One-Way-Trip Time Measurement and Retransmission Policy for Congestion Control in Multipath Transmission Control Protocol (M/TCP)

6L－02

Kultida Rojviboonchai Naoki Watanabe Hitoshi Aida
Department of Frontier Informatics, The University of Tokyo

1. Introduction

The Internet seamlessly provides information and communication services. The most popular Internet applications such as WWW, electronic mail and file transfer use the reliable service provided by TCP. Studying congestion control mechanisms in TCP, therefore, becomes an essential part to build high performance and high reliability network.

Hayashi et al. suggest TCP communication using multiple routes to improve network reliability. Simulation results [1] have been reported that network throughput can be improved with the use of distribution mode and less packet loss possibility can be achieved by using duplication mode. In owing to carry out the communication, Multipath Transmission Control Protocol (M/TCP) [2] has recently been introduced as an alternative TCP option. Although there have been several general studies of TCP communication using multiple routes, no congestion control mechanisms has yet been properly designed. In this paper, congestion control mechanisms in M/TCP are introduced. We consider how to estimate Round-Trip Time (RTT), which is a significant part to calculate Retransmission Timeout (RTO). We also describe retransmission policy suitable for M/TCP communication.

This paper is organized as follows. First, the proposed congestion control mechanisms including One-Way-Trip Time (OWTT) measurement and retransmission policy are presented in section 2. Then, section 3 describes implementation issues. Finally, we present conclusion and future work in section 4.

2. Proposed Congestion Control Mechanisms in M/TCP

M/TCP is multipath connection (Figure 1). A data segment and the corresponding ACK are probably transferred via different paths. Therefore, the conventional RTT measurement and retransmission policy [3] are not sufficient and need to be considered in detail.

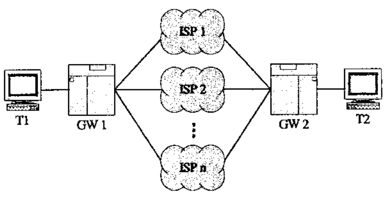

Figure 1: Multipath Data Transmission

The new OWTT measurement and retransmission policy can be handled by using multi-route option (Figure 2) proposed in [2].

Option Kind	Option Length	Route ID	Route ID Reply
Sequence Reply			
8 bits	8 bits	8 bits	8 bits

Figure 2: Multi-Route Option

2.1 One-Way-Trip Time (OWTT) Measurement

To assign RTO for each transmitting segment more precisely, OWTT measurement allows sender to estimate time intervals of forward path and reverse path separately. When the sender receives an ACK containing *Route ID Reply* = x and *Route ID* = y, the time interval of forward path via route x, $OWTT_{xf}$, and the time interval of reverse path via route y, $OWTT_{yr}$, are calculated as

$$Err = M_{xy} - (OWTT_{xf} + OWTT_{yr}) \quad (1)$$
$$M_{xf} \leftarrow OWTT_{xf} + Err/2 \quad (2)$$
$$M_{yr} \leftarrow OWTT_{yr} + Err/2 \quad (3)$$

where M_{xy} is the measured time interval between sending a particular sequence number via route x and receiving the corresponding ACK via route y, M_{xf} is the measured time interval of forward path via route x and M_{yr} is the measured time interval of reverse path via route y.

Based on the original TCP specification, $OWTT_{xf}$ and $OWTT_{yr}$ is updated and RTO is determined as follows.

$$OWTT_{xf} \leftarrow \alpha OWTT_{xf} + (1-\alpha)M_{xf} \quad (4)$$
$$OWTT_{yr} \leftarrow \alpha OWTT_{yr} + (1-\alpha)M_{yr} \quad (5)$$

For Distribution Mode:

$$RTO_x = (OWTT_{xf}+\max(OWTT_r))\beta \quad (6)$$

For Duplication Mode:

$$RTO_x = (\max(OWTT_f)+\max(OWTT_r))\beta \quad (7)$$

where α is a smoothing factor, β is a delay variance factor, RTO_x is retransmission timeout value of the next segment transmitted via route x, $\max(OWTT_f)$ is the maximum value among the time intervals of forward path and $\max(OWTT_r)$ is the maximum value among the time intervals of reverse path.

