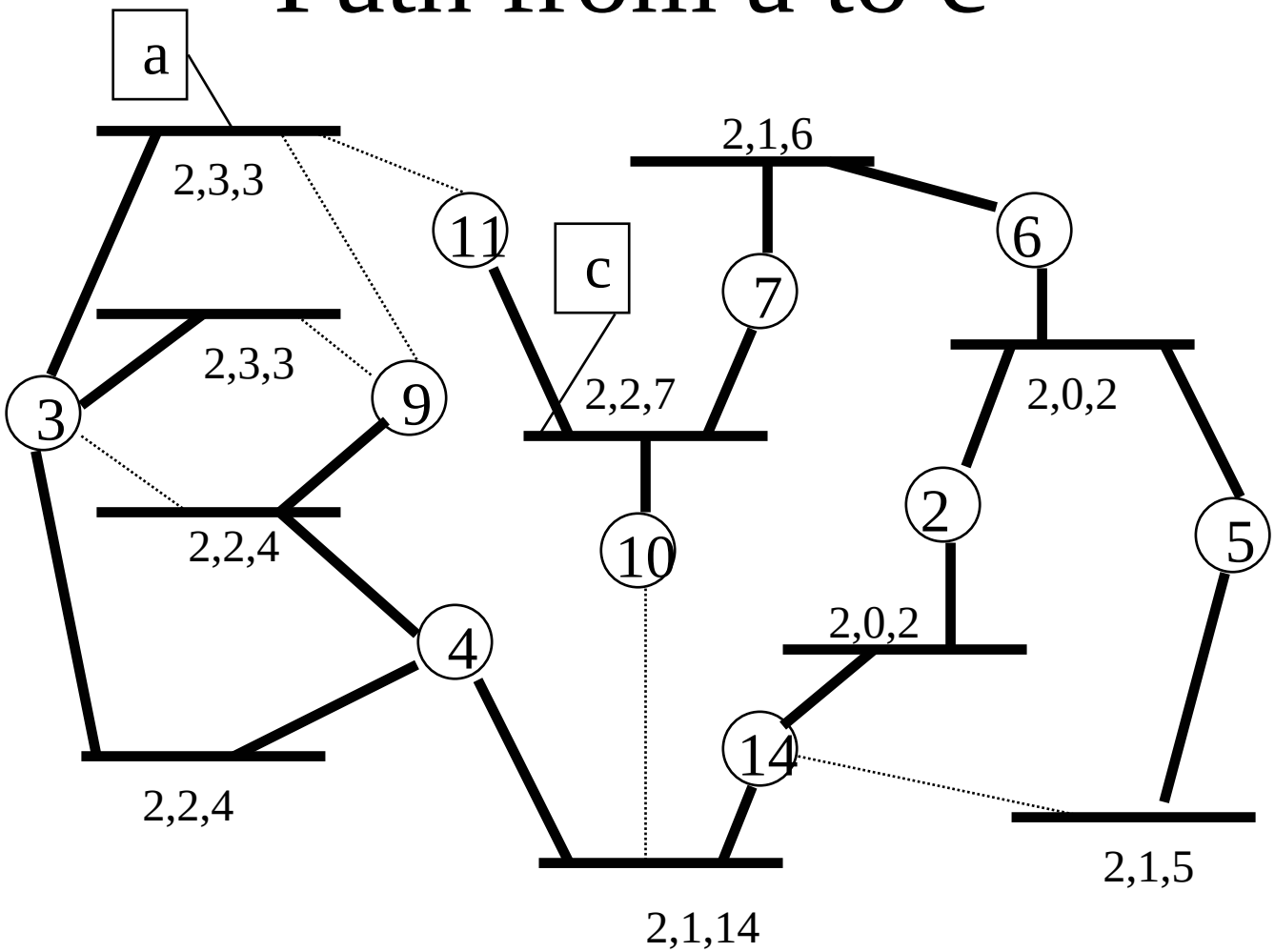
Rbridges: Transparent Routing

Radia Perlman
radia.perlman@sun.com

Problems with Bridges

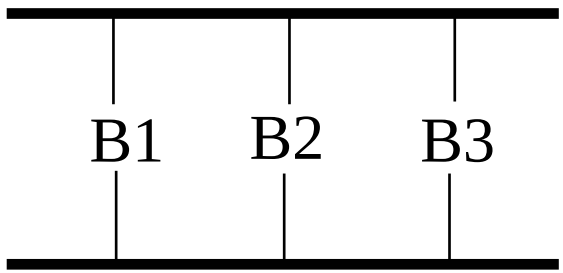
- Routes are not optimal (spanning tree)
  - STA cuts off redundant paths
  - If A and B are on opposite side of path, they have to take long detour path
- Temporary loops really dangerous
  - no hop count in header
  - proliferation of copies during loops
  - So, should be conservative in transition

Path from a to c



Why loops are a disaster

- No hop count
- Exponential proliferation



Why bridges are slow to start forwarding

- Temporary loops might cause meltdown
- Can’t (except in certain special cases, like a port to an endnode) know if turning on a link might cause temporary loop
- Simple solution: wait before turning on link, so other bridges can turn off links first
- People want instant failover (but they don’t want meltdowns)

