Proceedings of the 7th International Conference on Applied Informatics
Eger, Hungary, January 28–31, 2007. Vol. 2. pp. 403–408.

TCP dynamics and congestion control on asymmetric lines*

Péter Orosz[a], János Sztrik[a], Che Soong Kim[b]

[a]Faculty of Informatics, University of Debrecen
e-mail: oroszp@unideb.hu, jsztrik@inf.unideb.hu

[b]Department of Industrial Engineering, Sangji University
e-mail: dowoo@mail.sangji.ac.kr

Abstract

A growing number of network computers are connected to the Internet through asymmetric connections such as ADSL, CableNet, Satellite link and other broadband solutions. However, the asymmetric nature of such connections may have significant impact on the performance of TCP transmission as upstream bandwidth can be easily overloaded. In this situation TCP's self-clocking mechanism may be misled when TCP ACK packets are delayed due to the congested upstream link that will result degradation in TCP throughput. Purpose of the present letter is to show the effect of asymmetric physical lines on TCP dynamics based on our empirical measurement results. Therefore, we addressed the evaluation of alternative TCP congestion control mechanisms as well.

Keywords: TCP, congestion control, ACK, asymmetric bandwidth, traffic dynamics.

1. Introduction

ADSL is a data communication technology that enables faster data transmission over copper phone lines that a conventional voice-band modem can provide. The available bandwidth is greater in one direction (downlink) than the other (uplink). There are technical reasons of the implementation of asymmetric bandwidth on long distance connections. More crosstalk is likely to be experienced between the local loops that are connected and aggregated to the DSLAM with high density interfaces. Therefore, the uplink signal is the weakest at the noisiest part of the local loop. Another limiting factor is the available frequency range for upstream

*Research is supported by Hungarian Scientific Research Fund-OTKA K 60698/2006 and KOSEF-HAS Bilateral Scientific Cooperation under grant KOSEF-F01-2006-000-1004-0.

traffic that is much narrower than that of the downstream which enables lower uplink bandwidth.

We can easily point out how lower uplink bandwidth limits the performance of TCP in some common traffic situation. As we know TCP's congestion control aims to effectively utilize network bandwidth [1]. Congestion control feature of TCP is actually a self clocking mechanism where reception of ACK triggers next packet transmission. ACKs spend any time in buffer queues during their transit through the network path, their arrivals may be delayed moreover, arrival time of ACKs can get closer to each other (ACK burst) that actually misleads self-clocking mechanism on sender side to send more data than the network can handle which will cause congestion and loss of efficiency [2].

2. Investigated TCP variants

2.1. TCP BIC

BIC solves the performance problem of TCP in fast long distance networks. Accordingly, the main feature of BIC-TCP is its unique window growth function [3]. TCP severely underutilizes the available bandwidth because of its slow response to available bandwidth. BIC solves this problem by modifying TCP window increase function. It can scale its window increase function to be more aggressive without diminishing the fairness properties of TCP too much. This binary search technique allows the window increase to be logarithmic.

2.2. TCP Reno

Reno is the implementation of Van Jacobson research and was the default congestion control scheme until recently.

2.3. TCP Vegas

Vegas is based on Reno and tries to track the sending rate through looking at variances to the RTT along with other enhancements.

2.4. TCP Westwood

Westwood relies on mining the ACK stream for information to help it better set the congestion control parameters [4]: Slow Start Threshold (ssthresh) and Congestion Window (cwin).

3. Testbed

- End-point with 512/128 Kbit/sec ADSL connection towards the ISP's DSLAM.

- Server box with 100 Mbit/sec full-duplex LAN connection. 1 Gbit/sec backbone bandwidth (between the ISP and the university network).
- Characteristics of the measurement data:
- FTP transmission of large binary files (both directions)
- Software environment: Linux based system - Fedora Core 6, kernel v2.6.18-1
- FTP server: PureFTPd v1.0.21-9
- Traffic analysis tool: Ethereal v0.99.0, tcpdump

Linux kernel (version 2.6.13+) supports pluggable TCP congestion control, therefore our measurements were performed based upon this kernel feature:

/sbin/sysctl -w net.ipv4.tcp_congestion_control=bic|cubic|westwood|reno|hybla|htcp|vegas

Normally, TCP memory buffers are set by the kernel at boot stage. According to the network link bandwidth and latency, we calculated the BDP value of the applied physical link so to determine optimal buffer sizes. Based upon these values we could tune TCP buffer parameters (`tcp_mem,tcp_rmem,tcp_wmem,tcp_app_win,tcp_sack,tcp_wmax,tcp_rmax`) if required.

