Comparative Study of TCP Protocols

Dr. Junaid Arshad, Tabinda Ashraf, Noor-ul-Sabah
tabinda983@gmail.com
Department of Computer Science & Engineering, University of Engineering and Technology, Lahore

Abstract
TCP is the most extensively used protocol for trustworthy communication. New TCP variants are studied and proposed by researchers, due to its extensive need, making an attempt to boosts its behavior towards congestion to form it use the foremost on the market information measure whereas conserving a logical level of fairness towards different protocols. Evaluation and Comparison of the performance of foremost recent TCP deployed in general OS, is main objective of this paper. We fastidiously compare the TCP variants to analyze them on the basis of different parameters like, ThroughPut, Fairness (intra-, inter), RTT Fairness, Algrothims, BandWidth and LossRatio. Investigation shows that if the buffer size (No. of Packets) is small then ThroughPut will be lower. However, protocols act in a different way, where they attain different values with awfully little variations. Three TCP variants CUBIC, Compound and NewReno are fair to additional TCP transportation and deliver the identical intra fairness above wireless links.

Keywords:
HS-TCP, TCP Hybla,H-TCP, RTFC, BIC, TCP Real, CUBIC, Jersey, Saleable TCP, Agile-SD, TCP Africa, Zeta, TCP Fusion, TCP Compound, TCP Tahoe, TCP Illinois, NewReno, YeAH, Throughput, Loss Ratio, Fairness Index

1. Introduction

Most Internet Applications use Transmission Control Protocol (TCP). It complements Internal Protocol (IP) complete set is called TCP/IP. TCP does not rely on underlying network's feedback. So it is capable of most reliable and stable delivery of data packets. TCP depends only on sender and receiver ends. Due to this, it is also known as host-to-host or end-to-end protocol. Most Internet applications like email, file transfer, remote administration and World Wide Web uses TCP.

TCP's first idea was given in 1947 by Khan and Cerf. After that, TCP is being used in many Operating Systems and it is also being tested in real scenarios. Network technology advancement has given rise to many problems and new scenarios i.e., underutilization of bandwidth, network congestion, avoidable retransmission, non-congestion loss, out-of-order delivery and unfair share. Because of this, TCP behavior was revised by the scholars and many TCP variants were introduced. Every variant try to resolve some problems, some continue over congested and low connections and some make use of complete bandwidth by attaining advanced throughput and some attempt to be reasonable [1].

All variants are mostly different from one another so they are classified into different categories i.e., wireless, high-speed, low priority and satellite. A TCP variant that is suitable for high-BDP networks may not be suitable for wireless networks.

So, these TCP variants must be compared in terms of different parameters to know their advantages and disadvantages. So, we have chosen to examine performance of most extensively used TCP variants in current Operating Systems. In this paper we have compared different TCP variants like HS-TCP, TCP Hybla,H-TCP, RTFC, BIC, TCP Real, CUBIC, Jersey, Saleable TCP, Agile-SD, TCP Africa, Zeta, TCP Fusion, TCP Compound, TCP Tahoe, TCP Illinois, NewReno and YeAH. Among all these TCP protocols, three protocols are being used in most updated and recent operating systems i.e., Cubic TCP is being used in Linux Kernel version, Windows 2008 and Vista uses TCP Compound and Windows XP uses NewReno. This paper shows differences between them in terms of loss ratio, fairness, throughput over high-BDP networks. This paper is helpful for the investigators to increase performance of current TCP variants [2].

Manuscript received February 5, 2025
Manuscript revised February 20, 2025
https://doi.org/10.22937/IJCSNS.2025.25.2.21

This paper presents a comparative analysis of different TCP variants. In section II, we have discussed the advantages and drawbacks of TCP variants and which issue of the previous variant is solved by the new one. In section III, TCP variants are compared in terms of throughput, loss ratio, RTT-fairness and inter-fairness. Section IV concludes all our discussion.

