Problem statement and architecture draft

TBD

Presented by
Erik Nordmark
since TBD isn't available

Background

- Our charter calls for a problem statement and architecture document
  - To clarify to ourselves as well as to those outside the WG
- We currently do *not* have an initial draft for such a document
  - Looking for a volunteer author/editor
- This presentation tries to jumpstart this
- Does it cover what is important without unecessary detail?

Trill Overview

- Problem Statement
- Architecture
- Threat Analysis
- Impact on LAN service model, if any

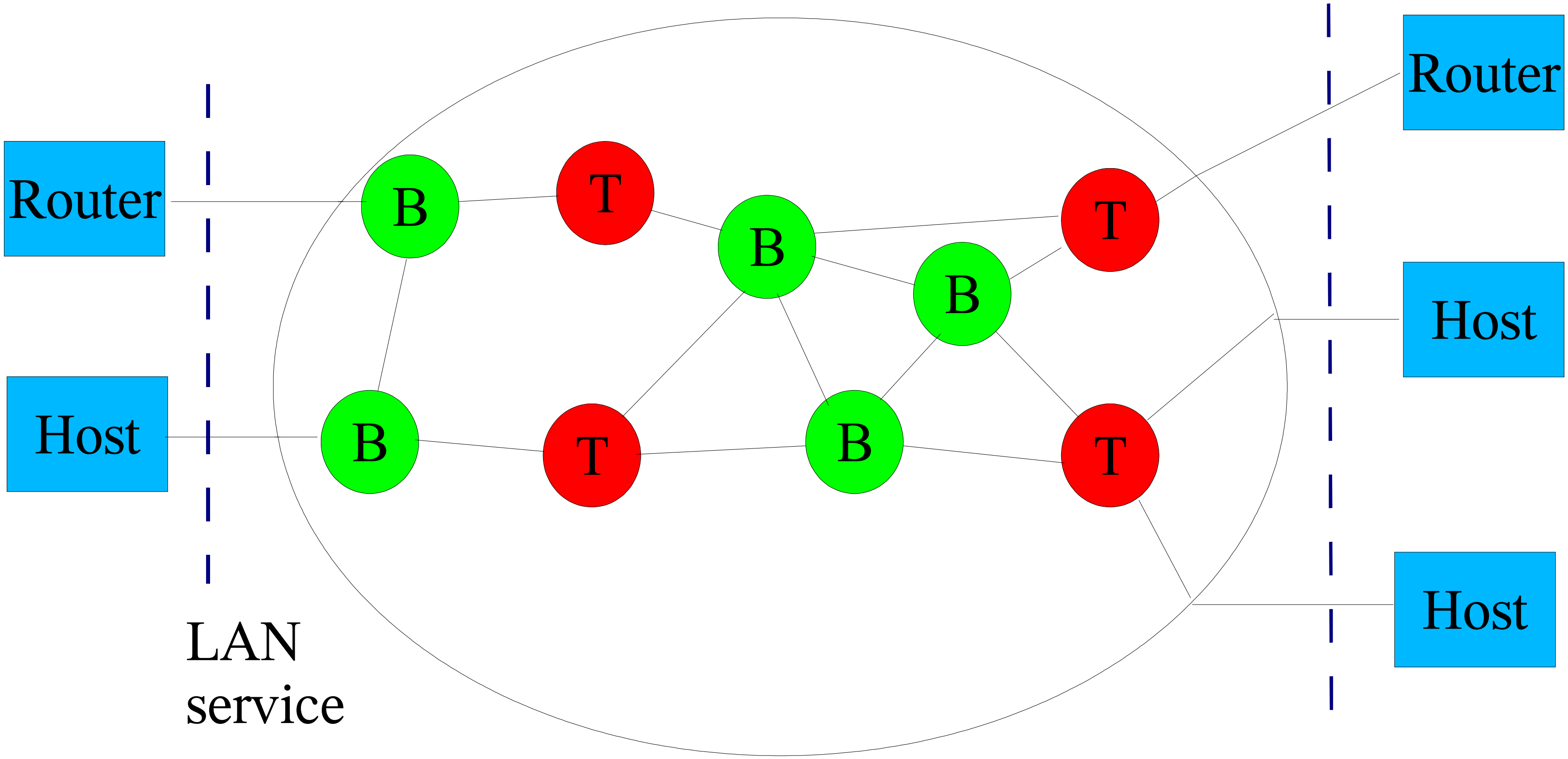
Problem Statement

- The TRILL WG will design a solution for shortest-path frame routing in multi-hop IEEE 802.1-compliant Ethernet networks with arbitrary topologies, using an existing link-state routing protocol technology.
- Initially be based on draft-perlman-rbridge-03.txt.

Properties

- The design should have the following properties:
  - Minimal or no configuration required
  - Load-splitting among multiple paths
  - Routing loop mitigation (possibly through a TTL field)
  - Support of multiple points of attachment
  - Support for broadcast and multicast
  - No significant service delay after attachment
  - No less secure than existing bridged solutions

Transparent for hosts, routers, and bridges



Architecture

- Forwarding based on safe header
- Coexist with existing bridges
- Trill core runs a link state routing protocol
- Elements to address scalability

