Marsland Press
The Journal of American Science 2009:5(2) 119-128

Building Multihoming Awareness into FAST TCP: A Simulation Analysis

M. Junaid Arshad, M. Saleem Mian
Department of Computer Science and Engineering, University of Engineering and Technology, Lahore, Pakistan

Abstract:
The remarkable growth of high-capacity network environments and communication systems (from point-to-point connections or multipoint-to-point or multihomed structures) reveals that these growing technologies (such as Grid computing environments) are now requesting improved and advanced network transport-layer level features. An end-to-end transport layer multihoming using concurrent multipath transfer (CMT) of data is an efficient approach that will be able to meet the demand for required bandwidth and connectivity, but the current multihomed-aware protocols (like SCTP or pTCP) are not designed for high-capacities and large-latencies networks, they often show throughput degradation while sending and sharing huge data files over long-distance WANs. It has been shown that SCTP-CMT is more sensitive to receiver buffer (rbuf) constraints, and this rbuf blocking problem causes considerable performance problems if multiple paths are utilized simultaneously. In this research paper, we demonstrate the weakness of SCTP-CMT rbuf constraints and, we then identify that rbuf blocking problem in SCTP multihoming is mostly due to its loss-based nature for detecting network congestion. We present a simulation-based performance comparison of FAST TCP versus SCTP in high-speed networks. The objective of this article is threefold: to discuss rbuf blocking problems in SCTP-CMT; to describe some proposed transport protocols (like FAST TCP) that solve a number of throughput issues; and finally, to gain new insight into these protocols and thereby suggest avenues for future research. [Journal of American Science 2009;5(2):119-128] (ISSN: 1545-1003)

Keywords: FAST TCP; SCTP-CMT; Multihoming; Transport Protocols; Receiver Buffer

1. Introduction

The Internet is the network of large-scale group of connected computers around the world that sends out data using packet switching technique based on the TCP/IP stack. With a continuous improvement in the field of communication technologies and infrastructures by means of enhancing the functionalities of the existing protocols, the Internet has achieved the massive success and popularity. Since, over the time with the growing and accelerating progresses in communication patterns and wildly demands for spare capacity and connectivity, the Internet in almost every aspect frequently experiences modifications and changes in order to bring up-to-date.

During the last few years, a countless fast improvements have been observed in the area of Grid applications, peer-to-peer networks and distributing systems. Because of continual

developments in network computing, storage space equipments, and fast communication devices, along with the development of world-wide and national Grid infrastructures [DeFanti et al., 2003], thus demanding a proper transport protocol with some enhanced features, with the purpose of realizing the idea of distributed cooperation, such as transfer a huge quantity of application data and reachability to distant resources (storage capacities and/or computing facilities) through a high-speed WANs.

In this regard, first it would be significant to illustrate the limitations in the existing loss-based congestion control protocols (like TCP [Allman et al., 1999] and SCTP [Stewart et al., 2000; Iyengar et al., 2004]) when a large amount of data is shares and transferred over a WANs. The key issues we come-across, and aimed to resolve, is that the existing loss-based protocols don't scale to this regime, that is to say, they are not well appropriate for the future high-speed and long-distance networks, which motivates the design of new distributed algorithms for large bandwidth-delay product networks (i.e., FAST TCP [Jin et al., 2003, Wei et al., 2007]). The algorithms used by TCP and SCTP for controlling the network congestion are founded on RFC 2581 [Allman et al., 1999] and RFC 2960 [Stewart et al., 2000], respectively. Their key mechanisms are Slow Start and congestion avoidance phase, and they all make use of the AIMD (additive increase/multiplicative decrease) approach [Jacobson, 1988] that additively grows the congestion window to obtain the offered network bandwidth and immediately reduces the congestion window, as the network available limit is reached and network congestion is detected through packet losses (hence achieving a low utilization of the network bottleneck-link).

But this limitation is avoided by the delay-based approach of FAST TCP; it is one of such algorithms that are designed for high-bandwidths large-latencies networks, it tries to quickly balanced networks into effective, stable and reasonable operating positions. FAST TCP congestion control mechanism reacts to both queuing-delay and packet loss, since loss of packet simply gives one bit of knowledge regarding the network congestion point, while network delay is a persistent measure and in principal offers additional knowledge concerning the network (which in turn provides efficient link utilization).