2. Literature Review

2.1 NewReno

TCP Reno had FastRecovery problem that occurs when multiple packet loss occurs. This has reduced the performance of TCP Reno in densely crowded networks. TCP NewReno was developed in 1999 by Floyd and Henderson and was modified in 2004 by Floyd et al. and then in 2012 by Henderson. TCP NewReno overcomes the NewReno's problem. In this protocol, one can exit from FastRecovery state when complete data from the initial cwnd is being acknowledged that intellects the partial data ACKs. It separates new data ACKs from partial data ACKs. The new data ACK specifies successful delivery of information that was sent before the loss detection. The partial ACK specifies other data losses in the initial cwnd. It is not good for High Speed Networks [1].

2.2 Scalable TCP (STCP)

Scalable TCP was introduced in 2003 by Kelly. It was designed to get rid of the problems of current congestion control algorithms. These problems were raised due to bandwidth growth in high-speed networks. STCP has achieved good network utilization with higher BDP (Bandwidth Delay Product) and it has not caused any bad influence on the current traffic. STCP is just a sender side modified form of TCP congestion control algorithm. It is executed in Linux [2]. Results at that period showed that STCP would have minor effect on current network traffic and would boost the performance of transferring data in high-speed networks. α, β is being used in loss-based STCP congestion control algorithm. In this ($0 < \alpha < 1$) and ($0 < \beta < 1$). If congestion is not detected, this protocol modifies its cwnd after getting each ACK in one RTT by α. If congestion occurs, congestion window is decreased by β.

Cwnd=cwnd+ α α=0.01

Cwnd=cwnd-(β*cwnd) β=0.125

2.3 BIC-TCP

RTT-unfairness issue in STCP and HS-TCP was resolved in BIC-TCP that was introduced in 2004 by Xu et al. Let us have two TCP flows that are going to share one bottleneck and the loss is detected synchronously. If both are HS-TCP flows then the flow whose round trip time is smaller will have 4.56 times larger network share. And if both of them are STCP flows, the flow with smaller RTT will have all bandwidth of network. So, new protocol (BIC-TCP) was designed to solve this RTT-unfairness issue. It window growth function is not so good particularly for short distance or low speed networks. It depends on RTT measurements and can have poor inter-fairness. It is complicated because of many modes of algorithm i.e., max probing, binary search increase, Smin and Smax [3].

2.4 CUBIC TCP

Now-a-days, CUBIC TCP is the default TCP Algorithm. It was introduced in 2008 by Rhee and Ha. It is being used in maximum Linux OS. CUBIC TCP is the updated version of current TCP variants. To increase high-BDP network's scalability, it uses cubic function of increase in congestion window instead of the linear function. CUBIC TCP uses BIC algorithm and HTCP's cubic function of congestion window.

$$w=\mathrm{C}\left(\Delta-\sqrt[3]{\frac{\beta*w\mathrm{ma}}{C}}\right)^3+w\mathrm{max}$$

In this equation, $w_{\max}$ is the size of congestion window just before the detection of previous loss, β is multiplicative decrease function in FastRecovery and C is a predefined constant [2]. It has higher scalability and fairness both RTT and intra because of Rapid Convergence, Limited Slow Rate and RTT Independence. In preliminary stage of window increase, cubic function's right branch calculates target window $w_{\max.}$ In case the loss is detected and the loss is momentary and $w_{\max}$ is not reached yet, then congestion window is amplified conferring to

cubic function's both left and right branches. Its throughput is better than NewReno's throughput. After standard TCP, it is the second mostly implemented TCP variant in Linux OS. It also has some drawbacks i.e. available bandwidth's under-utilization and production of large number of packet losses [1].

2.5 High-speed TCP (HS-TCP)

HS-TCP was introduced in 2003. It was designed for large sized congestion windows. Standard TCP's less appropriate performance over high speed networks was resolved in this protocol. It is also based on loss congestion control algorithm. It does not cause any risk i.e., congestion collapse because behavior of standard TCP is not changed in this protocol. Congestion window is decreased or increased in this protocol by α(w) and β(w) and it is just modified form of sender-side. α(w) and β(w) values can differ from 1 and 0.5, (when cwnd=< 38 packet) to 70 and 0.1, (when cwnd>= 84k packets). HS-TCP has higher throughput in high-speed networks but its sharing fairness is not as much of standard TCP sharing fairness. HS-TCP also causes bursty packet loss problem in High Bandwidth Delay Product Networks because of standard Slow Start. HS-TCP has limited its Slow Start to 100 packets to solve this issue. This weakness of HS-TCP is known as "Limited Slow Start" [1].

