"Latency-based approaches for TCP/IP networking"
Two ad-hoc techniques for short TCP transfers

Giulia Rizzoli[†]

***Abstract*—Over the years several techniques were investigated to reduce latency, that can be implemented to a wide range of applications through the Internet [1]: among them short TCP transfers. Particularly relevant is the development of strategies that can make the TCP/IP networking more latency-sensitive while keeping the benefits of its protocol such as reliability.**

***Index Terms*—Latency, TCP/IP, TCP Fast Open, Early Retransmit**

I. Introduction

Latency is a measure of the responsiveness of an application [1]. It tends to be cumulative over the communication sessions and grows rapidly until disaster scenarios as network failure occurs. Formerly TCP/IP protocol does not provide a delay guarantee, especially relevant for real-time traffic flows with high rates. Besides that, even in applications with a much lower data rate such as web transfer, the user experience might be intermittent and unnatural due to undesired delays, therefore further mechanism are needed. In general short TCP transfers involve frequent communication setup for few data exchanges. Delays can be introduced at various stages of communication. In TCP end-point interaction two aspects can be taken into account to accomplish latency reduction: improving the 3-way handshake when resuming a session and an early detection in packet loss recovery.

II. Methods

Initialization of a session in TCP consists of a 3-way handshake (3WHS) between parties, usually client to server. The client sends first an opening packet (SYN), then waits for the server acknowledge (SYN-ACK) and in turn, it sends an acknowledge back. This costs each time a Round Trip Time (RTT) before actual data can be sent. Server to client model is analogous, with half RTT difference, therefore only the former is considered.

TCP Fast Open (TFO) [2]: The main idea of this technique is to bypass the 3WHS mechanism when the client wants to resume a previously initialized session. This can be implemented by sending to the client a TFO cookie that can be used to verify that an early handshake occurred. The cookie can be passed together with the server's SYN-ACK, and it consists of a MAC tag generated by the server with a variable length of 6 to 18 bytes, including 2 fixed bytes that specify kind and length [2]. Its lifetime is managed by the server and it can expire at any time to provide more security. The client caches the cookie and sends it along with the SYN packet when a new connection needs to be opened. This allows the client to send data before the 3WHS, in other words without waiting for the server response, resulting in one RTT saves per resumed connection. If the cookie is invalid and/or expired then the server acknowledges only for the SYN, starting a new connection as in normal TCP. The TFO protocol does not revise the maximum amount of data exchanged, governed by TCP's congestion control in RFC5681 [3], but it provides saves in latency for resumed sessions, therefore it is particularly suitable for short transactional connections as web transfers, where idle periods are frequently followed by a new TCP connection setup. Further improvement can be reached while combining TFO with Transport Layer Security (TLS) False Start [4], which saves an additional RTT while exchanging encryption keys during the initial handshake.

Packet loss is associated with error-prone and congested networks: recovery contributes to make the protocol reliable, however congestion control might be an additional source of delay, among that the time needed to detect loss. In TCP Fast Retransmit, a packet is considered lost after three duplicate ACKs, which is usually less than the Retransmission Timeout (RTO, often $\sim 500ms$), estimated as the average measured RTT plus a soft margin to account variability. Duplicate ACKs are sent when an out-of-order segment is received.

Early Retransmit (ER) [5]: This approach attempts to reduce the number of the required duplicate ACKs to trigger fast retransmission. It is particularly significant when the congestion window is small, thus indicating more severe congestion. Then, in case of loss, out-of-order segments are also slowed down, followed by delayed ACKs that may affect the responsiveness of Fast Retransmit. This might result in the RTO timer to expire first. In order to prevent the described situation, the Segment-based Early Retransmit algorithm establishes that when the receive window does not permit new segments to be transmitted or there are no unsent data, then the duplicate ACKs threshold is:

$$ER\_thresh = \#\, outstanding\, segments\ - 1$$

which should be always lower than three, meaning that less than four outstanding segments are allowed. If the latter

[†]Department of Information Engineering, University of Padova, email: giulia.rizzoli.1@dei.unipd.it
Special thanks to RWTH Aachen University.

or the previous conditions are not satisfied, the standard supported retransmission criteria should be used. Furthermore the segment-based approach requires to track the boundaries of the segments (not automatically done by TCP), and it can be implemented by keeping a circular list. The combination of ER with Selective Acknowledgment (SACK) [6] variant of TCP adds more robustness, especially in the case in which only a few segments need to be transmitted. In SACK an ACK timer is also kept, producing a sufficient number of duplicate ACKs to trigger Early Retransmit. ER is particularly suitable when the amount of data to send is not sufficient to fill the congestion window, thus it might be a good solution for Web servers in which the number of exchanged segments per connection is low.

III. Concluding Remarks

This study aimed to exploit latency reduction techniques suitable for TCP/IP networking that might be addressed for short TCP transfers. TFO saves 1 RTT for each resumed session at the cost of up to 18 bytes overloading. It might not be suitable for all the applications because it requires cookies-enabling. Moreover, TFO gives no advantages when the connections' frequency between two parties is not very high. The combination with TLS False Start might be difficult for some implementations (e.g peer-to-peer) [4]. ER instead tries to fasten loss detection, in particular when fast retransmission fails. It comes with the additional cost of keeping state of the outstanding segments on the sender side. Its performance decrease when persistent reordering of segments is combined with applications that do not send data constantly because it results in needless retransmission. Other techniques such as Non-Congestion Robustness [7] might deal better with segments reordering. In conclusion, the use of TFO and ER might be an adequate solution to reduce latency in TCP/IP networking, but there is still room for improvement.

References

[1] B. Briscoe, A. Brunstrom, A. Petlund, D. Hayes, D. Ros, J. Tsang, S. Gjessing, G. Fairhurst, C. Griwodz, and M. Welzl, "Reducing internet latency: A survey of techniques and their merits," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 3, pp. 2149–2196, 2014.

[2] Y. Cheng, J. Chu, S. Radhakrishnan, and A. Jain, "Tcp fast open," *RFC 7413 (Experimental), Internet Engineering Task Force*, 2014.

[3] T. Henderson, S. Floyd, A. Gurtov, and Y. Nishida, "The newreno modification to tcpâ€™s fast recovery algorithm," *RFC6582*, 2012.

[4] A. Langley, N. Modadugu, and B. Moeller, "Transport layer security (tls) false start," *Internet Draft*, 2010.

[5] M. Allman, K. Avrachenkov, U. Ayesta, J. Blanton, and P. Hurtig, "Early retransmit for tcp and stream control transmission protocol (sctp)," *Internet RFCs, ISSN 2070-1721, RFC*, vol. 5827, 2010.

[6] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow, "Tcp selective acknowledgment options," tech. rep., RFC 2018, 1996.

[7] S. Floyd, S. Bhandarkar, and A. N. Reddy, "Improving the robustness of tcp to non-congestion events,"