TRILL Working Group
Internet Draft
Expires: August 2006

R. Perlman
Sun
J. Touch
USC/ISI
February 13, 2006

Rbridges: Base Protocol Specification
draft-perlman-trill-rbridge-protocol-00.txt

Status of this Memo

Abstract

RBridges provide the ability to have an entire campus, with multiple physical links, look to IP like a single subnet. The design allows for zero configuration of switches within a campus, optimal pair-wise routing, safe forwarding even during periods of temporary loops, and the ability to cut down on ARP/ND traffic. The design also supports VLANs, and allows forwarding tables to be based on RBridge destinations (rather than endnode destinations), which allows

internal routing tables to be substantially smaller than in conventional bridge systems.

Table of Contents

1. Introduction

In traditional IPv4 and IPv6 networks, each link must have a unique prefix. This means that a node that moves from one link to another must change its IP address, and a node with multiple links must have multiple addresses. It also means that a company with many links (separated by routers) will have difficulty making full use of its IP

address block (since any link not fully populated will waste addresses), and IP routers require significant configuration. Bridges avoid these problems because bridges can transparently glue many physical links into what appears to IP to be a single LAN.

However, bridge routing via the spanning tree using the layer 2 header has some disadvantages:

- The spanning tree limits which links can be used, and therefore concentrates traffic onto selected links
- Forwarding based on a header without a TTL is dangerous, because temporary loops might arise due to topology changes, lost spanning tree messages, or components such as repeaters coming up)
- Routes cannot be pair-wise shortest paths, but instead whatever path remains after the spanning tree eliminates redundant paths

We define the term "campus" to be the set of links connected by any combination of RBridges and bridges. A campus appears to IP nodes to be a single subnet.

This document presents the design for Rbridges (routing bridges), which combines the advantages of bridges and routers. Like bridges, RBridges are zero configuration, and are transparent to IP nodes. Like routers, RBridges forward on pair-wise shortest paths, and do not have dangerous behavior during temporary loops. RBridges have the additional advantage that they can optimize ARP (IPv4) and ND (Ipv6) by avoiding the broadcast/multicast behavior of the queries.

RBridges are fully compatible with current bridges as well as current IPv4 and IPv6 routers and endnodes. They are as invisible to current IP routers as bridges are, and like routers, they terminate a bridged spanning tree.

The main idea is to have RBridges run a link state protocol amongst themselves. This enables them to have enough information to compute pairwise optimal paths for unicast, and to calculate distribution trees for delivery of packets to unknown destinations, or multicast/broadcast packets.

RBridges must learn the location of endnodes. They learn the location and layer 2 addresses of attached nodes from the source address of data frames, as bridges do. Additionally, in order to facility proxy ARP or proxy ND optimizations, RBridges also learn the (layer 3, layer 2) addresses of attached IP nodes from ARP or ND replies.

Once an RBridge learns the location of a directly attached endnode, it informs the other RBridges in its link state information.

RBridge forwarding can be done, as with a router, via pairwise shortest paths.

To mitigate the temporary loop issues with bridges, RBridges must always forward based on a header with a hop count. Although the hop count will quickly discard looping frames, it is also desirable not to spawn additional copies of frames. This can be accomplished by having RBridges specify the next RBridge recipient while forwarding across a shared-media link.

Frames must be encapsulated as they travel between RBridges for several reasons:

1. to prevent source MAC learning from frames in transmit

2. so that the frames can be directed towards the egress RBridge. This enables forwarding tables of RBridges to be sized with the number of RBridges rather than the total number of nodes in the common broadcast domain

3. so that frames in transit can include a hop count (for links, like Ethernet, that do not already contain a hop count)

In order to coexist with Ethernet bridges on Ethernet links, frames in transit on Ethernet links must be encapsulated with an Ethernet header. The outer header of an RBridge-forwarded frame must look, to an Ethernet bridge on the path between two RBridges, like the header of a normal frame that the bridge will forward.

