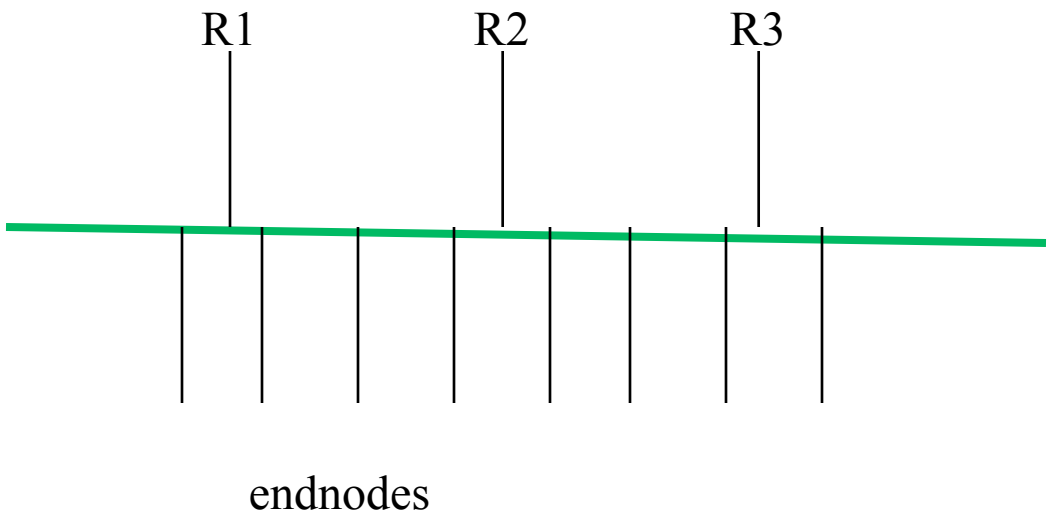
Overview of TRILL Active-Active Goals, Challenges, and Proposed Solutions

Radia Perlman
radiaperlman@gmail.com

Don’t we have “appointed forwarders”?



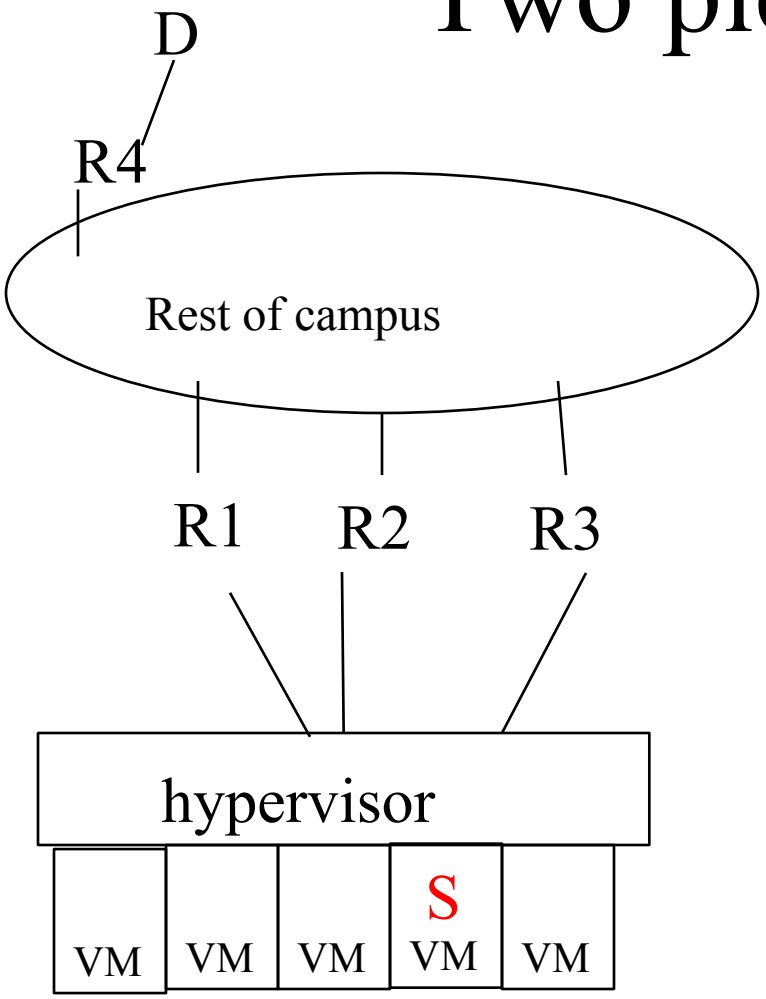
R1, R2, and R3 can all forward. DRB assigns work, based on VLANs
But requires R1, R2, R3 all carefully coordinated, all see all packets