Bridge meltdowns

- They do occur (a Boston hospital)
- Lack of receipt of spanning tree msgs tells bridge to turn on link
- So if too much traffic causes spanning tree messages to get lost…
  - loops
  - exponential proliferation of looping packets

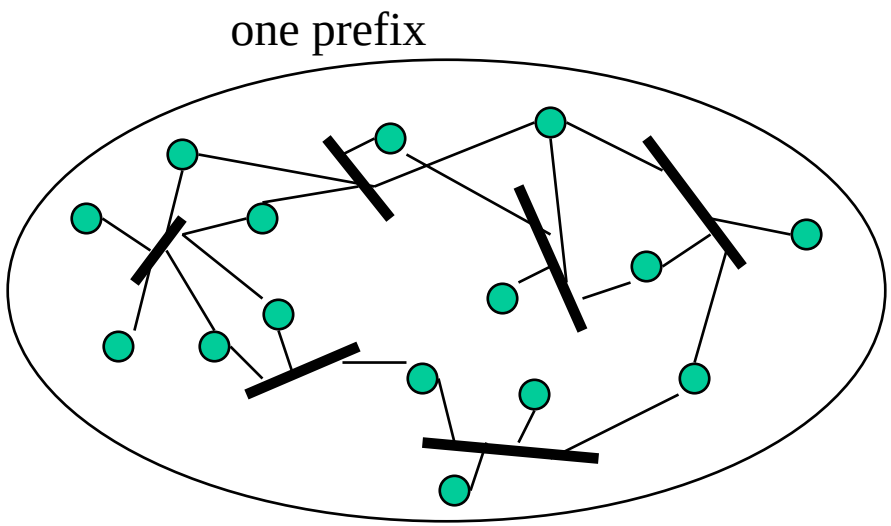
Why are there still bridges?

- Why not just use routers?
  - Bridges plug-and-play
  - Endnode addresses can be per-campus
- IP routes to links, not endnodes
  - So IP addresses are per-link
  - Need to configure routers
  - Need to change endnode address if change links

True “level 1” routing

- CLNP addresses had two parts
  - “area” (14 bytes…)
  - node (6 bytes)
- An area was a whole multi-link campus
- Two levels of routing
  - level 1: routes to exact node ID within area
  - level 2: longest matching prefix of “area”

CLNP areas



CLNP level 1 routing

- Depended on protocol “ES-IS”
  - endnodes periodically multicast presence to rtrs
  - (also, rtrs periodically multicast to endnodes)
- Rtrs tell each other, within area, location of all endnodes in area
- IS-IS originally designed for CLNP. “Level 2” was to longest prefix. “Level 1” was to exact match of bottom 6 bytes.

“Level 1 routing” with IP

- IP has never had true level 1 routing
- Each link has a prefix
- Multilink node has two addresses
- Move to new link requires new address
- Bridging is used to create a campus in which all nodes share the same prefix
- But bridging isn’t as good as routing

What we'd like, part 1: replace bridging with Rbridging

- keep transparency to endnodes
- keep plug-and-play
- have best paths
- eliminate problems with temporary loops
  - have a hop count
  - don't exponentially proliferate packets
- then can converge optimistically (like rtrs)

What we'd like, part 2: true "level 1 routing" for IP

- allow plug-and-play campus sharing a prefix
- allow optimal routing
- don't require any endnode changes (e.g., implement ES-IS)
- work for IPv4 and IPv6

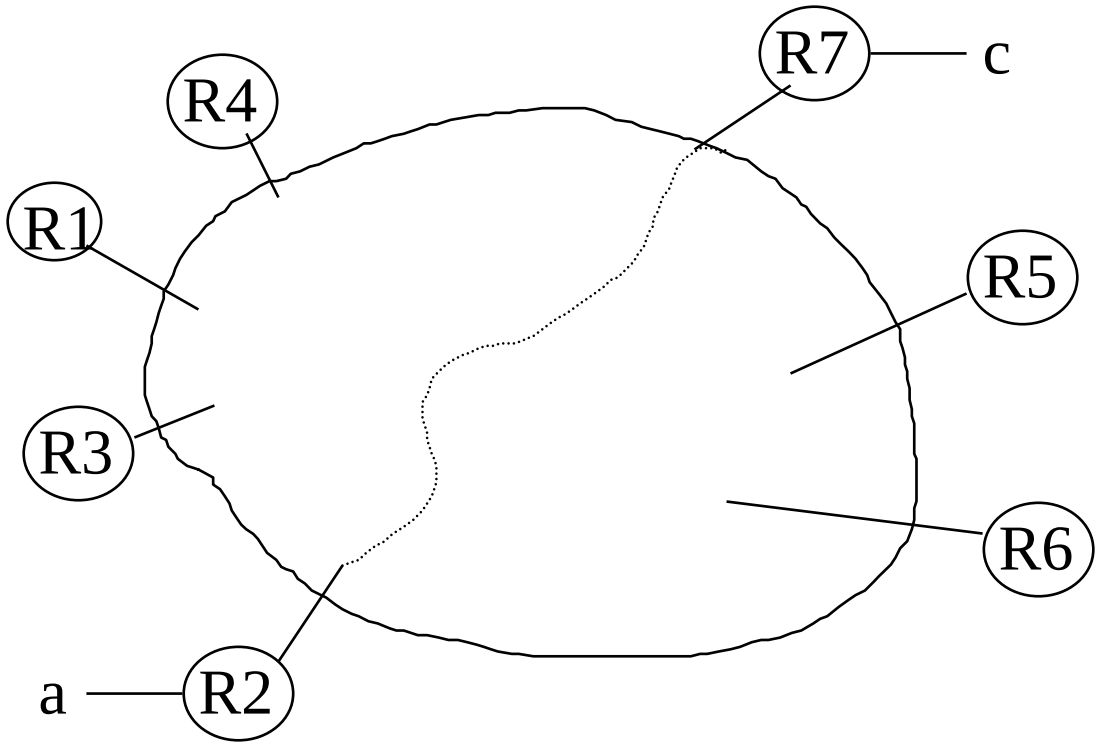
Rbridges

- Compatible with today's bridges and routers
- Like routers, terminate bridged LAN
- Like bridges, glue LANs together to create one IP subnet (or for other protocols, a broadcast domain)
- Like routers, optimal paths, fast convergence, no meltdowns
- Like bridges, plug-and-play

Rbridging layer 2

- Link state protocol among Rbridges (so know how to route to other Rbridges)
- Like bridges, learn location of endnodes from receiving data traffic
- But since traffic on optimal paths, need to distinguish originating traffic from transit
- So encapsulate packet to destination Rbridge

Rbridging



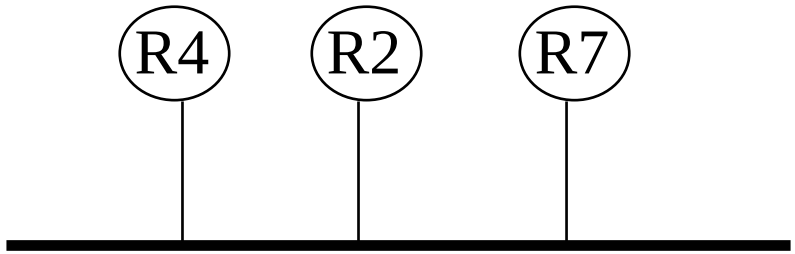
Encapsulation Header

S=Xmitting Rbridge D=Rcving Rbridge pt="transit"	hop count	original pkt (including L2 hdr)

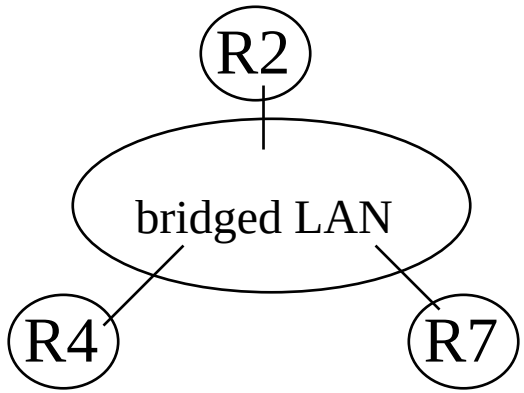
- Outer L2 hdr must not confuse bridges
- So it's just like it would be if the Rbridges were routers
- Need special layer 2 destination address
  - for unknown or multicast layer 2 destinations
  - can be L2 multicast, or any L2 address provided it never gets used as a source address

Rbridges and Bridges

Seems like:



Actually can be:



Endnode Learning

- On shared link, only one bridge (DR) can learn and decapsulate onto link
  - otherwise, a “naked” packet will look like the source is on that link
  - have election to choose which Rbridge
- When DR sees naked pkt from S, announces S in its link state info to other Rbridges

Pkt Forwarding

- If D known: encapsulate and forward towards D
- Else, send to “destination=flood”, meaning send on spanning tree
  - calculated from LS info, not sep protocol
  - each DR decapsulates

Rbridging IP

- Rbridging at layer 2 will do it
- Optimization: locally answer ARPs
  - learn (layer 3, layer 2)
  - pass that in link state info
- Another optimization for IP: shorter endnode cache timer (since can ping)

Alternative for IP

- Some router hardware doesn’t like to learn on data packets (“fast path”)
- Encapsulation not too desirable
- For IP packets, we can avoid both the above
- Forward like IP, using IP hdr
  - learn from ARP replies
  - decrement hop count in IP hdr
  - L2 hdr: Rbridge to Rbridge

Avoiding encapsulation for IP

- On-campus IP destination
  - forward based on IP header
  - learn from ARP replies
  - if destination unknown, flood ARP query
- Off-campus IP destination
  - forward based on layer 2 destination

Conclusions

- Looks to routers like a bridge
  - invisible, plug-and-play
- Looks to bridges like routers
  - terminates spanning tree, broadcast domain

Conclusions, cont’d

- Much better replacement for bridging
  - optimal paths
  - still plug and play and transparent
  - fast convergence
  - no meltdowns
- For IP
  - allows plug-and-play single-prefix campus