Result of BDP calculation for the test link:
The latency *(D, RTT)* of the ADSL link was 25 ms. Maximum physical bandwidth *(B)* was 512 Kbit/sec on downlink and 128 Kbit/sec on uplink direction.

$$B \times D = 0.025 \times 64\,\mathrm{KB} = 1.6\,\mathrm{Kbytes}$$

The following generic rule shows the optimal TCP receiver buffer size derived from BDP:

$$\textit{Receiver buffer} >= RTT \times BW$$

As default buffer sizes are far larger than the calculated BDP value, we left all TCP parameters on their default values. Selective ACK feature was turned on.

4. Test method

Our test had three phases where we generated TCP-based FTP traffic between the server and the client. We reached a 100% downlink load on one flow. In uplink direction only ACKs belonging to that TCP flow were passed through. At the second phase we initiated an independent TCP flow to the opposite direction, from the client towards the server machine. Accordingly, uplink load easily reached 100%. Nevertheless, TCP throughput has been squarely determined by the arrival time of ACK packets. They are transmitted through the uplink with latency due to the congested link. In this phase, throughput of downstream TCP flow falls down to 70% of the available physical bandwidth, depending on the applied congestion control. By initiating another TCP flow from client to server – resulting

greater congestion on the uplink – further dramatic degradation was experienced on downstream side. Effective throughput now falls down to an average of 40%. Under these circumstances, physical downlink bandwidth cannot be effectively utilized by TCP. Accordingly, uplink load has a direct effect on downstream TCP performance. The investigated congestion control mechanisms performed similarly at the first phase while they showed different results within the last two ones.

Considering the asymmetric nature of ADSL, active network users (who generate considerable upstream traffic such as video conferencing, VoIP, sending large Emails, etc.) may face with TCP performance problem on downlink whilst it is actually not congested. For an extreme example, TCP Vegas throughput is far the lowest as long as any TCP flow appears on uplink. Its effective throughput falls down dramatically and became equal to or under the uplink throughput rate. As we know, Vegas monitors at the variance of RTT to tune the window size. In congested uplinks the ACK packets may have an arrival time fluctuating in a quite large interval that misleads Vegas algorithm. Conventional Reno algorithm is known by its relatively slow response the available bandwidth and novel high speed TCP variants operate with more aggressive window growth algorithms.