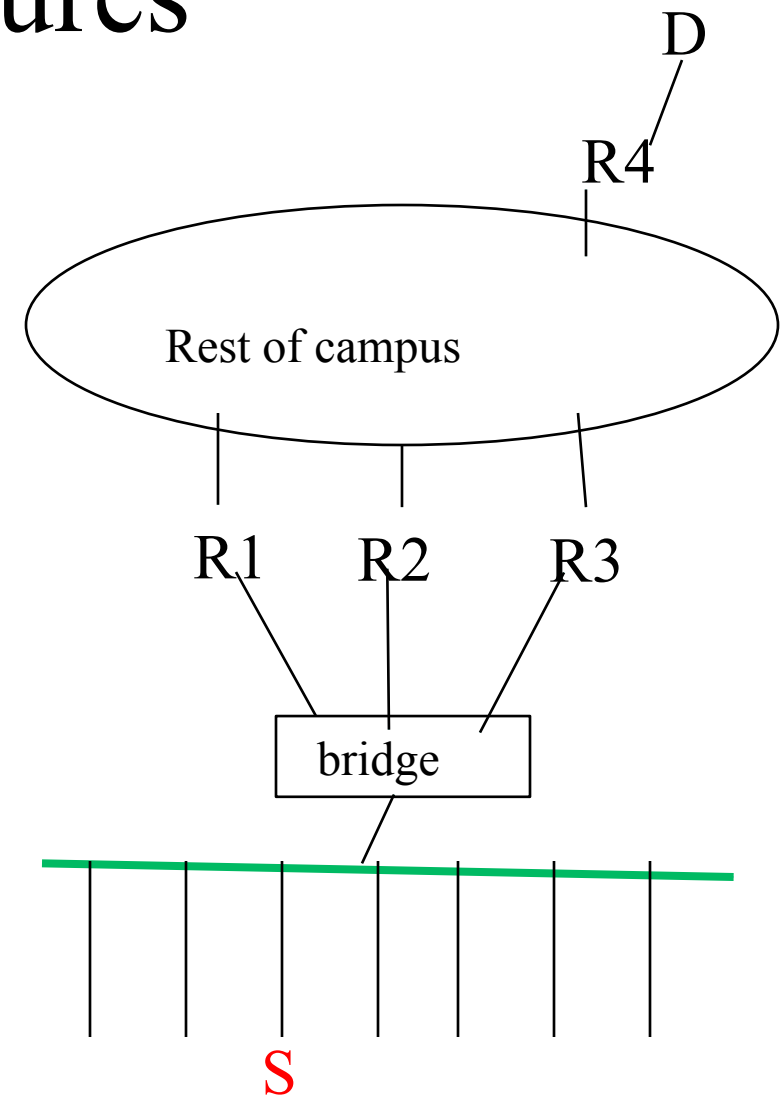
The active-active stuff being discussed in different

The “active-active” stuff

- Multiple RBridges, {R1, R2, R3} attached to a bunch of endnodes
- But when a packet is forwarded from the bunch of endnodes, only one of {R1, R2, R3} sees that packet
- And {R1, R2, R3} cannot easily talk to each other (they are not on a common link)

Two pictures





(Presumed) Rules for forwarding upwards

- Same “flow” go to the same next hop
- Otherwise, basically random
- Multidestination might go to any of the next hops
- And nothing is forwarded (by hypervisor or bridge) between the up-links

Goals (“would be nice”)

- Probably we won’t find any solutions that meet all the goals

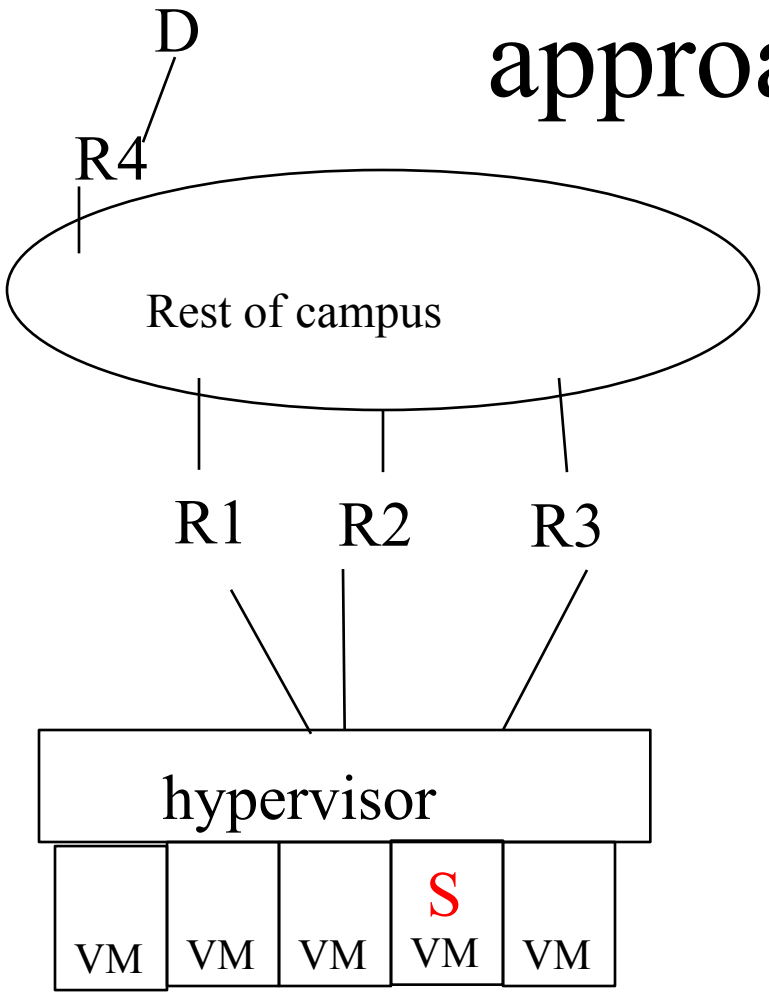
Goals ("would be nice")

