Improving SRP with Techniques from RSVP

Rodney Cummings

National Instruments

Those who don't know history…

- Desired features for future PAR to improve SRP
  - Scalability: Need to refresh large number of streams
  - Reliability: Important for many time-sensitive applications
- IETF RSVP (RFC 2205) analogous to SRP
  - RSVP refresh of soft-state had two problems:
    - Scalability: Large number of flows each refresh period
    - Reliability: RSVP message loss causes delay to next refresh
- RFC 2961: RSVP Refresh Overhead Reduction Extensions
  - Extends refresh to fix problems

Overview of RSVP Overhead Reduction

- RSVP Path/Resv analogous to SRP Talker/Listener
  - Exchanged by neighbors (hop-by-hop)
- RSVP soft-state refresh analogous to SRP LeaveAll
  - Scalability problem suggests longer refresh period
  - Reliability problem suggests shorter refresh period
- Solutions in RFC 2961
  - Reliability: Add ability to detect when state is synced (steady)
  - Scalability: Once synced, brief summary each refresh
  - Optional features (backwards compatible)
  - Optimized for point-to-point links

Point-to-point Optimization

- Proposal in this presentation optimizes SRP on point-to-point links
  - Assumes MRP declare/register is 1-to-1
- Other point-to-point optimizations exist today in 802.1
  - MRP point-to-point subset (802.1Q, subclause 10.6)
  - ISIS-SPB (802.1aq, clause 27 introduction)
- Ideally, shared media (e.g. 802.11) will provide point-to-point abstraction to 802.1 higher layer protocols
  - Enables fully optimized protocols on shared media

MRP Versioning

Compatibility for MRP-based Protocols

- 802.1Q-2011 subclause 10.8.3.5, Required handling of protocol versions
  - Receive MRPDU version lower than implemented → interpret as lower version
  - Receive MRPDU version higher than implemented → discard unrecognized AttributeType & AttributeEvent
- New does not break old
- Benefits of new available to contiguous new devices
  - End-stations or bridges

SRP Overhead Reduction

- MMRP, MVRP, and MSRP currently version 0
- Proposal: Add “Overhead Reduction” as MRP feature
  - MRP state machines change for Overhead Reduction
  - MSRPv0 disallows Overhead Reduction
  - MSRPv1 requires Overhead Reduction
  - Additional AttributeType value in v1; ignored by v0
- Consider v1 for MMRP, MVRP, and MIRP as well

Reliability

RSVP Reliability (Ack)

1. Declare-side transmits message (Path or Resv)
   - Contains Message ID analogous to SRP Stream ID
   - Contains 'Epoch' to detect power cycles
2. When register-side receives message, transmits Ack
   - Ack contains Message ID and Epoch only
3. Declare-side waits for receipt of Ack
   - If wait times out, re-transmit message (back to step 1)
     - Next timeout will be 2X longer than current timeout
     - If reach limit on number of re-transmit, give up (Path/Resv failure)
   - If Ack received, state for that message is synced (done)

- Goal: Sync state quickly, but avoid too much traffic

802.1 Bridge: Reliable in Traffic Bursts

- Higher layers “attached as separate end-stations”
  - 802.1Q, 8.13.9, Figure 8-11

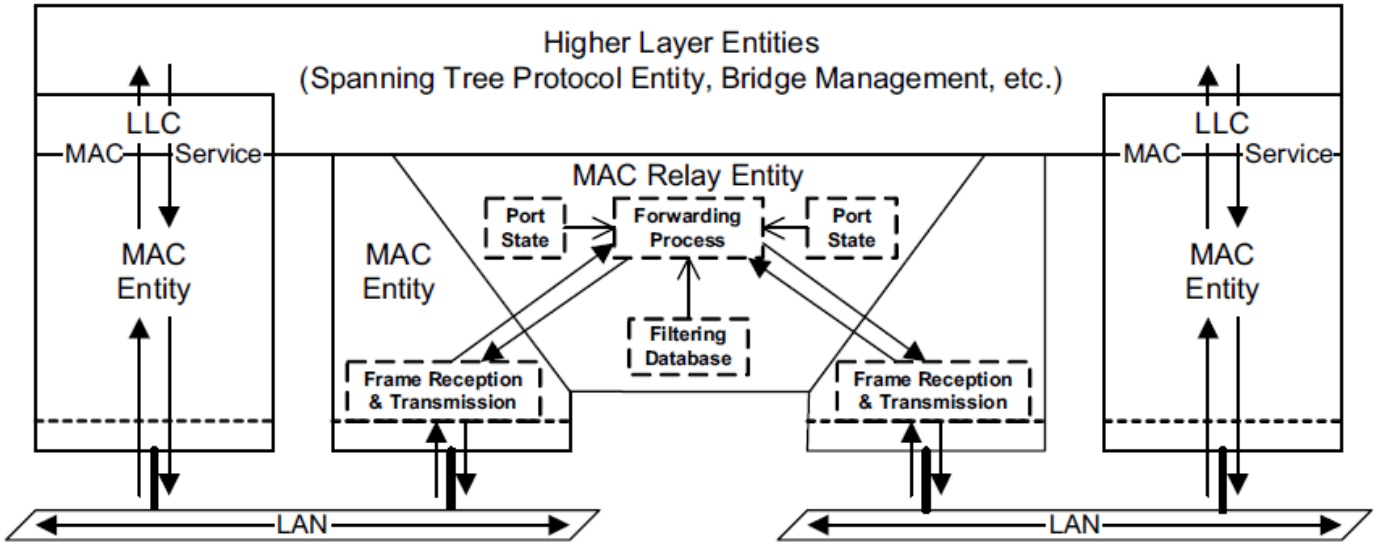


Figure 8-11—Logical points of attachment of the Higher Layer and Relay Entities

- Proposal: Clarify higher layer egress priority
  - E.g. lower than time-sensitive, higher than best-effort

MRP Reliability

- 802.1Q, 10.4 h)
  - “MRP is resilient in the face of single packet loss”
  - State machines transmit twice
- RSVP technique is resilient to multiple packet loss
  - May occur more often on shared media
  - Desired for Ethernet as well

Applying RSVP Reliability to SRP

1. Declare-side transmits MRPDU (Join, Leave, …)
   - ProtocolVersion v1, but otherwise same MRPDU as v0
2. When v1 register-side receives MRPDU, transmits Ack
   - Ack uses same AttrType, same value for lookup by declare
     - Value comparison exists in v0; value avoids new ‘Epoch’ concept
   - Ack specified as new bit
     - Ignored by v0 declare-side
     - Bit encoding to-be-determined (6 AttrEvent values already)
3. V1 declare-side waits for receipt of Ack
   - If wait times out, re-transmit MRPDU (back to step 1)
   - If Ack received, state for that MRPDU is synced (done)

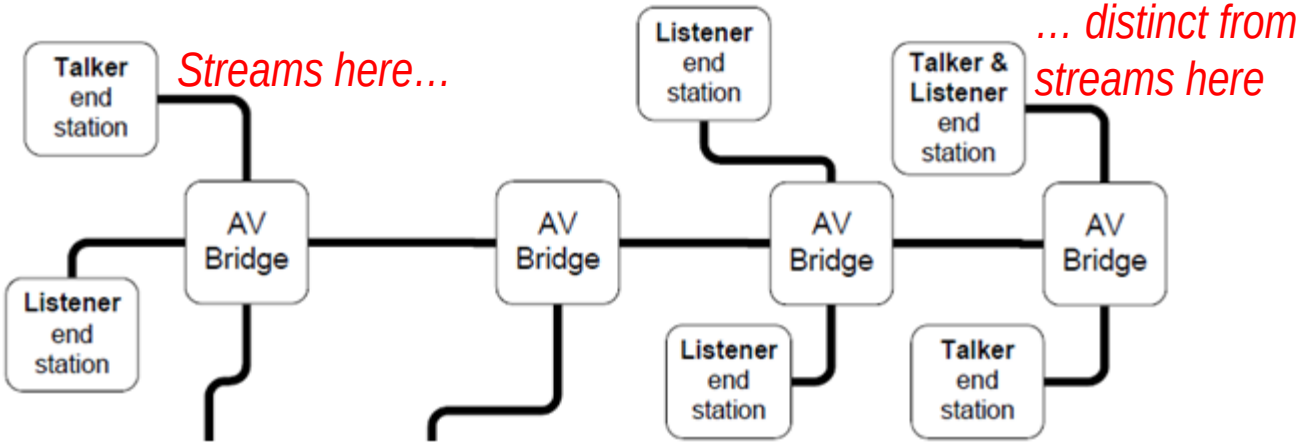
Scalability

RSVP Scalability (Summary Refresh)

- Declare or register side transmits Srefresh message
  - PDU contains ‘Epoch’ plus list of Message ID
  - Srefresh period is faster than normal refresh (Path/Resv)
  - If receive Srefresh with Message ID not synced, transmit ‘Nack’ for that Message ID
- Normal refresh (Path/Resv) continues for
  - Message IDs not in Srefresh
  - Message IDs that were Nack’d

Review of SRP's Normal Refresh

- 'All' attributes to refresh are distinct per hop



- On LeaveAll, all attributes for that hop are cleared
  - Must be re-declared before LeaveTimer expires
  - Limits scalability as shown in Dave Olsen's presentation
- Unlike RSVP, SRP normal refresh applies to all
  - Assumption: New SRPv1 summary refresh must apply to all

Applying RSVP Scalability to SRP

- Proposal: Hash solution from Dave's presentation, improved to use the Reliability proposal (Ack)
  - Perform when both neighbors are v1
- Restart LeaveAllTimer when receive MRPDU
  - Similar to v0's restart on !rLA; v1 restarts on !r<any>
  - Postpone refresh while end-stations make changes
  - Goal: Ensure neighbors are synced prior to refresh
- When LeaveAllTimer expires, transmit new Summary
  - Summary = steady-state hash of all attribute values
  - No need to transmit Stream IDs, because 'all' are synced

SRPv1 Summary Details

- Learn from ISIS-SPB Agreement Digest (802.1aq-2012, 28.4)
  - Optimizes refresh of soft-state for routing
    1. Run MD5 on parameters (value) of each edge
    2. Add all MD5 together to form digest value
    3. PDU contains number of edges, and digest value
  - Benefits
    - MD5 guards against value corruption (somewhat)
    - When add/subtract edge, simply add/subtract single MD5
- Additional AttributeType for Digest in MSRPv1 (e.g. 5)
  - Same calculation as ISIS-SPB, using attribute values
- When Digest received, if no match,
  revert to Leave All behavior (normal refresh)

Improve LeaveTime

- When we revert to normal refresh (LeaveAll), LeaveTime is not sufficient for large number of streams
- Proposal: Since we obtain number of streams from Digest, use that to calculate v1 LeaveTime
  - E.g. LeaveTime = 600ms * NumAttributes

Thank you