) MPLS and GMPLS Traffic Engineering", RFC 5316, December 2008.
- [RFC5392] Chen, M., Zhang, R., and X. Duan, "OSPF Extensions in Support of Inter-Autonomous System (AS) MPLS and GMPLS Traffic Engineering", RFC 5392, January 2009.
- [RFC5441] Vasseur, JP., Zhang, R., Bitar, N., and JL. Le Roux, "A Backward-Recursive PCE-Based Computation (BRPC) Procedure to Compute Shortest Constrained Inter-Domain Traffic Engineering Label Switched Paths", RFC 5441, April 2009.

[RFC5920] Fang, L., "Security Framework for MPLS and GMPLS Networks", RFC 5920, July 2010.

[RFC5921] Bocci, M., Bryant, S., Frost, D., Levrau, L., and L. Berger, "A Framework for MPLS in Transport Networks", RFC 5921, July 2010.

Authors' Addresses

So Ning
Tata Communications

Email: ning.so@tatacommunications.com

Dave McDysan
Verizon
22001 Loudoun County PKWY
Ashburn, VA 20147

Email: dave.mcdysan@verizon.com

Eric Osborne
Cisco

Email: eosborne@cisco.com

Lucy Yong
Huawei USA
5340 Legacy Dr.
Plano, TX 75025

Phone: +1 469-277-5837
Email: lucy.yong@huawei.com

Curtis Villamizar
Outer Cape Cod Network Consulting

Email: curtis@occnc.com

RTGWG
Internet-Draft
Intended status: Informational
Expires: December 22, 2012

S. Ning
Tata Communications
A. Malis
D. McDysan
Verizon
L. Yong
Huawei USA
C. Villamizar
Outer Cape Cod Network
Consulting
June 20, 2012

Composite Link Use Cases and Design Considerations
draft-symmvo-rtgwg-cl-use-cases-01

Abstract

This document provides a set of use cases and design considerations for composite links.

Composite link is a formalization of multipath techniques currently in use in IP and MPLS networks and a set of extensions to multipath techniques.

Note: symmvo in the draft name is the initials of the set of authors: So, Yong, McDysan, Malis, Villamizar, Osborne. This paragraph will be removed when/if this document is adopted as a WG item.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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."

This Internet-Draft will expire on December 22, 2012.

Copyright Notice

Copyright (c) 2012 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 Simplified BSD License.

Table of Contents

| | |
|---|----|
| 1. Introduction | 3 |
| 2. Conventions used in this document | 3 |
| 2.1. Terminology | 3 |
| 3. Composite Link Foundation Use Cases | 4 |
| 4. Delay Sensitive Applications | 7 |
| 5. Large Volume of IP and LDP Traffic | 7 |
| 6. Composite Link and Packet Ordering | 8 |
| 6.1. MPLS-TP in network edges only | 10 |
| 6.2. Composite Link at core LSP ingress/egress | 11 |
| 6.3. MPLS-TP as a MPLS client | 12 |
| 7. Security Considerations | 12 |
| 8. Acknowledgments | 13 |
| 9. References | 13 |
| 9.1. Normative References | 13 |
| 9.2. Informative References | 13 |
| Appendix A. More Details on Existing Network Operator Practices and Protocol Usage | 15 |
| Appendix B. Existing Multipath Standards and Techniques | 17 |
| B.1. Common Multipath Load Splitting Techniques | 18 |
| B.2. Simple and Adaptive Load Balancing Multipath | 19 |
| B.3. Traffic Split over Parallel Links | 20 |
| B.4. Traffic Split over Multiple Paths | 20 |
| Appendix C. Characteristics of Transport in Core Networks | 20 |
| Authors' Addresses | 22 |

1. Introduction

Composite link requirements are specified in [I-D.ietf-rtgwg-cl-requirement]. A composite link framework is defined in [I-D.so-yong-rtgwg-cl-framework].

Multipath techniques have been widely used in IP networks for over two decades. The use of MPLS began more than a decade ago. Multipath has been widely used in IP/MPLS networks for over a decade with very little protocol support dedicated to effective use of multipath.

The state of the art in multipath prior to composite links is documented in Appendix B.

Both Ethernet Link Aggregation [IEEE-802.1AX] and MPLS link bundling [RFC4201] have been widely used in today's MPLS networks. Composite link differs in the following characteristics.

1. A composite link allows bundling of non-homogenous links together as a single logical link.
2. A composite link provides more information in the TE-LSDB and supports more explicit control over placement of LSP.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2.1. Terminology

Terminology defined in [I-D.ietf-rtgwg-cl-requirement] is used in this document.

In addition, the following terms are used:

classic multipath:

Classic multipath refers to the most common current practice in implementation and deployment of multipath (see Appendix A). The most common current practice makes use of a hash on the MPLS label stack and if IPv4 or IPv6 are indicated under the label stack, makes use of the IP source and destination addresses [RFC4385] [RFC4928].

classic link bundling:

Classic link bundling refers to the use of [RFC4201] where the "all ones" component is not used. Where the "all ones" component is used, link bundling behaves as classic multipath does. Classic link bundling selects a single component link on which to put any given LSP.

Among the important distinctions between classic multipath or classic link bundling and Composite Link are:

1. Classic multipath has no provision to retain order among flows within a subset of LSP. Classic link bundling retains order among all flows but as a result does a poor job of splitting load among components and therefore is rarely (if ever) deployed. Composite Link allows per LSP control of load split characteristics.
2. Classic multipath and classic link bundling do not provide a means to put some LSP on component links with lower delay. Composite Link does.
3. Classic multipath will provide a load balance for IP and LDP traffic. Classic link bundling will not. Neither classic multipath or classic link bundling will measure IP and LDP traffic and reduce the advertised "Available Bandwidth" as a result of that measurement. Composite Link better supports RSVP-TE used with significant traffic levels of native IP and native LDP.
4. Classic link bundling cannot support an LSP that is greater in capacity than any single component link. Classic multipath and Composite Link support this capability but will reorder traffic on such an LSP. Composite Link can retain order of an LSP that is carried within an LSP that is greater in capacity than any single component link if the contained LSP has such a requirement.

None of these techniques, classic multipath, classic link bundling, or Composite Link, will reorder traffic among IP microflows. None of these techniques will reorder traffic among PW, if a PWE3 Control Word is used [RFC4385].

3. Composite Link Foundation Use Cases

A simple composite link composed entirely of physical links is illustrated in Figure 1, where a composite link is configured between LSR1 and LSR2. This composite link has three component links.

Individual component links in a composite link may be supported by different transport technologies such as wavelength, Ethernet VLAN. Even if the transport technology implementing the component links is identical, the characteristics (e.g., bandwidth, latency) of the component links may differ.

The composite link in Figure 1 may carry LSP traffic flows and control plane packets. Control plane packets may appear as IP packets or may be carried within a generic associated channel (G-Ach) [RFC5586]. A LSP may be established over the link by either RSVP-TE [RFC3209] or LDP [RFC5036] signaling protocols. All component links in a composite link are summarized in the same forwarding adjacency LSP (FA-LSP) routing advertisement [RFC3945]. The composite link is summarized as one TE-Link advertised into the IGP by the composite link end points. This information is used in path computation when a full MPLS control plane is in use. The individual component links or groups of component links may optionally be advertised into the IGP as sub-TLV of the composite link advertisement to indicate capacity available with various characteristics, such as a delay range.

