Multilevel TRILL

draft-perlman-trill-rbridge-multilevel-00.txt

Radia Perlman

radiaperlman@gmail.com

Intel Labs

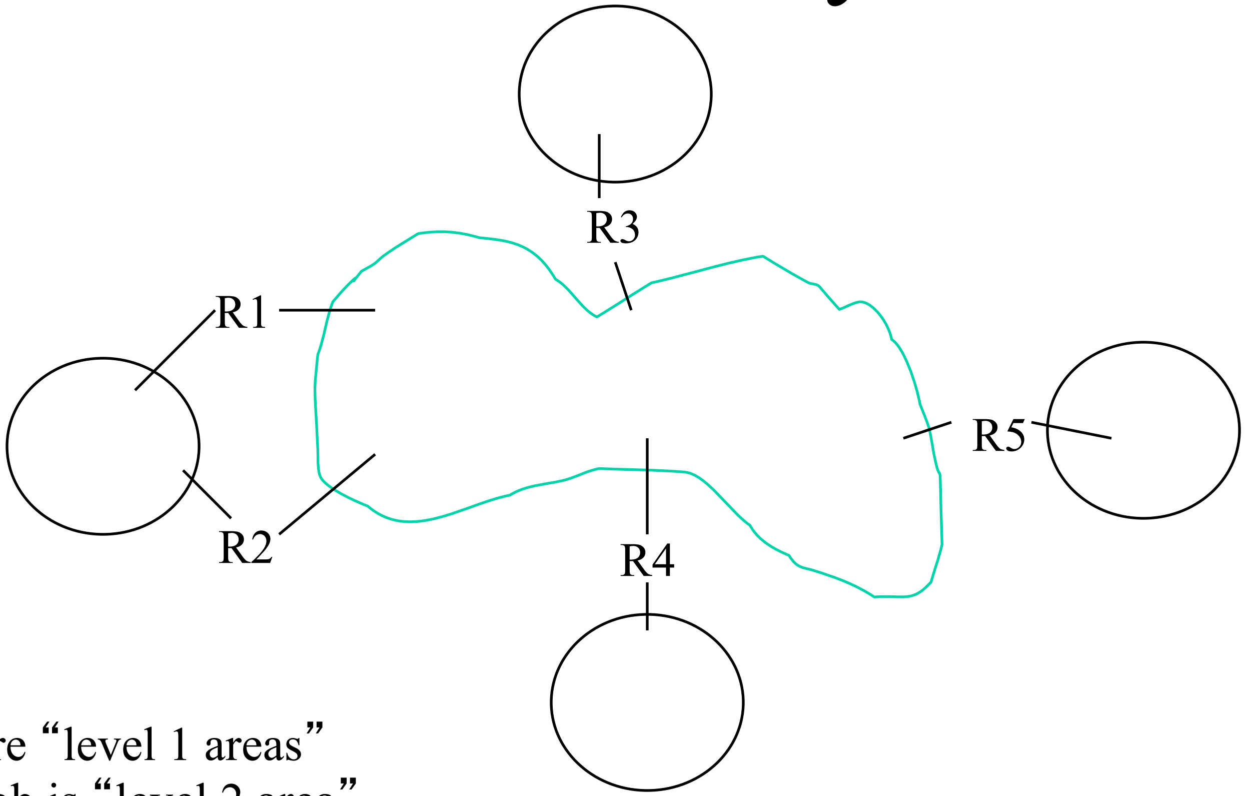
Potential scalability issues

- routing computation load
- volatility of LSP database
  - too much control traffic
  - database in unconverged state too often
- size of LSP database (too much memory)
- running out of nicknames
- size of broadcast domain using up capacity
- size of endnode learning table (MAC,nickname)

The red ones are not addressed by multilevel

- routing computation load
- volatility of LSP database
  - too much control traffic
  - database in unconverged state too often
- size of LSP database (too much memory)
- running out of nicknames
- size of broadcast domain using up capacity
- size of endnode learning table (MAC,nickname)

Hierarchy



circles are "level 1 areas"
green blob is "level 2 area"
connection between areas is through "border RBridge", attached to both levels
Level 2 area must be physically intact

Area Addresses

- IS-IS messages have “area address” field
- TRILL says “must be zero”
- Area address had special significance in CLNP, for which IS-IS was originally designed
- For TRILL, the only reason is to ensure two level 1 areas don’t get accidentally merged

Alternative strategies for dealing with area addresses?