Inside that header is a shim header that RBridges will add to the frame that will contain:

- the ingress-RBridge (in the case of a broadcast/multicast/unknown destination frame), or egress-RBridge (in the case of a unicast frame to a known destination)

- a hop count

Inside the shim header is the original frame, as injected into the campus.

RBridges must also support VLANs.

A VLAN is a broadcast domain. That means that a layer 2 broadcast (multicast) frame sent to a VLAN must only be delivered to links that are in that VLAN. A frame for a particular VLAN may transit any link on the campus, but an unencapsulated VLAN frame must only be delivered to links that RBridges know (for example, through configuration) support that VLAN.

There are several types of frames which RBridges must deliver within a broadcast domain:

1. frames for unknown destinations

2. frames for layer 2 multicast addresses derived from IP multicast addresses

3. frames for layer 2 broadcast/multicast frames which are not derived from IP multicast addresses

4. ARP/ND queries

If a frame belongs in a particular VLAN, the frame must be delivered only to links in that VLAN. This is true for both broadcast/multicast frames, and unicast frames.

RBridges will calculate a distribution tree for each ingress RBridge, which we will refer to as the "ingress RBRidge tree". In theory, RBRidges could have calculated a single spanning tree for the entire campus. However, it was decided that the additional computation necessary to compute ingress RBridge trees was warranted because:

1. it optimizes the distribution path and (almost always) the cost of delivery when the number of destination links is a subset of the total number of links. Delivery is only to a subset of links in the case of VLANs and IP multicasts

2. for unknown destinations, out-of-order delivery is minimized because in the case where a flow starts before the location of the destination is known by the RBridges, the path to the destination through the per-ingress-RBridge tree will be the same as the path directly to the destination

RBridges will not use the bridge spanning tree algorithm to calculate the ingress RBridge trees. Instead, the trees are calculated based on the link state information. Therefore the tree calculation is done without requiring any additional exchange of information between RBridges.

2. Detailed Rbridge Design

2.1. Link State Protocol

Running a link state protocol among RBridges is straightforward. It is the same as running a level 1 routing protocol in an area, with endnode addresses being layer 2 addresses rather than, say, IP addresses. IS-IS is natural choice for a link state protocol because it is easy in IS-IS to define new TLVs for carrying new information, and because IS-IS can be done with zero configuration. All that is required to run IS-IS is for each RBridge to have a unique 6-byte system ID, which can be any of the RBridge's MAC addresses.

2.1.1. Separate Instances

The instance of IS-IS that RBridges will implement is separate from any routing protocol that IP routers will implement, just as the spanning tree messages are not implemented by IP routers.

To prevent potential confusion between an IS-IS instance being run by IP routers and the IS-IS being run by RBridges, RBridge routing messages will be sent to a different layer 2 multicast address than IS-IS routing messages. The RBridge IS-IS instance is also differentiated by having a distinct, contant "area address" (the value 0) that would never appear as a real IS-IS area address.

2.1.2. Multiple Rbridge IS-IS Instances

There are two types of information that are carried in RBridge link state information; "core-RBridge information", and "endnode information". In theory this information could all be contained in one instance of RBridge IS-IS. However, since endnode information for a particular VLAN only needs to be known to RBridges that are connected to links configured to be in that VLAN, it was decided that each RBridge R1 will run a "core" instance of IS-IS for the core RBridge information, and an instance per VLAN that R1 is attached to, for the endnode information for those VLANs.

