Resilient State Management in Large Scale Networks

Yangcheng Huang and Saleem N. Bhatti

Department of Computer Science, University College London,
London WC1E 6BT UK
{y.huang, s.bhatti}@cs.ucl.ac.uk

Abstract. This paper describes, briefly, ongoing research on resource reservation state management, including research motivations and initial design.

1 Introduction

A fundamental challenge in designing protocols in large scale networks is how to manage a large amount of state information to deal with failures but with acceptable cost in protocol overhead. In the Resource reSerVation set-up Protocol (RSVP) [1], a soft state mechanism has traditionally been used to achieve state consistency, but its high overhead makes it infeasible for large-scale deployment [8]. This research is to propose a framework for reservation state management to maximize RSVP's performance, including reliability in delivering control messages and resilience in restoring state from inconsistency while minimizing protocol overhead and complexity.

2 Motivations

The standard RSVP [1] takes a soft-state approach in its design. In order to maintain its state information, RSVP nodes send periodic RSVP refresh messages for each existing RSVP session. In the absence of refresh messages, the RSVP state information would automatically time out and be deleted. The original RSVP relies heavily on a soft-state approach in state maintenance, including: 1) detecting state inconsistency and recovering from internal state corruption and failure; periodic PATH and RESV refresh messages contain state information for each existing RSVP session; a RSVP node verifies its RSVP state with refresh messages to detect/recover state errors; 2) achieving reliability in control message delivery; PATH and RESV state installation messages are transmitted as best-effort traffic under the assumption that any loss of control messages would be recovered from periodic refresh messages.
However, soft state mechanisms may not be the best choice because:

- The communication overhead due to such periodic refreshes increases linearly with the number of active RSVP sessions [6];
- In some RSVP extensions [2] [6], a timer mechanism is widely deployed to achieve robustness and resilience. However, different extensions may have correlations and conflicts in timer configuration. For example, the timer interval of a Srefresh (summary refresh) message should be longer than that of standard refresh

H. de Meer and N. Bhatti (Eds.): IWQoS 2005, LNCS 3552, pp. 350 – 352, 2005.

message; when a standard refresh message is sent, a corresponding summary refresh should not be sent during the same refresh period [2];
- Recent efforts [3] in state mechanisms show that a simple soft state approach does not compete with a mixed hard/soft state approach in performance: "a soft-state approach coupled with explicit removal substantially improves the degree of state consistency while introducing little additional signalling message overhead" [3]. Further efforts in analyzing and improving its performance are desirable.

3 Towards a Resilient State Management Framework

3.1 Internal Failure Detection Based on State Consistency Arbiter

Existing methods for internal state corruption detection are based on soft-state synchronization, such as refresh messages in standard RSVP [1] and neighbouring state verification mechanisms in RSVP extensions [6] [7]. To reduce dependence on soft state mechanism, we propose an asynchronous state consistency verification mechanism in which no periodic state updating is required; state consistency verification only happens when there are state changes; once state inconsistency is detected, state recovery process will be initiated. The mechanism is described as follows.

State consistency arbiter could be any node in or outside the RSVP session. The role of an arbiter is to listen to state updates from every RSVP node, simulate the global state of the RSVP session and detect inconsistency. In the absence of inconsistencies, it works in silent mode: only collecting state digests from RSVP nodes. Once inconsistency is detected, the arbiter either notifies the RSVP node to initiate state recovery/synchronization, or create a corrected state digest and transmit it to all nodes.

Whenever detecting any state change, every RSVP node computes a digest for all active RSVP sessions and sends the digest to the arbiter; when receiving digest information from a RSVP node, the arbiter compares the digest information with other digest information it already holds to judge the consistency.

The state digest mechanism in this approach is similar with that in state compression [6]. The key difference between the state compression and arbiter architecture is that the former adopts a soft state mechanism and sends periodic state digests to neighbouring RSVP nodes, while the latter adopts asynchronous state verifying methods and sends state digests to a centralized arbiter only when state is changed. Compared with state verification through exchanging neighbouring nodes' state information, this approach has a global view of RSVP sessions and could solve the problem of simultaneous failure among neighbouring nodes.

3.2 Dynamic Refresh Timer

In existing soft state protocols, the values of the timer intervals are chosen by "matching empirical observations with desired recovery and response times" [4]. The fixed-intervals mechanism has no consideration of network status in terms of failure rate; it adapts neither to the wide range of link qualities that exist in large scale networks, nor to fluctuations in rate of failure occurrence over time.

We propose an adaptive approach in which values of timer intervals adapt dynamically to real-time link status based on failure feedback. The essential mechanisms required to realize this dynamic timer approach are: 1) fast failure detection and reporting mechanisms to signal end hosts to adjust timer intervals; 2) dynamic adjustment of a sender's refresh rate so that state failure can be recovered very quickly after it occurs, whilst overhead is kept low when there is no error.

3.3 Reliable Control Message Delivery

Loss of control messages may cause delay in RSVP setup or state inconsistency [9]. So, it is desirable that RSVP control messages are reliably delivered. However, we argue that per-session ACK-based mechanism [2] [5] are costly and inefficient to handle, causing bursts of RSVP requests, especially for short RSVP sessions.

We propose a summary acknowledgement scheme (S_ACK), which guarantees control message delivery for a list of RSVP sessions. In our scheme, we verify the delivery of trigger messages after multiple messages have been transmitted.

4 Conclusions and Current Status

This paper describes briefly ongoing PhD research on state management; simulations and experiments are being carried out; therefore more results will be available soon.

References

1. L. Zhang, S. Deering, D. Estrin, S. Shenker, D. Zappala: RSVP: A New Resource Reservation Protocol. IEEE Network, Vol. 7, Sept. 1993
2. L. Berger, et al: RSVP Refresh Overhead Reduction Extensions. RFC 2961, April 2001
3. Ping Ji, Zihui Ge, Jim Kurose, Don Towsley: A Comparison of Hard-state and Soft-state Signaling Protocols. Proc. ACM SIGCOMM, 2003
4. P. Sharma, et al: Scalable timers for soft state protocols. Proc. IEEE Infocom, Japan, 1997
5. P. Pan, H. Schulzrinne: Staged Refresh Timers for RSVP. Proc. IEEE Globecom, 1997
6. L. Wang, et al: A New Proposal for RSVP Refreshes. Proc. IEEE ICNP, 1999
7. D. Awduche, et al: RSVP-TE: Extensions to RSVP for LSP Tunnels. RFC 3209, Dec 2001
8. A. Mankin, et al: Resource ReSerVation Protocol (RSVP) Version 1 Applicability Statement: Some Guidelines on Deployment. RFC2208, Sept 1997
9. O. Komolafe, J. Sventek: An Evaluation of RSVP Control Message Delivery Mechanisms. Proc. IEEE HPSR, Phoenix, Arizona, USA, April 2004