- Change TRILL to “area address is defaulted to zero”
- Leave “area address” field as zero, and invent new TLV, ignored by old RBs, that can be used in case two new RBs are interconnected
- Don’t worry about accidental interconnection of areas

Some things to worry about

- Assigning nicknames
- Advertising reachability of external information
- Advertising filtering information across areas (VLAN, IP multicast reachability)
- Computation of multi-area trees
- Computation of RPF information
- Compatibility with old RBs

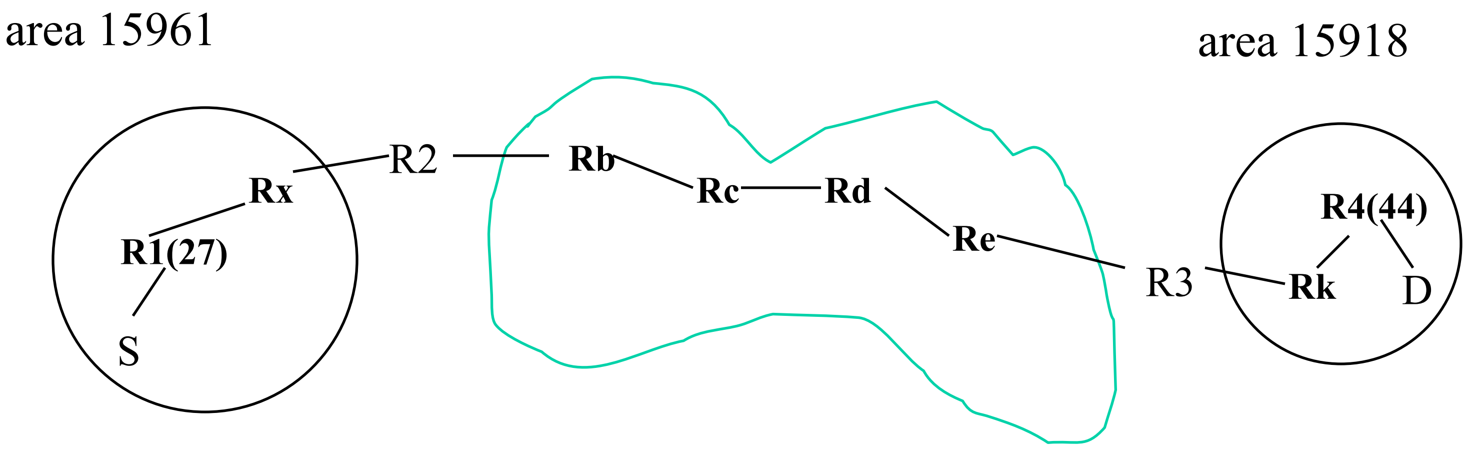
What does “compatibility” mean?