The core-RBridge information, which is carried in the core-RBridge instance, is:

1. the system IDs of RBridges which are neighbors of RBridge R1, and the cost of the link to each of those neighbors

2. VLANs directly connected to R1

The endnode information for VLAN A, which is carried in the VLAN A IS-IS instance injected by R1, contains:

1. L2INFO: layer 2 addresses of nodes on a VLAN A link attached to R1 which have transmitted frames but have not transmitted ARP or ND replies (i.e., these are not known to be IP nodes)

2. L3and2INFO: layer 3, layer 2 addresses of IP nodes attached to R1, which R1 has learned through ARP/ND replies emitted by endnodes on an attached VLAN A link. For data compression, only the portion of the address following the campus-wide prefix need be carried. (This is a more important optimization for IPv6 than for IPv4)

If R1 has learned endnode E's location first from a data packet (and therefore has included E's layer 2 address in the L2INFO, and later E transmits an ARP/ND reply, R1 MUST include E in the L3andL2INFO, and MAY remove E from L2INFO.

Given that RBridges must already support delivery only to links within a VLAN (for multicast or unknown frames marked with the VLAN's tag), the same mechanism is used to advertise endnode information solely to RBridges within a VLAN.

The per-VLAN instance of IS-IS will appear to the RBridges to consist of a single link. R1 will originate a VLAN-A-specific IS-IS frame. All RBridges will recognize the frame as a VLAN A multicast frame (even if they are not connected to VLAN A), and prune the ingress-R1 tree so as to only deliver the frame along branches with VLAN A links. This is the same behavior core RBridges would have for any VLAN A multicast/broadcast/unknown destination frame. RBridges that are connected to VLAN A links will, in addition to forwarding along the ingress RBridge tree, process the frame in their VLAN-A IS-IS instance.

Thus suppose that RBridges R1, R2, and R3 are all on VLAN A, on links scattered throughout the campus. The VLAN A IS-IS will appear to be a single link (broadcast domain) with R1, R2, and R3 as neighbors. The only information carried in the instance is the endnode information for VLAN A. The other RBridges on the campus facilitate delivery within the VLAN A broadcast domain, and therefore may be on the path between R1 and R2, but will treat the VLAN A instance link state frames as ordinary datagrams.

The way that RBridges distinguish which IS-IS instance the link state information is for is based on the VLAN tag in the inner header.

2.2. Ingress Rbridge Tree Calculation

Some frames (e.g., to unknown destinations, or multicast destinations) will need to be delivered to multiple links. To optimize delivery in the case where not all links are to receive the frame (e.g., an IP multicast or a VLAN-tagged frame), and to avoid out-of-order delivery when location of the destination is discovered after a flow starts up, RBridges calculate a tree per ingress RBridge, and deliver a frame along that distribution tree. The ingress RBridge trees will be calculated based on the link state information distributed in the core IS-IS instance.

In IS-IS, each "node" that initiates a link state packet has an ID. If the "node" is a router, initiating the link state information on behalf of itself, the ID is the router's system ID, concatenated with the constant 0. If the "node" is a pseudonode, i.e., a shared link, then one RBridge, say R1, on the link, is elected Designated RBridge, and R1 initiates a link state packet on behalf of the pseudonode. In this case the ID of the pseudonode is a 7-byte quantity which R1 can be sure is unique within the campus, usually the 6-byte system ID of R1, concatenated with a byte chosen by R1 to differentiate this pseudonode from any other link for which R1 might also be Designated RBridge.

So, the link state information consists of a bunch of nodes, each with a unique (within the campus) ID, and the connectivity between these nodes. It is essential that all RBridges calculate the same set of ingress RBridge trees. In the case of encountering multiple equal cost paths in the tree calculation, some tie breaker must be used to ensure that all RBridges calculate the same tree.

When choosing where to attach a node to the tree being calculated, the tie-breaker is the ID of the parent to which the node will be attached (if a node can be attached to either parent P1 or P2 with the same cost, choose P1 if P1's ID is lower than P2).

2.3. Pruning the Ingress Rbridge Tree

Packets which must be flooded (e.g., multicasts, unknown destinations), are flooded along a tree rooted at the ingress RBridge, and pruned based on whether there are potential receivers downstream. In the case of a VLAN-tagged packet, it need not be delivered if no RBridges along a branch of that tree (rooted at the ingress RBridge) have RBridges participating in that VLAN. In the case of a multicast derived from an IP multicast, IGMP snooping by RBridges might further narrow which links have potential receivers.