Another approach that is used for improving the end-to-end throughput and link redundancy is a transport layer multihoming [Ohta, 2002]. Multihoming is the ability of a host or site to access remote destination via more than one upstream connection, usually from different providers. SCTP supports concurrent multipath transfer (CMT) [Iyengar et al., 2004] of data between the multihomed hosts, but the existing TCP [Allman et al., 1999] and its variant (such as FAST TCP) do not support multihoming. In the early days of the Internet extensive use of multi-homing was not practicable by reason of cost restrictions, but nowadays, network IP addresses form the several service providers have turned into ordinary and affordable things. Now, by taking advantages of low-cost of Internet-access and network-interfaces, the content providers are establishing wired and wire-less connections for having simultaneous connectivity via multiple ISP's, for getting high capacity and redundancy purpose.

Thus, we believe that a transport layer protocol right for huge data shares/transfers efficiently over high-speed long-distance networks, for example, as in Grid computing should have at least the following properties:

- A transport protocol that has an effective performance as well as robustness, not only in

local transfer of small files but also in high-speed long-distance transfer of extremely large files, along with the capability for independent up-gradation of its components.

- A transport protocol that has the ability for transferring of data through concurrent multipath using multihoming or some other means.
- A transport protocol that could run on the same Internet infrastructure (with minimum modifications) we have today.

Since the protocols performance evaluations are addressed in several research articles [Floyd, 2003; Kelly, 2002; Jin et al., 2003; Bullot et al.] suggesting revisions to the standard TCP's AIMD mechanisms, but with the vast development of wide-area Grid computing environments which are connected through fast optical fiber links, there is still continuing examinations to assess the existing set of transport protocols for high-bandwidth networks to select the optimal protocol. In this study, our purpose is not to choose a victor because many of the protocols are still in progress. Somewhat, we anticipate to get in detail knowledge of the situations wherein different protocols would work better, and the sources of performance degradation.

The remainder of this paper is organized as follows. In Section 2, first we present an overview of the protocols [i.e., SCTP and FAST TCP] and then we present our experimental setup and progressively analyze the behavior of FAST TCP compared to SCTP by using a simple network topology in ns-2 [VINT Project, NS2]. In Section 3, we delineate the rbuf blocking problem in SCTP-CMT [Iyengar et al., 2005] and identify the dilemma degrading its performance in the presence of a bounded receive buffer. Finally in Section 4, we present the conclusions of this work.

2. FAST TCP vs. SCTP in high-speed networks

SCTP [Stewart et al., 2000] is a new reliable session-oriented transport protocol operating on top of the Internet Protocol (IP). SCTP and TCP use the basic AIMD algorithm to adjust their windows sizes. These loss-based protocols achieve congestion control effectively in the present slow-rate data networks but they may operate inefficiently in such networks environment where the data paths have large bandwidth-delay product (BDP) characteristics; because the AIMD approach is extremely conventional and not intended for big window size data streams. Initially, it consumes much time for an optimal size of window source to recover after a back-off, as a result the available link bandwidth is not efficiently used [Floyd, 2003], secondly it identifies network congestion actually as soon as packets are lost in the network.

FAST TCP [Jin et al., 2003] is an alteration to the standard TCP congestion control approach for large-delay high-bandwidth communication networks. The congestion control using delay-based algorithm (i.e., FAST TCP) in contrast is fundamentally different from AIMD approach; it exploits queuing-delay as well as packet loss as the key for network congestion indication and its benefit over loss-based approach is small at low speed, but significant at high speed networks.

The congestion window (cwnd) update algorithm of FAST TCP calculates the precise cwnd size derived from the present measurement of queuing-delay whenever reliable round-trip time (RTT) estimations are available, i.e., *queuing_delay = average_RTT – base_RTT.*

On the basis of average RTT and average queuing-delay the FAST TCP periodically recalculates its cwnd according to (1) as described in [Jin et al., 2003].

