Bufferbloat Problem

CSC458 Computer Networks Programming Assignment 2

Author: Tingfeng Xia, University of Toronto, mailto:tingfeng.xia@mail.utoronto.ca
Date: Wednesday 25^{th} November, 2020

Abstract Bufferbloat refers to the existence of excessively large and frequently full buffers inside a network. [Sivov and Sokolov, 2012]

Most TCP congestion control algorithms rely on packet drops to determine the available bandwidth between two ends of a connection.[Allman and Paxson, 2009, El-Sayed et al., 2011] In general, TCP congestion control algorithms speed up the data transfer (via increasing congestion window size) until packets start to drop, then slow down the transmission rate. Under ideal conditions, we expect an equilibrium speed to be reached after a period of time of adjustments.

In a fast to slow transition hop, bufferbloat can easily occur. Let's consider the internet topology illustrated in the assignment handout (Assignment Topology). TCP will continue to increase the cwnd size since packets sent out are being buffered inside the intermediate router s_0 and with no packet being dropped. It will only decrease the cwnd size when buffer of s_0 is saturated, but that is already too late. In other words, the buffer in the intermediate router has turned the packet drops into an **un-timely** indication of congestion, which is bad since TCP rely on timely communication of congestion via packet drops.

Bufferbloat causes increase in queueing delay, and thus causes end users to experience increase in latency, which is the sum of transmission delay, processing delay, and queueing delay. [Sivov and Sokolov, 2012] Bufferbloat also causes jitters and decreases the overall throughput of the network.

Methods We emulate our Assignment topology using mininet. Then, we simultaneously perform the following three tasks

- start a long lived TCP flow sending data from h_1 to h_2, and
- send 1 ping per 0.1 second from h_1 to h_2, and
- spawn an web server on h_1, and download the webpage from h_1 once every two seconds.

This simulation is repeated for three different queue sizes, $Q = 5/20/100$ pkts. Then, for each max queue size, we plot the time-series of the long-lived TCP flow's cwnd, the RTT reported by ping, the webpage download time, and the queue size at the router s_0.

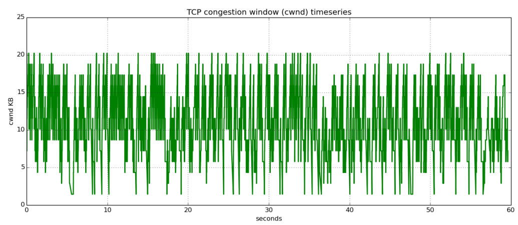
Results Figures

Discussion For the sake of mitigating the bufferbloat problem, we can try

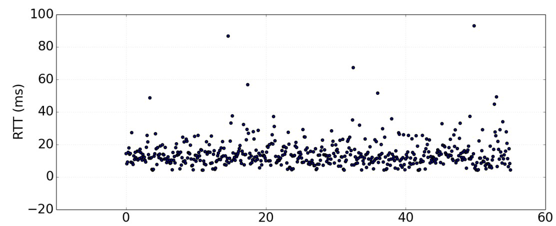
- The probably simplest approach is to decrease the buffer sizes at each hop. This way, when congestion happen, buffers fill up quickly, and then rely on packet drops as a

Queue Size	5	20	100
Mean(ms)	0.67138	0.50437	1.27632
Stddev(ms)	0.54124	0.12420	0.54989

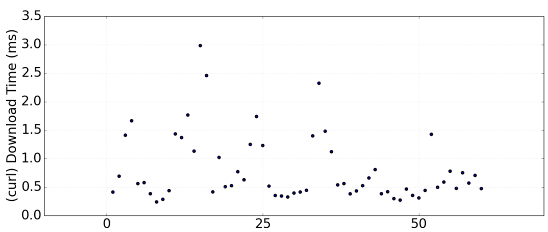
Table 1: Mean and standard deviation in fetch time for all three experiments.



