This design provides the ability to have an entire campus, with multiple physical links, look to IP like a single subnet. This allows zero configuration of the switches within the campus, and allows nodes to move around within the campus without changing IP.
addresses. This capability is often provided today with bridges. Bridges do accomplish this goal. However, bridges have disadvantages: routing is confined to a spanning tree (precluding pair-wise shortest paths), the header on which the spanning tree forwards has no hop count, spanning tree forwarding in the presence of temporary loops spawns exponential copies of packets, nodes can have only a single point of attachment, and the spanning tree, in order to avoid temporary loops, is slow to start forwarding on new ports. The design in this paper avoids these disadvantages of bridges while maintaining the advantages. This design works for both IPv4 and IPv6.

This document is a work in progress; we invite you to participate on the rbridge mailing list at http://www.postel.org/rbridge

Table of Contents

1. Introduction .................................................. 3
2. Detailed RBridge Design ....................................... 5
   2.1 Link State Protocol ....................................... 5
   2.2 Spanning Tree ............................................. 5
   2.3 Designated Bridge ......................................... 6
   2.4 Learning endnode location .................................. 6
   2.5 Distributed ARP query ..................................... 7
   2.6 Forwarding header ......................................... 7
3. RBridge Addresses, parameters, and constants .................. 9
4. Handling non-IP packets ....................................... 10
5. Handling on-campus IP Packets ................................ 11
6. Handling off-campus IP packets ................................ 12
7. Handling ARP Queries .......................................... 13
8. Issues ......................................................... 14
   8.1 Avoiding encapsulation in some cases ..................... 14
      8.1.1 Avoiding encapsulation for on-campus IP packets ... 14
      8.1.2 Avoiding encapsulation for off-campus IP packets ... 14
   8.2 Effects on L3 TTL ......................................... 14
   8.3 Using L3 Encapsulation ..................................... 15
   8.4 Topology Issues ........................................... 15
9. Security Considerations ........................................ 16
10. Conclusions .................................................. 17
11. Acknowledgments .............................................. 18
12. References .................................................. 19
   12.1 Normative References .................................... 19
   12.2 Informative References ................................... 19
       Authors’ Addresses .......................................... 20
       Intellectual Property and Copyright Statements ........... 21
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 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 concentrates traffic onto selected links, is slow to bring new connectivity on-line because temporary loops are very dangerous (because there is no hop count in the header and there may be exponential proliferation of packets during loops), and routes cannot be pair-wise shortest paths, but instead whatever path remains after the spanning tree eliminates redundant paths.

There have been proposals for having routers within a campus automatically number links with distinct IP subnet numbers. Although this makes a campus plug-and-play, it requires a large number of IP subnet numbers, a node must change its address if it moves to a different link, and addresses of nodes might fluctuate as the topology changes and links must be renumbered.

NB: the term ‘campus’ needs to be clearly defined. A campus refers to a set of links connected by either Rbridges or bridges. In other words, the campus is terminated by traditional IP routers, in the same way that an IP subnet would be terminated by an IP router. A campus will look to IP nodes like a single IP subnet, whether the interconnection of the links is done with bridges, Rbridges, or some combination of the two.

This proposal introduces R Bridges [Pe04] (Routing Bridges), which allow transparent interconnection of many links without the disadvantages of bridges.

R Bridges are fully compatible with current bridges as well as current IPv4 and IPv6 routers. 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 R Bridges run a link state protocol amongst themselves (IS-IS is ideal, since its TLV encoding easily allows new information to be carried in link state information, as this proposal requires).
The next step is for RBridges to learn the location of endnodes. They can learn the location and layer 3 addresses of IP endnodes from ARP replies (IPv4) or ND messages (IPv6) ([RFC1122], [RFC1812], etc.). It may also be necessary to learn layer 2 addresses of nodes (for support of protocols that are not carried inside IP headers, and for transporting IP packets destined off the campus to a specific IP router). These can be learned through receipt of data packets, as bridges do.

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. RBridges could also maintain MPLS paths between themselves, and route packets on an MPLS path.