- All up-links active
- If S sends to distant node D, D → S traffic should enter S via link closest to D, regardless of which up-link was used for path S → D
- Have D → S traffic take same path as S → D traffic (note: directly conflicts with above goal)
- R4 (or D) shouldn't keep changing its mind about which RB S is connected to
- Packets for a flow should stay in order

Goals ("would be nice")

- No need to change entire campus at once (perhaps only need to change {R1, R2, R3}, maybe R4.
- Works with all existing silicon
- RPF check on multi-destination works (doesn't falsely drop packets)

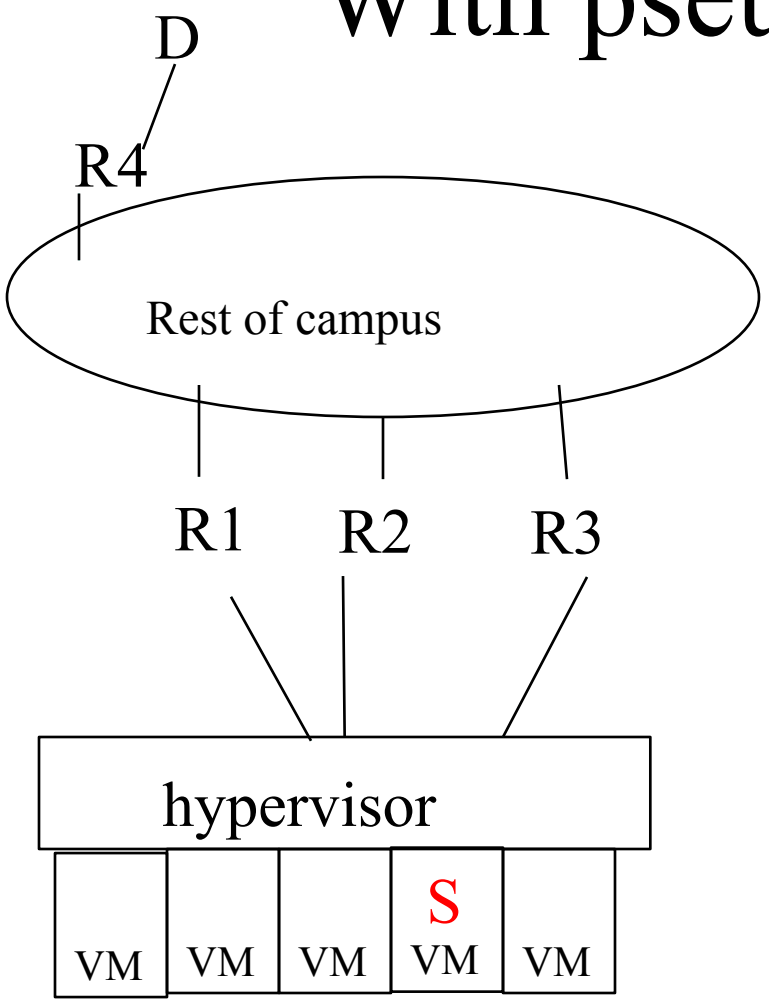
What's wrong with naïve approach?



When S sends via R1, "first RB" field=R1, etc.