The actual spanning tree to forward along is chosen based on the ingress-RBridge, whose identity is contained in the shim header. Say the ingress RBridge is Ri. Suppose RBridge Rc knows that the set of links {L1, L2, L3} is in the ingress-Ri spanning tree.

If the frame is received on link L3, the frame may be forwarded to links L1 and L2. However, Rc can limit distribution of the frame if Rc knows there are no interested receivers along a branch.

The way this is done is that Rc first calculates the Ri tree, determining that, say, links {L1, L2, and L3} are contained in that tree. Furthermore, since this is an ingress RBridge tree, distribution is unidirectional. So Rc will know that for this tree, all traffic will be received on, say, L1, and transmitted out L2 and L3. Now Rc must calculate, for each of the output links (L2 and L3), the set of destinations that should be forwarded onto that link.

For each of L2, and L3, and for each VLAN and for each IP multicast address (as determined by IGMP snooping), Rc must indicate which of those addresses have receivers downstream from that link.

So Rc will know that {L2, and L3} are output links for the Ri-ingress-spanning tree. For each of the output links for this ingress Rbridge tree, Rc keeps a list of layer 2 multicast addresses derived from IP multicasts, and learned through IGMP snooping, and the set of VLAN tags, which are reachable through that output link on this ingress-RBridge tree.

If Rc receives a multicast frame from L1, it selects the spanning tree based on the ingress RBridge as indicated in the shim header. Then it prunes based on the destination address in the original frame. Then it forwards on the remaining output links for that ingress RBridge tree that have receivers.

For each link for which Rc is Designated, Rc additionally checks to see if it should decapsulate the frame and send it to the link.

2.4. Designated Rbridge

One RBridge on each link needs to be elected to have special duties. This elected RBridge is known as the Designated RBridge. IS-IS already holds such an election.

The Designated RBridge is the one on the link that will learn and advertise the identities of attached endnodes, encapsulate and forward frames that originate on that link to the rest of the campus, decapsulate and forward frames onto that link received from other

RBridges, initiate a distributed ARP when an ARP query is received for an unknown destination, and answer ARP queries when the target node is known.

2.5. Learning Endnode Location

RBridges learn endnode location from data frames. They learn (layer 3, layer 2) pairs (for the purpose of supporting ARP/ND optimization) from listening to ARP or ND replies.

This endnode information is learned by the DR, and distributed to other RBridges through the link state protocol.

2.6. Forwarding Behavior

2.6.1. Receipt of a Native Packet

R1 receives a native (i.e., not RBridge-encapsulated) unicast frame. R1 knows that this is a native frame because the Ethertype is not "RBridge encapsulated frame". The destination in the layer 2 header is D, the source is S.

R1 inserts a VLAN tag if required, according to the same rules as bridges do.

Once the VLAN (if any) is established, the layer 2 address of D is looked up in the destination table to find the egress RBridge R2, or discover that D is unknown.

If D is known, with egress R2, then R1 encapsulates the packet, with R2 indicated in the shim header as egress RBridge. In the outer header, R1 puts "R1" as source, and next hop RBridge (in the path to R2) as "destination", and "encapsulated RBridge packet" as the Ethertype.

If D is unknown, R1 encapsulates the packet, with "R1" indicated as ingress RBridge in the shim header, and outer header with source=R1, destination = "all-RBridges".

2.6.2. Receipt of an In-transit Packet

RBridge R1 receives an encapsulated frame (as indicated by Ethertype="Rbridge-encapsulated).

2.6.2.1. Flooded Packet

If the destination in the outer header is "all-RBridges", then R1 forwards along the ingress RBridge tree indicated by the shim header.

If the frame's inner header indicates it is for a specific VLAN, links in that indicated ingress RBridge tree that do not lead to links in that VLAN are pruned for this packet. Furthermore, if the frame contains an IP multicast packet, then R1 only forwards on branches that have learned, through IGMP, have receiver on those links for this IP multicast.

