Analysis of FAST TCP for Multiple-Streams Implementation

S. A. Awan[1], M. J. Arshad[2], S. S. Muhammad[3]

[1,2,3]Computer Science and Engineering Department, UET Lahore, Pakistan
[1]sarfaraz.awan@hotmail.com

Abstract-From congestion control point of view, TCP variants are divided in two main categories, one using loss based congestion control algorithm and second, using delay based congestion control algorithm. FAST TCP is one of the TCP variants which uses delay based congestion control algorithm and it is suitable for large distance and high bandwidth delay product networks. TCP and all its variants use single byte-stream within its connection which don't adequately support request concurrency of applications that become a reason of head of line blocking. So, single streamed transport protocols have overall negative effect on application performance. In this paper the use of multistream mechanism in a single connection of FAST TCP is proposed, which provide two benefits. First is to reduce data delivery latency between transport and application layer. Second is to reduce head of line blocking. All possible issues are identified which could be faced while incorporating multistream mechanism into FAST TCP and solution of those issues are analyzed.

Keywords-Multistreaming, Head of Line Blocking, FAST TCP, Loss Based Congestion Control (LBCC), Delay Based Congestion Control (DBCC)

I. INTRODUCTION

Transmission Control Protocol (TCP) [i] and Stream Control Transmission Protocol (SCTP) [ii] are reliable transport protocols. These protocols ensure ordered and error free transfer of user data across the network with appropriate congestion control.

SCTP was introduced in 2000 [ii]. The latest specification of SCTP were published in 2007 [iii]. It has several distinct features such as partial reliable data transfer, multihoming, multistreaming and dynamic address reconfiguration.

TCP provides reliable, in-order, error free and flow control services for data transmission between the applications running on end nodes with appropriate mechanism to avoid network congestion. It is one of the important components of internet protocol suit [iv]. It faces some problems such as HOL blocking and denial of service due to which its performance is suffered [v-viii].

End to end congestion control algorithms used by transport layer protocols are divided in two categories i.e. loss-based and delay-based congestion control algorithm. Congestion control technique was proposed in 1988 [ix] and afterward a number of improvements were made to refine this technique [x-xv]. Congestion Control techniques had improved the Internet performance as it grew up in size, speed, connectivity and load. Performance of Transport protocols which use loss based congestion control technique suffers on high bandwidth delay product network paths [xvi-xx].

Transport protocols using delay based congestion control technique performs better on large bandwidth delay product network compare to the protocols that use delay based congestion control technique. TCP-Vegas [xxi], FAST TCP [xxii, xxiii] and TCP-LP [xxiv] use delay based congestion control techniques. TCP [I], SCTP [iii], TCP New Reno [xxv], Scalable TCP [xvi], BIC TCP [xix], High-Speed TCP [xvii], TCP-XM [xxvi] and CUBIC [xxvii] use loss based congestion control technique.

Applications need an efficient transport layer protocol for reliable end to end communication. Some applications which are running on one end host have to transmit number of independent messages/objects across the network in response of the request generated from the other end host. TCP serialized these messages over its single bytestream. It delivers data to the application in an order. So, if any transport protocol data unit (TPDU) is misplaced during data transmission then subsequent TPDUs received are buffered and not forwarded to the application layer until the misplaced TPDU is received. Lost TPDU of independent application message suspends the data delivery from the rest of independent messages to the application. It is called head of line (HOL) blocking. Head-of-line blocking is faced by all those transport layer protocols which do not logically separate application's independent message [xxviii]. Multiple streams at transport layer is one of the solutions to mitigate HOL blocking issue. When a separate stream is used for transmission of an independent application message then that message is processed and displayed separately from the rest of the messages. Stream within a connection is a logical separation for independent user message and have its own sequence space. Data order in each stream is maintained separately regardless of data order in other streams. Data from each stream is handed over to the application according

to its own data order. So multistream is a most appropriate technique for the transmission of independent application messages [xxviii, xxix].

In this paper a working of FAST TCP (a transport layer protocol), its advantages and limitations are described. All possible issues are identified which could be faced while incorporating multistream mechanism into FAST TCP and solution of those issues are analyzed.

This paper is organized as follows. In section II we describe the published work related to HOL elimination techniques. In section III FAST TCP overview is given. In section IV HOL blocking issue in the context of FAST TCP is discussed. In section V, we discuss the issues which must be addressed during design and implementation of multistream feature using FAST TCP. In section VI, conclusion is presented.