5. Measurement results

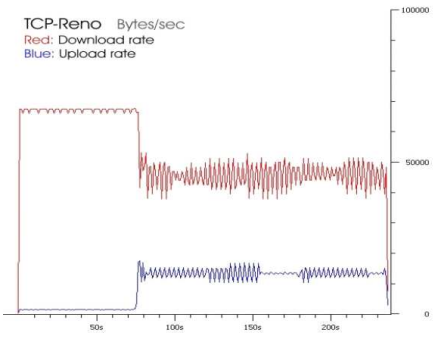

Figure 1: TCP Reno throughput

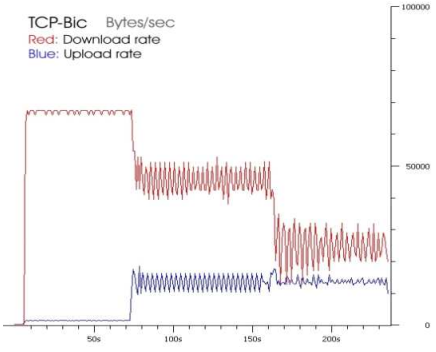

Figure 2: TCP BIC throughput

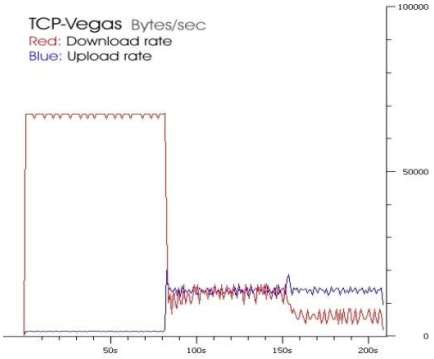

Figure 3: TCP Vegas throughput

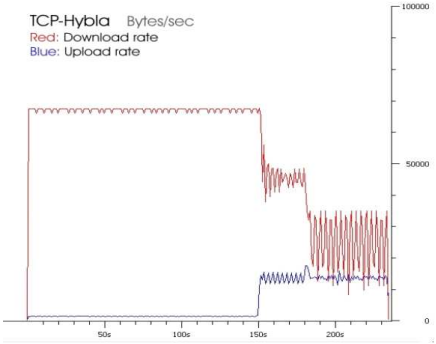

Figure 4: TCP Hybla throughput

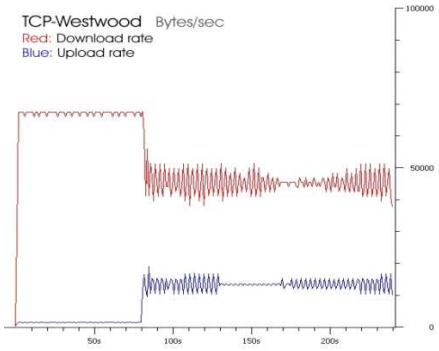

Figure 5: TCP Westwood throughput

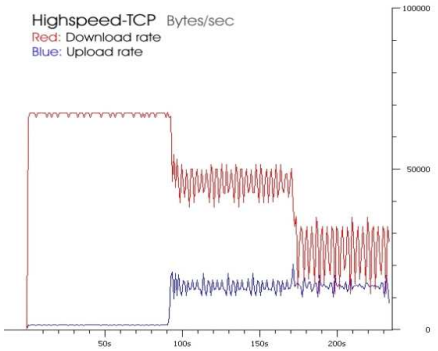

Figure 6: Highspeed TCP throughput

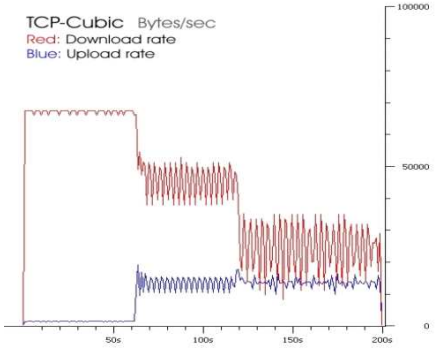

Figure 7: TCP Cubic throughput

6. Summary

- Uplink of ADSL connection has a very low bandwidth that can be easily overloaded therefore it may seriously degrade downstream TCP performance.
- TCP self-clocking may be misled into window size degradation by uplink congestion.
- ACK compression on the uplink may cause downlink congestion.
- TCP-Vegas' window growth function has no benefit on asymmetric connections therefore overall downstream TCP performance is dramatically degraded in congested uplink situation.
- HTCP/Bic/Cubic showed significant throughput fluctuation as soon as the uplink became congested due to its more aggressive window growth function.
- Size of TCP memory buffers are set by the kernel at boot stage. According to network link bandwidth and latency, BDP should be calculated so to

determine optimal TCP buffer sizes. Based upon these values TCP buffer parameters can be tuned for optimal performance.

References

[1] Allman, M., Paxson, V., Stevens, W., TCP congestion control, RFC 2581 (RFC2581), http://www.faqs.org/rfcs/rfc2581.html

[2] Mogul, Jeffrey C., Observing TCP Dynamics in Real Networks, ACM Press, 1992

[3] TCP Bic/Cubic reference pages: http://www.csc.ncsu.edu/faculty/rhee/export/bitcp/index_files/Page703.htm

[4] TCP Westwood reference pages: http://www.cs.ucla.edu/NRL/hpi/tcpw/