Staged Refresh Timers for RSVP

Ping Pan
Henning Schulzrinne
Roch Guerin
Background

• RSVP uses *soft state*:
  – reservations will disappear by themselves if not being refreshed;
  – advantage 1: avoid orphan reservations
  – advantage 2: quick adaptation to route changes
  – explicit tear-down messages to speed up the removal of reservations
Background

- Unreliable RSVP control message delivery:
  - periodic refresh between hops;
  - cleanup timer: a state is deleted if no refresh messages arrive before the expiration of a cleanup timer interval.
Motivation

• Packet loss problem in the Mbone:
  – 1-2% on average;
  – 20% or more occasionally.

• If the first RSVP message is lost due to congestion:
  – no PATH or RESV re-transmitting until the next refresh cycle (30 seconds by default).
  – no retransmission for tear-down messages; the default timeout is 90 seconds.
... Motivation

- Why not increase the refresh rate?
- A problem with hop-by-hop refresh:
  - do not propagate unchanged refresh messages.
  - for example ...
(until B’s refresh cycle)
... Motivation

• Why do we need reliable and fast RSVP message delivery?
  – End system multimedia application requirement: the first few seconds may be critical.
  – Service policy requirement: The delay of RSVP delivery may cause billing and accounting problems.
Terminology

• Sending and Receiving nodes
• Trigger and Refresh Messages:
  – trigger messages: generated due to state changes. Need to be delivered immediately after state changes are detected.
  – refresh messages: replicated messages to maintain states. Could be sent very infrequently.
Operation Overview

- Send trigger messages with echo-request.
- Retransmit the message until the echo-reply is received.
- The retransmission interval is governed by a staged refresh timer.
- Scale back the refresh rate if the echo-reply is received.
Staged Refresh Timer

• Each sending node has the following tunable parameters:
  – $R_f$: the initial fast refresh interval. Default value is 3 seconds.
  – $R_s$: the slow refresh interval (after echo-reply). Default is 15 minutes.
  – $R$: fixed refresh interval. 30 sec by default.
  – $\Delta$: an incremental value. 0.3 by default.
Staged Refresh Timer (2)

- After sending a trigger message:
  - unless the echo-reply is received, schedule retransmission after $R_f \times (1+\Delta) \times R_f \times (1+\Delta)^2 \times R_f$, ...
  - if the echo-reply is received, switch the refresh rate to $R_s$.
  - When $(1+\Delta)^l R_f$ reaches to $R$, refresh PATH/RESV with $R$, and stop sending tear-down messages.
Staged Refresh Timer (3)

A new RSVP timer algorithm:

```
If (R_k < R)
    R_k -> R_k (1+\Delta)
    send out a refresh message
    wake up in state k after R_k seconds;
    exit
else
    R_k -> R
    if (the state k is a tear-down message)
        clean up state k;
        exit;
    else
        send out state k after R_k seconds;
        exit
```
Basic Properties

• hop-by-hop;
• minor addition to the RSVP protocol;
• backward compatible;
  – does not require the proposed scheme to be implemented on the receiving nodes.
• small operating overhead.
Special Considerations (1): tear-down messages

- Release the resource, and mark the state as *closing*.
- Use the state info for retransmission;
- Remove the state *only* after
  - the echo-reply is received,
  - or the refresh interval has changed to the fixed interval $R$. 
Special Considerations (2): operation in NBMA

- Problem: for a multicast session, a sending node *does not* know the total number of receiving nodes for PATH or PATHTEAR at an egress interface.
- Therefore, cannot switch to a longer refresh timer Rs based on having received echo-replies.
Operation in NBMA:
PATH message
Operation in NBMA: PATH

- Solution 1: Query ARP or MARS server to find out the exact number of receiving nodes. Switch to Rs after receiving replies from all receiving nodes.
- Solution 2: PATH is used for traffic advertisement. So don’t apply staged refresh timer for PATH messages.
Operation in NBMA: PATHTEAR
Operation in NBMA: PATHTEAR

- A sending node knows all the receiving nodes that have made reservations.
- Generate PATHTEAR with staged refresh timer until replies are received from all known nhop nodes.
Evaluation
Reduced Message Loss Probability

– Assume the message loss probability for a single message is 20%. The *accumulative* probability that no reservation is established after half minute is reduced to $3 \times 10^{-4}$ compared with $4 \times 10^{-2}$ with the current fixed timer.

– For loss rate of 2%, the failure probabilities become $3 \times 10^{-9}$ and $4 \times 10^{-4}$, respectively.
Evaluation
Reduced Protocol Overhead
(150 bytes per message)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>60 min</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixed Refresh</td>
<td>300</td>
<td>18,000</td>
</tr>
<tr>
<td>Slewed Refresh</td>
<td>300</td>
<td>1,950</td>
</tr>
<tr>
<td></td>
<td>(slew.rate = 0.3)</td>
<td></td>
</tr>
<tr>
<td>Staged Refresh (no reply)</td>
<td>900</td>
<td>18,600</td>
</tr>
<tr>
<td>Staged Refresh (with reply)</td>
<td>300</td>
<td>900</td>
</tr>
</tbody>
</table>
Conclusion

- Simple
- Backward Compatible