Problem: R4 will return via same up-link (possibly not optimal one), and R4 will keep switching its endnode table for S if traffic from S comes via R2, R3,…

With pseudonode



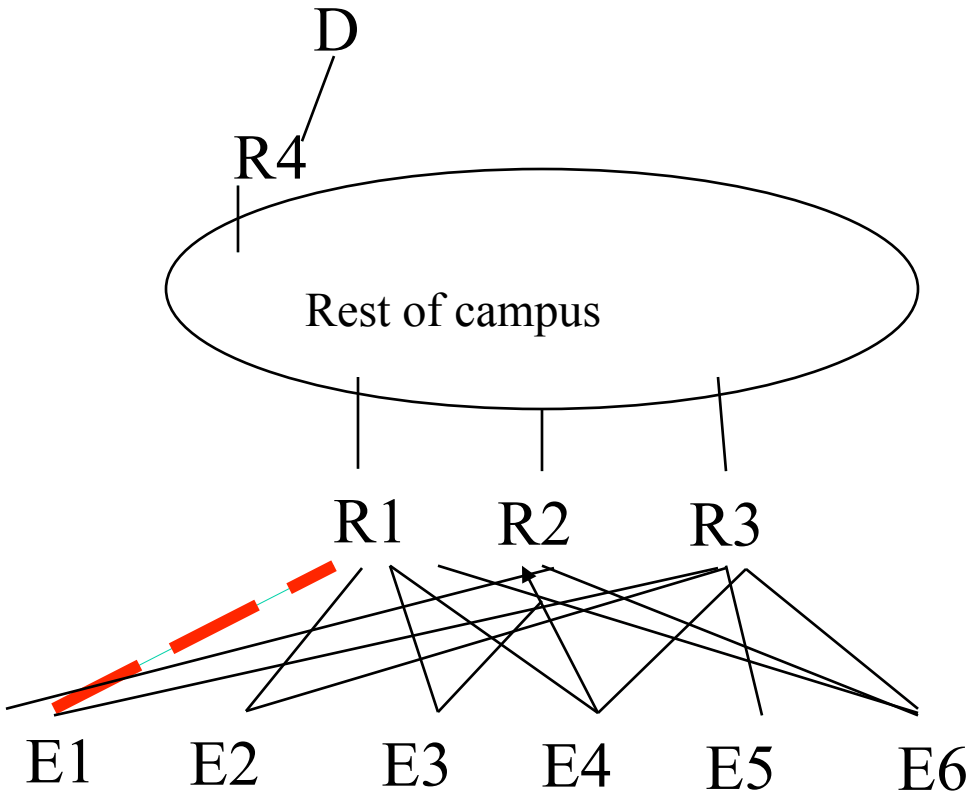
R1, R2, R3 agree (somehow) on a pseudonode nickname for the set of MACs reachable from {R1, R2, R3}, let's say "79"

Always encapsulate with 1st RB=79

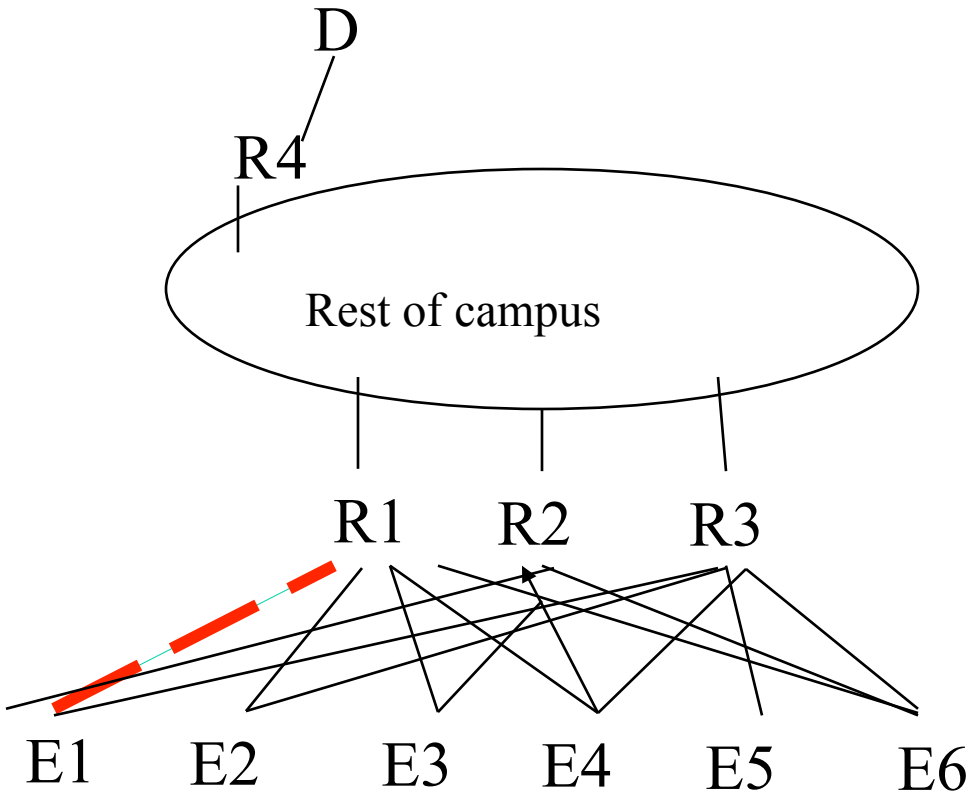
Pseudonode

- R1, R2, R3 claim they are attached to “79”
- Use “79” as ingress when receive from their uplink
- All endnodes attached to R1, R2, R3 look like they are reachable via 79

Problem: What if E1's link to R1 dies?