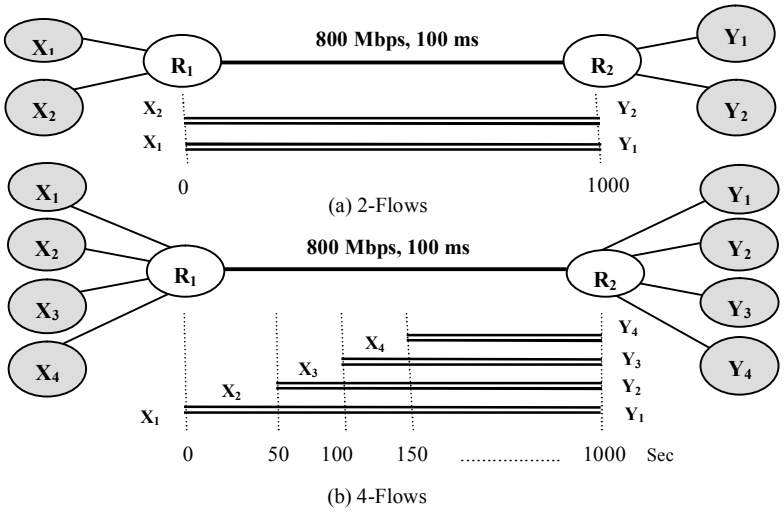

Fig.1 Network topology used in the simulations with active periods of the flows

$$w\_new \leftarrow \min\left\{2w, (1-\gamma)w\_current + \gamma\left(\left(\frac{baseRTT}{RTT}\right)w\_old + \alpha(w\_current, qdelay)\right)\right\} \quad (1)$$

In the next subsections, we present the arrangement of our investigational comparison of the protocol's performance or behavior by means of queuing-delay, packet loss and application throughput for the period of data file transfers.

2.1 Experimental setup

In this section, we briefly describe the experiments carried out to compare the performance of FAST TCP and SCTP protocols in single-homed high-speed networks. We used *ns-2* network simulator as the basis for our protocols comparison and performance evaluation. We used FAST TCP's *ns-2* module [Cui et al.], ver_1.1 (SACK introduced) and for SCTP, we used the University of Delaware's module [Caro et al.]. We conducted two set of simulations to compare the protocol's performance based on the network topology: i) with the two (flows) sender and receiver pairs ($X_1 \leftrightarrow Y_1$, $X_2 \leftrightarrow Y_2$) shown in Fig. 1(a) and, ii) with four (flows) sender and receiver pairs ($X_1 \leftrightarrow Y_1$, $X_2 \leftrightarrow Y_2$, $X_3 \leftrightarrow Y_3$, $X_4 \leftrightarrow Y_4$) shown in Fig. 1(b).

To see the difference between FAST TCP and SCTP, we simulated same link with different number of flows having the bottleneck link capacity of 800Mbps with drop-tail queuing, and the buffer size of 3000 packets with a fixed packet length of 1500 bytes. A router monitor's module recorded the queue size every 0.2 second and packet loss was set to 0%. We ran each set of simulations for 1000 seconds and data transfer was done using FTP. For FAST TCP, in all of our experiments the parameter value alpha (α) was set to 200 packets for each flow.

2.2 Results and discussions

We present a performance comparison of both the protocols through simulation results and discuss the protocols behavior in this section. In the first set of simulations, there were two flows (sources) sharing a router with a common propagation delay of 100ms, which initiated and ended at the same times as shown in Fig. 1(a). In the second set of simulations, there were four flows having the equal propagation delay of 100ms, they

entered and left the network at different times as depicted in Fig. 1(b).

We ran each set of simulations under each of two protocols (SCTP and FAST TCP) and presented the aggregate throughput, the queue size, cwnd and the total number of packets dropped at the bottleneck link. Fig. 2 and Fig. 3 show the simulation results for SCTP and FAST TCP, respectively, when two flows were used. Similarly, the simulation results in Fig. 4 and Fig. 5 show the aggregate throughput, the queue size, cwnd and the total number of packets dropped at the bottleneck link for SCTP and FAST TCP, respectively, when four flows were used.

SCTP's trajectories in Fig. 2 (2-flows) show that for the duration of slow-start phase, there is no in advance information of the existing link-bandwidth which can be exercised to end the exponential increase of the SCTP's windows. Thus, we observe that SCTP's sources grow their cwnds until the available bandwidth is exceeded and they use progressively more router buffers until they start losses packets by overflowing the bottleneck queue, since all these losses are only caused by network congestion at the routers; we don't adjust the loss-rate in our simulations (i.e., not a single packet loss is observed by reason of bit errors).

