Protocols For High-Speed Networks With High Bandwith*RTT Product

Petr Holub

Introduction

In current IP based network TCP protocol is widely used as general purpose reliable data transfer protocol. While TCP with standard Tahoe and Renoe congestion control was flawlessly working for either slow networks or for fast networks with small round-trip time (RTT) it faces it's limits when applied to current long distance high speed networks with high RTT. This poster summarizes state-of-art protocols and developments for such networks.

To improve networking performance on this environment there are three major strategies: TCP tuning and new TCP based protocols, non-TCP based protocols, and generic strategies for bulk data movement.

TCP Tuning

Most of current implementation of feature TCP Reno congestion control algorithm usually with some improvements of "hackish" kind (e.g. it is well known fact that Linux kernel doubles internally window size compared to window size requested). Both Tahoe and (New)Reno algorithms use packed drop as a sign of congestion while newer Vegas algorithm tries to avoid congestion by monitoring RTTs and linearly decreasing packet rate if congestion is imminent. One simple methods for improving performance is enlarging TCP window size. Problem with standard TCP connection on high-speed network with long delay can be illustrated on following example [3]: achieving 10 Gbps sustained rate on network with 100 ms RTT (that is rather low RTT in global networking environment) and 1500 bytes segment (standard Ethernet without jumbo frames) size would require average congestion window of 83,300 segments and packet drop at most every 5.10^9 packet, i.e. 1⅔ hours which is clearly not very realistic presumption. TCP with standard Van Jacobson's congestion control mechanism [1] uses additive increase $cwnd = cwnd + (1/cwnd)$ for each packet received and multiplicative decrease $cwnd = 0.5 * cwnd$ for each packet loss considered to be congestions (AIMD).

TCP instrumentation that allows for protocol auto-tuning and improved fine-grained manual tuning was implemented by Web100 [2] and Net100 [3] projects for Linux. Web100 has auto-tuning features using user-land daemon and offers some MIB based monitoring. Net100 projects further develops Web100 features by interfacing Netlogger to achieve better performance.

TCP Based Protocols

Other simple approach based on standard TCP was to use multiple parallel streams. This type of transport was first realized by MulTCP project and it was adopted by several pragmatic applications for bulk data transfers (see section below).

ScalableTCP [3] uses "almost standard" additive increase and exponential decrease mechanism but adds two constants to it: $cwnd = cwnd + a$ for additive increase and $cwnd = cwnd - b*cwnd$ for exponential back-off. Use of $a = 0.01$ and $b = 0.125$ is be motivated by considering Scalable TCP's impact on legacy traffic, bandwidth allocation properties, flow rate variance, convergence properties, and control theoretic stability. Comparison of standard TCP vs. Scalable TCP behavior is illustrated in Fig. 1. Very similar to Scalable TCP is GridDT approach [5].

HSTCP [4] uses similar principle but it makes constants a and b variables depending on $cwnd$: $a(w) = 2*hw^2*hp*b(w)/(2 - b(w))$, $b(w) = (hd - 0.5)*(log(w) - log(W))/(log(W1) - log(W)) + 0.5$. Because of rather complicated nature of there expressions prototype implementation pre-calculates a and b values for some range of window sizes.

Very useful tool for obtaining optimal throughput on the network is proposed IP extension

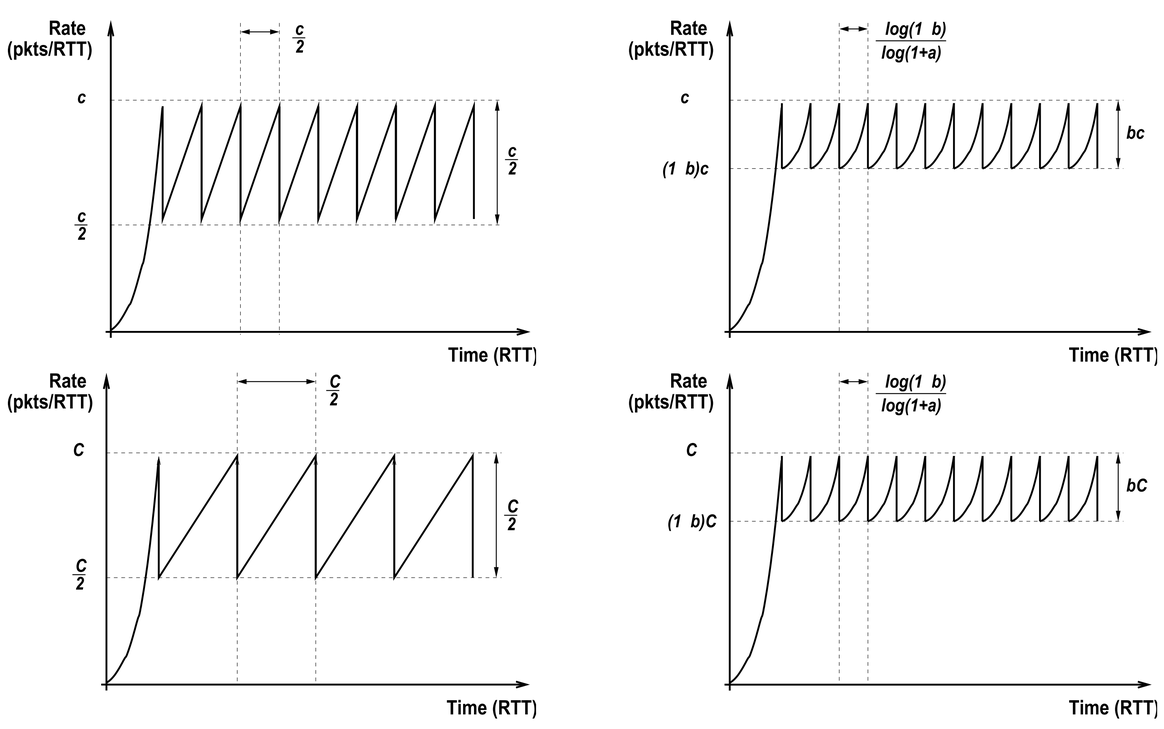

Fig. 1. Traditional vs. Scalable TCP

by Ramakrishnan, Floyd and Black called Early Congestion Notification (ECN) [6]. It provides different method of congestion notification compared to traditional loss-based method. Many of new protocol proposals and TCP improvements build on ECN. There's however danger with relying solely on ECN without any other mechanism (at least verifying that ECN is not erased underway as specified in [6]) since on some router/firewalls ECN field can be erased and congestion information gets lost.

Examples of TCP-based protocols based on ECN usage are Fast AQM Scalable TCP (FAST) and ECN [7] and TCP–E [8]. TCP–E for example tries to use ECN with correctly configured AQM on the router and it requires modification of TCP stack at the receiver: instead of reflecting ECN bit in each ACK until sender backs off the congestion window it has to reflect ECN bit in ACK message only once. It suggests to freeze congestion window when ECN flagged ACK arrives (which differs from original ECN proposal which suggests to react to ECN notification in the same way as to the congestion, i.e. to back-off congestion window by ½); this should provide almost 100% link utilization as with fully utilized link all ACK will have ECN bit set. It also requires small number of random (Bernoulli) losses at router AQM to improve temporal unfairness (these losses can occur instead offsetting ECN bit) which induces multiplicative decrease and results in convergence to temporarily fair behavior.

Another interesting improvement of TCP is QuickStart (QS) [9]. It proposes new 4-byte long option header which has 2 fields: QS TTL and Initial rate (suggestion is to use packets per 0.1 s). Sender willing to use QS sets QS TTL to random value and Initial rate to desired value. All routers on the way to receiver that understand and approve QS decrement QS TTL by 1 and decrease Initial rate if needed. Receiver sends feedback in SYN/ACK packet so sender knows if all the routers on the way participated, has RTT measurement. So sender sets initial adequate congestion window and than uses AIMD as usual. Disadvantage of Quickstart is that it requires modification to IP layer.

Non-TCP Based Protocols

There are also some new protocol designs completely free of TCP heritage. For example XCP (eXplicit Congestion control Protocol) [10] and STP (Scheduled Transfer Protocol) [11]. STP protocol is very simple and therefore it is easy to implement hardware acceleration into hardware network interfaces. There is implementation of such an acceleration for Linux [12].

Strategies For High Speed Bulk Transfers

In present grid environment there's strong need for high-speed data movement strategies because parts of grid are usually widely distributed all around the world and connected with high bandwidth lines with high RTT. Because networking stack modification is not always feasible or desirable, there is number of strategies for high speed networks in such environment implemented on application level using current TCP and UDP protocols.

GridFTP [13] is very common software for bulk data transfer in grid environment as it is both well performing and integrated with Globus grid middleware. From networking point of view it uses multiple parallel TCP streams to improve transmission characteristics.

bbFTP [14] also uses mutli-stream TCP and it is optimized for transmission of files larger than 2 GB. Furthermore it uses big windows (RFC 1323), on-the-fly data compression, automatic retry, customizable time-outs, transfer simulation, AFS authentication integration, and RFIO interface.