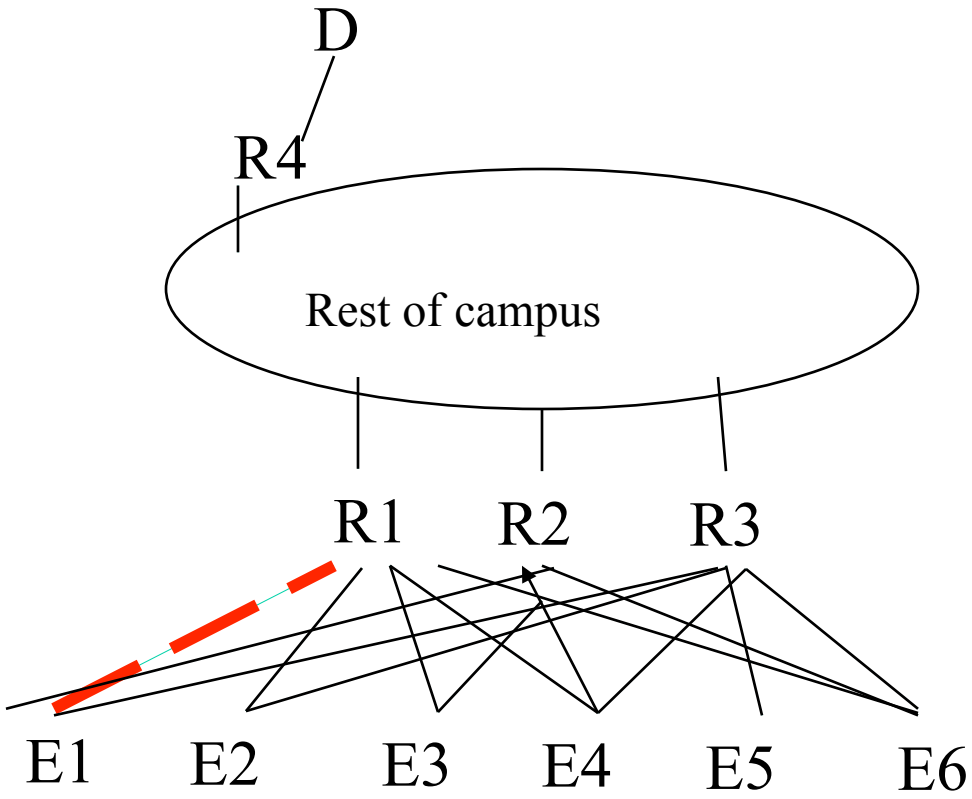


Problem: What if E1's link to R1 dies?



If R3 uses pseudonode "79" when sending to D, return traffic to E1 might go via R1, and fail

Problem: What if E1's link to R1 dies?