For distribution mode, $\max(OWTT_r)$ is used since the sender does not recognize via which route the corresponding ACK will be sent back. Similarly, $\max(OWTT_f)$ is used for duplication mode since the sender does not recognize via which route the data will arrive at the receiver. Moreover, it would be better to overestimate than underestimate the RTT, which may lead to unnecessary retransmissions.

2.2 Retransmission Policy

There are two ways for a sender to assume a packet to be lost and subsequently perform a retransmission.

2.2.1 Retransmission Timeout

The algorithm for managing a retransmission timer for M/TCP communication is introduced as follows:

1. Retransmission timer and congestion window *(cwnd)* should be managed independently for each route.
2. Every time a packet containing data is sent to a route (including a retransmission), if the timer of the route is not running, start it running so that it will expire after RTO seconds (for the current RTO value of the route).
3. When all outstanding data of a route has been acknowledged, turn off the retransmission timer of the route.
4. When an ACK is received that acknowledges the new data, restart the retransmission timers of all routes indicated by *Route ID Reply* field so that they will expire after their own RTO seconds.

When the retransmission timer of a route expires, the sender may do as follows.

5. Retransmit the earliest segment sent to the route that has not been acknowledged by the M/TCP receiver and set *cwnd* of the route to one segment. The M/TCP sender may retransmit the missing segment to a route, which satisfies equation (8) and has the minimum value of $OWTT_f$.

$$\min(cwnd, rwnd) - unack \geq \text{MSS} \qquad (8)$$

where *rwnd* is the current receiver's advertised window, *unack* is the number of data bytes sent but has not been acknowledged yet and MSS is Maximum Segment Size.

6. If the M/TCP sender performs a retransmission to the route that the retransmission timeout expires, set RTO of the route to 2*RTO (Exponential backoff). For other routes, the missing segment will be sent normally.
7. Start the retransmission timer, such that it will expire after RTO seconds.

2.2.2 Fast Retransmit Algorithm

When M/TCP receiver transmits an ACK for every other data segments, an ACK will contain many pairs of *Route ID Reply* and *Sequence Reply* corresponding to all segments arriving at the receiver. Fast retransmit and fast recovery algorithms for M/TCP communication can be implemented together as follows:

1. When the sender receives three pairs whose *Route ID Reply* values are the same and *Sequence Reply* values are more than *sequence number* of the earliest unacknowledged segment sent to the route, set slow start threshold *(ssthresh)* of the route to one-half of the minimum of *cwnd* and *rwnd.*
2. Retransmit the earliest unacknowledged segment to all routes according to equation (8) and set *cwnd* of the route to *ssthresh* plus 3 times the segment size.
3. Each time another ACK containing the same *Route ID Reply* and *Sequence Reply* that value is more than the earliest unacknowledged segment arrives, increment *cwnd* of the route by the segment size and transmit a packet if allowed by the new value of *cwnd.*
4. When the next ACK arrives that acknowledges new data including the earliest unacknowledged segment, set *cwnd* to *ssthresh* (the value in step 1).

3. Implementation Issues

A basic configuration with a single source and two transmission paths are considered. Performance studies of proposed congestion control mechanisms in M/TCP are conducted using the network simulator. The simulator contains implementations of Node and TCP Reno. First, the exclusive gateway is implemented by extending the existing Node source code. M/TCP is implemented by modifying the existing TCP-Reno source code to include the new OWTT measurement and retransmission policy schemes.

4. Conclusion and Future Work

In order to carry out M/TCP communication, congestion control mechanisms need to be designed properly. In this paper, we propose the new OWTT measurement and retransmission policy suitable for M/TCP communication. The algorithms and implementation issues are described. As future work, we plan to implement proposed congestion control mechanisms and evaluate its performances by comparing to TCP Reno.

References

[1] H. Hayashi, S. Yamasaki, N. Morita, H. Aida, M. Takeichi and N. Doi, “Performance Evaluation of Data Transfer on Multiple Routes through Internet”, Trans. IEICE B, Vol. J84-B, No.3, pp. 514-522, 2001. (Japanese)

[2] 中村成貴、“複数経路コネクション指向プロトコル M/TCP の提案”、東京大学大学院工学系研究科情報工学専攻修士論文、2001.

[3] V. Paxson, M. Allman, “Computing TCP's Retransmission Timer”, RFC 2988, Nov. 2000.