We also observe that, as more number of SCTP competing sources join the network, stability becomes worse for this loss-based protocol that produce more oscillations in its congestion windows and queue size, and increase packet loss in the network as shown in Fig. 4.

On the other hand under similar conditions, FAST TCP consistently does better than SCTP in terms of throughput, stability with zero packet loss at the bottleneck, because each source aims to keep the equal number of packets inside the queue in equilibrium so that each competing source equally shares the bottleneck link bandwidth as shown in Fig. 3 and Fig. 5. These figures show clearly that FAST achieves a better aggregate throughput (seeing as they can maintain the network link around full utilization).

As a comparison, SCTP's flows (Fig. 2 and Fig. 4) deliberately produce packet losses since they swing between under utilization and full utilization. This fact can be explained by the following description in Fig. 4 (4-flows), as the first SCTP flow $X_1 \leftrightarrow Y_1$ was started at time zero, for the duration of the initial slow-start, there was no in advance information of the existing link bandwidth, so this exponential increase of the window stop too early (when the network is far from congestion) and of course it will spend a much time by following the linearly increase to reach at some finest cwnd size (if there was only one flow), but at time 50 and 100 seconds the two more SCTP flows $X_2 \leftrightarrow Y_2$ and $X_3 \leftrightarrow Y_3$ joined the network and started their windows increasing too, as a consequence, increase the sending rates blindly (at what time the network is getting ready to congestion) and the cwnds will frequently grow until the available bandwidth is exceeded.

As the windows size increase we wait for the aggregate throughput to raise as well however the aggregate throughput cannot move up further than the offered link-bandwidth, this is because, any development into the cwnd size further than this point simply causes the segments occupying the space of router buffer at the bottleneck. And hence it then started packets drop at the network when the queue size exceeded the available router buffer capacity as shown in Fig. 4 in the region between 0 and 102.8 seconds. Similar, cwnds reductions are observed at time 200 seconds (when last SCTP flow $X_4 \leftrightarrow Y_4$ at time 150 seconds joined the network) and later at time 800 seconds.