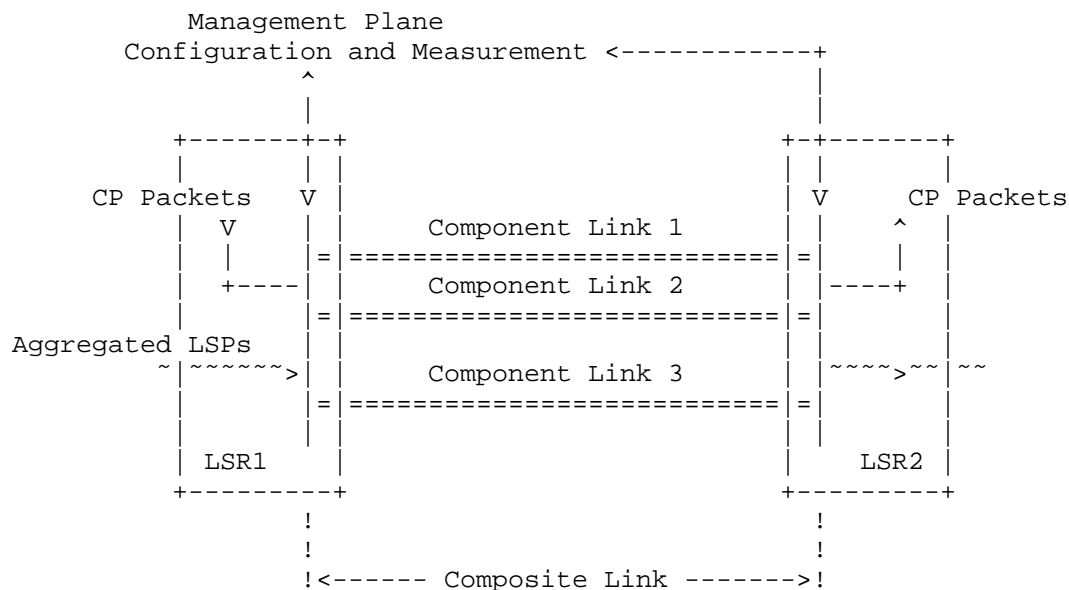


Figure 1: a composite link constructed with multiple physical links between two LSR

[I-D.ietf-rtgwg-cl-requirement] specifies that component links may themselves be composite links. Figure 2 shows three three forms of component links which may be deployed in a network.

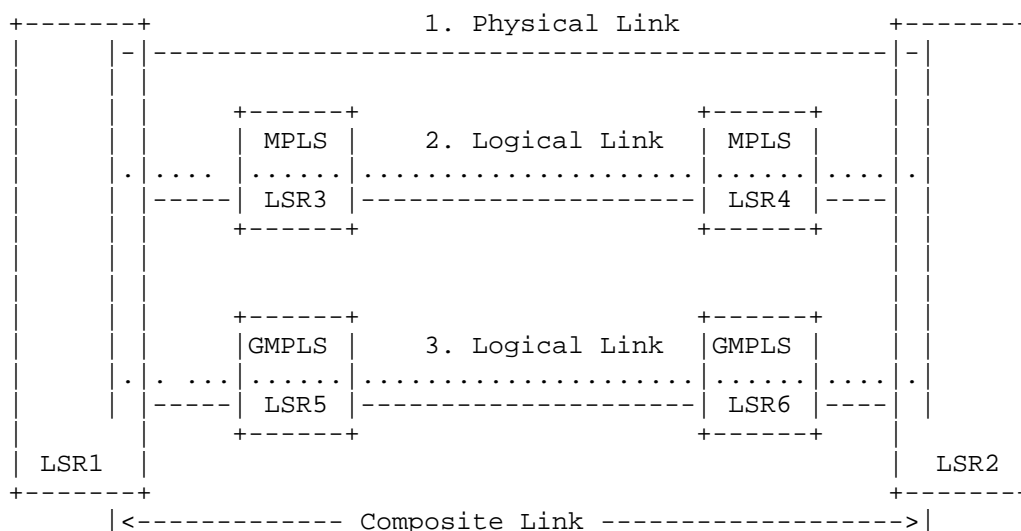


Figure 2: Illustration of Various Component Link Types

The three forms of component link shown in Figure 2 are:

1. The first component link is configured with direct physical media.
2. The second component link is a TE tunnel that traverses LSR3 and LSR4, where LSR3 and LSR4 are the nodes supporting MPLS, but supporting few or no GMPLS extensions.
3. The third component link is formed by lower layer network that has GMPLS enabled. In this case, LSR5 and LSR6 are not the nodes controlled by the MPLS but provide the connectivity for the component link.

A composite link forms one logical link between connected LSR and is used to carry aggregated traffic [I-D.ietf-rtgwg-cl-requirement]. Composite link relies on its component links to carry the traffic over the composite link. The endpoints of the composite link maps incoming traffic into component links.

For example, LSR1 in Figure 1 distributes the set of traffic flows including control plane packets among the set of component links. LSR2 in Figure 1 receives the packets from its component links and sends them to MPLS forwarding engine with no attempt to reorder packets arriving on different component links. The traffic in the opposite direction, from LSR2 to LSR1, is distributed across the set of component links by the LSR2.

These three forms of component link are only example. Many other examples are possible. A component link may itself be a composite link. A segment of an LSP (single hop for that LSP) may be a composite link.

4. Delay Sensitive Applications

Most applications benefit from lower delay. Some types of applications are far more sensitive than others. For example, real time bidirectional applications such as voice communication or two way video conferencing are far more sensitive to delay than unidirectional streaming audio or video. Non-interactive bulk transfer is almost insensitive to delay if a large enough TCP window is used.

Some applications are sensitive to delay but unwilling to pay extra to insure lower delay. For example, many SIP end users are willing to accept the delay offered to best effort services as long as call quality is good most of the time.

Other applications are sensitive to delay and willing to pay extra to insure lower delay. For example, financial trading applications are extremely sensitive to delay and with a lot at stake are willing to go to great lengths to reduce delay.

Among the requirements of Composite Link are requirements to advertise capacity available within configured ranges of delay within a given composite link and the support the ability to place an LSP only on component links that meeting that LSP's delay requirements.

The Composite Link requirements to accommodate delay sensitive applications are analogous to diffserv requirements to accommodate applications requiring higher quality of service on the same infrastructure as applications with less demanding requirements. The ability to share capacity with less demanding applications, with best effort applications being the least demanding, can greatly reduce the cost of delivering service to the more demanding applications.

5. Large Volume of IP and LDP Traffic

IP and LDP do not support traffic engineering. Both make use of a shortest (lowest routing metric) path, with an option to use equal cost multipath (ECMP). Note that though ECMP is prohibited in LDP specifications, it is widely implemented. Where implemented for LDP, ECMP is generally disabled by default for standards compliance, but often enabled in LDP deployments.

Without traffic engineering capability, there must be sufficient capacity to accommodate the IP and LDP traffic. If not, persistent queuing delay and loss will occur. Unlike RSVP-TE, a subset of traffic cannot be routed using constraint based routing to avoid a congested portion of an infrastructure.

In existing networks which accomodate IP and/or LDP with RSVP-TE, either the IP and LDP can be carried over RSVP-TE, or where the traffic contribution of IP and LDP is small, IP and LDP can be carried native and the effect on RSVP-TE can be ignored. Ignoring the traffic contribution of IP is certainly valid on high capacity networks where native IP is used primarily for control and network management and customer IP is carried within RSVP-TE.

Where it is desireable to carry native IP and/or LDP and IP and/or LDP traffic volumes are not negligible, RSVP-TE needs improvement. The enhancement offered by Composite Link is an ability to measure the IP and LDP, filter the measurements, and reduce the capacity available to RSVP-TE to avoid congestion. The treatment given to the IP or LDP traffic is similar to the treatment when using the "auto-bandwidth" feature in some RSVP-TE implementations on that same traffic, and giving a higher priority (numerically lower setup priority and holding priority value) to the "auto-bandwidth" LSP. The difference is that the measurement is made at each hop and the reduction in advertised bandwidth is made more directly.

6. Composite Link and Packet Ordering

A strong motivation for Composite Link is the need to provide LSP capacity in IP backbones that exceeds the capacity of single wavelengths provided by transport equipment and exceeds the practical capacity limits acheivable through inverse multiplexing. Appendix C describes characteristics and limitations of transport systems today. Section 2 defines the terms "classic multipath" and "classic link bundling" used in this section.