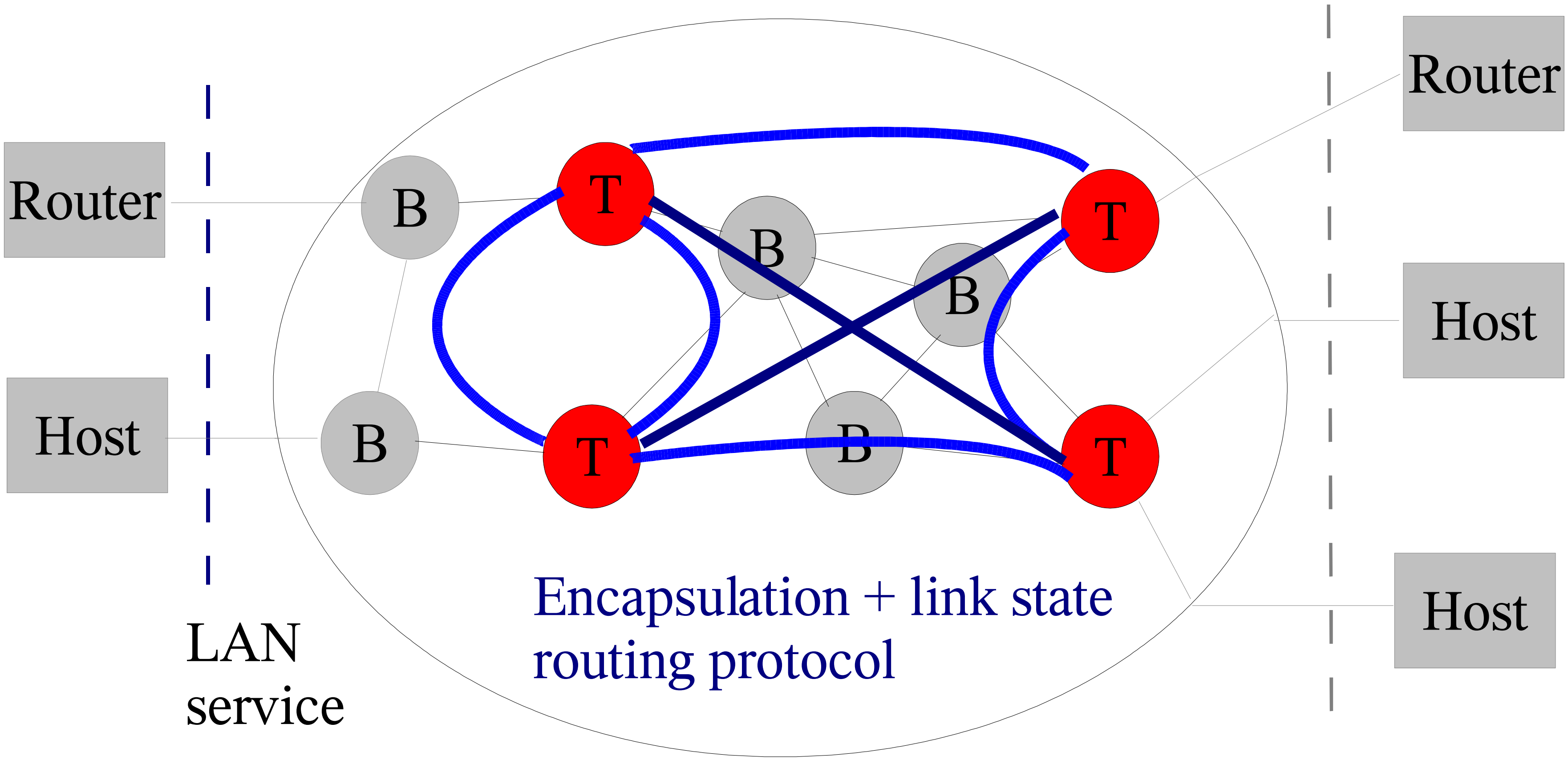
Forwaring based on a safe header

- TTL in the encapsulation header
- Encapsulation header with next hop rbridge address
  - in addition to egress rbridge
  - Prevents packet profileration during a temporary loop

Coexist with bridges

- Encapsulate across core; outer header is Ethernet
- Edge: elect designated rbridge to forward to/from link
- Edge: when an rbridge becomes a designated one, send "topology change" message to bridges on edge to flush their learning tables
  - Details to be worked out in interaction with 802.1D at the edge

TRILL overlay approach



Link-state routing protocol in core

- All rbridges know how to reach all over via shortest path
- Per VLAN spanning trees can computed without further protocol messages
- Per-ingress rbridge spanning trees can be computed for optimizing IP multicast distribution ("IP" because IGMP filtering info available) [no additional signaling needed to compute tree]
- Per ingress rbridge spanning tree used for unknown destinationsto prevent misordering which switching to shortest-path

Scalability elements

- Core forwarding table only with rbridges (i.e. not endnodes)
- VLAN endnode information only needs to be known to rbridges directly connected to links in that VLAN

Optimizations for IP

- [This might not be part of the architecture]
- Rbridges pass around l2/l3 pairs to enable proxy arp/nd
  - Perhaps later – not in current charter
- IGMP/MLD snooping
  - Just like a L2 switch

Optimizations for wireless

- [This might be a result of the architectur, but not part of the architecture proper?]
- By default endnode learning just like bridges
  - Look at source MAC address
- Can optimize when there are L2 “associations” as in 802.11 AP
  - Rbridges can then proactively inform everybody of the hosts new location (without the host having to send any packets)

Threat Analysis

- First, do no harm
  - Not any worse than in a bridged network today
- Explore ways we can do better
  - Need to look at both core – securing link state routing protocol – and at edge – learning host's location
  - Likely to require configuration of rbridges

Conclusion

- What's missing?
- Can we make this into a concise document?
  - Around 10 pages of content?
- Volunteers?