Detecting the Sawtooth pattern of router buffer occupancy due to congestion control in TCP flows

EE384Y Project Proposal, Spring 2006
Pratibha Gupta

Motivation

The AIMD congestion control in TCP is implemented by changing the congestion window size per connection. In practice, however there hasn't been evidence of the sawtooth behavior of buffer occupancy as predicted by the protocol. Internet routers typically contain buffers larger than the delay-bandwidth product, $\overline{RTT}xC$, where $\overline{RTT}$ is the average round trip time, and C the link capacity. Subsequent work has shown that these buffer sizes are over-provisioned. Current implementations of operating systems set a maximum congestion window size of between 10 (in Windows XP) to 40 (in Linux 2.4 and 2.6). In addition, many Internet flows today are short-lived, and end with the packets still buffered in routers along the way for retransmission. This project aims to systematically study the buffer occupancy along routers to determine if the sawtooth pattern of the contention window size is ever seen, and re-assess optimal buffer sizes in the light of the findings.

Background

It has been shown in [2] that the aggregate window size of desynchronized TCP flows follows a Gaussian distribution about the average congestion window size, and high link utilization can be achieved by using smaller buffers of size $(\overline{RTT}xC)/\sqrt{n}$, where n is the number of flows . With an analysis using Chernoff bounds that assumes independent flows, tighter bounds have been obtained. [4] proposes a non-work conserving randomized scheduling algorithm requiring buffer sizes O(log(Nd)) where N is the number of packets, and d is the dilation, or network diameter. It further extends the analysis to obtain a constant queue size bound of $\log(c+d)\ 2^{O(\log^*(c+d))}$, where c is the maximum congestion on a link. In [1], the authors show that in an over-provisioned network, O(log$W$) buffers are sufficient, where W is the window size of each flow.

With the bottleneck in achieving high throughput in TCP flows often being the setting of the sender and receiver buffer sizes, and not the available bandwidth of the links, it is an interesting question as to whether lower bounds on the buffer sizes can be derived, taking these into account. In addition, analysis for more accurate self-similar traffic models and UDP flows [15] that may in future start to dominate the Internet traffic present interesting areas for exploration.

Tasks / Deliverables

The project will start with surveying existing tools for network monitoring and measurements used in auto-tuning - Web100 [8], net100 [9], NTAF [10], PlanetLab [11], DRS [14] and others [12] [15] built on standard Unix utilities like iperf, pathdiag, pathchar, tcptrace etc.
Then various traffic scenarios will be implemented for monitoring TCP flows on links of varying RTT, bandwidth, competing stream traffic and congestion control / avoidance protocols. This may involve writing additional tools that permit accurately reporting the buffer occupancy along the routers in the path.
If a sawtooth pattern is detected, the timescales and duration of flows that cause it will be estimated, and the effect of buffer sizing on the congestion control mechanism will be analyzed. Factors such as short-lived flows, over-provisioned buffer sizing and premature start of congestion avoidance mode at a small window due to overflow in short queues during the TCP slow start will be studied.

Schedule

4/17 – 4/23	Survey of existing tools, methodology and preliminary measurements
4/24 – 4/30	Designing TCP traffic scenarios to induce congestion on low-delay links, extension to high-delay, high bandwidth links, on available resources [13]
5/1– 5/14	Implement additional experiments and tools to detect evidence of AIMD behavior in CW size and buffer occupancy
5/15 – 5/21	Based on findings, review experimental methodology, or analyze factors that could result in saturation of CW size. Progress Report and feedback.
5/22 – 5/28	Analyze the impact of AIMD parameters such `rmem_max`, `wmem_max`, `tcp_rmem`, `tcp_wmem`, `maxsockbuf`, `tcp_adv_win_scale`, and congestion avoidance algorithms [7], `tcp_moderate_rcvbuf`, `tcp_sack` on buffer occupancy.
5/29 – 6/4	Based on findings, establish relationship between optimal buffer size, RTT, link capacity and congestion avoidance parameters. Documentation and report.

Team Role / Collaboration

The idea for this project was initially proposed by Guido Appenzeller, and Yashar Ganjali. I will be keen to collaborate with them and get insights from their prior work in the course of this project. By teaming with other interested students in EE384Y, and based on the findings, the scope of the proposed project could be expanded.

References

[1] "Routers with very small buffers", M. Enachescu, Y. Ganjali, A., N. McKeown, and T. Roughgarden, Proceedings of the IEEE INFOCOM'06, April 2006

[2] "Sizing Router Buffers", Guido Appenzeller, Isaac Keslassy and Nick McKeown, ACM SIGCOMM 2004, Portland, August 2004; Stanford HPNG Technical Report TR04-HPNG-060800

[3] "Part I: Buffer sizes for core routers", Damon Wischik, Nick McKeown, ACM SIGCOMM Computer Communication Review 2005 , Volume 35 , Issue 3, pp75 - 78

[4] "Packet routing and job-shop scheduling in O(congestion+dilation) steps", Leighton, Maggs, Rao, Combinatorica, Vol. 14, No. 2, 1994, pp. 167-180.

[5] "Buffer sizing for congested Internet links", A. Dhamdhere, H. Jiang, C. Dovrolis, INFOCOM 2005, Vol 2, pp1072 – 1083

[6] "Open Issues in Router Buffer Sizing", A. Dhamdhere, C. Dovrolis, ACM SIGCOMM Computer Communications Review (editorial section), January 2006.

[7] IETF RFCs 2581, 2861, 2914, 3649, 3742

[8] Web100 http://www.web100.org/
M. Mathis, J Heffner and R Reddy, "Web100: Extended TCP Instrumentation for Research, Education and Diagnosis", ACM Computer Communications Review, Vol 33, Num 3, July 2003

[9] net100 http://www.csm.ornl.gov/~dunigan/netperf/index.html

[10] LBNL Network Tools Analysis Framework http://dsd.lbl.gov/DIDC/NTAF/

[11] PlanetLab http://www.planet-lab.org/

[12] Jeffrey Semke, Jamshid Mahdavi, and Matthew Mathis, "Automatic TCP Buffer Tuning", ACM SIGCOMM Computer Communications Review, Vol 28, No. 4, Oct 1998.

[13] Virtual Network System (VNS), http://yuba.stanford.edu/vns/
"The virtual network system", M. Casado, N. McKeown, Proc. of the ACM SIGCSE 2005

[14] "On estimating end-to-end network path properties", Mark Allman, Vern Paxson, ACM SIGCOMM 99, Vol 29, Issue 4

[15] SLAC IEPM
http://www-iepm.slac.stanford.edu/monitoring/bulk/window-vs-streams.html
http://www-iepm.slac.stanford.edu/monitoring/load/