For purpose of discussion, consider two very large cities, city A and city Z. For example, in the US high traffic cities might be New York and Los Angeles and in Europe high traffic cities might be London and Amsterdam. Two other high volume cities, city B and city Y may share common provider core network infrastructure. Using the same examples, the city B and Y may Washington DC and San Francisco or Paris and Stockholm. In the US, the common infrastructure may span Denver, Chicago, Detroit, and Cleveland. Other major traffic contributors on either US coast include Boston, northern Virginia on the east coast, and Seattle, and San Diego on the west coast. The capacity of IP/MPLS links within the shared infrastructure, for

example city to city links in the Denver, Chicago, Detroit, and Cleveland path in the US example, have capacities for most of the 2000s decade that greatly exceeded single circuits available in transport networks.

For a case with four large traffic sources on either side of the shared infrastructure, up to sixteen core city to core city traffic flows in excess of transport circuit capacity may be accommodated on the shared infrastructure.

Today the most common IP/MPLS core network design makes use of very large links which consist of many smaller component links, but use classic multipath techniques rather than classic link bundling or Composite Link. A component link typically corresponds to the largest circuit that the transport system is capable of providing (or the largest cost effective circuit). IP source and destination address hashing is used to distribute flows across the set of component links as described in Appendix B.3.

Classic multipath can handle large LSP up to the total capacity of the multipath (within limits, see Appendix B.2). A disadvantage of classic multipath is the reordering among traffic within a given core city to core city LSP. While there is no reordering within any microflow and therefore no customer visible issue, MPLS-TP cannot be used across an infrastructure where classic multipath is in use, except within pseudowires.

These capacity issues force the use of classic multipath today. Classic multipath excludes a direct use of MPLS-TP. The desire for OAM, offered by MPLS-TP, is in conflict with the use of classic multipath. There are a number of alternatives that satisfy both requirements. Some alternatives are described below.

MPLS-TP in network edges only

A simple approach which requires no change to the core is to disallow MPLS-TP across the core unless carried within a pseudowire (PW). MPLS-TP may be used within edge domains where classic multipath is not used. PW may be signaled end to end using single segment PW (SS-PW), or stitched across domains using multisegment PW (MS-PW). The PW and anything carried within the PW may use OAM as long as fat-PW [RFC6391] load splitting is not used by the PW.

Composite Link at core LSP ingress/egress

The interior of the core network may use classic link bundling, with the limitation that no LSP can exceed the capacity of a

single circuit. Larger non-MPLS-TP LSP can be configured using multiple ingress to egress component MPLS-TP LSP. This can be accomplished using existing IP source and destination address hashing configured at LSP ingress and egress, or using Composite Link configured at ingress and egress. Each component LSP, if constrained to be no larger than the capacity of a single circuit, can make use of MPLS-TP and offer OAM for all top level LSP across the core.

MPLS-TP as a MPLS client

A third approach involves modifying the behavior of LSR in the interior of the network core, such that MPLS-TP can be used on a subset of LSP, where the capacity of any one LSP within that MPLS-TP subset of LSP is not larger than the capacity of a single circuit. This requirement is accommodated through a combination of signaling to indicate LSP for which traffic splitting needs to be constrained, the ability to constrain the depth of the label stack over which traffic splitting can be applied on a per LSP basis, and the ability to constrain the use of IP addresses below the label stack for traffic splitting also on a per LSP basis.

The above list of alternatives allow packet ordering within an LSP to be maintained in some circumstances and allow very large LSP capacities. Each of these alternatives are discussed further in the following subsections.

6.1. MPLS-TP in network edges only

Classic MPLS link bundling is defined in [RFC4201] and has existed since early in the 2000s decade. Classic MPLS link bundling place any given LSP entirely on a single component link. Classic MPLS link bundling is not in widespread use as the means to accomodate large link capacities in core networks due to the simplicity and better multiplexing gain, and therefore lower network cost of classic multipath.

If MPLS-TP OAM capability in the IP/MPLS network core LSP is not required, then there is no need to change existing network designs which use classic multipath and both label stack and IP source and destination address based hashing as a basis for load splitting.

If MPLS-TP is needed for a subset of LSP, then those LSP can be carried within pseudowires. The pseudowires adds a thin layer of encapsulation and therefore a small overhead. If only a subset of LSP need MPLS-TP OAM, then some LSP must make use of the pseudowires and other LSP avoid them. A straihtforward way to accomplish this is with administrative attributes [RFC3209].

6.2. Composite Link at core LSP ingress/egress

Composite Link can be configured only for large LSP that are made of smaller MPLS-TP component LSP. This approach is capable of supporting MPLS-TP OAM over the entire set of component link LSP and therefore the entire set of top level LSP traversing the core.

There are two primary disadvantage of this approach. One is the number of top level LSP traversing the core can be dramatically increased. The other disadvantage is the loss of multiplexing gain that results from use of classic link bundling within the interior of the core network.

If component LSP use MPLS-TP, then no component LSP can exceed the capacity of a single circuit. For a given composite LSP there can either be a number of equal capacity component LSP or some number of full capacity component links plus one LSP carrying the excess. For example, a 350 Gb/s composite LSP over a 100 Gb/s infrastructure may use five 70 Gb/s component LSP or three 100 Gb/s LSP plus one 50 Gb/s LSP. Classic MPLS link bundling is needed to support MPLS-TP and suffers from a bin packing problem even if LSP traffic is completely predictable, which it never is in practice.

The common means of setting composite link bandwidth parameters uses long term statistical measures. For example, many providers base their LSP bandwidth parameters on the 95th percentile of carried traffic as measured over a one week period. It is common to add 10-30% to the 95th percentile value measured over the prior week and adjust bandwidth parameters of LSP weekly. It is also possible to measure traffic flow at the LSR and adjust bandwidth parameters somewhat more dynamically. This is less common in deployments and where deployed, make use of filtering to track very long term trends in traffic levels. In either case, short term variation of traffic levels relative to signaled LSP capacity are common. Allowing a large overallocation of LSP bandwidth parameters (ie: adding 30% or more) avoids overutilization of any given LSP, but increases unused network capacity and increases network cost. Allowing a small overallocation of LSP bandwidth parameters (ie: 10-20% or less) results in both underutilization and overutilization but statistically results in a total utilization within the core that is under capacity most or all of the time.

The classic multipath solution accommodates the situation in which some composite LSP are underutilizing their signaled capacity and others are overutilizing their capacity with the need for far less unused network capacity to accommodate variation in actual traffic levels. If the actual traffic levels of LSP can be described by a probability distribution, the variation of the sum of LSP is less

than the variation of any given LSP for all but a constant traffic level (where the variation of the sum and the components are both zero).

There are two situations which can motivate the use of this approach. This design is favored if the provider values MPLS-TP OAM across the core more than efficiency (or is unaware of the efficiency issue). This design can also make sense if transport equipment or very low cost core LSR are available which support only classic link bundling and regardless of loss of multiplexing gain, are more cost effective at carrying transit traffic than using equipment which supports IP source and destination address hashing.

6.3. MPLS-TP as a MPLS client

Accommodating MPLS-TP as a MPLS client requires a small change to forwarding behavior and is therefore most applicable to major network overbuilds or new deployments. The change to forwarding is an ability to limit the depth of MPLS labels used in hashing on the label stack on a per LSP basis. Some existing hardware, particularly microprogrammed hardware, may be able to accommodate this forwarding change. Providing support in new hardware is not difficult, a much smaller change than, for example, changes required to disable PHP in an environment where LSP hierarchy is used.

The advantage of this approach is an ability to accommodate MPLS-TP as a client LSP but retain the high multiplexing gain and therefore efficiency and low network cost of a pure MPLS deployment. The disadvantage is the need for a small change in forwarding.

7. Security Considerations

This document is a use cases document. Existing protocols are referenced such as MPLS. Existing techniques such as MPLS link bundling and multipath techniques are referenced. These protocols and techniques are documented elsewhere and contain security considerations which are unchanged by this document.