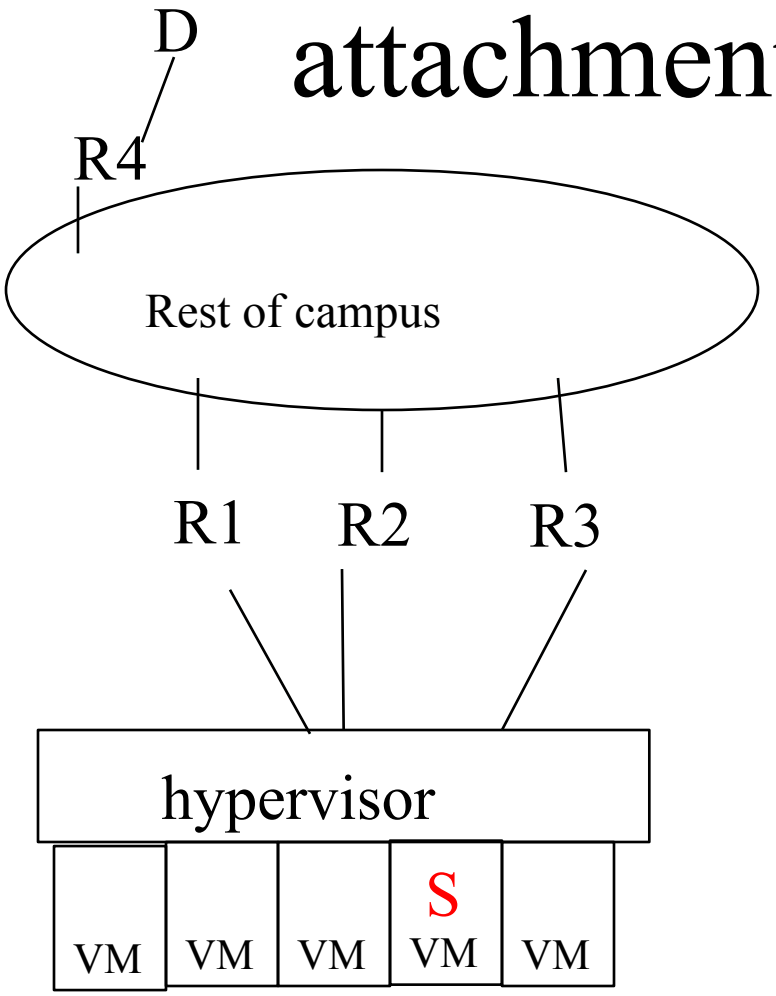


How could R3 detect this, even if there was something sensible for it to do?

Solutions?

- Ignore the problem: “hardly ever happens”
- Have R1 notice somehow and tunnel traffic for E1 to R2 or R3 (even though R1 is still connected to “79” for other endnodes)
- Don’t use pseudonode, and have distant RBs learn multiple addresses for each E, as in
  - E1 reachable via R2 (timestamp), and R3 (timestamp)
  - E2 reachable via R1 (timestamp), R2 (timestamp), and R3 (timestamp)
- ??? Any other possibilities?

Picture for "learn multiple attachments for S"



R4 keeps, for S
S located at
- R1/ last seen T1
- R2/ last seen T2
- R3/last seen T3

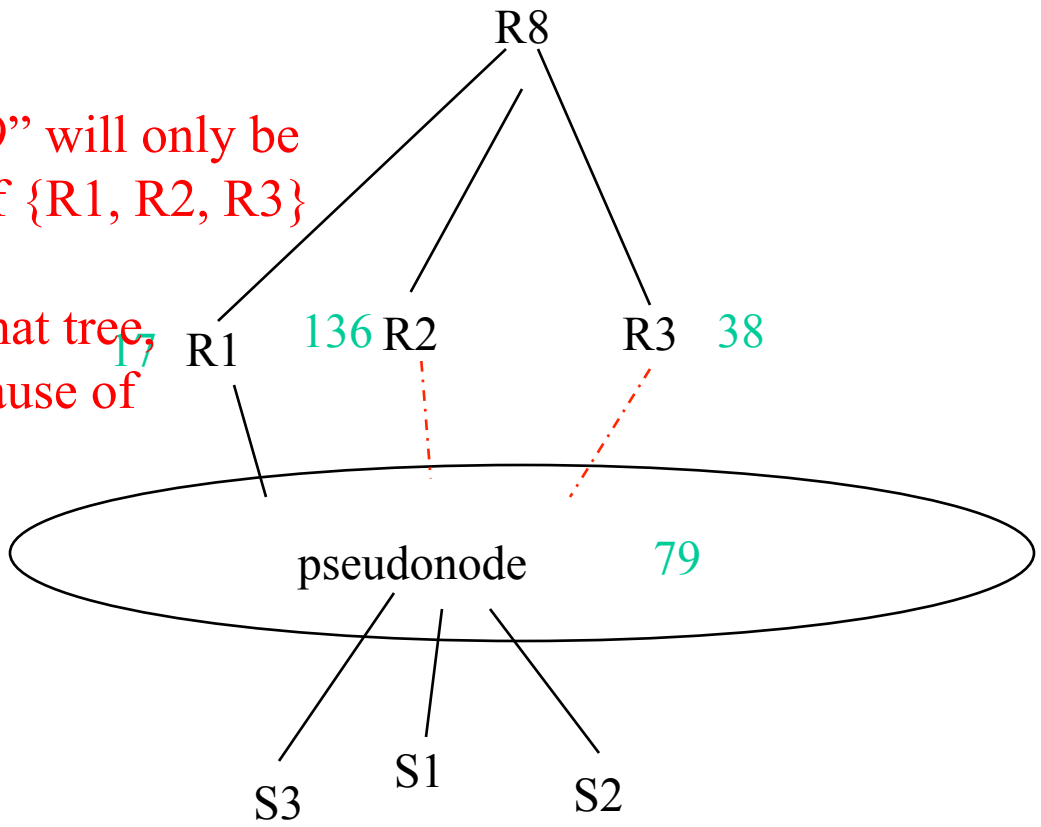
And R1, R2, R3 don't use "79", they use their own nicknames