II. Related Work

In this section techniques proposed to mitigate HOL blocking problem are described.

Preethi et al. [xxviii] discussed unique features of SCTP i.e. multihoming, mulstistreaming and four way handshake. Preethi stated that by these features, SCTP has overcome three limitations of TCP i.e. HOL problem, network failure and SYN attacks. In this paper authors described different areas where multistream feature could be beneficial.

Muhammad Junaid and Muhammad Saleem [xxx] in their paper made a comparative study between SCTP and FAST TCP (Loss based vs. delay based approach) in high speed networks and through simulation results showed that performance benefit of FAST TCP in the form of throughput and stability over SCTP is small at low-speed, but dominant at high-speed and large bandwidth networks.

Preethi et al. in their work discussed that using several TCP connections in parallel to avoid HOL blocking problem affects HTTP throughput. Several TCP connections raise fairness issue. There is an overhead associated with each TCP connection i.e. it must be established, maintained and closed separately. Moreover, each TCP connection has to recover from packet loss independently. Preethi showed that there is a negative correlation between HTTP throughput and number of parallel TCP connections [xxxi].

So, the use of several TCP connections for application's independent messages is not a comprehensive solution of head of line blocking issue.

Seung et al. proposed a transport protocol Dynamic Multi-stream TCP (DMS-TCP) [xxxii] to support scalability over a high speed and large bandwidth network. Authors used multiple streams within a single connection keeping in view scalability, friendliness and fairness attributes. Protocol manages the streams according to the available bandwidth. The main idea of the protocol was that throughput can be increased by increasing the number of streams. Proposed concept was simulated by using ns-2 simulator.

Authors used multiple streams and showed that it increases throughput.

III. FAST TCP

Congestion control algorithm used by FAST TCP is very useful for high speed and long latency network. This protocol consumes large bandwidth in long distance network and as a result provides high throughput [xx, xxii]. FAST TCP has four components of its congestion control algorithm; (1) data control (2) window control (3) burstiness control (4) estimation component. Data control component takes the decision about which packets to be transmitted, window control component decides about the number of packets to be transmitted and burstiness control component decides at what time these packets will be transmitted. All of the decisions depend on the provided information of estimation component [xxiii].

1) Estimation Component

Each data packet transmitted over the network gives two feedback information (1) loss or no loss indication (2) queuing delay. Round Trip Time (RTT) is calculated upon receipt of positive acknowledgement. This RTT is utilized to compute average round trip time and minimum round trip time. Window control uses average RTT and minimum RTT to calculate window size. Data control component uses loss indication on receipt of negative acknowledgement [xxiii].

2) Data Control

It selects data to send from three types of packets: unsent packets, packets lost during transmission, and unacknowledged packets. New packets are dispatched when no data is lost and acknowledgement of old packets is received. This is termed as self-clocking. In case of recovery of lost data, one of the three actions may be performed: resend packets which are lost, send new packets or resend older packets which are not acknowledged or lost. Data control also decides to mix packets from these data pools [xxiii].

3) Burstiness Control

This component tracks available bandwidth by smooth transmission of packets. It is important for networks having huge bandwidth delay products to measure available bandwidth. The loss rate and long queues may be generated by tremendous burstiness. FAST TCP uses two approaches of burstiness control: window pacing and burstiness reduction. How many packets to be transmitted and to reduce the burst size when timescale becomes lesser than one round trip time. Congestion window is expanded to the target by

window pacing. Thus burstiness control follows an appropriate scheduling overhead to reduces burstiness [xxiii].

4) Window Control

FAST TCP uses queuing delay to measure congestion and window adjustment. Queuing delay as a congestion indicator has two benefits i.e. (1) it is an excellent source to predict congestion (2) it has multi-bit information. Queuing delay manages the flow of data as per the capacity of the network link, which helps to maintain stability as network capacity boosts. FAST TCP changes congestion window as per the average RTT value [xxiii]. FAST TCP alters its congestion window according to the space available in a buffer [xxiii].

A. Advantages of FAST TCP

FAST TCP has advantages over other loss based transport protocols such as SCTP and TCP. It detects congestion using queuing delay and packet loss [xx] and in case of large delay and high bandwidth environment, it removes the limitation of TCP. FAST TCP's congestion control technique maintains window size and avoids fluctuation.

Congestion detection in delay based congestion control algorithm is better than loss based congestion control algorithm because delay means buffers filling is started whereas loss means buffers are completely filled. So, delay based congestion control protocols can overcome the limitations of loss based congestion control protocols.