- Ideally, just new RBs need to be aware of multilevel
- Ideally the data plane should not change (e.g., format of forwarding table, how to look up RPF information on multidestination frames, etc.)
- Perhaps special features in border RBs

Aggregated or Unique Nicknames

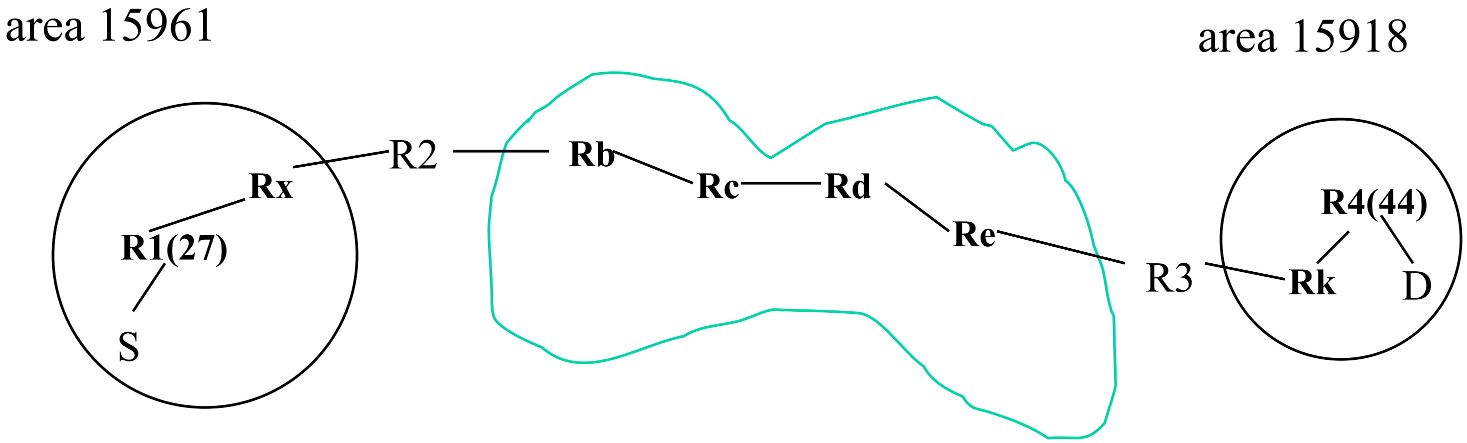
- I'll describe two approaches
  - Unique nicknames: Each RB in entire campus has a unique nickname
  - Aggregated nickname: All nicknames in an area are represented outside the area as a single aggregated nickname

How aggregated nicknames work



S transmits packet to D
R1 encapsulates with TRILL header
ingress=27, egress=15918
R2 replaces ingress with 15961
R3 replaces egress with 44

How aggregated nicknames work



S transmits packet to D
R1 transmits ingress=27, egress=15918
R2 transmits ingress=15961, egress=15918
R3 transmits ingress=15961, egress=44

What if D is unknown?

- If unknown by R2, need to do multi-area multidestination frame
- If R1 did know the right area, but R3 doesn’t know D, then R3 needs to turn it into “unknown unicast” within D’s area

Designated Border RB (DBRB)