This document also describes use cases for Composite Link, which is a work-in-progress. Composite Link requirements are defined in [I-D.ietf-rtgwg-cl-requirement]. [I-D.so-yong-rtgwg-cl-framework] defines a framework for Composite Link. Composite Link bears many similarities to MPLS link bundling and multipath techniques used with MPLS. Additional security considerations, if any, beyond those already identified for MPLS, MPLS link bundling and multipath techniques, will be documented in the framework document if specific to the overall framework of Composite Link, or in protocol extensions

if specific to a given protocol extension defined later to support Composite Link.

8. Acknowledgments

Authors would like to thank [no one so far] for their reviews and great suggestions.

In the interest of full disclosure of affiliation and in the interest of acknowledging sponsorship, past affiliations of authors are noted. Much of the work done by Ning So occurred while Ning was at Verizon. Much of the work done by Curtis Villamizar occurred while at Infinera. Infinera continues to sponsor this work on a consulting basis.

9. References

9.1. Normative References

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

9.2. Informative References

[I-D.ietf-rtgwg-cl-requirement]
Villamizar, C., McDysan, D., Ning, S., Malis, A., and L. Yong, "Requirements for MPLS Over a Composite Link", draft-ietf-rtgwg-cl-requirement-04 (work in progress), March 2011.

[I-D.so-yong-rtgwg-cl-framework]
So, N., Malis, A., McDysan, D., Yong, L., Villamizar, C., and T. Li, "Composite Link Framework in Multi Protocol Label Switching (MPLS)", draft-so-yong-rtgwg-cl-framework-04 (work in progress), June 2011.

[IEEE-802.1AX]
IEEE Standards Association, "IEEE Std 802.1AX-2008 IEEE Standard for Local and Metropolitan Area Networks - Link Aggregation", 2006, <<http://standards.ieee.org/getieee802/download/802.1AX-2008.pdf>>.

[ITU-T.G.694.2]
ITU-T, "Spectral grids for WDM applications: CWDM wavelength grid", 2003,

<<http://www.itu.int/rec/T-REC-G.694.2-200312-I>>.

[ITU-T.G.800]

ITU-T, "Unified functional architecture of transport networks", 2007,
<<http://www.itu.int/rec/T-REC-G.800-200709-I>>.

[ITU-T.Y.1540]

ITU-T, "Internet protocol data communication service - IP packet transfer and availability performance parameters", 2007, <<http://www.itu.int/rec/T-REC-Y.1540/en>>.

[ITU-T.Y.1541]

ITU-T, "Network performance objectives for IP-based services", 2006, <<http://www.itu.int/rec/T-REC-Y.1541/en>>.

[RFC1717] Sklower, K., Lloyd, B., McGregor, G., and D. Carr, "The PPP Multilink Protocol (MP)", RFC 1717, November 1994.

[RFC2475] Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., and W. Weiss, "An Architecture for Differentiated Services", RFC 2475, December 1998.

[RFC2597] Heinanen, J., Baker, F., Weiss, W., and J. Wroclawski, "Assured Forwarding PHB Group", RFC 2597, June 1999.

[RFC2615] Malis, A. and W. Simpson, "PPP over SONET/SDH", RFC 2615, June 1999.

[RFC2991] Thaler, D. and C. Hopps, "Multipath Issues in Unicast and Multicast Next-Hop Selection", RFC 2991, November 2000.

[RFC2992] Hopps, C., "Analysis of an Equal-Cost Multi-Path Algorithm", RFC 2992, November 2000.

[RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, December 2001.

[RFC3260] Grossman, D., "New Terminology and Clarifications for Diffserv", RFC 3260, April 2002.

[RFC3809] Nagarajan, A., "Generic Requirements for Provider Provisioned Virtual Private Networks (PPVPN)", RFC 3809, June 2004.

[RFC3945] Mannie, E., "Generalized Multi-Protocol Label Switching (GMPLS) Architecture", RFC 3945, October 2004.

- [RFC4201] Kompella, K., Rekhter, Y., and L. Berger, "Link Bundling in MPLS Traffic Engineering (TE)", RFC 4201, October 2005.
- [RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.
- [RFC4385] Bryant, S., Swallow, G., Martini, L., and D. McPherson, "Pseudowire Emulation Edge-to-Edge (PWE3) Control Word for Use over an MPLS PSN", RFC 4385, February 2006.
- [RFC4928] Swallow, G., Bryant, S., and L. Andersson, "Avoiding Equal Cost Multipath Treatment in MPLS Networks", BCP 128, RFC 4928, June 2007.
- [RFC5036] Andersson, L., Minei, I., and B. Thomas, "LDP Specification", RFC 5036, October 2007.
- [RFC5586] Bocci, M., Vigoureux, M., and S. Bryant, "MPLS Generic Associated Channel", RFC 5586, June 2009.
- [RFC6391] Bryant, S., Filsfils, C., Drafz, U., Kompella, V., Regan, J., and S. Amante, "Flow-Aware Transport of Pseudowires over an MPLS Packet Switched Network", RFC 6391, November 2011.

Appendix A. More Details on Existing Network Operator Practices and Protocol Usage

Often, network operators have a contractual Service Level Agreement (SLA) with customers for services that are comprised of numerical values for performance measures, principally availability, latency, delay variation. Additionally, network operators may have Service Level Sepcification (SLS) that is for internal use by the operator. See [ITU-T.Y.1540], [ITU-T.Y.1541], RFC3809, Section 4.9 [RFC3809] for examples of the form of such SLA and SLS specifications. In this document we use the term Network Performance Objective (NPO) as defined in section 5 of [ITU-T.Y.1541] since the SLA and SLS measures have network operator and service specific implications. Note that the numerical NPO values of Y.1540 and Y.1541 span multiple networks and may be looser than network operator SLA or SLS objectives. Applications and acceptable user experience have an important relationship to these performance parameters.

Consider latency as an example. In some cases, minimizing latency relates directly to the best customer experience (e.g., in TCP closer is faster). In other cases, user experience is relatively insensitive to latency, up to a specific limit at which point user

perception of quality degrades significantly (e.g., interactive human voice and multimedia conferencing). A number of NPOs have a bound on point-point latency, and as long as this bound is met, the NPO is met -- decreasing the latency is not necessary. In some NPOs, if the specified latency is not met, the user considers the service as unavailable. An unprotected LSP can be manually provisioned on a set of to meet this type of NPO, but this lowers availability since an alternate route that meets the latency NPO cannot be determined.

Historically, when an IP/MPLS network was operated over a lower layer circuit switched network (e.g., SONET rings), a change in latency caused by the lower layer network (e.g., due to a maintenance action or failure) this was not known to the MPLS network. This resulted in latency affecting end user experience, sometimes violating NPOs or resulting in user complaints.

A response to this problem was to provision IP/MPLS networks over unprotected circuits and set the metric and/or TE-metric proportional to latency. This resulted in traffic being directed over the least latency path, even if this was not needed to meet an NPO or meet user experience objectives. This results in reduced flexibility and increased cost for network operators. Using lower layer networks to provide restoration and grooming is expected to be more efficient, but the inability to communicate performance parameters, in particular latency, from the lower layer network to the higher layer network is an important problem to be solved before this can be done.

Latency NPOs for point-to-point services are often tied closely to geographic locations, while latency for multipoint services may be based upon a worst case within a region.

Section 7 of [ITU-T.Y.1540] defines availability for an IP service in terms of loss exceeding a threshold for a period on the order of 5 minutes. However, the timeframes for restoration (i.e., as implemented by pre-determined protection, convergence of routing protocols and/or signaling) for services range from on the order of 100 ms or less (e.g., for VPWS to emulate classical SDH/SONET protection switching), to several minutes (e.g., to allow BGP to reconverge for L3VPN) and may differ among the set of customers within a single service.

The presence of only three Traffic Class (TC) bits (previously known as EXP bits) in the MPLS shim header is limiting when a network operator needs to support QoS classes for multiple services (e.g., L2VPN VPWS, VPLS, L3VPN and Internet), each of which has a set of QoS classes that need to be supported. In some cases one bit is used to indicate conformance to some ingress traffic classification, leaving only two bits for indicating the service QoS classes. The approach

that has been taken is to aggregate these QoS classes into similar sets on LER-LSR and LSR-LSR links.