2.6 Hamilton TCP (H-TCP)

H-TCP was designed at Hamilton Institute in 2004 by D. Leith. It is good for long distance and high speed networks. It is also a loss-based congestion control protocol. It is more reasonable and fair as compared to than conventional TCP. α(Δ) determines increase in the cwnd for each Round Trip Time. On arrival of every non-duplicate ACK cwnd increases by α(Δ)/w. Δ is elapsed time since last congestion signal [3]. The increased function is defined as:

$\alpha(\Delta)=1+10(\Delta-\Delta_{low})+0.5*(\Delta-\Delta_{low})^2$

Δ_{low} has a predefined value. If $\Delta < \Delta_{low}$, $\alpha(\Delta) = 1$.

H-TCP reduces its cwnd by RTT_{ratio} if $\gamma<0.2$

$RTT_{ratio} = RTT_{min}/RTT_{max}$

$$\gamma = \left| \frac{B(k) - B(k-1)}{B(k-1)} \right|$$

B (k) is estimated throughput.

B (k-1) is previous estimated loss event.

2.7 TCP Africa

Africa stands for Adaptive and Fair Rapid Increase Congestion Avoidance. This protocol was offered in 2005 by King et al. It resolves high-BDP networks issues. The combined effect of HS-TCP scalability in congestion-free case and NewReno's attributes in congestion case have succeeded to get better performance as compared to current TCP protocols. This protocol is loss-delay-based algorithm. It uses TCP Vegas algorithm. In this algorithm, network estimated buffer Δ is compared with a predefined constant α. Little buffering space is indicated if $\Delta < \alpha$ and in this case TCP-Africa is switched to fastmode and FastRecovery and Congestion Avoidance Algorithm is applied. β(w) and α(w) calculates the decreased and increased. In other case, TCP-Africa is swapped to slowmode. In slowmode NewReno rules are applied that increments by one after receiving every ACK and decrements by splitting the cwnd 50:50 after detecting loss. TCP-Africa has better bandwidth utilization in high bandwidth delay product networks. It loss ratio is less as compared to STCP and HS-TCP. It intra, inter and RTT fairness is high. It is not being applied in real OS [6].

2.8 TCP-illinois

This sender-side-modified protocol was introduced in 2008 Liu et al. at UIUC. Standard TCP's AIMD algorithm is modified in this protocol. Delay and loss are used as congestion signals to decrement or increment cwnd. Its performance is high than the standard TCP and network bandwidth is shared fairly. If congestion is not detected, TCP-illinois appraises its cwnd on each ACK arrival in one RTT by (α/cwnd). Otherwise, cwnd is decreased by (β*cwnd).

$Cwnd = cwnd + (\alpha/cwnd) \quad (0 \le \alpha \le 1)$

$Cwnd=cwnd-(\beta*cwnd) \quad (0.125 \le \beta \le 0.5)$

2.9 Compound TCP (C-TCP)

CTCP was introduced in 2006 by Tan and Song (2006). This is a loss-delay based TCP. This TCP increases the bandwidth utilization over high bandwidth delay product networks by combining 2 modes of HS-TCP and NewReno. α is a predefined constant in CTCP which is compared with estimated Δ. If Δ surpasses α, this protocol decreases the value of W_{fast} by ζ which is a predefined constant.

$W_{fast} = W_{fast} - (\zeta * \Delta)$

This calculated W_{fast} is added to final cwnd.

$W = W_{reno} + W_{fast}$

W_{fast} is leveled transaction from fastmode of HS-TCP to slowmode of NewReno. When CTCP exceeds the threshold, it shows a convex curve but TCP-Africa increases linearly. But performance of both protocols is same. TCP Vegas generates RTT estimation problem and because of this, CTCP is very sensitive to RTT measurements. Due to this CTCP can be unfair to some extent. It is being used in Microsoft Windows operating systems and it is the most commonly used congestion control algorithm [2].