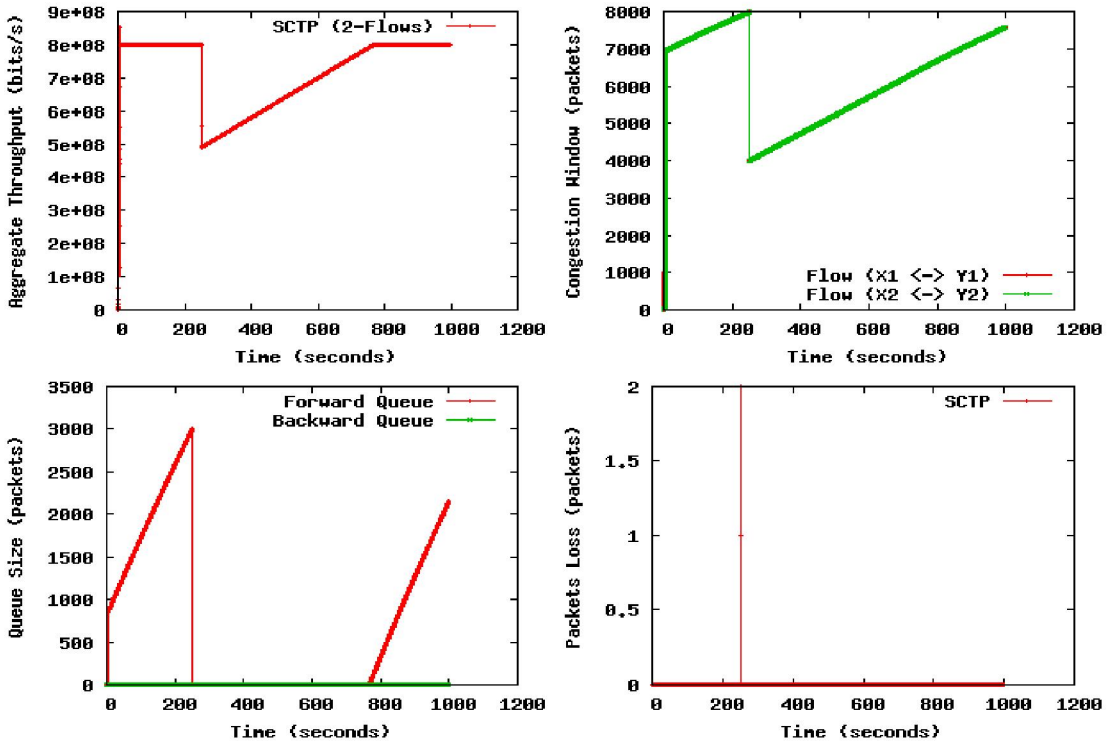

Fig. 2 SCTP's trajectories with 2-flows

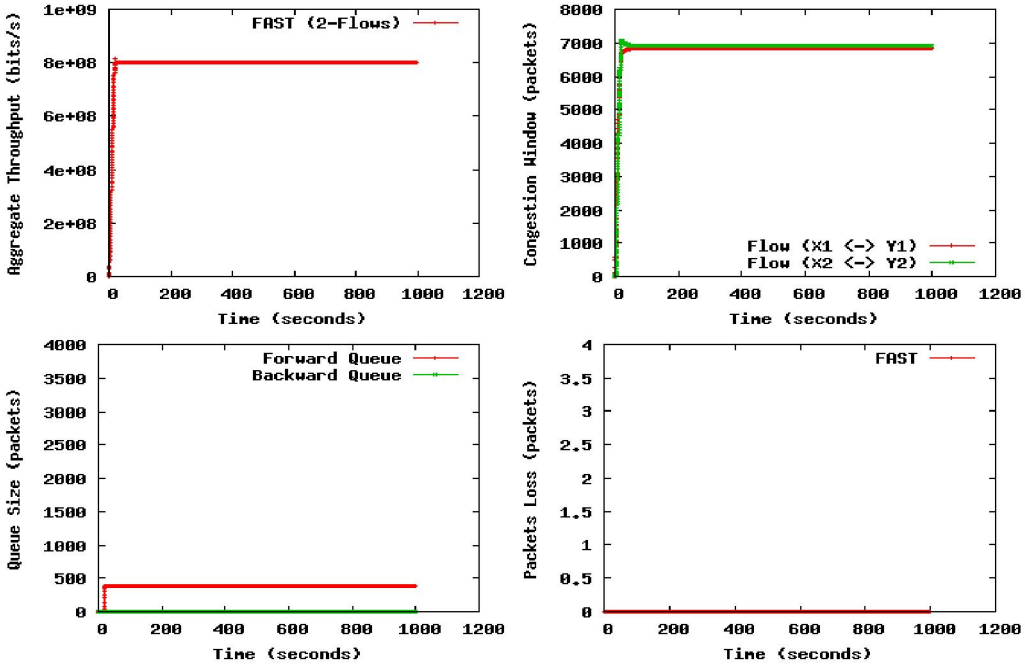

Fig. 3 FAST TCP's trajectories with 2-flows

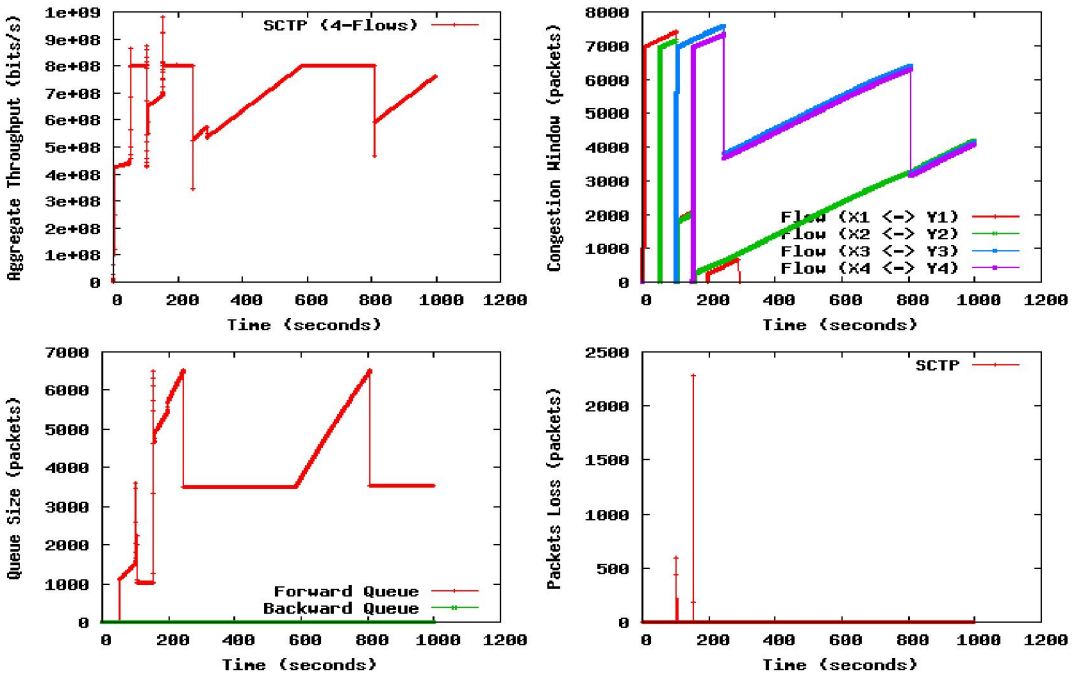

Fig. 4 SCTP's trajectories with 4-flows

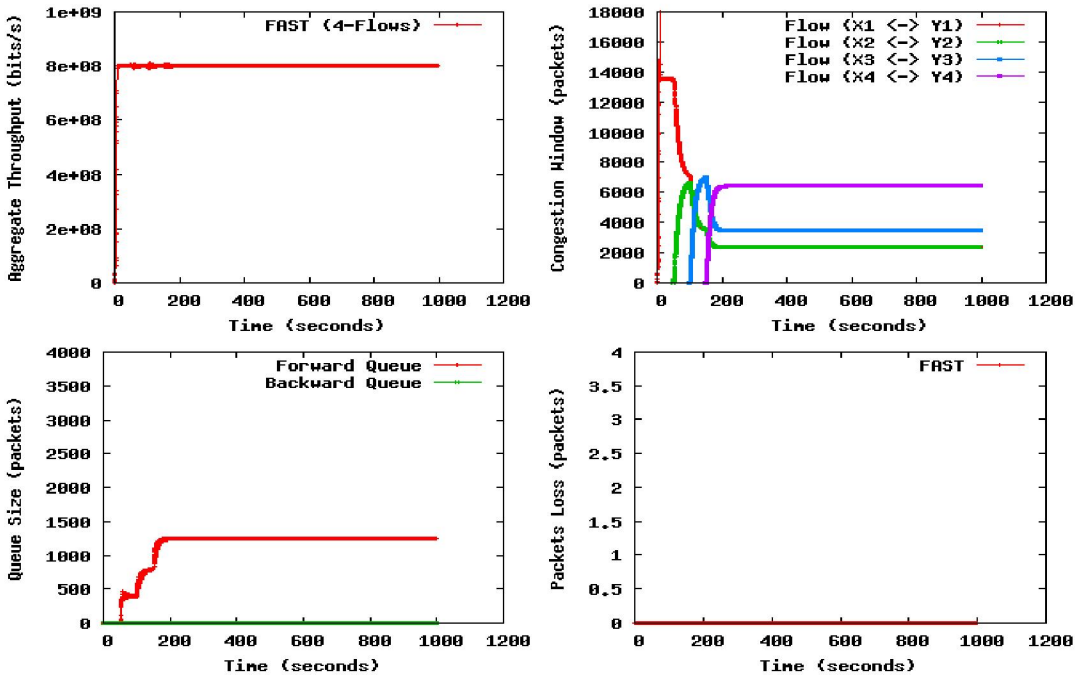

Fig. 5 FAST TCP's trajectories with 4-flows

3. Receive buffer blocking in SCTP-CMT multihoming

3.1 Problem overview

SCTP is relatively new transport layer protocol that natively supports multihoming. It is an IETF standards-track protocol, which hasn't yet been largely deployed in the Internet regardless of its several advantages over standard Internet Protocols (UDP and TCP); however, the research on extending SCTP to CMT using multihoming is currently in progress [Iyengar et al., 2004].