To prevent the temporary loop issues with bridges, RBridges must always forward based on a header with a hop count, and must avoid packet proliferation by only forwarding in one direction, and specifying the intended next recipient while the packet is in transit. Because current L2 protocols do not always support TTls (notably Ethernet does not), this may require the use of an encapsulation header with a TTL field, e.g., an outer IP wrapper or a shim layer.

An Rbridge uses multiple devices to emulate an L2 bridge, using an internal fabric of tunnels and routing independent of to the transited traffic. Transited traffic is usually encapsulated at the Rbridge system ingress in either an L2 or L3 + L2 header that directs the traffic towards the Rbridge system egress. Rbridges are similar to Recursive Routers, which provide similar transit to emulate a single L3 router, in that case using L3 + L2 encapsulation ([To01],[To03]).
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. IS-IS is a more appropriate choice than OSPF in this case because it is easy in IS-IS to define new TLVs for carrying new information. However, the instance of IS-IS that RBridges will implement will be separate from any routing protocol that IP routers will implement, just as the spanning tree messages are not implemented by IP routers.

To keep the instances separate, RBridge routing messages should be sent to a different layer 2 multicast address than IS-IS routing messages. Alternatively, they can be differentiated by having a different "area address", where, in order to keep RBridges configuration-free, the RBridge area address would be a constant for all RBridges, and would not be one that would ever appear as a real IS-IS area address.

Additional information that RBridge link state information will carry is:

- layer 2 addresses of non-IP nodes within the campus
- (layer 3, layer 2) addresses of IP nodes within the campus. For data compression, perhaps only the portion of the address following the campus-wide prefix need be carried. This will be more of an issue for IPv6 than for IPv4.

2.2 Spanning Tree

There will be cases when RBridges may need to send packets to all links. These cases include:

- layer 2 multicast or broadcast packets
- distributed RBridge layer 3 address location query

In this case the packets must be sent through a spanning tree. However, there is no need to implement a separate spanning tree protocol in addition to the link state protocol. Instead, the link state information can be used to create a single spanning tree
throughout the campus. This is done by choosing the RBridge with lowest ID, and calculating the Dijkstra tree with that RBridge as Root.
In the case of multiple equal cost links, some tie-breaker must be used to ensure that all RBridges calculate the same spanning tree. We suggest using the ID of the parent as the tie breaker (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).

In the case of multicast L2 addresses, the rbridge may treat these as broadcast, or may include existing techniques for emulating multicast at L2, i.e., snooping IGMP and/or PIM-SM packets to configure an internal, L2 multicast tree.

2.3 Designated Bridge

It is useful for one RBridge on each link to have special duties. Thus one RBridge per link should be elected Designated RBridge. IS-IS already holds such an election.

The Designated RBridge is the one on the link that will learn the identities of attached endnodes, initiate a distributed ARP when an ARP query is received for an unknown destination, and answer ARP queries when the target node is known.

2.4 Learning endnode location

There are several mechanisms for learning endnode location. RBridges could learn, like bridges do, from data packets. If this is done, it is essential that this learning only occur on the source’s link. Otherwise RBridges on transit links, as well as other RBridges on the destination’s link, will be confused and think that the source resides on their link.

If learning is to occur on data packets, this confusion can be prevented by marking packets in transit, and ensuring that only the Designated RBridge learn endnode locations, and only the Designated RBridge forwards packets onto the LAN (and removes the transit mark).

This form of learning needs to be done to learn layer 2 addresses of nodes that are speaking protocols that are not carried in IP frames.

For packets carried in IP frames, it is not necessary to learn in the data path. Instead destination locations can be learned either from ARP replies (or ND discovery) or from link state information.

If Designated RBridge R receives an IP packet for D, and D is on-campus (D’s prefix is the campus prefix), and D is unknown to R, R initiates a "distributed ARP query". Likewise if R receives an ARP
query for target D, R initiates a distributed ARP query.
2.5 Distributed ARP query

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 by Designated RBridge R2, R2 initiates a link state update to inform all the other RBridges of D’s location and layer 2 address.