2.10 YeAH TCP

It is the abbreviation of "Yet Another High-speed" TCP. It was proposed in 2007 by Baiocchi et al. YeAH is like CTCP and TCP Africa. In this protocol, network delay is predicted by RTT estimation and loss detection. STCP and NewReno are combined in YeAH. In every RTT, congestion window is increased by one and if loss is detected, congestion window is decreased by half when 3 duplicated ACKs are received. YeAH have two thresholds (α, φ) whose values are predefined. If ($Q/RTT_{min} < \varphi$) and ($\Delta < \alpha$), YeAH behaves like CTCP by switching to fastmode. On other hand, YeAH is switched to NewReno slowmode. Intra-fairness, inter-fairness, RTT-fairness and efficiency of YeAH is higher than other protocols in higher BDP-networks. It also has disadvantage of RTT transmission due to Vegas algorithm [6].

2.11 TCP Fusion

TCP Fusion was proposed in 2007 by Kaneko et al. It has combined DUALs queuing delay, Westwood's achievable rate and network buffering estimation using Vegas algorithm. In this protocol, Fusion can be switched into three modes. This switching depends on queuing delay's threshold value. Fastmode is applied when predefined threshold value is greater than queuing delay and congestion window value is increased by Westwood's estimation fraction. Congestion window is decreased by no. of packets, if threshold value is three times less than current queuing delay. Congestion window is neither increased nor decreased if the value of queuing delay is between 1 and 3 times of the threshold. Its fairness and bandwidth utilization is greater as compared to other protocols. It has some disadvantages like threshold value is calculated manually.

2.12 TCP Hybla:

One more variant of TCP is Hybla. It is offered to improve the channel Goodput / output in high speed networks. The main goal of Hybla is to eradicate castigation of TCP that combine an extraordinary-latency telluric or outpost wireless connection, because of their greater RRT (Round Trip Times). It stalks from a logical assessment of the CWD, which recommends the essential amendments to eliminate the enactment reliance on Round Trip Time. Error Rates (ER) of the channels are increasing very rapidly. So this can be significantly diminished with the help of methods like Forward Error Correction. Many methods or techniques had been suggested to recompense the inaccurate attribution of congestion, and to offer improved CW organization and also impartiality from Round Trip Time. The Hybla has advantage of address lengthy Round Trip Time. It is much superior as compare to other variants of the TCP in case of greater latency and greater ER. Great work has been done in case of Congestion Control with the help of Hybla for greater latency with greater ER [4].

2.13 TCP Tahoe:

One more variant of TCP is Tahoe; it deals with slow start and congestion window (CWND) and when damage ensues, FR (Fast Retransmit) is delivered, partial of the recent CWND is keep back in sense of slow start threshold after that slow start activates once more from its early CWND. Here the time reaches when CWND touches the slow start

threshold and then TCP alters the CAA (Congestion Avoidance Algorithm), here every fresh acknowledgment upsurges the CWND by (Slow Start+ Slow Start)/ Congestion Window. So the consequences come in form of undeviating upsurge of the CWND [6].

2.14 Agile SD TCP:

Agile-SD could be a UNIX operating system-based Congestion management algorithmic program (CCA) that is meant for the important Linux kernel. It's a receiver-side algorithmic program services a loss-based methodology retaining an original method, referred to as Agility Factor. it's been projected by Mohamed A. Alrshah et al. to extend the information measure utilization over high-speed and short-distance networks (low-BDP networks) like native space networks or Fiber Optic network, particularly once the pragmatic buffer size is tiny. It's been estimated by evaluating its recital to Compound-TCP and CUBIC by means of NS-2 machine. It advances the overall recital up to fifty fifth in term of average outturn [7].

2.15 Zeta TCP:

Zeta-TCP identifies the congestions from each the latency and loss rate methods, and put on completely changed CWND methods supported the probability of the congestions to exhaust the possibilities the throughput. It additionally contains a number of alternative enhancements to accurately find the packet losses, dodging retransmission break retransmission; and hasten the inward traffic. Zeta-TCP denotes to a collection of exclusive TCP algorithms besieged to enhance the recital of TCP, despite whether or not the peer is Zeta-TCP or the other TCP protocol heap, in alternative words, to be well-matched with the prevailing TCP algorithms [6].