Labeled LSPs and use of link layer encapsulation have been standardized in order to provide a means to meet these needs.

The IP DSCP cannot be used for flow identification since RFC 4301 Section 5.5 [RFC4301] requires Diffserv transparency, and in general network operators do not rely on the DSCP of Internet packets. In addition, the use of IP DSCP for flow identification is incompatible with Assured Forwarding services [RFC2597] or any other service which may use more than one DSCP code point to carry traffic for a given microflow.

A label is pushed onto Internet packets when they are carried along with L2/L3VPN packets on the same link or lower layer network provides a mean to distinguish between the QoS class for these packets.

Operating an MPLS-TE network involves a different paradigm from operating an IGP metric-based LDP signaled MPLS network. The multipoint-to-point LDP signaled MPLS LSPs occur automatically, and balancing across parallel links occurs if the IGP metrics are set "equally" (with equality a locally definable relation).

Traffic is typically comprised of a few large (some very large) flows and many small flows. In some cases, separate LSPs are established for very large flows. This can occur even if the IP header information is inspected by a LSR, for example an IPsec tunnel that carries a large amount of traffic. An important example of large flows is that of a L2/L3 VPN customer who has an access line bandwidth comparable to a client-client composite link bandwidth -- there could be flows that are on the order of the access line bandwidth.

Appendix B. Existing Multipath Standards and Techniques

Today the requirement to handle large aggregations of traffic, much larger than a single component link, can be handled by a number of techniques which we will collectively call multipath. Multipath applied to parallel links between the same set of nodes includes Ethernet Link Aggregation [IEEE-802.1AX], link bundling [RFC4201], or other aggregation techniques some of which may be vendor specific. Multipath applied to diverse paths rather than parallel links includes Equal Cost MultiPath (ECMP) as applied to OSPF, ISIS, or even BGP, and equal cost LSP, as described in Appendix B.4. Various multipath techniques have strengths and weaknesses.

the term Composite Link is more general than terms such as Link Aggregation which is generally considered to be specific to Ethernet and its use here is consistent with the broad definition in [ITU-T.G.800]. The term multipath excludes inverse multiplexing and refers to techniques which only solve the problem of large aggregations of traffic, without addressing the other requirements outlined in this document, particularly those described in Section 4 and Section 5.

B.1. Common Multipath Load Splitting Techniques

Identical load balancing techniques are used for multipath both over parallel links and over diverse paths.

Large aggregates of IP traffic do not provide explicit signaling to indicate the expected traffic loads. Large aggregates of MPLS traffic are carried in MPLS tunnels supported by MPLS LSP. LSP which are signaled using RSVP-TE extensions do provide explicit signaling which includes the expected traffic load for the aggregate. LSP which are signaled using LDP do not provide an expected traffic load.

MPLS LSP may contain other MPLS LSP arranged hierarchically. When an MPLS LSR serves as a midpoint LSR in an LSP carrying other LSP as payload, there is no signaling associated with these inner LSP. Therefore even when using RSVP-TE signaling there may be insufficient information provided by signaling to adequately distribute load based solely on signaling.

Generally a set of label stack entries that is unique across the ordered set of label numbers in the label stack can safely be assumed to contain a group of flows. The reordering of traffic can therefore be considered to be acceptable unless reordering occurs within traffic containing a common unique set of label stack entries. Existing load splitting techniques take advantage of this property in addition to looking beyond the bottom of the label stack and determining if the payload is IPv4 or IPv6 to load balance traffic accordingly.

MPLS-TP OAM violates the assumption that it is safe to reorder traffic within an LSP. If MPLS-TP OAM is to be accommodated, then existing multipath techniques must be modified. Such modifications are outside the scope of this document.

For example, a large aggregate of IP traffic may be subdivided into a large number of groups of flows using a hash on the IP source and destination addresses. This is as described in [RFC2475] and clarified in [RFC3260]. For MPLS traffic carrying IP, a similar hash can be performed on the set of labels in the label stack. These

techniques are both examples of means to subdivide traffic into groups of flows for the purpose of load balancing traffic across aggregated link capacity. The means of identifying a set of flows should not be confused with the definition of a flow.

Discussion of whether a hash based approach provides a sufficiently even load balance using any particular hashing algorithm or method of distributing traffic across a set of component links is outside of the scope of this document.

The current load balancing techniques are referenced in [RFC4385] and [RFC4928]. The use of three hash based approaches are described in [RFC2991] and [RFC2992]. A mechanism to identify flows within PW is described in [RFC6391]. The use of hash based approaches is mentioned as an example of an existing set of techniques to distribute traffic over a set of component links. Other techniques are not precluded.

B.2. Simple and Adaptive Load Balancing Multipath

Simple multipath generally relies on the mathematical probability that given a very large number of small microflows, these microflows will tend to be distributed evenly across a hash space. Early very simple multipath implementations assumed that all component links are of equal capacity and perform a modulo operation across the hashed value. An alternate simple multipath technique uses a table generally with a power of two size, and distributes the table entries proportionally among component links according to the capacity of each component link.

Simple load balancing works well if there are a very large number of small microflows (i.e., microflow rate is much less than component link capacity). However, the case where there are even a few large microflows is not handled well by simple load balancing.

An adaptive load balancing multipath technique is one where the traffic bound to each component link is measured and the load split is adjusted accordingly. As long as the adjustment is done within a single network element, then no protocol extensions are required and there are no interoperability issues.

Note that if the load balancing algorithm and/or its parameters is adjusted, then packets in some flows may be briefly delivered out of sequence, however in practice such adjustments can be made very infrequent.

B.3. Traffic Split over Parallel Links

The load splitting techniques defined in Appendix B.1 and Appendix B.2 are both used in splitting traffic over parallel links between the same pair of nodes. The best known technique, though far from being the first, is Ethernet Link Aggregation [IEEE-802.1AX]. This same technique had been applied much earlier using OSPF or ISIS Equal Cost MultiPath (ECMP) over parallel links between the same nodes. Multilink PPP [RFC1717] uses a technique that provides inverse multiplexing, however a number of vendors had provided proprietary extensions to PPP over SONET/SDH [RFC2615] that predated Ethernet Link Aggregation but are no longer used.

Link bundling [RFC4201] provides yet another means of handling parallel LSP. RFC4201 explicitly allow a special value of all ones to indicate a split across all members of the bundle. This "all ones" component link is signaled in the MPLS RESV to indicate that the link bundle is making use of classic multipath techniques.

B.4. Traffic Split over Multiple Paths

OSPF or ISIS Equal Cost MultiPath (ECMP) is a well known form of traffic split over multiple paths that may traverse intermediate nodes. ECMP is often incorrectly equated to only this case, and multipath over multiple diverse paths is often incorrectly equated to ECMP.

Many implementations are able to create more than one LSP between a pair of nodes, where these LSP are routed diversely to better make use of available capacity. The load on these LSP can be distributed proportionally to the reserved bandwidth of the LSP. These multiple LSP may be advertised as a single PSC FA and any LSP making use of the FA may be split over these multiple LSP.

Link bundling [RFC4201] component links may themselves be LSP. When this technique is used, any LSP which specifies the link bundle may be split across the multiple paths of the LSP that comprise the bundle.

Appendix C. Characteristics of Transport in Core Networks

The characteristics of primary interest are the capacity of a single circuit and the use of wave division multiplexing (WDM) to provide a large number of parallel circuits.

Wave division multiplexing (WDM) supports multiple independent channels (independent ignoring crosstalk noise) at slightly different

wavelengths of light, multiplexed onto a single fiber. Typical in the early 2000s was 40 wavelengths of 10 Gb/s capacity per wavelength. These wavelengths are in the C-band range, which is about 1530-1565 nm, though some work has been done using the L-band 1565-1625 nm.

The C-band has been carved up using a 100 GHz spacing from 191.7 THz to 196.1 THz by [ITU-T.G.694.2]. This yields 44 channels. If the outermost channels are not used, due to poorer transmission characteristics, then typically 40 are used. For practical reasons, a 50 GHz or 25 GHz spacing is used by more recent equipment, yielding 80 or 160 channels in practice.