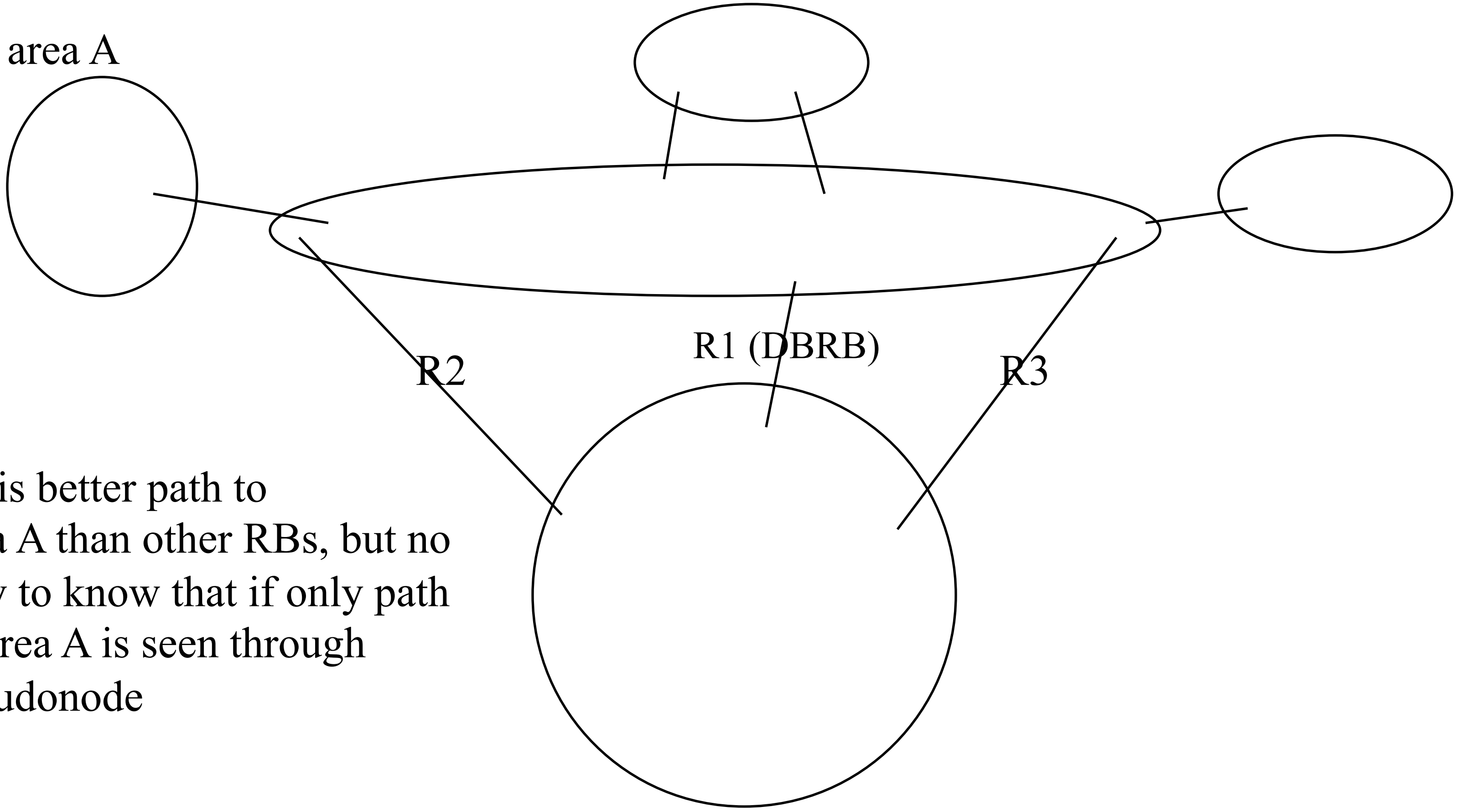
- Need to elect a single DBRB for each area
- Can’t be usual “Hello” mechanism, since border RBs in area A need not be actual neighbors
- So, do that through LSP database
  - Advertise “I am a border RB, with priority x”
- DBRB (R1) gives area a pseudonode
  - R1 announces pseudonode representing area A into level 2
  - R1 announces pseudonode representing entire rest of world, into level 1 area A
  - Announces, with pseudonode, all relevant information (reachable VLANs, IP multicast information, all external nicknames

More optimal (unicast) paths to other areas

- Especially since all the border RBs are not physically on the same link, it could be that some RBs are far better paths to certain areas than the DBRB (pseudonode)
- If the only information about reachability of outside information is through the pseudonode, then won't find much more optimal paths to other areas

Optimal paths, cont' d

- Obviously, hierarchy hides best paths
  - there' s best paths to area, and best paths to individual RBs within area
  - there' s a tradeoff between optimal paths and scalability of information
- But, especially in aggregated nickname case, it wouldn' t be unreasonable to have some border RBs announce their (much better) path to certain areas, in addition to having DBRB announce reachability of all areas through the pseudonode



R2 is better path to
area A than other RBs, but no
way to know that if only path
to area A is seen through
pseudonode