Zeta-TCP agreements the subsequent enhancements mainly:

- Congestion dodging supported each latency and loss methods.
- Amended loss-detection rule.
- Converse management.

2.16 Data center TCP (DC-TCP):

DC control TCP uses ECN to reinforce the Transmission management Protocol congestion control formula. It's utilized in DC networks. Where the quality congestion control formula is merely ready to observe the presence of congestion, Data center CTCP, by means of ECN, is ready to measure the level of congestion. Data Center TCP alters the protocol headset to perpetually impart the precise ECN pattern of entering packets at the price of flouting a purpose that's intended to reserve signaling liableness. It creates a Data Center Protocol dispatcher at risk of loss of acknowledgments from the receiver that it's no machinery to observe or address [4].

2.17 TFRC:

TFRC may be a TCP-Friendly, congestion management protocol that aims to contend equally for information measure with transmission control protocol drifts [7].

2.18 TCP Real:

Transmission control protocol-Real services a receiver sloping and dimension built congestion management machinery that increases TCP recital above high speed networks and above uneven ways [5].

2.19 TCP Jersey:

Transmission control protocol-Jersey may be a new TCP theme that emphases on the potential of the transport machinery to differentiate the wireless from congestion package losses [7].

2.20 Vegas TCP:

Congestion avoidance is implemented in TCP Vegas instead of sensing congestion first and after that lessening channel congestion. In this protocol, basic RTT is calculated and then it is compared with RTT of packet with recently received ACK. If RTT is greater than basic RTT, sending window is decreased and if RTT is smaller than basic RTT, sending window is increased [5].

3. Comparative Study:

Comparison study has been done to check the behavior of Bandwidth, Fairness (intra-, inter-), Throughput, Algorithms, RTT fairness and Loss

Ratio for TCP variants, NewReno, S-TCP, BIC-TCP, HS-TCP, CUBIC, H-TCP, TCP-Africa, TCP Illinois, C-TCP, YeAh TCP, TCP Fusion, Hybla, TCP Tahoe, Agile SD-TCP, Zeta-TCP, DC-TCP, TFRC, TCP Real, TCP Jersey and VTCP. The objective of this paper was to make a comparative study of all TCP variants for high speed networks. From the Figures, it's clear that, by increase in the value of buffer size (number of packets), throughput also increases. So it can be concluded that, the greater buffer size is the larger ThroughPut. Thus, to completely use the High Speed Bandwidths, all the current TCP variants quiet need additional enhancement to extend their capability.

Moreover, BIC, CUBIC, HTCP, Illinois, Agile SD, Zeta, DC-TCP, TCP Real and TCP jersey are best with respect to fairness (intra-, inter-) as compare to other TCP variants. In most of the cases, CUBIC overwhelms all other variants, so we can say that CUBIC (loss-based) is the best one. HS-TCP (loss based), TCP jersey (loss/delay based) and YeAh (loss/delay based) exhibits best enactment in many scenarios [14].

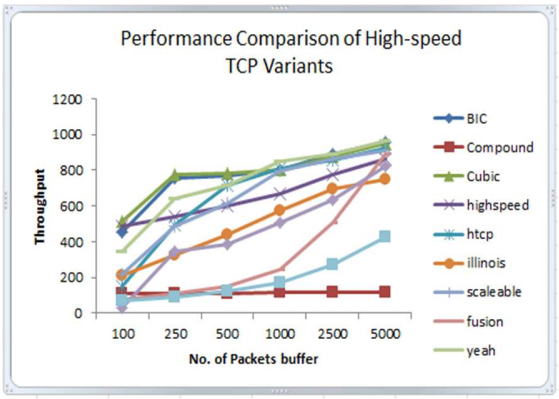

Figure 1: Performance Comparison of TCP Variants in terms of throughput vs No. of packets Buffer