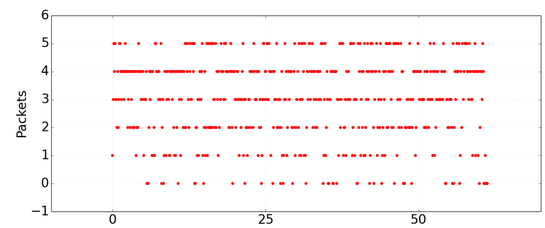
(a) Long-lived TCP flow's cwnd



(b) RTT reported by ping

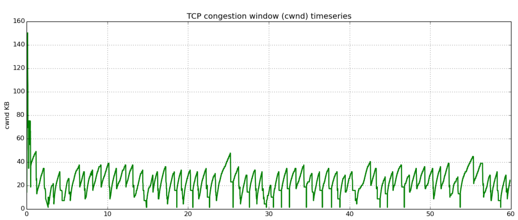


(c) Webpage download time

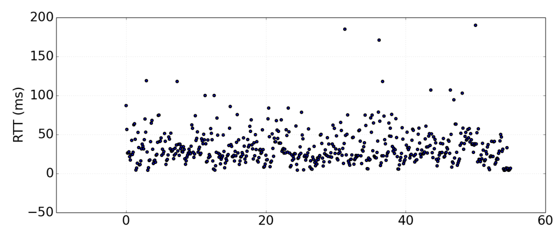


(d) Queue size at the router

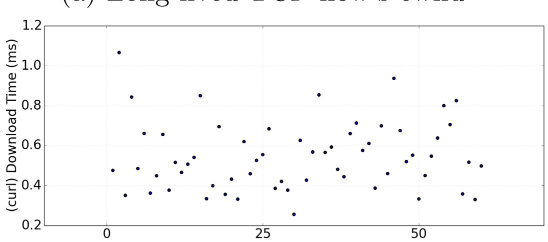
Figure 1: Long-lived TCP flow's cwnd, RTT reported by ping, webpage download time, and queue size at the router with max buffer size of 5.



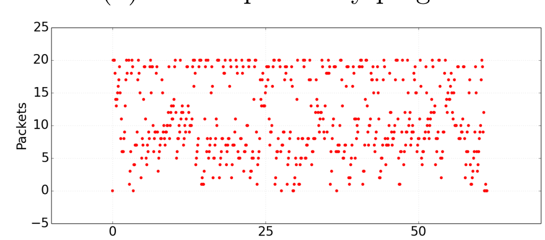
(a) Long-lived TCP flow's cwnd



(b) RTT reported by ping

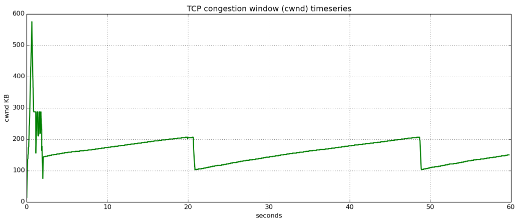


(c) Webpage download time

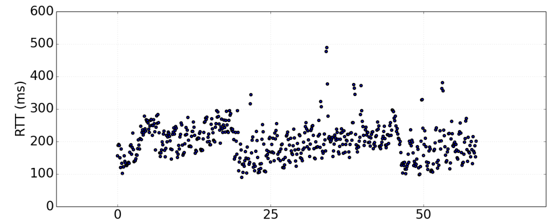


(d) Queue size at the router

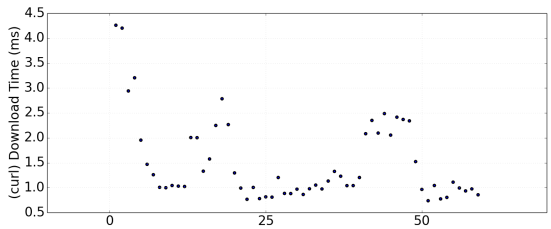
Figure 2: Long-lived TCP flow's cwnd, RTT reported by ping, webpage download time, and queue size at the router with max buffer size of 20.



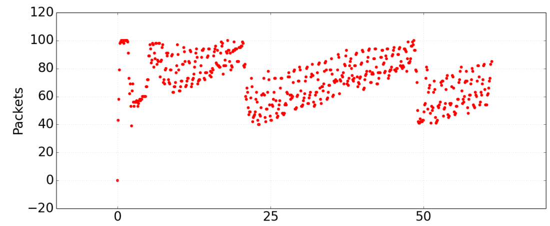
(a) Long-lived TCP flow's cwnd



(b) RTT reported by ping



(c) Webpage download time



(d) Queue size at the router

Figure 3: Long-lived TCP flow's cwnd, RTT reported by ping, webpage download time, and queue size at the router with max buffer size of 100.

timely indication of congestion. This approach is, however, not optimal. We originally introduced buffers to deal with bursts of packets, and make networking more smooth in general. Reducing buffer size will cost us the ability to deal with bursts in the network.

- Use a delay based congestion avoidance algorithm rather than delay based. In this way, excessive buffering, even when there is no drop in packets, will signal that a congestion is occurring. We can then control the cwnd size based on this information. [Sivov and Sokolov, 2012]

References

[Allman and Paxson, 2009] Allman, M. and Paxson, V. (2009). Tcp congestion control. Rfc, RFC Editor.

[El-Sayed et al., 2011] El-Sayed, A., HAGGAG, S., and EL-FESHAWY, N. (2011). A survey for mechanisms for tcp congestion control. *International Journal of Research and Reviews in Computer Science (IJRRCS)*, 02:676–682.

[Sivov and Sokolov, 2012] Sivov, A. and Sokolov, V. (2012). The bufferbloat problem and tcp: Fighting with congestion and latency.