Optimal rts vs scalability

- Most scalable:
  - No external nicknames appear in link state database… just “I am a border RB”
  - If nickname not in your area, send to nearest border RB
- Most optimal rts: each border RB in area A:
  - reports into l2, cost to each area A nickname
  - reports into A, cost to each external nickname (or, its cost to each border RB in other areas, and flood into A, LSPs from each border RB in other areas, what their cost is to each nickname inside their areas)

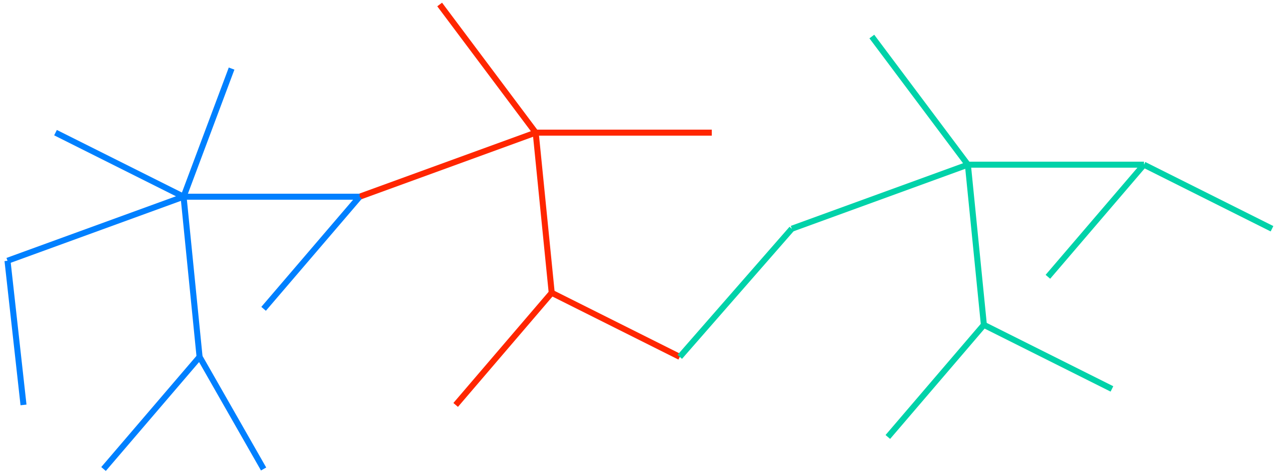
My preference (probably)

- I’d go with more scalable, with a compromise of
- DBRB (R1) reports all information on pseudonode, and another border RB (R2) in area A only reports cost to external nicknames if:
  - R2 is configured to report that nickname, or perhaps
  - The cost from R2 to that nickname is much shorter (with “much” a parameter) than path reported through pseudonode

Trees

- Note that if each area computes a tree, and exactly one border RB connects an area to level 2, that the result will be a multilevel tree