In SCTP-CMT, the multihomed receiver keeps a single receive buffer which's shared across all the paths (sub-flows), and it consumes data purely in order, regardless of the destination addresses they're directed to. The transmission rate of an SCTP sender is bounded by the peer-receiver window together with the relevant destination's congestion window.

It has been shown in [Iyengar et al., 2005] that when more than one paths are employed for concurrent multipath data transfer : (i) the path having a poor quality or greater packet-loss rate puts-down the whole throughput of a receive buffer constrained CMT association by blocking the receive buffer or peer-rwnd and, (ii) it also degrades performance increasingly with increasing difference into end to end delay combinations on all the paths utilized during this association and, (iii) in the environment of very small end-to-end delay, it also increases the receive buffer blocking problem in SCTP-CMT.

This receive buffer blocking causes considerable throughput deficiency if data is transferred through multiple paths simultaneously. Moreover, larger the difference between the paths (due to delays and/or loss-rates differences) also increases the rbuf blocking in SCTP-CMT.

3.2 Impact of receive buffer blocking on CMT due to network congestion-based losses

In this section, we study the impact of network congestion-based losses on rbuf-blocking in CMT. During the concurrent multipath transfer of data when a path undergoes failure (due to congestive losses or non-congestive losses), its outstanding data has to be recovered by means of a retransmission timeout (RTO), which in turn causes receive buffer blocking for the period of the timeout, thus, the possibilities of receive buffer blocking are greater during periods of missing packet's recovery through retransmissions. Since each timeout results in the reduction of congestion window at the sender, and causes idle time (that is sender not sending data), that ultimately resulting in throughput degradation.

Although, several retransmissions policies [Iyengar et al., 2004; Iyengar et al., 2005] are suggested to reduce the rbuf-blocking problem in SCTP-CMT at transport layer, but rbuf blocking problem cannot be eliminated. We also demonstrate this problem in the next Section 3.2.1, through network simulations and analysis the performance of SCTP-CMT during the occurrence of a bounded receive buffer. We then identify that reducing (or eliminating) the number of packet losses will reduce the rbuf blocking problem in SCTP-CMT, *but in the real Internet it is not possible for the SCTP-CMT to avoid from these losses (mostly due to congestion) due to its loss-based congestion detection mechanisms.* We then will identify and come to the conclusion that rbuf blocking problem in SCTP-CMT multihoming is mostly due to its loss-based nature for detecting congestion.

As the loss of packet can be occurred due to several reasons including over-saturated networks links, poor-quality of the signal at the network, defective networking hardware and damaged packet

discarded during transfer (etc). But the most common reason for packet loss is the network congestion and this congestion is sensed by the loss-based protocols through the packet loss indication. It means the network congestion is only sensed when packet is actually lost by the loss-based protocols (like SCTP), as we have shown in the single-path scenario (Fig. 2 – Fig. 5) and we also observed that such kinds of problems are far away from FAST TCP, because it uses queuing-delay rather than packet-loss probability as a metric for detecting the network congestion.

Also the Fig. 2 – Fig. 5 clearly show that the congestion avoidance techniques of FAST TCP at work, moreover, how its throughput adjusts to the varying environments on the network. FAST TCP policy is to regulate the sender's transmission rate in an effort to maintain a little amount of data packets in the routers buffers alongside the network pathway so that it does not go beyond the bandwidth-delay product (BDP) of the connection plus the no. of buffers on the bottlenecks. Such approach provides FASTTCP the capability to expect network congestion and stabilize its sending rate consequently in such manner that there are *little or zero packet losses*. We believe that if packets are sent through multiple paths (having different traffic-load distribution) simultaneously to destination using end-to-end multihoming, the packets are highly likely to arrive in the order they were initially sent (preventing rbuf from blocking), and this belief is based on the experiments reported in this paper for comparing the protocols and also in [Junaid et al., 2008];.

Therefore, we argue that– under CMT (which uses two congested paths) FAST TCP will perform much better than SCTP in high-speed multihomed networks under the same finite receive-buffer size due to its delay-based congestion control mechanisms. To end with, we motivate delay-based approach (i.e., FAST TCP) as a congestion control mechanism used for implementing the end-to-end transport layer multihoming for parallel data transfer (in high-speed long-distance networks) rather than other loss-based congestion control protocols.