The distributed ARP query must be sent to a (new, to be assigned) layer 2 multicast address. The fields it must contain are:

Layer 2 header:

- destination = newly defined 12 multicast address
- source = transmitting RBridge (replaced hop by hop)
- protocol type = same as encapsulated RBridge

Body:

- TTL (for safety if the RBridge spanning tree has temporary loops)
- target IP address (IPv4 or IPv6)

Intermediate RBridges decrement the above TTL, and replace the source RBridge with their own layer 2 address on the outgoing interface.

2.6 Forwarding header

It is essential that RBridges coexist with ordinary bridges. Therefore, a packet in transit must look to ordinary bridges like an ordinary layer 2 packet. For packets to IP destinations on the campus, it is not necessary for packets to be encapsulated, since routing could be done on the IP header. However, this would result in the TTL being decremented by the RBridges, which would look different to customers than if the campus were connected by bridges. Therefore, it is likely we will opt for encapsulating all packets as they traverse the campus.

Although the forwarding header must look like an ordinary layer 2 header to bridges, it must be differentiable from ordinary layer 2
packets by R Bridges. For this we need a new layer 2 protocol type ("Ethertype"). An encapsulated packet would look as follows:
The encapsulation header contains:

- L2 destination = next RBridge
- L2 source = transitting Rbridge (the most one that most recently handled this packet)
- protocol type = "to be assigned...RBridge encapsulated packet"
- TTL = starts at some value and decremented by each RBridge. Discarded if=0

Note that the outer L2 destination is the next RBridge rather than the destination’s L2 address. This prevents proliferation of packets, since a single RBridge destination is specified. The alternative would be that multiple R Bridges might decide to forward the packet, creating extra copies.
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 packet.

A layer 2 multicast address must be assigned for use as the destination address in distributed ARP queries.
4. Handling non-IP packets

RBridges must learn the source’s location based on receipt of data packets, just like a bridge would. However, only the first RBridge must see the source address; otherwise, since the packet is routed by a pairwise shortest path, intermediate RBridges and bridges will be confused about the location of the source.

Therefore, the first RBridge (and only the Designated RBridge on the source’s link) encapsulates the packet with an encapsulation header. The specified next RBridge, R2, will look up the layer 2 destination in the inner header to determine the forwarding direction. Then R2 will replace the layer 2 source and destination addresses in the outer header with R2 as source and next RBridge as destination, decrement the TTL, and forward the packet. If R2 is the Designated RBridge on the destination’s link, R2 removes the outer header and forwards the packet exactly as transmitted by the source.
5. Handling on-campus IP Packets

Here, RBridges forward based on the layer 3 header. If the layer 3 header is used, there is the advantage that the campus can encompass links with incompatible layer 2 addresses. This enables IP nodes in the campus to communicate even if they speak incompatible layer 2 protocols. However, it will not allow two such nodes to communicate if they are not speaking IP, unless the layer 2 protocols are sufficiently similar that RBridges can translate the headers. Such functionality is beyond the scope of this document, however.

It might also be nice to eliminate the inner layer 2 header. However, future uses might be made if the original layer 2 header were preserved where possible (where source and destination were on compatible layer 2 links). For instance, it might be nice to update ARP caches based on receipt of data packets.

Each intermediate RBridge that receives an on-campus IP packet looks up the layer 3 destination address in its forwarding table, and replaces the source and destination addresses in the outer layer 2 header, and decrements the encapsulation header’s TTL. If the TTL is 0, the packet is discarded. If this Rbridge is the Designated RBridge on the destination’s link, this RBridge removes the encapsulation header and forwards the packet onto the destination’s link.
6. Handling off-campus IP packets

Here, RBridges must forward based on the destination in the original layer 2 header, because the endnode must be able to choose which router to send off-campus packets to. In particular, an IP router must be able to forward to another IP router across the campus.

So such packets are handled the same way as non-IP packets.
7. Handling ARP Queries

If the target address is unknown, initiate a distributed ARP query. If the target address is known, reply with a proxy ARP reply, giving the target’s true layer 2 address.

When initiating a distributed ARP query (or IPv6 neighbor solicitation) remember the address of the requesting node. When the information is discovered, respond to the requester.
8. Issues

8.1 Avoiding encapsulation in some cases

8.1.1 Avoiding encapsulation for on-campus IP packets

In theory, on-campus IP packets need not be encapsulated with an additional layer 2 header. The original layer 2 header can be discarded and replaced with one where the layer 2 destination is replaced by the next RBridge, and the source layer 2 address is replaced by something that will not confuse bridge learning (since packets will be injected into each segment from unpredictable directions because shortest path routes will be used).