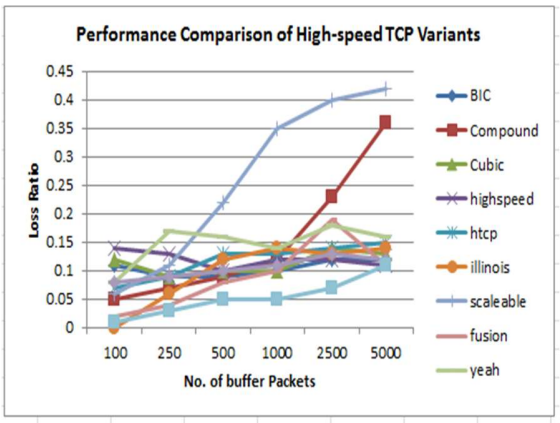

Figure 2: Performance Comparison of TCP Variants in terms of Loss Ratio vs No. of packets Buffer

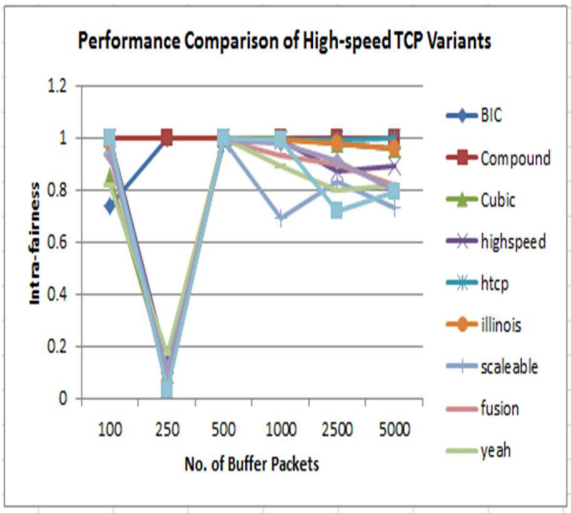

Figure 3: Performance Comparison of TCP Variants in terms of Intra-fairness vs No. of packets Buffer

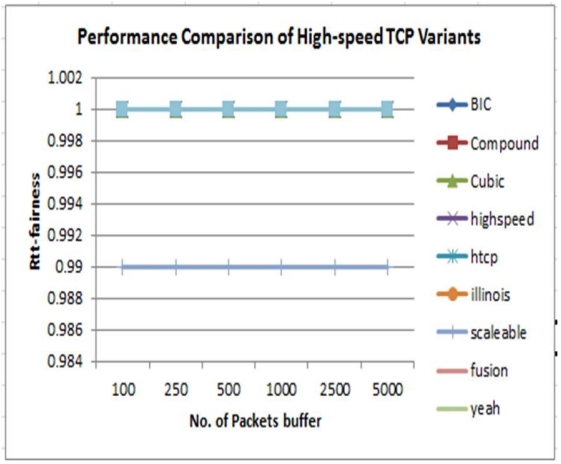

Figure 4: Performance Comparison of TCP Variants in terms of RTT-fairness vs No. of packets Buffer

TCP variants can be compared on the basis of different parameters such as:

1. **Variety of acknowledgments received:** the quantity of acknowledgments received is associate degree pointer of the in transmission of packages. The larger the quantity of acknowledgment packages, larger is the information communicated during a specified time. That the variety of acknowledgments expected will alright be used as an enactment metric so as to check the 2 protocol variants.
2. **Throughput:** It is defined because the quantity of helpful info that's supplied for each unit time not including protocol overhead and resent information packets. It's considered as a enactment metric rather than outturn, as a result of outturn isn't a well-known factor once managing protocol overhead. It's usually checked at a allusion below the network layer.
3. **LossRatio:** The ratio b/w no. of lost packets to the sent packets is lossratio.
4. **Intra Fair and RTT fair:** To make a check on the concurrency and flow of the TCP's bandwidth and time, intrafair and RTT fair are used respectively [14]

Table 1: Comparison Table of TCP variants

Sr. No.	TCP Variant	Founded	Approach	Bandwidth	Fairness (intra-,inter-)	Throughput	Algorithm	RTT fairness	Loss Ratio
1	NewReno	1999	Loss Based	Low bandwidth	Low Fairness	Poor	AIMD	Oscillating fairness	Lowest
2	S-TCP	2003	Loss Based	High bandwidth	Low Fairness	Low	AIMD	Oscillating fairness	Highest
3	BIC TCP	2004	Loss Based	High bandwidth	High Fairness	Best	AIMD	Best Case	Stable
4	HS-TCP	2003	Loss Based	High bandwidth	Low Fairness	Low	AIMD	Oscillating fairness	Stable
5	CUBIC	2008	Loss Based	High bandwidth	High Fairness	Best	AIMD	Best Case	Stable
6	H-TCP	2004	Loss Based	High bandwidth	High Fairness	High	AIMD	Best Case	Stable
7	TCP Africa	2005	Loss/ Delay Based	Average bandwidth	Low Fairness	Low	Vegas algorithm	Oscillating fairness	Stable
8	TCP-illinois	2006	Loss & Delay Based	Average bandwidth	High Fairness	High	AIMD	Best Case	Stable
9	C-TCP	2006	Loss/ Delay Based	High bandwidth	Proportional	High	AIMD	Best Case	Highest

10	YeAH TCP	2007	Loss/ Delay Based	High bandwidth	Low Fairness	Best	Vegas algorithm	Oscillating fairness	Stable
11	TCP Fusion	2007	DUALs queuing delay Based	Average bandwidth	Low Fairness	Low	Vegas algorithm	Oscillating fairness	Stable
12	Hybla	-----	Loss Based	High bandwidth	Average Fairness	Average	Slow start and congestion avoidance	Best Case	Stable
13	TCP Tahoe	1988	Loss Based	Low Bandwidth	Low fairness	Low	Modified AIMD with fast recovery mechanism	Low fairness	Low
14	Agile-SD TCP	2015	Loss Based	High Bandwidth	High Fairness	High	CCA with agility factor	High fairness	Stable
15	Zeta-TCP	----	Latency & Loss Based	High Bandwidth	High fairness	High	Improved loss detection with reverse control	High fairness	Low
16	DC-TCP	-----	Multi bit signal	High Bandwidth	High fairness	High	Modified ECN	Best case	Stable
17	TFRC	-----	Loss Based	No retransmission	Minimum delay	Stable	AIMD algorithm	High fairness	Average
18	TCP Real	-----	Rate Based	High Bandwidth	High fairness	High	AIMD with advance error detection & classification	High fairness	Low
19	TCP Jersey	-----	Loss/ Delay	Average Bandwidth	High fairness	High	Available bandwidth estimation algorithm with congestion warning	High fairness	Low
20	Vegas TCP	1999	2-bit Signal	High Bandwidth	Proportional	High	Modified Congestion Control Algorithm	High fairness	Least

4. Conclusion:

In this paper we investigate the behavior of Bandwidth, Fairness (intra-, inter-), Throughput, Algorithms, RTT fairness and Loss Ratio for TCP variants, NewReno, S-TCP, BIC-TCP, HS-TCP, CUBIC, H-TCP, TCP-Africa, TCP Illinois, C-TCP, YeAh TCP, TCP Fusion, Hybla, TCP Tahoe, Agile SD-TCP, Zeta-TCP, DC-TCP, TFRC, TCP Real, TCP Jersey and VTCP. The objective of this paper was to make a comparative study of all TCP variants for high speed networks. From the Figures, it's clear that, by increase in the value of buffer size (number of packets), throughput also increases. So it can be concluded that, the greater buffer size is the larger ThroughPut. Thus, to completely use the High Speed Bandwidths, all the current TCP variants quiet need additional enhancement to extend their capability. Moreover, BIC, CUBIC, HTCP, Illinois,

Agile SD, Zeta, DC-TCP, TCP Real and TCP jersey are best with respect to fairness (intra-, inter-) as compare to other TCP variants. In most of the cases, CUBIC overwhelms all other variants, so we can say that CUBIC (loss-based) is the best one. HS-TCP (loss based), TCP jersey (loss/delay based) and YeAh (loss/delay based) exhibits best enactment in many scenarios. Though, throughput is the number of positively received packets per unit time. The ratio b/w no. of lost packets to the sent packets is lossratio. To make a check on the concurrency and flow of the TCP's bandwidth and time, intrafair and RTT fair are used respectively.