FAST TCP shows better stability and throughput at the bottleneck by foreseeing the network congestion at early stage by calculating the difference of expected data transfer rate and actual data transfer rate. In reaction of this network congestion prediction FAST TCP readjusts its data transmission rate so that packet losses could be minimized or eliminated. This change in data transmission rate decreases the number of packets in router buffer.

By considering the advantage of FAST TCP, it is selected for incorporating multistream feature for simultaneous transferring of independent application messages over a high speed and long distance network

B. Limitations of FAST TCP

According to some studies there is a minor correlation between increased delay and network congestion losses [xxxiii-xxxv]. Delay based congestion control algorithm as implemented in FAST TCP assumes that RTT is a best indicator to foresee congestion and to avoid it. These studies [xxxiii-xxxv] indicate that congestion control on delay based technique does not guarantee congestion free transmission. It is found that with the considerable increase in RTT, 7-18% loss events occur [xxxiv]. This congestion due to increase in RTT may become the reason of HOL Blocking which may block transmission of independent application messages over single stream FAST TCP.

C. FAST TCP vs. TCP

Differences between FACT TCP and TCP are described in the following three ways.

- FAST TCP decreases packet level fluctuations.
- FAST TCP senses the network congestion by calculating the queuing delay. Queuing delay is more reliable than loss probability in high speed and long distance networks.
- FAST TCP has stable flow control and obtains proportional fairness in equilibrium which does not affect long flows [xxiii].

IV. HOL Blocking and FAST TCP

Web applications usually have to transfer independent messages between end nodes over the network. A reliable transport protocol is required by these applications for data transmission over the internet. Most of the applications use TCP at transport layer for its data flow. TCP provides a sequential flow over a single bytestream for transferring its data. TCP serialized the independent messages of the application and place on the stream serially.

In case, a TPDU of any independent message is lost then the receiver will not hand over the received TPDUs of other independent messages to the upper layer until the lost TPDU is not received.

It is called Head-of-Line (HOL) blocking problem which happens because no logical segregation is done by TCP between independent application objects during transmission of data.

HOL blocking occurs by unnecessary filling of transport layer buffer on receiver side. TCP places those TPDUs in receiver buffer whose sequence of arrival is not in order. As the data in receiver buffer becomes in order on receipt of missing TPDU, the transport layer handovers this in order data to the application layer. Buffer filling is not required for those independent messages whose TPDUs are not lost. Buffer size is directly proportional with the loss probability and the number of application's independent message to be sent. High loss rates and low bandwidth raise HOL blocking [xxviii].

More than one TCP connections are used by the browsers to remove HOL blocking problem. Independent object requests are dispersed among these TCP connections. HOL blocking is handled up to certain level by using multiple TCP connections but it could not be removed altogether [xxxvi].

SCTP has a feature of multistreaming which is helpful to remove HOL blocking. The Loss-based congestion control algorithm of SCTP affects its performance in the network of large Bandwidth-Delay Product. Congestion control with delay based technique resolves the problems of loss based

congestion control in large Bandwidth-Delay Product networks.

FAST TCP uses delay based congestion control algorithm but it uses single byte stream to transmit data across the network.

By incorporating multistream feature, FAST TCP can overcome the issue of HOL blocking while transferring independent application objects. Implementing multistreaming using FAST TCP may raise several questions to be addressed.

V. Issues of Implementing Multistreaming Using FAST TCP

Few points are being highlighted which must be addressed during the implementation of multistreaming feature using FAST TCP.

- How FAST TCP exchanges the information about the number of streams to be opened at connection setup time?
- How the data order can be maintained in each stream?
- How multistream feature in FAST TCP improves cumulated throughput?
- Should a separate or shared congestion control be used for each stream?
- How the traffic scheduling should be managed on multiple stream?
- Flow control separate or shared for each stream?
- How the stream priority given at application level be enforced?
- How the receiver window (rwnd) for multiple streams will be used at sender side?
- Will the buffer space for each stream be separate or shared?
- How the sequence space will be used for all the streams within a connection?

A. Addressing main issues of Implementing Multistreaming Using FAST TCP

We present useful suggestions to resolve main issues of implementing multistreaming using FAST TCP.

1) Connection Setup

Connection establishment is the first step for data communication in FAST TCP. Connection is established by a three-way handshake [i]. One node of FAST TCP initiates this procedure and other FAST TCP node responds. Multistreaming is not the feature of current FAST TCP implementation that's why the stream's information is not exchanged between end nodes during establishing a connection.

The information about the required number of stream for the transmission of application's independent objects should be exchange between end nodes.

This information can be exchanged by two ways (1) at the time of connection setup similar to SCTP [ii] (2) streams are opened after connection setup similar to Structured Stream Transport (SST) [xxxvii].

That required number of streams will be negotiated between end hosts at connection setup time and afterward more streams could be opened or existing streams could be closed.

It is being proposed that FAST TCP will negotiate the number of streams required to be opened between end hosts at connection setup time and afterward more streams could be opened or existing streams could be closed. There will be a bidirectional data flow in these streams within a single connection.

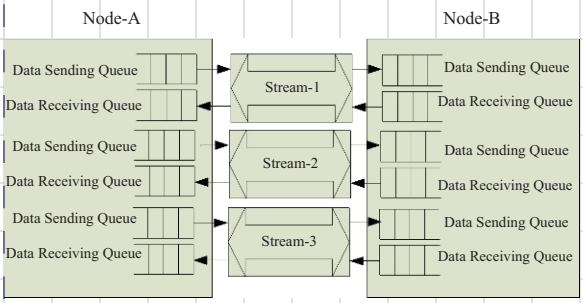

Fig.1. A single connection with multiple streams

In Fig. 1 a single connection having three streams with bidirectional data flow among two end nodes is shown.

2) Preserve Data Order within each Stream

Sequence Number is given to all data segments for in-order data delivery before transmitting it on a single bytestream of FAST TCP. But this sequence number is not adequate to keep in sequence data delivery within multiple streams.

To maintain data sequence in each stream, Stream Sequence Number (SSN) will be added in FAST TCP header. It will be required to support multistream feature in FAST TCP.

SSN will make sure in sequence delivery of each independent object's packet to application layer without considering the sequence of delivery in remaining streams. A separate stream will be opened for each independent object of an application. Separate stream number and separate stream sequence number will be maintained for this stream in order to maintain in sequence delivery of TPDUs of independent application object.

Each independent object of an application is assigned a separate stream having a unique stream number and SSN within that stream make sure in-order data delivery.

However, the blockage in any stream due to the loss of TPDU will not stop the data flow in remaining streams.

3) Increase Aggregate Throughput

Multiple streams in FAST TCP will remove the HOL Blocking problem during congestion, as a result throughput will be increased.

4) Congestion Control

FAST TCP has a single stream for data transfer across the network. Its delay based congestion control mechanism operates at packet level and flow level [xx].

Congestion control algorithm of FAST TCP has four parts (1) data control (2) window control (3) burstiness control (4) estimation components. Selection of packets for transmission is made by data control, window control decides about the number of packets to send and burstiness control decides when these packets will be sent. These three parts relies on the estimation component information for its decisions [xxiii].

When estimation component receives acknowledgement of sent packet, it calculates two things one weather packet is lost or not second queuing delay. In FAST TCP, the acknowledgement of data packet is used to compute RTT and this RTT is used to compute average RTT and minimum RTT. Window control uses the values of minimum RTT and average RTT for data flow control [xxiii].

Queuing delay is a difference of observed RTT and base RTT. It serves as a congestion measure in FAST TCP and used to maintain the number of packets in flight within a network. Queuing delay is inversely proportional to the data sending rate.

There are three types of data on sender side (1) lost packets (2) new packets (3) unacknowledged packets. Data control takes from these three types of data. [xxiii].

Congestion control mechanism of FAST TCP operates on single data stream. But multistreamed FAST TCP will use a shared congestion control technique at packet and flow level. On loss feedback data control part will not select data from the lost packet's stream. If the queuing delay is increased then lesser number of packets from each stream will be selected for transmission.

5) Scheduling of Traffic on Multiple Streams

FAST TCP with multistreaming will face an issue of data selection from multiple streams. Multistreaming is a feature of SCTP but the recent standard [iii] doesnot illustrate the procedure of data selection from different streams. The errata to the standard [xxxviii] propose round robin technique for data selection from multiple streams which have pending data. Different choice is made by different implementations e.g. FreeBSD selected round robin selection technique and Solars selected first-come-first-serve selection technique from the streams in its stack. Each implementation has its own choice but they share “one size fits all” philosophy. This approach confines the advantages of multistreaming.

The problem can be solved if multistream FAST TCP could provide different data selection technique in its implementation. Application will select specific data selection technique by providing information to the transport protocol.

VI. Conclusion and Future Work

FAST TCP is a delay-based transport layer protocol which achieves high utilization of available bandwidth without filling the intermediate buffers. It becomes the reason of queueing delay, as loss based algorithms do. This study highlighted the limitation faced by TCP and all its variants including FAST TCP. The single byte-stream doesn't adequately support request concurrency of applications which become a reason of head of line blocking. The use of multistream mechanism in a single connection of FAST TCP at transport layer has been proposed. The use of multistream mechanism reduces the data delivery latency between transport and application layer and head of line blocking. All possible issues are identified which could be faced while incorporating multistream mechanism into FAST TCP. Also the solutions of those issues have been analyzed in this paper. This work provides guidelines to develop design level details for multistream FAST TCP.

References

[i] J. Postel, "Transmission control protocol," in RFC 793, ed, 1981.[Online]. Available: https://tools.ietf.org/html/rfc793

[ii] R. Stewart, Q. Xie, K. Morneault, C. Sharp, H. Schwarzbauer, T. Taylor, et al., "Stream Control Transmission Protocol," in RFC 2960, ed, 2000.[Online]. Available: https://tools.ietf.org/html/rfc2960

[iii] R. Stewart, "Stream Control Transmission Protocol," in RFC 4960, ed, 2007.[Online]. Available: https://tools.ietf.org/html/rfc4960

[iv] K. R. Fall and W. R. Stevens, TCP/IP illustrated, volume 1: The protocols: addison-Wesley, 2011.

[v] D. E. Comer, Internetworking with TCP/IP, 2000, 4th ed. vol. 1: Prentice Hall, 2000.

[vi] W. M. Eddy, "Defenses against TCP SYN flooding attacks," The Internet Protocol Journal, vol. 9, pp. 2-16, 2006.

[vii] C. M. Kozierok, The TCP/IP guide: a comprehensive, illustrated Internet protocols reference: No Starch Press, 2005.

[viii] S. Kang and M. Fields, "Experimental Study of the SCTP compared to TCP," Electrical Engineering Department, Texas A&M University, USA, 2003.

[ix] V. Jacobson and M. J. Karels, "Congestion avoidance and control," in SIGCOMM '88

Symposium proceedings on Communications architectures and protocols, Stanford, USA, 1988, pp. 314-329.

[x] V. Jacobson, R. Braden, D. Borman, M. Satyanarayanan, J. Kistler, L. Mummert, *et al.*, "TCP extensions for high performance," in *RFC 1323*, ed, 1992.[Online]. Available: https://www.ietf.org/rfc/rfc1323.txt

[xi] W. Stevens, "TCP Slow Start Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms," in *RFC 2001* ed, 1997.[Online]. Available: https://tools.ietf.org/html/rfc2001

[xii] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow, "TCP Selective Acknowledgment Options," in *RFC 2018*, ed, 1996.[Online]. Available: https://tools.ietf.org/html/rfc2018

[xiii] J. C. Hoe, "Improving the start-up behavior of a congestion control scheme for TCP," in *Proc. on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM'96)*, NY, USA, 1996, pp. 270-280.

[xiv] M. Allman, V. Paxson, and W. Stevens, "TCP congestion control," in *RFC 2581*, ed, 1999.

[xv] S. Floyd and T. Henderson, "The NewReno Modification to TCP's Fast Recovery Algorithm," in *RFC 2582*, ed, 1999.[Online]. Available: https://tools.ietf.org/html/rfc2582

[xvi] T. Kelly, "Scalable TCP: Improving performance in highspeed wide area networks," *ACM SIGCOMM Computer Communication Review,* vol. 33, pp. 83-91, 2003.

[xvii] S. Floyd. HighSpeed TCP for large congestion windows [Online]. Available: http://www.icir.org/floyd/hstcp.html

[xviii] R. Shorten and D. Leith, "H-TCP Protocol for High-Speed Long Distance Networks," in *Proc. 2nd Workshop on Protocols for Fast Long Distance Networks. Argonne, Canada*, 2004.

[xix] L. Xu, K. Harfoush, and I. Rhee, "Binary increase congestion control (BIC) for fast long-distance networks," in *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, 2004, pp. 2514-2524.