Another problem with pseudonode: RPF check on multicast

Multidestination frames, pseudonode nickname, and the RPF check

For each tree, “79” will only be attached to one of {R1, R2, R3}

If R2 injects on that tree, R8 will drop because of RPF check



Potential solutions (assuming R3 not attached to any tree, but R1 and R2 are)

- R3 refuses to use link(s) to “79” at all (disables its port)
- R3 continues to work, but only for unicast; if a packet must be multidestination from “79”, R3 tunnels to R1 or R2
- On multicast, R3 sends, but uses “R3” for ingress instead of “79”
- Use a bit in the TRILL header to mean “I’m in multiple places” (turn off MAC flip-flop panic, or keep multiple RB attachments)
- Let’s look at pros and cons of each approach

R3 disables the port completely

- Pro: simple
- Con: very sad

R3 tunnels multidestination to R1 or R2

- Pro:
  - Simple
  - Doesn’t change anyone except {R1, R2, R3}
- Cons:
  - Maybe some silicon doesn’t support this?
  - Extra hops

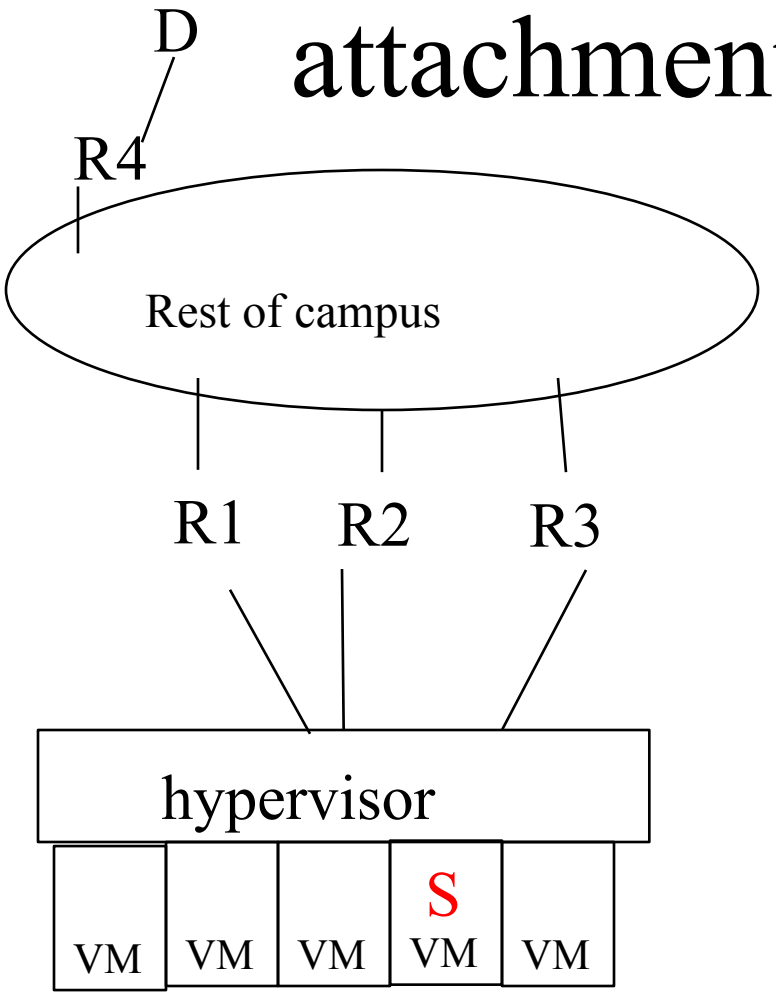
R3 uses its own nickname for multidestination ingress

- Pros
  - Simple
  - Doesn't change anyone except {R1, R2, R3}
- Cons
  - (distant) R8 sometimes learns (S,79) (on unicast), sometimes (S,R3) (multidestination through R3)

No pseudonode; learn multiple attachments

- Change (all) edge RBs to cope with E being attached to multiple places (R1, R2, and R3)
- Keeps separate timestamp for each learned attachment
- When sending to S, choose any (say nearest, or load split based on flows) of R1, R2, R3

Picture for "learn multiple attachments for S"



R4 keeps, for S
S located at
- R1/ last seen T1
- R2/ last seen T2
- R3/last seen T3