ACKNOWLEDGEMENT

This research was funded by Department of Computer Engineering, University of Engineering and Technology, Lahore. We are grateful to our Supervisor, Dr. Junaid Arshad, for providing consistent support throughout our research work.

References

[1] M. A. O. M. A. B. &. H. Z. M. Alrshah, "Comparative study of high-speed Linux TCP variants over high-BDP networks," *Journal of Network and Computer Applications,* pp. 66-75, 2014.

[2] V. &. D. R. Bhanumathi, "TCP variants-A comparative analysis for high bandwidth-delay product in mobile adhoc network," *In Computer and Automation Engineering (ICCAE), 2010 The 2nd International Conference,* vol. 2, pp. 600-604, 2010, February.

[3] I. R. H. F. S. &. K. A. Abdeljaouad, "Performance analysis of modern TCP variants: A comparison of Cubic, Compound and New Reno," *Communications (QBSC), 2010 25th Biennial Symposium,* pp. 80-83, 2010, May.

[4] J. L. D. &. M. M. Cloud, "Network Coded TCP (CTCP) Performance over Satellite Networks," 2013.

[5] G. A. I. M. &. J. K. Abed, "A comparison and analysis of congestion window for HS-TCP, Full-TCP, and TCP-Linux in long term evolution system model," *Open Systems (ICOS), 2011 IEEE Conference,* pp. 358-362, 2011, September.

[6] D. A. &. A. G. Hayes, "Revisiting TCP congestion control using delay gradients," *In International Conference on Research in Networking,* pp. 328-341, 2011, May.

[7] J. B. S. A. M. M. A. M. R. Abed, "COMPARISON OF HIGH SPEED CONGESTION CONTROL PROTOCOLS," *International Journal of Network Security & Its Applications (IJNSA),* vol. 4, 2012, September.

[8] S. J. S. K. R. &. R. S. Trivedi, "Comparative performance evaluation of TCP Hybla and TCP Cubic for satellite communication under low error conditions," *Comparative performance evaluation of TCP Hybla and TCP Cubic for satellite cIn Internet Multimedia Services Architecture and Application (IMSAA), 2010 IEEE 4th International Conference ,* pp. 1-5, 2010, December.

[9] P. &. P. P. Tomar, "A Comprehensive Analysis and Comparison of TCP Tahoe, TCP Reno and TCP Lite.," *International Journal of Computer Science and Information Technologies (IJCSIT),* vol. 5, pp. 2467-2471, 2011.

[10] M. A. O. M. A. B. &. H. Z. M. Alrshah, "Agile-SD: a Linux-based TCP congestion control algorithm for supporting high-speed and short-distance networks.," *Journal of Network and Computer Applications,* pp. 181-190, 2015.

[11] "https://en.wikipedia.org/wiki/Zeta-TCP".

[12] M. G. A. M. D. A. P. J. P. P. P. B. S. M. Alizadeh, "Data center tcp (dctcp)," *CM SIGCOMM computer communication review,* vol. 40, pp. 63-74, 2010, August.

[13] J. D. R. &. M. M. Widmer, "A survey on TCP-friendly congestion control," *IEEE network,* vol. 5, pp. 28-37, 2011. May.

[14] C. &. T. V. Zhang, "TCP-real: improving real-time capabilities of TCP over heterogeneous networks.," *In Proceedings of the 11th international workshop on Network and operating systems support for digital audio and video,* pp. 189-198, 2011, January.

[15] K. T. Y. &. A. N. Xu, "TCP-Jersey for wireless IP communications," *IEEE Journal on Selected Areas in Communications,* vol. 4, pp. 747-756, 2014.

[16] A. &. S. H. Sawarkar, " Performance Analysis of TCP Variants," *International Journal of Computer Science and Network Security (IJCSNS),* vol. 4, 2016.

[17] "https://en.wikipedia.org/wiki/TCP_congestion_control".

Manuscript received February 5, 2025
Manuscript revised February 20, 2025
https://doi.org/10.22937/IJCSNS.2025.25.2.21