The disadvantages of this approach are:

- the IP header’s TTL would be decremented by each RBridge, making the customer aware that bridges have been replaced by RBridges, and possibly breaking IP protocols that expect the TTL not to be decremented over an L2 system
- the original layer 2 addresses might need to be preserved for some conceivable uses

8.1.2 Avoiding encapsulation for off-campus IP packets

Likewise, in theory, off-campus IP packets need not be encapsulated. The TTL in the IP header can be decremented. The same disadvantages as for on-campus IP packets apply, including the concerns on the impact of decremented TTL on other IP protocol behavior. However, there is the additional disadvantage that since the actual layer 2 destination has to be preserved end-to-end there is the danger of packet proliferation if multiple RBridges decide to forward the packet, which can occur while the topology is adjusting.

8.2 Effects on L3 TTL

In general, an Rbridge should have no effect on a Layer 3, e.g., IP TTL field, since the Rbridge is a Layer 2 device. The TTLs which ensure loop-free operation in an Rbridge system should occur in the encapsulation header, and not affect any of the headers of the packet passed through the Rbridge system. The Rbridge should do nothing to transited packets other than that which would be done by an equivalent L2 system.
8.3 Using L3 Encapsulation

Rbridges may use L3, e.g., IP encapsulation to provide a routable internal address and a loop-check indicator. This allows the Rbridge system to use L3 routing algorithms, e.g., OSPF, using existing L3 implementations. As with any Rbridge system, packets are forwarded only within the preconfigured Rbridge system. Intermediate L2 bridges are allowed whether L2 or L3 encapsulation is used. L3 encapsulation processing – including ICMP handling, fragmentation, etc., are well-defined (e.g., RFC2003).

In this case, the L3 encapsulation should not decrement the TTL of the transited packet, since (as per RFC2003) the Rbridge system would not be considered a forwarding (i.e., L3) ‘tunnel’. Further, changing the IP TTL would potentially affect the reachability of all 1’s broadcast or multicast, which would not reach the full L2 subnet.

The primary disadvantage to L3 encapsulation is the increased overhead of encapsulation (e.g., adding both an L3 and subsequent outer L2 header) and complexity of providing L2 services (broadcast notably) within the L3 subnet (RFC1122, RFC1812). Note that L3 supports fragmentation and reassembly for tunnels, notably both for IPv4 and IPv6 encapsulation. Reassembly would be required at the egress, which increases the load on the egress Rbridge in tracking and storing the fragments, but the resulting transited packet is generally transparent to the process. The primary effect would be if there were a large amount of reordering (increasing the reassembly load) or high packet loss (resulting in failed reassembly and thus lost packets). In the latter case, packet loss is amplified because of the lack of fate sharing of the fragments of a single transited packet.

8.4 Topology Issues

It may be possible for an rbridge system to forward the same encapsulated packet over the same physical link multiple times. This could occur when rbridge systems overlap, or when the tunnels of an rbridge system are not explicitly matched to the underlying topology.

In this case, loops are still avoided because internally an rbridge campus uses a loop-free routing protocol, and externally the rbridge campus acts like a single bridge in the outer L2 spanning tree system. The repeating use of individual links may affect performance, but is strictly not avoidable and does not affect correctness.
9. 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.
10. 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.
11. Acknowledgments

We anticipate that many people will contribute to this design, and invite you to join the mailing list at http://www.postel.org/rbridge
12. References

12.1 Normative References


12.2 Informative References

November 1990.


Authors’ Addresses

Radia Perlman
Sun Microsystems

Phone:  
Fax:  
EMail: Radia.Perlman@Sun.COM
URI:

Joe Touch
USC/Information Sciences Institute

Phone:  
Fax:  
EMail: touch@isi.edu
URI:

Alper Yegin
Samsung Advanced Institute of Technology

Phone:  
Fax:  
EMail: alper.yegin@samsung.com
URI: