MPLS Traffic Engineering

RSVP Extensions

Lou Berger (lberger@fore.com)

December, 1998

Orlando IETF

MPLS-TE RSVP Extensions

- Extensions motivated by MPLS framework and traffic engineering requirements
  - Extensions documented in
    - draft-ietf-mpls-framework-02.txt
    - draft-ietf-mpls-rsvp-lsp-tunnel-00.txt
- Extensions presented at last IETF

– Tunnel identification	- Label Object
– Tunnel parameter negotiation	- Session-Attribute Obj.
– Routing policy distribution	- Explicit-Route Obj
– Routing debugging information	- Record-Route Obj.
– Initial scalability improvements	- Aggregate Message

- Not covered at last IETF

– Additional scalability improvements	- Refresh extensions

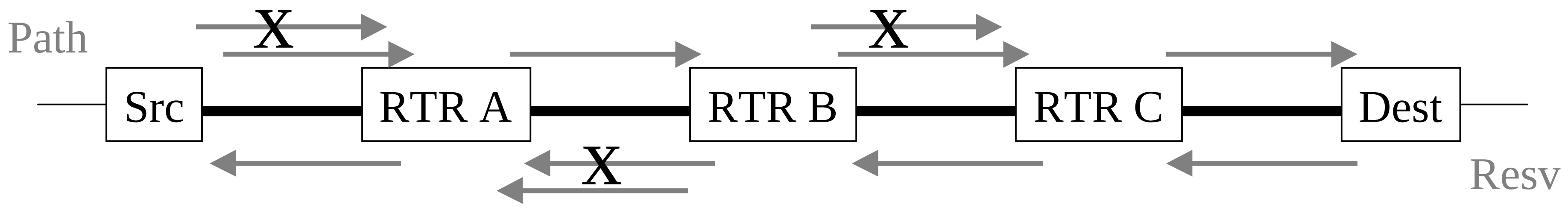
Relevant MPLS Requirements

- Scalability
  - Must be able to support O(n^2) Label Switched Paths
    - For non-merging solutions
    - N is number of edge routers
      - 100 edge routers ==> O(10,000) reservations
      - 300 edge router s ==> O(100,000) reservations
    - All sessions will be unicast
    - Multicast is for further study
  - Key RSVP implications
    - Refresh message rate
    - Processing overhead per refresh message
- Network resiliency
  - Rapid failure detection
  - Bounded setup/teardown time
- Key RSVP implication
  - Latency of end-to-end state synchronization
  - Reliability of messages

Issue: Refresh Message Processing

- Senders must regenerate messages for each installed state
- Receivers must parse whole message
  - To determine if new message or refresh
- Limits scaling to large number of sessions
  - Parsing requirements and message rates are issues
  - Each Path and Resv must be independently refreshed
  - Example:
    With 100,000 sessions and a 30 second refresh interval
    3,333 messages per second must be generated and
    3,333 messages per second must be parsed
- Longer refresh intervals are not a cure!
  - Increasing refresh interval hurts failure detection and recovery

Issue: Latency and Reliability



- Setup issue
  - Worst case setup time is tied to
    Number of hops, Refresh Interval, Loss rate
  - Recovering from lost setup message tied to refresh rate
    - Loss may occur at every hop -- In both directions
    - Multiple losses are possible
- Teardown issue
  - Recovering from lost Tear messages is tied to refresh rate
    - State must be timed-out
  - Resources remain unavailable to other users
- Shorter refresh intervals are not a cure!
  - Decreasing refresh interval increases refresh processing overhead

Proposed Solution: Message_ID Extension

- Composed of two new objects
  - MESSAGE_ID and MESSAGE_ID ACK
  - Both may be carried in any type of RSVP message
  - MESSAGE_ID ACK Object may also be carried in ACK message
- Objects used to support:
  - Reduced processing for refresh messages
    via a 24-bit identification of represented state
  - Reliable message delivery
    via requested acknowledgements
  - Refresh elimination
    for desired messages
    - Requires notification from routing on route change
    - Requires other neighbor failure detection mechanism
      such as information from routing or HELLO Extension

MESSAGE_ID (continued)

- Object Format:

```
  0                   1                   2                   3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |     Flags     |                  Message ID                   |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- Message_ID field
  – A sender generated value that uniquely identifies message
- Flags
  – ACK_Desired - indicates sender willing to accept an ACK
  – Last_Refresh - indicates that message will not be refreshed
    - Once acknowledged
- ACK Flags (for MESSAGE_ID ACK Objects)
  – No_Refresh - indicates refreshes are not needed for message
- Extra refresh required to maintain state in face of message loss
  – Tear must be reliably sent if receiver not expecting refreshes

MESSAGE_ID Multicast Restrictions

- Avoiding ACK implosion
  - Responders wait a random interval prior to acknowledging
- When number of next-hops not known
  - Should only expect a single Ack
    - Means “fast retransmit” until 1st Ack received
  - MUST ignore No_Refresh flag
    - Means using standard RSVP refresh processing
- When new receivers cannot be identified
  - Should only expect a single Ack
  - MUST ignore No_Refresh flag
- When all receivers do not request No_Refresh
  - MUST ignore No_Refresh flag

Proposed Solution: Hello Extension

- Used to detect failures in neighboring RSVP nodes
  - Required when not using RSVP’s refresh processing
  - When no other mechanism available
- Composed of:
  - STATE_SET Object
  - Hello message
  - Hello Ack message
- Hello and Hello Ack message each allow message receiver to detect reset/failure of sender

Hello Extension (continued)

Supports:

- Failure detection
  - Via no response and reset of "instance" value
- Use of Hello failure detection by one side or both
  - All implementations supporting MESSAGE_ID MUST be able to answer Hellos but are not required to generate them
- Independent failure detection rates
  - Messages will end up being generated by sender with lower rate
- Explicit support for multiple interfaces using same IP address
  - "Instance" values passed on a per LIH basis
  - Aimed at explicit support for unnumbered links and RSVP tunnels
    - Unnumbered links could be supported by single "instance" value coupled with physical link information
  - Open issue: should explicit LIH support be removed?

Compatibility

Both extensions are fully backward compatible:

- MESSAGE_ID Class uses value of form 10bbbbbb
  - Per RFC 2205 classes with values of this form must be ignored and not forwarded by nodes not supporting the class.
  - Non-supporting receivers will silently ignore object
  - Senders will see no ACK and therefore continue with standard RSVP refresh processing
- Hello related Class uses values of form 0bbbbbbb
  - Per RFC 2205, this is an “Unknown Object Class”
  - Non-supporting receivers will ignore message or respond with error
  - Senders will see no Hello ACK, and therefor are prohibited from setting No_Refresh flag.