The early optical modulation techniques used within a single channel yielded 2.5Gb/s and 10 Gb/s capacity per channel. As modulation techniques have improved 40 Gb/s and 100 Gb/s per channel have been achieved.

The 40 channels of 10 Gb/s common in the mid 2000s yields a total of 400 Gb/s. Tighter spacing and better modulations are yielding up to 8 Tb/s or more in more recent systems.

Over the optical is an electrical encoding. In the 1990s this was typically Synchronous Optical Networking (SONET) or Synchronous Digital Hierarchy (SDH), with a maximum defined circuit capacity of 40 Gb/s (OC-768), though the 10 Gb/s OC-192 is more common. More recently the low level electrical encoding has been Optical Transport Network (OTN) defined by ITU-T. OTN currently defines circuit capacities up to a nominal 100 Gb/s (ODU4). Both SONET/SDH and OTN make use of time division multiplexing (TDM) where the a higher capacity circuit such as a 100 Gb/s ODU4 in OTN may be subdivided into lower fixed capacity circuits such as ten 10 Gb/s ODU2.

In the 1990s, all IP and later IP/MPLS networks either used a fraction of maximum circuit capacity, or at most the full circuit capacity toward the end of the decade, when full circuit capacity was 2.5 Gb/s or 10 Gb/s. Beyond 2000, the TDM circuit multiplexing capability of SONET/SDH or OTN was rarely used.

Early in the 2000s both transport equipment and core LSR offered 40 Gb/s SONET OC-768. However 10 Gb/s transport equipment was predominantly deployed throughout the decade, partially because LSR 10GbE ports were far more cost effective than either OC-192 or OC-768 and became practical in the second half of the decade.

Entering the 2010 decade, LSR 40GbE and 100GbE are expected to become widely available and cost effective. Slightly preceding this transport equipment making use of 40 Gb/s and 100 Gb/s modulations

are becoming available. This transport equipment is capable of carrying 40 Gb/s ODU3 and 100 Gb/s ODU4 circuits.

Early in the 2000s decade IP/MPLS core networks were making use of single 10 Gb/s circuits. Capacity grew quickly in the first half of the decade but more IP/MPLS core networks had only a small number of IP/MPLS links requiring 4-8 parallel 10 Gb/s circuits. However, the use of multipath was necessary, was deemed the simplest and most cost effective alternative, and became thoroughly entrenched. By the end of the 2000s decade nearly all major IP/MPLS core service provider networks and a few content provider networks had IP/MPLS links which exceeded 100 Gb/s, long before 40GbE was available and 40 Gb/s transport in widespread use.

It is less clear when IP/MPLS LSP exceeded 10 Gb/s, 40 Gb/s, and 100 Gb/s. By 2010, many service providers have LSP in excess of 100 Gb/s, but few are willing to disclose how many LSP have reached this capacity.

At the time of writing 40GbE and 100GbE LSR products are being evaluated by service providers and content providers and are in use in network trials. The cost of components required to deliver 100 GbE products remains high making these products less cost effective. This is expected to change within years.

The important point is that IP/MPLS core network links have long ago exceeded 100 Gb/s and a small number of IP/MPLS LSP exceed 100 Gb/s. By the time 100 Gb/s circuits are widely deployed, IP/MPLS core network links are likely to exceed 1 Tb/s and many IP/MPLS LSP capacities are likely to exceed 100 Gb/s. Therefore multipath techniques are likely here to stay.

Authors' Addresses

So Ning
Tata Communications

Email: ning.so@tatacommunications.com

Andrew Malis
Verizon
117 West St.
Waltham, MA 02451

Phone: +1 781-466-2362
Email: andrew.g.malis@verizon.com

Dave McDysan
Verizon
22001 Loudoun County PKWY
Ashburn, VA 20147

Email: dave.mcdysan@verizon.com

Lucy Yong
Huawei USA
5340 Legacy Dr.
Plano, TX 75025

Phone: +1 469-277-5837
Email: lucy.yong@huawei.com

Curtis Villamizar
Outer Cape Cod Network Consulting

Email: curtis@occnc.com

Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: July 11, 2013

B. Zhang
J. Shi
The University of Arizona
J. Dong
M. Zhang
Huawei
Januray 10, 2013

Power-aware Routing and Traffic Engineering: Requirements, Approaches,
and Issues
draft-zhang-greenet-01

Abstract

Energy consumption of network infrastructures is rising fast. There are emerging needs for power-aware routing and traffic engineering, which adjust routing paths to help reduce power consumption network-wide. This document gives a high-level analysis on the basic requirements, approaches, and potential issues in power-aware routing and traffic engineering.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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."

This Internet-Draft will expire on July 11, 2013.

Copyright Notice

Copyright (c) 2013 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 Simplified BSD License.

Table of Contents

| | |
|--------------------------------------|----|
| 1. Introduction | 3 |
| 2. Requirements | 4 |
| 3. Approaches | 4 |
| 4. Issues | 6 |
| 4.1. Hardware Support | 6 |
| 4.2. Software Support | 8 |
| 4.3. Impacts on Protocols | 9 |
| 4.4. Network Monitoring | 11 |
| 5. Summary | 12 |
| 6. IANA Considerations | 12 |
| 7. Security Considerations | 12 |
| 8. Informative References | 12 |
| Authors' Addresses | 13 |

1. Introduction

Driven by exponential growth of Internet traffic, networks worldwide are expanding their infrastructures at a fast pace by deploying more high-capacity, power-hungry routers, which also leads to increasing energy consumption. Besides operational costs and environmental impacts, the ever-increasing energy consumption has become a limiting factor to long-term growth of network infrastructure, due to challenges in power delivery to and heat removal from router components as well as the hosting facilities [Gupta03] [Epps06].

Today's ISP networks have redundant routers and links, over-provisioned link capacity, and load-balancing traffic engineering. As a result, routers and links operate at full capacity all the time with low average usage, typically less than 40% of link utilization. This practice makes networks resilient to traffic spikes and component failures, but also makes networks far from energy efficient. Though advances in hardware design have made individual routers more energy efficient over the years, there is still has a long way to go before routers become energy-proportional. Recently researchers have started to look beyond a single router or linecard for network-wide solutions towards energy proportionality. Power-aware routing and traffic engineering has been proposed to improve network's energy efficiency, for example, aggregating traffic onto a subset of links and putting the other links with no traffic into sleep. As demonstrated in several research works, this approach has the potential to save a significant amount of energy [GreenTE] [Nedevschi08] [Chabarek08]. Designing a practical protocol, however, has been challenging, because making routing protocols power-aware brings fundamental changes to the routing system and the entire network, thus it is a complicated task with involvement of hardware support, protocol design and operations, and network monitoring.

This document gives a high-level analysis on power-aware routing and traffic engineering, including solution requirements, existing approaches, and potential issues. Power-aware routing and traffic engineering exploits the over-provisioned feature of networks, so there will be some impact on network performance and resilience. In order to save energy without impacting network performance and resilience too much, certain requirements should be met. When a power-aware approach is implemented, new issues may arise, and should be addressed to make the solution practical. While there are many aspects of energy efficient networks, this document focuses on intra-domain routing within a single ISP.

2. Requirements

The high-level idea of power-aware routing or traffic engineering is to adjust routing paths based on traffic level. When traffic level is high, use more links to carry the traffic; when traffic level is low, merge traffic to a small number of links so that other links can be put to sleep or reduce rate in order to save power. The fundamental requirement for any energy-efficient network solution is to save power without negative impacts on network operations, which can be broken down as follows.

- o The network should retain enough resiliency against node and/or link failures.
- o The network should have enough spare standby capacity or be able to react quickly enough to traffic spikes in order to minimize packet losses due to links/routers being in low power states.
- o QoS metrics such as end-to-end delay should be kept at a desired level.
- o The operation of other protocols should not be interrupted, or at least other protocols should be able to adapt without being broken after links/routers change their power states.