Other interesting approach is Tsunami protocol [15]. It uses out-of-band TCP connection for control purposes and UDP for data channel. Transfer parameters negotiation, retransmission requests, end of transmission negotiation are run via TCP channel. Data sender that is called server in the Tsunami protocol steers sending speed using inter-packet delay based on client's (data receiver) requests. It also polls client for retransmission requests periodically before sending further data. Tsunami uses exponential increase and exponential back-off and doesn't collapse when just low packet loss appears. There are many tunable parameters: speedup/slowdown factors, error threshold, maximum retransmission queue, retransmission request interval etc.

Both bbFTP and Tsunami are favorite protocols for testing performance of current networks and also for some production runs in grid environment.

Internet Backplane Protocol (IBP) has been designed as middleware for distributed storage. It features routing on it's own application layer and many other features but is not so interesting from protocol point of view.

Conclusions

Currently it seems that for general traffic some modifications of TCP stack will improve transfer behavior on high-speed networks (with mostly high RTTs). Of these Scalable TCP or similar modification can be seen as the most convenient because it modifies TCP stack only (no changes in other layers of the stack nor network required) and it is sufficient to modify sender's stack only if minimalistic modifications are required.

References

[1] Stevens W. R. *TCP/IP Illustrated, Vol. 1: The Protocols.* Addison-Wesley Reading, MA, 1994.
[2] Web100 project. http://www.web100.org/. Net100 project. http://www.net100.org/
[3] Kelly T. "Scalable TCP: Improving Performance in Highspeed Wide Area Networks." December 2002. http://www-lce.eng.cam.ac.uk/~ctk21/scalable/
[4] Floyd S. "HighSpeed TCP and Quick-Start for Fast Long-Distance Networks", PFLDnet2003 Workshop, CERN, 2003. http://datatag.web.cern.ch/datatag/pfldnet2003/slides/floyd.pdf
[4b] Li T. L., Fairey G. "Implementing High Speed TCP (aka Sally Floyd)." October 2002. http://icfamon.dl.ac.uk/papers/DataTAG-WP2/reports/ppt/20021001-Yee.ppt
[5] Ravot S., ``GridDT", PDFLnet workshop 2003, February 3--4, 2003, CERN, Geneva, Switzerland http://datatag.web.cern.ch/datatag/pfldnet2003
[6] Ramakrishnan K., Floyd S., Black D. "The Addition of Explicit Congestion Notification (ECN) to IP." RFC 3168, September 2001. ftp://ftp.isi.edu/in-notes/rfc3168.txt
Floyd S. "ECN (Explicit Congestion Notification) in TCP/IP." http://www.icir.org/floyd/ecn
[7] Jin C., Wei D., Low S. H., Buhrmaster G., Bunn J., Choe D. H., Cottrell R. L. A., Doyle J. C., Newman H., Paganini F., Ravot S., Singh S. "FAST – Fast AQM Scalable TCP." http://netlab.caltech.edu/FAST/
[8] Kamra A., Misra V., Towsley D. "Achieving High Throughput in Low Multiplexed, High Bandwidth, High Delay Environments", PDFLnet workshop 2003, February 3–4, 2003, CERN, Geneva, Switzerland http://datatag.web.cern.ch/datatag/pfldnet2003
[9] Jain A., Floyd S. "Quick-Start for TCP and IP." http://www.ietf.org/internet-drafts/draft-amit-quick-start-02.txt and http://www.icir.org/floyd/quickstart.html
[10] Katabi D., Handley M., Rohrs C. ``Congestion Control for High Bandwidth-Delay Product Networks." http://www.ana.lcs.mit.edu/dina/XCP/
[11] http://www.sgi.com/peripherals/networking/st_whitepaper.pdf and
[12] Pekka Pietikäinen. "Hardware-Assisted Networking Using Scheduled Transfer Protocol On Linux". University of Oulu, Oulu, Finland. Diploma thesis. 2001.
[13] GridFTP in Globus Toolkit, http://www.globus.org
[14] Farrache G. "bbFTP", IN2P3 computing Center in Lyon, France. http://doc.in2p3.fr/bbftp/
[15] Wallace S. *et al.* "Tsunami File Transfer Protocol." ANML, Indiana University, USA. http://www.indiana.edu/~anml/anmlresearch.html