TRILL OAM
draft-ietf-trill-rbridge-oam-02.txt

IETF 82 – Paris
March, 2012

David Bond, Meenakshi Kaushik,
Thomas Narten, Santosh (Sunny) Rajagopalan

Moving Forward With OAM

- No defined core OAM for TRILL yet...
  - Implementors need guidance and a roadmap on OAM ASAP or risk of missing “window of opportunity” for a industry-wide approach
- WG MUST identify baseline OAM components to be finalized and completed ASAP
- We have multiple OAM related documents being proposed:
  - draft-ietf-trill-rbridge-oam-02.txt
  - draft-yizhou-trill-multi-destination-ping-01.txt
  - draft-tissa-trill-oam-03.txt
  - draft-rohit-trill-proactive-oam-00.txt
- Suggest taking a requirements-oriented approach
  - What functionality is required (i.e., MUST)?
  - What is nice-to-have (but can be optional and/or defined in a separate/later document?)
  - Once we agree on required functionality, what is best engineering solution for satisfying the requirement?
- Take the best parts of various drafts and combine into a complete OAM suite

Changes to ietf-trill-rbridge-oam

- Removed error codes that were not needed (e.g. Corrupt frame)
  - But some of these really ought to go into the MIB
- Added path sharing traceroute with 'real' data being sent.
- Added previous hop information TLV.
- Made most TLVs optional to allow hardware/fast path implementations where this information might not be available.
- Changed Next Hop Nickname TLV into Next Hop Information TLV since next hop might not always reserve a nickname. The new TLV includes the next hop system id.
- Moved the values table to the message format section and converted from table to list.
- Numerous minor typo corrections and wording clarifications.

Additions to MIB?

- Removed error codes from OAM document that did not seem appropriate
  - Does not seem useful to return error messages saying “corrupt frame” or “unknown egress RB”
  - But, those sorts of errors should presumably be counted/logged.
  - Isn't that what MIBs are tailor-made for?
- Believe we should add some counters to MIB
  - But sense some pushback on making further changes to the MIB because it's “almost done”

Need Multi-Destination Ping

- Seems defficient not to have multi-destination ping
- Working to merge in yizhou-trill-multi-destination-ping
  - Based on existing OAM framework using channel mechanism
  - One document seems better than two
  - Didn't make it into the revised document

OAM Requirements (assertion)

- Verifying path liveness (Ping)
- Path Discovery (Traceroute)
  - All possible paths
  - A path (but not all paths)
  - The path data looking like X would follow
  - Unpruned tree discovery
  - Pruned tree discovery
- Fault finding (Traceroute)
- Fault Isolation (Traceroute)
- Fault Notification (BFD)

Required Functionality?

- Known Unicast Ping
  - From RB1, ping RB2
  - Verifies path liveness between RB1 and RB2
- Known Unicast Traceroute (between RB1 and RB2)
  - Shows path between RB1 and RB2
  - Variations:
    - Simple Traceroute (show a path)
    - All Possible Routes Traceroute
    - Fate-sharing Traceroute (path followed by particular flow)
- Known multi-Destination Traceroute (Un-pruned)
- Known multi-Destination Traceroute (Pruned)

High-Level Decisions to Make

- RBridge Channel vs. ICMP?
- Useful but could be considered as an addition to a basic OAM suite.
  - Address-Binding Verification
  - MAC address discovery
  - End-Station Attachment Point Discovery
  - Traffic Triggered Monitoring

Trill-Specific OAM vs ICMP

- Should TRILL OAM use TRILL-specific OAM or attempt to leverage ICMP?
  - TRILL does not currently require an IP Stack – are we willing to mandate an IP stack?
  - Not all traffic is IP (e.g., FCoE)
  - IP/ICMP proposal assumes one can use L2 header fields without effecting ECMP
    - not necessarily a valid assumpion

Address-Binding Verification

- Allows an operator to query the IP Address to MAC Address or MAC Address to IP Address mappings of VMs
  - Not a TRILL-specific problem
  - Aren't there already existing tools for this, and if not, why should TRILL define them?

MAC Address Discovery

- Identify MAC – RBridge bindings
- Similar to functionality already provided by ESADI
  - Why does TRILL OAM need to do anything special here?
  - Why not snoop ESADI traffic and build such a table if necessary, then use SNMP to query the specific RB for additional associated info

End-Station Attachment Point Discovery

- Allows discovering, RBridge, interface information, VLAN, virtual Tags, etc, associated with a given IP Address.
- Potentially useful, but
  - Is this an essential part of core TRILL OAM?
  - Would it be reasonable to pursue this as an optional feature in a separate document?

Traffic Triggered Monitoring

- Monitor and analyze existing traffic crossing TRILL
- This would presumably involve monitoring traffic that traverses a TRILL campus
  - Monitoring is of an end-to-end nature, TRILL just happens to be a component.
  - What is it that makes this a TRILL problem?
  - Better solved outside of TRILL?
  - What does TRILL need to do to support such external functionality?

BACKUP

Required Unicast Functionality

- Which of the following are required functionality?
- Known Unicast Ping
  - From RB1, ping RB2
  - Verifies path liveness between RB1 and RB2
- Known Unicast Traceroute (between RB1 and RB2)
  - Shows path between RB1 and RB2
  - Simple Traceroute
  - All Possible Routes Trace
  - Fate-sharing Traceroute
- Multi-Destination Traceroute (Un-pruned)
- Multi-Destination Traceroute (Pruned)

Known Unicast Ping

- Simple ping using RBridge Channel to test connectivity
- Limitation:
  - Does not provide any way to mimic real data as this is only meant for a simple connectivity test and not for path discovery/fault isolation
  - But still useful for most cases
-

Known Unicast Traceroute

- Three Variants
  - Simple Traceroute
    - Uses RBridge Channel
    - Simple single path discovery between two RBridges
  - All Possible Paths Trace
    - Uses RBridge Channel but violates that drafts by setting the locally administered bit and incrementing destination mac over N traceroutes to test all ECMP paths
    - After N traceroutes there is a reasonable certainty that all routes have been covered
    - RBridge Channel Ethertype used to ensure frames to leave TRILL campus

Known Unicast Traceroute

- Fate Sharing Traceroute
  - Sends mimicked data (a real inner frame with hop count limited)
  - Data will not leak because frames will never be egressed since hop count errors are trapped to OAM module and the RBridge originating the traceroute will stop sending frames once the egress RBridge has been hit
  - Additionally zeroing the check sum provides another layer of protection if a frame ever did leak (e.g. if an RBridge did not implement the OAM standard)

Multi-Destination Traceroute (Un-pruned)

- Uses RBridge Channel with multi-destination bit set
- No leaking since DST MAC is ALL_EGRESS_RBRIDGES

Multi-Destination Traceroute (Pruned)

- Uses Tissa's approach but is not IP-centric. Uses any payload
  - Inner DA is multicast address
- ECMP is not an issue, because path is determined entirely by the choice of shared tree.
- To prevent frame leakage outside of TRILL:
  - Proposal: set the inner source MAC address to ALL_EGRESS_RBRIDGES and require egress RBridges MUST NOT egress such frames outside TRILL