In addition, for links for which R1 is Designated, R1 decapsulates the packet and transmits the packet onto those links (unless the packet is IP multicast or VLAN-tagged, and the packet does not belong on that link).

If the frame belongs in VLAN A, (based on the presence of a tag in the inner header) then R1 (the ingress RBridge) looks up D's location in R1's table of VLAN A endnodes.

If the native frame's destination is a layer 2 multicast, then if the frame is a BDPU, the RBridge drops the frame.

If the native frame's destination is "all-RBridges" with Ethertype "IS-IS", then R1 processes the link state packet.

If the packet is an IGMP announcement, which will be transmitted to an IP-derived layer 2 multicast address of "all IP routers", then the RBridge learns, based on the "ingress RBridge" in the shim header, the mapping between egress RBridges and IP multicast address listeners.

2.6.2.2. Unicast Packet

If the destination in the outer header is not R1, then R1 drops the frame.

If the shim header indicates R1 is the egress RBridge, then R1 extracts the inner frame and forwards it onto the link containing the destination, or processes the packet if the destination in the inner frame is R1.

Else, R1 looks up the egress RBridge R2 indicated in the shim header, in its forwarding table, and forwards the packet towards R2, by replacing the outer header with one with source=R1,

destination=nexthop RBridge towards R2, and Ethertype "encapsulated RBridge".

2.7. IGMP Learning

RBridges learn, based on seeing IGMP packets, which multicast addresses should be forwarded onto which links.

IGMP messages have to be forwarded throughout the campus, since IP routers in the broadcast domain also need to see these messages.

IGMP messages are forwarded by RBridges throughout the campus like any layer 2 multicast. They are recognized by having an IP message type=2 in the IP header. In addition, they are processed by RBridges in order to extract, from announcements, what egress RBridges have receivers for which groups.

2.8. Combined Bridge/Rbridge

RBridges do not participate in the bridge spanning tree protocol. The only thing RBridges do with regard to bridge spanning tree is recognize BPDUs and drop them.

In some cases it might be advantageous for the Designated RBridge to be the Root of the bridge spanning tree calculated on a link (where "link" as perceived by the RBridges might actually be a collection of segments connected via bridges). Having the bridge spanning tree on that link rooted at the RBridge will optimize paths that go between a node on that link and a node off the link to another link within the campus. However, this may make intra-link paths worse, or paths between a node on the link and an IP router also on that link.

If it is desired for the spanning tree to be rooted at the Designated RBridge, this can be accomplished by implementing a colocated bridge/RBridge. This would be equivalent to two boxes: a bridge directly connected to the link, and a point-to-point link to the RBridge.

The bridge portion of the logically combined box would change its priority to the numerically lowest value if the colocated RBridge is elected Designated RBridge on that link.

Note that this section is only an implementation possibility. RBridges are not required to be implemented as combined with bridges. The only point of this section is that RBridges MUST recognize and drop BPDUs.

2.9. Forwarding Header on 802 Links

It is essential that RBridges coexist with ordinary bridges. Therefore, a frame in transit must look to ordinary bridges like an ordinary layer 2 frame. However, it must also be differentiable from a native layer 2 frame by RBridges. To accomplish this, we use a new layer 2 protocol type ("Ethertype").

A frame in transit on an 802 link will therefore have two 802 headers, since the original frame (including the original 802 header) will be tunneled by the RBridges. But rather than just having an additional 802 header, we include additional information between the two headers; at least a hop count.