While this document focuses on routing and traffic engineering, it requires support from underlying hardware and system for energy management capability, which is the topic of the IETF Energy Management Working Group [EMAN-WG]

3. Approaches

In the last couple of years a number of power-aware protocols have been proposed in research. Instead of listing them individually, here we categorize the solutions along three different dimensions.

Link Sleep vs. Rate Adaptation

Sleeping and rate adaptation are two major ways to save energy in computer systems. Many hardware, including line cards and chassis, consumes a significant amount of power when they stand by without doing any actual work. When put into sleep mode, they will consume only a little power. Thus putting an idle component to sleep is a common way to save energy. If there is a need to use this component, it can be waken up and become usable after a transition time. The longer a component is in sleep mode, the more power saved. A power-aware protocol adjusts routing paths to increase the sleep time for

certain links in the network.

A network interface often supports multiple data rates. Operating at a lower data rate usually consumes less energy, though the actual rate-power curve varies from device to device. Rate-adaptation-based approaches operate interfaces at lower data rates when the traffic demand is low and increase the data rate when traffic demand is high. Thus the routers can save power during low utilization period.

These two approaches are also related in the case of "bundled links" [Fisher10]. A bundled link is a virtual link comprised of multiple physical links. A sleep-based approach can put some physical links into sleep to save power, which is same as conducting rate adaptation on the virtual link with adjustment unit of a physical link.

Configured vs. Adaptive

The key in power-aware routing and traffic engineering is to adjust routing paths in response to traffic changes, so that the power state of routers (or router components) will also change accordingly to achieve energy saving. Different approaches differ at the granularity of the adjustment.

Some approaches take the long-term traffic average as input, and output a routing configuration that is applied to the network regardless of short-term traffic variation. This is mostly useful when network traffic exhibits a stable, clear pattern, e.g., diurnal pattern where traffic is high during work hours and low during off hours. It can only exploit the target traffic pattern; it cannot react dynamically to short-term traffic changes to either save energy (by putting links to sleep) or avoid congestion (by waking links up), but the design and implementation should be simple.

Another type of approaches is to adapt to traffic changes dynamically on much smaller time granularity. This approach may be able to save more energy and have better performance because it is more responsive, but the design and implementation usually are more complicated. This approach needs to continuously collect traffic data in order to adjust routing dynamically. The adjustment may be done periodically or whenever significant traffic changes are observed.

Distributed vs. Centralized

In distributed solutions, routers make power-aware adjustment decisions, such as link sleep/wake-up and rate increase/decrease, locally without a central controller. These routers need to exchange information in order to achieve consistent network states.

Distributed approach fits the Internet operation model well but its design is the most challenging. Traditional routing does not respond to traffic variation while power-aware routing does, and it needs to do so without causing loops or congestions.

In centralized solutions, a controller computes the routing paths considering the network topology and traffic demand, and informs routers how to adjust their routing paths. A centralized server usually has more complete information, more computation power, and more memory and storage than routers, thus it may make better decisions than distributed approach. The server locates in the network NOC and can be backed up by server replicas. Nevertheless, this approach requires high reliability of the server.

Both distributed and centralized solutions may find their places in ISP networks. For example, centralized solution can be integrated into the Path Computation Element (PCE) framework [PCE-WG]. There can also be hybrid designs, e.g., using a centralized solution based on long-term traffic pattern, and distributed mechanisms to handle short-term traffic variations.

4. Issues

4.1. Hardware Support

In order to save power, routers and switches should support low power states, and make available control primitives to enter or leave low power states. To reduce the impact on network performance, routers and switches should have the ability to change power states quickly. These are the hardware support needed by power-aware protocols.

Sleeping State

Most sleep-based approaches require routers and switches, or a component of them such as a line card, to support sleeping state. While most components can go to sleep very quickly, they also need to be able to wake up quickly. Besides, entering and leaving sleeping state often incurs extra energy draw, which need to be kept small. Different designs may have different requirements of the transition time between power states. In uncoordinated sleeping approach, upstream routers intentionally buffer packets for a very short period of time to allow downstream routers longer sleep time. This approach can only allow a component to sleep for a few milliseconds, otherwise the buffering may cause too much extra delay. Hence this approach requires a very short transition time and low penalty power. In coordinated sleeping approaches, where routers coordinate on which paths to use and when to put links to sleep, a component usually can

sleep much longer, for seconds, minutes or even longer. Therefore their requirement on transition time and power is more relaxed.

Common energy management scheme at the individual component such as line card is sleep-on-idle (SoI) and wake-on-arrival (WoA). When a link is idle for a short period of time, it goes into sleep; when a packet arrives, it wakes up. Power-aware protocols manipulate traffic paths so that some links will have much longer idle time than default routing.

The hardware is also expected to minimize potential packet loss during the transition between power states. Especially in WoA, the first packet is susceptible to loss. The two ends of the link can coordinate, e.g., one end sends a dummy packet to the other end to inform about the link wakeup, or if they don't coordinate, the receiver end should have the capability to buffer incoming packets before the interface wakes up to process these packets.

Multiple Data Rates

CMOS based silicon supports Dynamic Voltage Scaling (DVS), so clocking an interface at a lower frequency, and operating at lower data rate can save considerable amount of energy. This calls for a need for router interfaces to support multiple data rates. If an interface could support more data rates and incur low penalty power on a change, there are more opportunity to save energy. Furthermore, it will also help if an interface supports different sending and receiving data rates.

The transition between different data rates needs be quick and on-the-fly. Most Ethernet cards supports auto-negotiation of data rates, which happens when a cable is plugged in and takes hundreds of milliseconds. Auto-negotiation is not suitable for changing data rate to save energy, because buffer would be filled up during the negotiation period and leads to packet loss. A fast mechanism for initiating and agreeing upon a link data rate change is necessary.

Electrical Damage

Many electronic devices are not designed to be turned on and off frequently. When a device is waken up, the in-rush current may damage or destroy a component and related circuits. Hardware that is more friendly to power management is needed.

Optical Component Support

Electrical components consumes much more energy than optical components in network routing infrastructure. Therefore, many power-

aware routing or traffic engineering approaches are designed with electrical devices in mind. However, the number of optical components in ISP networks can also be large. Care should be taken when adopting existing approaches to optical networks. For example, optical receivers cannot be turned off when WoA is needed. Furthermore, there is room for more power saving when optical components are explicitly considered in the approach. It's possible to turn off an optical receiver while maintaining the ability to wake it up when needed, by maintaining another route towards the other end that a control packet could be delivered.

4.2. Software Support

There are many different power-aware approaches. They need different input datasets, and generate different instructions. Software is required to collect necessary input, as well as deliver and execute resulting instructions.

Topology and Traffic

Many power-aware approaches require knowledge of global topology on a centralized server or on each router. It's fairly easy to satisfy this requirement by running a link state routing protocol such as OSPF. If a network running OSPF has OSPF areas configured, power-aware approaches can only be deployed within one area, or some other way to collect global topology is needed.

Many approaches require knowledge of link utilization on a local router, its neighbors, or all routers. Routers may need to maintain necessary counters to calculate this information, and exchange or announce them. Routers then need to categorize packets and maintain a separate set of counters for each interesting category. Some approaches such as GreenTE require network-wide traffic matrix. There are two ways to obtain this information: infer from link utilization, or collect directly. We can infer traffic matrix from global topology and link utilization by using gravity model and tomographic method [TM]. This method requires some computation power, but needs least amount of data exchange, so it is particularly useful when traffic matrix is only needed on a centralized server. However, the accuracy of this method is not guaranteed, especially when traffic engineering is in place that causes traffic pattern to deviate from the gravity model, or multicast is enabled which creates multiple copies of packets. We can also collect traffic matrix directly. There is a cost on ingress routers: an ingress router needs to identify the egress node, and maintain one counter per egress. Identifying egress is not an extra cost in many cases, because many approaches need to know egress to select a feasible route. Maintaining per-egress counters, as well as sending them to

the centralized server in centralized approaches, is a high cost.