And R1, R2, R3 don't use "79", they use their own nicknames

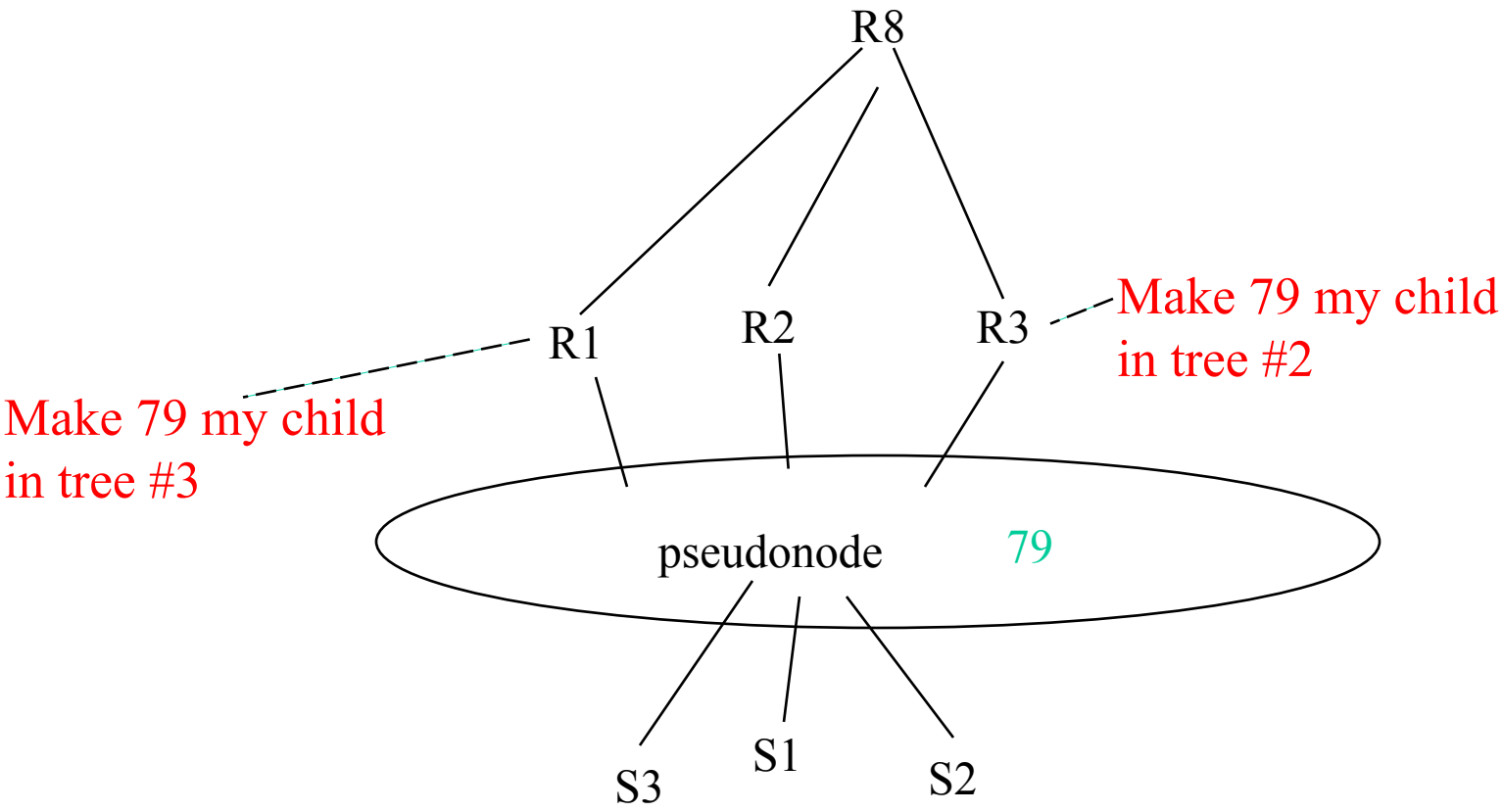
Learn Multiple attachments

- Pros
  - Less, or no, configuration required
  - Allows {R1, R2, R3} to use any multicast tree
  - No problem if E1's uplink to R1 fails
- Cons
  - Requires edge RBs to keep track of multiple attachment points for endnodes; and separately time them out; disable flip-flop panic

What's the "affinity" thing?

- It's a new TLV in IS-IS that says "for these trees, put this nickname as a child of me"

What does it do and what doesn't it do?



What does it do?

- If you have at least as many trees as up-links…
- And you configure everything properly…
- And all RBs in the campus implement this new thing…
- You will be assured that each of the uplinks has at least one tree to send on

What does it not do?

- Still have the problems mentioned earlier in the presentation
  - if fewer trees than uplinks
  - if misconfiguration
  - If one of the uplinks from some set of endnodes fails
- Requires as many trees as active uplinks. Each tree requires significant state and computation

And note: It requires all RBs in the campus to understand this new TLV and compute trees accordingly

Questions from me

- How many trees do people want?
- How many uplinks do people want?
- Do we care if an RB can't use all the campus trees?
- Do we care about misconfiguration?
- Are we worried about the problem of some uplinks failing?

Conclusions

- Lots of different aspects, and nothing addresses all of them at the same time…we can do mix and match
- No perfect solution