An encapsulated frame would look as follows:

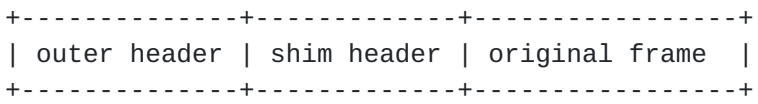

Figure 1 Encapsulated Frame

The outer header contains:

- L2 destination = next RBridge, or for flooded frames, a new (to be assigned) multicast layer 2 address meaning "all RBridges"

- L2 source = transmitting RBridge (the one that most recently handled this frame)

protocol type = "to be assigned...RBridge encapsulated frame"

The shim header includes:

- TTL = starts at some value and decremented by each RBridge. Discarded if=0

- egress RBridge (in the case of unicast), or ingress RBridge (in the case of multicast)

2.10. Format of the Shim Header

The format of the shim header is defined in draft-bryant-perlman-trill-pwe-encap-00. This is an MPLS-based format, although it will have the semantics stated above (that it contains a TTL and an ingress or egress RBridge). However, an RBridge ID is 6-bytes, and this format only allows for 19 bits to specify the RBridge ID.

Therefore, there is a distributed process piggybacked on the link state protocol, whereby RBridges choose 19-bit nicknames. This protocol is also specified in the internet draft draft-bryant-perlman-trill-pwe-encap-00.

2.11. Handling ARP/ND Queries

We will use the term "optimized ARP/ND response" to cover several possible behaviors an RBridge might utilize. Non-optimized behavior would consist of treating an ARP or ND query as an ordinary layer 2 broadcast/multicast, and send the query to all links in the campus, allowing the target to respond as to an ordinary ARP/ND query. This behavior is essential when the location of the target is unknown, although RBridges could suppress multiple queries to the same target within some amount of time.

When the target's location is assumed to be known by the first RBridge, it need not flood the query. Alternative behaviors of the first Designated RBridge that receives the ARP/ND query would be to:

1. send a response directly to the querier, with the layer 2 address of the target, as believed by the RBridge

2. encapsulate the ARP/ND query to the target's Designated RBridge, and have the Designated RBridge at the target forward the query to the target. This behavior has the advantage that a response to the query will be definitive. If the query does not reach the target, then the querier will not get a response

3. block ARP/ND queries that occur for some time after a query to the same target has been launched, and then respond to the querier when the response to the recently-launched query to that target is received

The reason not to do the most optimized behavior all the time is for timeliness of detecting a stale cache. Also, in the case of SEND, cryptography might prevent behavior 1, since the RBridge would not be able to sign the response with the target's private key.

It is not essential that all RBridges use the same strategy for which option to select for a particular query. However, once the first Designated RBridge decides on a strategy for a particular query, the other RBridges must carry that through. If the first RBridge responds directly to the querier, or blocks the query, then no other RBridges are involved.

If the first Designated RBridge R1 decides to unicast the query to the target's Designated RBridge R2, then R2 must decapsulate the query, and initiate an ARP/ND query on the target's link. When/if the target responds, R2 must encapsulate and unicast the response to R1, which will decapsulate the response and send it to the querier.

If the first Designated RBridge R1 decides to flood the query (which it MUST do if the target is unknown, but MAY do if it wants to assure freshness of the information), the query is encapsulated to be flooded through the indicated VLAN.

The distributed ARP query is carried by RBridges through the RBridge spanning tree. Each Designated RBridge, in addition to forwarding the query through the spanning tree, initiates an ARP query on its link(s). If a reply is received from the target by Designated RBridge R2, R2 initiates a link state update to inform all the other RBridges of D's location, layer 3 address, and layer 2 address, in addition to forwarding the reply to the querier.

It is the querier's Designated RBridge R1 that chooses which strategy to employ when seeing an ARP query.

Some mix of these strategies (responding directly, unicasting the query to the target's Designated RBridge, or flooding the query) might be the best solution. For instance, even if the target's location and (layer 3, layer 2) correspondence is in the link state information R1 received from R2, if the target's location has not been recently verified by R1 through a broadcast ARP/ND or unicast query to the target, then R1 MAY broadcast or unicast the query or respond directly. So for instance, RBridges could keep track of the last time a broadcast ARP/ND occurred for each endnode E (by any source, and injected by any RBridge). Let's say the parameter is 20 seconds. If a source S on RBridge R1's link does an ARP/ND for D, if R1 has not seen an ARP/ND for D within the last 20 seconds, R1 unicasts the query to force a reply from the target; otherwise it proxies the reply.