Traffic Splitting

Some approaches may route traffic between an ingress-egress pair along multiple paths, according to certain split ratio. To avoid out-of-order arrival which impacts TCP performance, traffic splitting is usually based on the hash of some fields in the packet header, such as source-destination IP pair. In a small network such as a company, there are big flows between some IPs, while there is little traffic on most other IPs. In this case, hash-based splitting has significant bias.

Timeliness of Solutions

Traffic engineering approaches take network topology and traffic information (in the form of link utilization or traffic matrix) as input, and outputs a solution including which links should be sleeping and what rate should links be operated on. Most traffic engineering approaches run on a centralized server. Traffic demand changes over time, and network topology may even change due to link failure. It takes time to collect traffic information from the entire network, and time is also consumed while computing the solution. Thus, the solution, when comes out, is based on network topology and traffic information of sometime earlier, and it may not still be applicable to current situation. Prediction of future traffic information may help in some situations.

4.3. Impacts on Protocols

Power-aware routing and traffic engineering is a tradeoff between energy consumption and network resilience. They save power by turning off or slowing down some links, which were previously over-provisioned to obtain better resilience. Any power-aware approach will cause loss of network resilience to some extent. Sleeping based approaches has another impact. Traditionally, a link is either up or down. An up link can transmit packets, and a down link cannot. A third state, sleeping, is added by power-aware protocols. A sleeping link cannot transmit packets right away, but it can be waken up when needed. The introduction of a third sleeping state has its impact on protocols that maintain their own states about network links.

Congestion after Traffic Surges

Traffic engineering approaches usually take traffic information at certain time, and a solution contains a routing scheme that could accommodate such traffic on a reduce topology with some links sleeping or operating at lower rate. This routing scheme usually

keeps link utilization under certain threshold, so that there is some safe margin in case traffic increases. However, because a solution is computed periodically, congestion is still possible when traffic increases to a level that exceeds the safe margin within one adjustment period. To address this issue, some method of fast readjustment is needed. When a traffic increase is observed, the routing scheme should be slightly changed to accommodate this traffic, probably waking up or increase rate on a few links.

Network Partition on Link or Node Failure

Many sleep based approaches will result in a topology with very low redundancy level. These reduced topologies are vulnerable to link and node failures, which are quite common in large networks. Those approaches should be improved by adding a constraint of redundancy level. A redundancy level of 2, which could protect from single link failure, is a reasonable value. It's possible to incorporate power aware feature into MRT to achieve energy saving while remain the network 2-disjoint [I-D.ietf-rtgwg-mrt-frr-architecture]. Once a partition is detected, it's easy to repair by waking up all sleeping links. But this causes a sudden increase on power consumption, which is sometimes undesirable. A local algorithm to select a subset of sleeping links that could repair the partition is needed. The selection doesn't need to be optimal, because waking up a small subset is much better than waking up all sleeping links.

Sleeping Link State in Routing Protocols

Sleeping links should be handled separately in routing protocols. A sleeping link should be advertised as up, probably with a tag stating it's sleeping. No HELLO messages should be sent over a sleeping link, so no HELLO messages could be received from a sleeping link. Missed HELLO messages on a sleeping link should not cause the link to be treated as down state. As a consequence, if a sleeping link fails, the failure would not be detected until the router attempts to wake it up. To detect a failure earlier, it may be desirable to wake up the link and probe it periodically (using a long interval such as every hour). No control message should be sent over a sleeping link. This may cause the network to converge slower than usual, because LSA flooding takes more hops. Fortunately most power-aware approaches have network diameter constraints, so convergence time should be comparable.

IP Multicast on Reduced Topology

IP Multicast works by building one or more trees on available links. If any link in a multicast tree goes to sleep, some receivers cannot receive multicast packets for a noticeable period of time, until IP

multicast automatically repairs the tree. So, if a link is part of a multicast tree, it should not be put to sleep.

One solution is keeping all links that are contained in multicast trees active. If there are many multicast trees that don't have much overlap, a major portion of links would be forced active by multicast, and power saving potential is greatly limited. Another solution is explicitly modifying multicast trees in a power-aware approach. This is not an easy way to go. There should be a delay constraint on each multicast tree, and there're possibly a large number of multicast trees. After a multicast tree is modified, utilization of multiple links will change.

A third solution is making IP multicast power-aware. When a multicast tree is being built, energy consumption is taken into account, such that IP multicast would attempt to use as few links as possible as long as delay constraint could be satisfied. After that, these links used by IP multicast will not go to sleep.

4.4. Network Monitoring

Network operators demand a monitoring solution when deploying anything. The most important metrics are: How much energy is saved? How much impact is there on network performance? Measurement of Energy Consumption. The IETF eman WG [EMAN-WG] is working on defining energy objects in network devices, and monitoring and controlling their states. When a device is running on full power state, the power demand is recorded as full power demand. When a power-aware approach is deployed, actual energy consumption is measured. The amount of saved energy is the full power demand multiplied by elapsed time during the measurement of actual energy consumption subtracted by actual energy consumption.

In centralized periodical adjustment approaches, the centralized server should have knowledge of current applied solution (which is based on previous traffic information) and current traffic information. It can then calculate what link utilization and delay would be when this traffic is routed on current applied solution, as well as the performance as if this traffic is routed without power-aware consideration. It's not trivial to measure the impact in other cases.

SNMP MIBs are needed to standardize monitoring. Software for operations products such as System Center Operations Manager needs to integrate power-aware routing and traffic engineering to existing IT monitoring architecture.

5. Summary

Power-aware routing and traffic engineering has great potential to improve network energy efficiency while maintain network services at desired levels. Its effectiveness, however, depends on various supports from hardware and software, and more importantly, protocol designs that address operational issues. This document is a first step towards developing practical power-aware protocols.

6. IANA Considerations

This document has no actions for IANA.

7. Security Considerations

This draft is a discussion on the Internet's necessity to follow an evolutionary path towards the future. There is no direct impact on the Internet security.

8. Informative References

[Chabarek08]

Chabarek, J. and et al. , "Power Awareness in Network Design and Routing", IEEE INFOCOM 2008.

[EMAN-WG]

"IETF Energy Management Working Group", 2012,
<<https://datatracker.ietf.org/wg/eman/>>.

[Epps06]

Epps, G. and et al. , "System Power Challenges", 2006,
<[http://www.slidefinder.net/c/cisco routing research/seminar august 29/1562106](http://www.slidefinder.net/c/cisco%20routing%20research/seminar%20august%2029/1562106)>.

[Fisher10]

Fisher, W. and et al. , "Greening Backbone Networks: Reducing Energy Consumption by Shutting Off Cables in Bundled Links", Green Networking 2010.

[GreenTE]

Zhang, M. and et al. , "GreenTE: Power-Aware Traffic Engineering", ICNP 2010.

[Gupta03]

Gupta, M. and S. Singh, "Greening the Internet", ACM SIGCOMM 2003.

[I-D.ietf-rtgwg-mrt-frr-architecture]

Atlas, A., Kebler, R., Envedi, G., Csaszar, A.,

Konstantynowicz, M., White, R., and M. Shand, "An Architecture for IP/LDP Fast-Reroute Using Maximally Redundant Trees", draft-ietf-rtgwg-mrt-frr-architecture-01 (work in progress), March 2012.

[Nedevschi08]

Nedevschi, S. and et al. , "Reducing Network Energy Consumption via Sleeping and Rate- Adaptation", USENIX NSDI 2008.

[PCE-WG]

"IETF Path Computation Element Working Group", 2012, <<https://datatracker.ietf.org/wg/pce/>>.

[TM]

Roughan, M., Thorup, M., and Y. Zhang, "Traffic Engineering with Estimated Traffic Matrices", IMC 2003.

Authors' Addresses

Beichuan Zhang
The University of Arizona

Email: bzhang@cs.arizona.edu

Junxiao Shi
The University of Arizona

Email: shijunxiao@cs.arizona.edu

Jie Dong
Huawei

Email: jie.dong@huawei.com

Mingui Zhang
Huawei

Email: zhangmingui@huawei.com