[xx] D. X. Wei, C. Jin, S. H. Low, and S. Hegde, "FAST TCP: motivation, architecture, algorithms, performance," *IEEE/ACM Transactions on Networking (ToN),* vol. 14, pp. 1246-1259, 2006.

[xxi] L. S. Brakmo, S. W. O'Malley, and L. L. Peterson, "TCP Vegas: New techniques for congestion detection and avoidance," in *SIGCOMM '94 Proceedings of the conference on Communications architectures, protocols and applications*, London, UK, 1994, pp. 24-35.

[xxii] C. Jin, D. Wei, and S. H. Low, "FAST TCP for high-speed long-distance networks," ed, 2003.

[xxiii] C. Jin, D. Wei, S. H. Low, G. Buhrmaster, J. Bunn, D. H. Choe, *et al.*, "FAST TCP: From theory to experiments," *IEEE network,* vol. 19, pp. 4-11, 2005.

[xxiv] A. Kuzmanovic and E. W. Knightly, "TCP-LP: low-priority service via end-point congestion control," *IEEE/ACM Transactions on Networking (TON),* vol. 14, pp. 739-752, 2006.

[xxv] S. Floyd and A. Gurtov, "The NewReno Modification to TCP's Fast Recovery Algorithm", RFC 3782," 2004.[Online]. Available: https://tools.ietf.org/html/rfc3782

[xxvi] K. Jeacle and J. Crowcroft, "Tcp-xm: Unicast-enabled reliable multicast," in *Computer Communications and Networks, 2005. ICCCN 2005. Proceedings. 14th International Conference on*, 2005, pp. 145-150.

[xxvii] S. Ha, I. Rhee, and L. Xu, "CUBIC: a new TCP-friendly high-speed TCP variant," *ACM SIGOPS Operating Systems Review,* vol. 42, pp. 64-74, 2008.

[xxviii] P. Natarajan, J. R. Iyengar, P. D. Amer, and R. Stewart, "SCTP: an innovative transport layer protocol for the web," in *Proceedings of the 15th international conference on World Wide Web (WWW '06)*, Edinburgh, Scotland, 2006, pp. 615-624.

[xxix] M. Scharf and S. Kiesel, "NXG03-5: Head-of-line Blocking in TCP and SCTP: Analysis and Measurements," in *Global Telecommunications Conference, 2006. GLOBECOM'06. IEEE*, 2006, pp. 1-5.

[xxx] M. J. Arshad and M. Saleem, "A simulation-based study of FAST TCP compared to SCTP: towards multihoming implementation using FAST TCP," *Journal of Communications and Networks,* vol. 12, pp. 275-284, 2010.

[xxxi] P. Natarajan, F. Baker, and P. Amer, "Multiple TCP Connections Improve HTTP Throughput Myth or Fact," in *IEEE IPCCC*, Arizona, USA, 2009.

[xxxii] S.-J. Seok, H.-J. Kim, K.-M. Jung, K.-H. Kim, and C.-H. Kang, "Dynamic Multi-stream Transport Protocol," in *Proc. of the 11th Asia-Pacific Symposium on Network Operations and Management: Challenges for Next Generation Network Operations and Service Management (APNOMS '08)*, Heidelberg, Berlin, 2008, pp. 287-296.

[xxxiii] J. Andren, M. Hilding, and D. Veitch, "Understanding end-to-end internet traffic dynamics," in *Global Telecommunications Conference (GLOBECOM) The Bridge to Global Integration*, Sydney, NSW, 1998, pp. 1118-1122.

[xxxiv] J. Martin, A. Nilsson, and I. Rhee, "Delay-based congestion avoidance for TCP," *IEEE/ACM Transactions on Networking (TON),* vol. 11, pp. 356-369, 2003.

[xxxv] S. Biaz and N. H. Vaidya, "Is the round-trip time correlated with the number of packets in flight?," in *Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement (IMC '03)*, NY, USA, 2003, pp. 273-278.

[xxxvi] Z. Wang and P. Cao. Persistent Connection Behavior of Popular Browsers [Online]. Available: http://pages.cs.wisc.edu/~cao/papers/persistent-connection.html

[xxxvii] B. Ford, "Structured streams: a new transport abstraction," in *Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM '07)*, Kyoto, Japan, 2007, pp. 361-372.

[xxxviii] R. Stewart, I. Arias-Rodriguez, K. Poon, A. Caro Jr, and M. Tuexen, "Stream Control Transmission Protocol (SCTP) specification errata and issues," in *RFC 4460*, ed, 2006.