When R2 forwards a unicast ARP/ND query, if the target does not respond, then R2 MAY replay the query, and if the target does not respond, R2 will remove the target from its link state information.

2.12. Assuring Freshness of Endnode Information

Designated RBridge R1 can ensure freshness of its endnode information by doing ARP/ND queries periodically to ensure that the endnodes are actually there. This can be a problem if the endnodes are in power-

saver mode, and this should be a configuration parameter on R1 as to whether R1 should "ping" the endnodes by doing ARP/ND queries.

3. Rbridge Addresses, Parameters, and Constants

Each RBridge needs a unique ID within the campus. The simplest such address is a unique 6-byte ID, since such an ID is easily obtainable as any of the EUI-48's owned by that RBridge. IS-IS already requires each router to have such an address.

A parameter is the value to which to initially set the hop count in the envelope. Recommended default=20.

A new Ethertype must be assigned to indicate an RBridge-encapsulated frame.

A layer 2 multicast address for "all RBridges" must be assigned for use as the destination address in flooded frames.

To support VLANs, RBridges (like bridges today), must be configured, for each port, with the VLAN in which that port belongs.

We may want a parameter to determine whether an RBridge should periodically do queries to ensure that the endnode information is fresh, and if so, with what frequency.

4. Security Considerations

The goal is for RBridges to not add additional security issues over what would be present with traditional bridges. RBridges will not be able to prevent nodes from impersonating other nodes, for instance, by issuing bogus ARP replies. However, RBridges will not interfere with any schemes that would secure neighbor discovery.

As with routing schemes, authentication of RBridge messages would be a simple addition to the design (and it would be accomplished the same way as it would be in IS-IS). However, any sort of authentication requires additional configuration, which might interfere with the perception that RBridges, like bridges, are zero configuration.

5. IANA Considerations

A new Ethertype must be assigned to indicate an RBridge-encapsulated frame.

A layer 2 multicast address for "all RBridges" must be assigned for use as the destination address in flooded frames.

6. Conclusions

This design allows transparent interconnection of multiple links into a single IP subnet. Management would be just like with bridges (plug-and-play). But this design avoids the disadvantages of bridges. Temporary loops are not a problem so failover can be as fast as possible, and shortest paths can be followed.

The design is compatible with current IP nodes and routers, and with current bridges.

7. Acknowledgments

In addition to Eric Grey, and Erik Nordmark, we anticipate that many people will contribute to this design, and invite you to join the mailing list at http://www.postel.org/rbridge

8. References

8.1. Normative References

[1] IEEE 802.1d bridging standard, "IEEE 802.1d bridging standard".

8.2. Informative References

[2] Bryant, S., Perlman, R., Atlas, Alk, Fedyk, D., "TRILL using Pseudo-Wire Emulation (PWE) Encapsulation", internet draft draft-bryant-perlman-trill-pwe-encap-00.

[3] Narten, T., Nordmark, E. and W. Simpson, "Neighbor Discovery for IP Version 6 (IPv6)", RFC 2461 (Standards Track), December 1998.

[4] Perlman, R., "RBridges: Transparent Routing", Proc. Infocom 2005, March 2004.

[5] Perlman, R., "Interconnection: Bridges, Routers, Switches, and Internetworking Protocols", Addison Wesley Chapter 3, 1999.

Author's Addresses

Radia Perlman
Sun Microsystems

Email: Radia.Perlman@sun.com

Joe Touch
USC/ISI
4676 Admiralty Way
Marina del Rey, CA 90292-6695
U.S.A.

Phone: +1 (310) 448-9151
Email: touch@isi.edu
URL: http://www.isi.edu/touch

Intellectual Property Statement

Disclaimer of Validity

Copyright Statement

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.