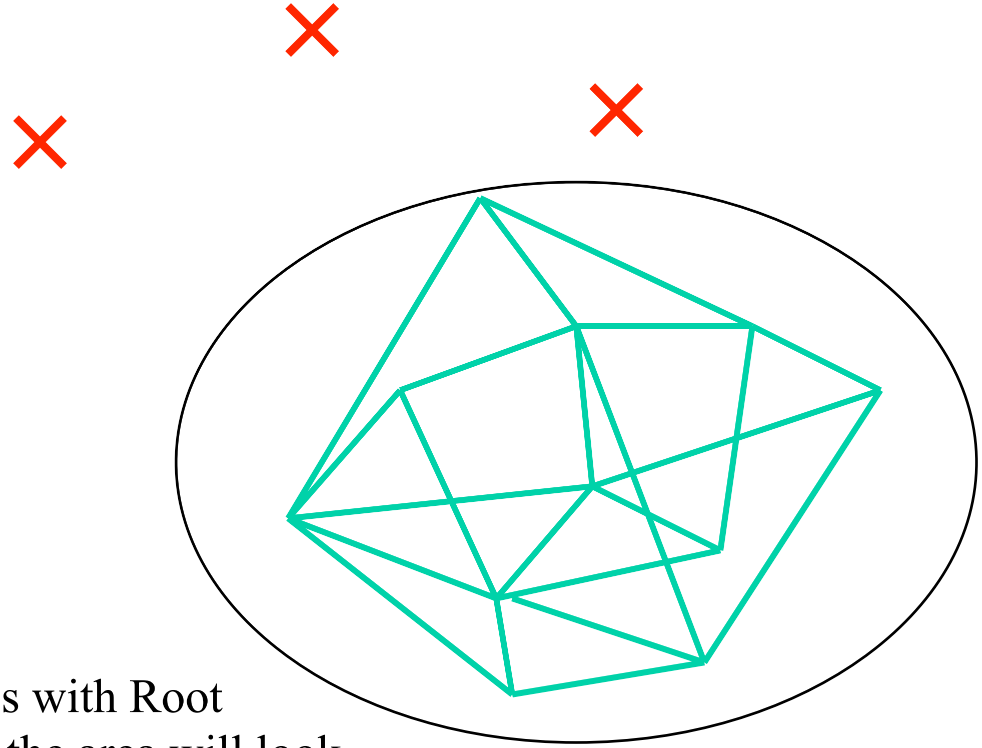
Multiarea trees



Tree Roots

- For maximum traffic spreading, you want to be able to choose multiple roots within the area
- If multilevel campus computed, say, 3 trees, and roots are all outside area A, then area A will have 3 trees all rooted at border RB

Tree roots



All trees with Root
outside the area will look
like they are rooted at the Border RB

Proposal

- Have each area calculate some number of trees, all rooted in that area
- The first RB chooses a tree (just like today)
- The border RB of that area has a mapping from (level 1 tree, level 2 tree), and replaces the “egress” field with a tree rooted in layer 2
- The border RB of each area maps the level 2 tree to a level 1 tree, rooted in the destination area

RPF state

- What R1 does for a TRILL tree T
  - R1 calculates which ports are in tree T
  - Each potential ingress RB advertises which trees it might choose as ingress
  - R1 places each potential ingress for T on one of R1's tree T ports
- For aggregated nicknames, this means total RPF state on a port R1 needs is at most (size of area + # of areas)
- Unique nicknames: total RPF state R1 needs on a port is potentially size of entire campus

Comparison

- Information needed to be passed into the area
  - aggregated nickname: one nickname per area
  - unique nickname: total # of RBridges, which of those will advertise which trees
- RPF state (previous slide)
- Forwarding table (size of area + # of areas, vs total # of RBs)

A potential way of somewhat taming unique nickname state

- Summarize with prefixes, or ranges
- This does change the way the forwarding table/RPF state would work (data plane change)
- And does further eat up nicknames
- And requires more configuration (since it's hard to change the ranges for an area if the area gets larger and needs more nicknames)

Another issue

- Suppose there are multiple border RBs
- Which of those will transition multidestination frames?
- R1, in center of area A, needs to know which border RB transitioned a frame, so as to properly calculate RPF state for all frames originating outside the area

My preference

- Have only a single RB transition all multidestination frames
- I don’t think it makes that much difference in terms of spreading load…there can still be lots of trees within the area

Alternative

- DBRB has to announce, in its LSP, what assignment it has made as to which border RBs will transition which multidestination frames
- If it' s per tree, then the RPF state is not changed from today
- If it' s per VLAN, or something else, then the RPF state looks different from today.
- Again, not sure why it would be important to spread the load of which border RB injects the multidestination frame

Summary

- Hopefully I’ve covered all the subtle issues to think about
  - – RPF state
  - – multiarea trees
  - – balancing act between optimality of routes vs control information