4. Conclusions and future work

In this research paper, we have studied the congestion control mechanisms of FAST TCP and SCTP protocols. We have conducted simple simulations to evaluate the performance of delay-based (FAST TCP) versus loss-based (SCTP) on high-speed networks (ns-2), and through these simulations we have shown that FAST TCP usually has the great performance under a similar network conditions.

In this study, we demonstrated the weakness of SCTP-CMT rbuf constraints and, we then exposed that rbuf blocking problem in SCTP-CMT multihoming was mostly due to its loss-based nature for detecting network congestion. Space restrictions naturally limit the number of results that we can show; however, the experimental results and survey presented in this research provide insight on design decisions for the future high-speed multihomed transport protocols.

In our forthcoming article [Junaid et al., 2010]; a number of issues will be discussed in attempting to develop such a transport layer protocol based on FAST TCP, which can transfer data parallel through multiple paths using end-to-end multihoming. The problem areas in this design and a brief introduction to each of the problems that are discussed in this research along with different alternatives will also be addressed. Moreover, the complex network and more experiments will be simulated to prove the practicability of this policy.

Acknowledgements

This work was supported by the Directorate of Research Extension and Advisory Services, University of Engineering and Technology (UET), Lahore-Pakistan.

References

Sally Floyd, "HighSpeed TCP for large congestion windows". Internet draft draft-floyd-tcp-highspeed-02.txt, work in progress, http://www.icir.org/floyd/hstcp.html, February 2003.

Tom Kelly, "Scalable TCP: Improving performance in highspeed wide area networks," Submitted for publication, http://www-lce.eng.cam.ac.uk/~ctk21/scalable/, Dec 2002.

M. Ohta, "The Architecture of End to End Multihoming," Internet-draft, IETF (Nov 2002), draft-ohta-e2e-multihoming-03.txt.

V. Jacobson. Congestion avoidance and control. ACM Computer Communication Review, 18:314–329, August 1988.

D.X. Wei, C. Jin, S.H. Low, and S. Hegde, "FAST TCP: motivation, architecture, algorithms, performance", to appear in IEEE/ACM Transactions on Networking, 2007.

C. Jin, D. Wei, and S. H. Low, FAST TCP: motivation, architecture, algorithms, performance, Tech. Rep. CaltechCSTR: 2003.010, Caltech, Pasadena CA, 2003, http://netlab.caltech.edu/FAST.

H. Bullot and L. Cottrell, "Tcp stacks testbed," www-iepm.slac.stanford.edu/bw/tcp-eval/.

J. R. Iyengar, K. C. Shah, P. D. Amer, and R. Stewart, "Concurrent multipath transfer using SCTP multihoming," in Proc. SPECTS, San Jose, CA, July 2004.

M. Junaid Arshad and M. Saleem Mian, Issues of Multihoming Implementation Using FAST TCP, International Journal of Computer Science and Network Security, VOL. 8, NO. 9, pp. 104-114, September 2008.

M. Allman, V. Paxson, and W.Stevens,"TCP Congestion Control," RFC2581, IETF, April 1999.

R. Stewart et al. Stream control transmission protocol. IETF RFC 2960, Oct. 2000.

VINT Project, Network Simulator ns-2, http://www.isi.edu/nsnam/ns/.

A. Caro and J. Iyengar, "ns-2 SCTP module," http://pel.cis.udel.edu.

T. Cui and L. Andrew, "FAST TCP simulator module for ns-2, version 1.1", http://www.cubinlab.ee.mu.oz.au/ns2fasttcp.

J. Iyengar, P. Amer, and R. Stewart. Receive Buffer Blocking In Concurrent Multipath Transport. In IEEE GLOBECOM, St. Louis, Missouri, November 2005.

J. Iyengar, P. Amer, and R. Stewart. Retransmission Policies For Concurrent Multipath Transfer Using SCTP Multihoming. In ICON 2004, Singapore, November 2004.

T. DeFanti, C. d. Laat, J. Mambretti, K. Neggers, and B. Arnaud, "TransLight: A global-scale LambdaGrid for e-science." Communications of the ACM, 47(11), November 2003.

Mohammad Junaid Arshad and Mohammad Saleem, A Simulation-Based Study of FAST TCP Compared to SCTP: Towards Multihoming Implementation Using FAST TCP, Journal of Communications and Networks, 2010 (Under-review).