Ensuring Better Network Utilization by Implementing Different TCP Protocols TCP Reno, TCP New Reno & TCP FACK

Boriya Punit H.[1] Ajani Mohit K.[2] Koringa Hiren C.[3]
[1,2,3] Department of Electronics & Communication
[1,2,3] Marwadi Education Foundation's Faculty of P.G. Studies & Research in Engineering & Technology

Abstract— Reliable transport protocols such as TCP are tuned to perform well in traditional networks where packet losses occur mostly because of congestion .TCP responds to all losses by invoking congestion control and avoidance algorithms, resulting in degraded end-to-end performance in systems. In this paper, we analyze and compare the different congestion control and avoidance mechanisms by Network Simulator 2 (NS2)which have been proposed for TCP/IP protocols, namely :TCP Reno, New-Reno, and TCP FACK implement different graphs of link capacity, throughput , packet loss rate , average queue ,no of packet dropped ,RTTP time and observe that which TCP protocol give the best result.

I. INTRODUCTION

TCP is a reliable connection oriented end-to-end protocol. It contains within itself, mechanisms for ensuring reliability by requiring the receiver the acknowledge the segments that it receives. The network is not perfect and a small percentage of packets are lost enroute, either due to network error or due to the fact that there is congestion in the network and the routers are dropping packets. We shall assume that packet losses due to network loss are minimal and most of the packet losses are due to buffer overflows at the router[1]. Thus it becomes increasingly important for TCP to react to a packet loss and take action to reduce congestion. TCP ensures reliability by starting a timer whenever it sends a segment. If it does not receive an acknowledgement from the receiver within the 'time-out' interval then it retransmits the segment. We shall start the paper by taking brief look at each of the congestion avoidance algorithms and noting how they differ from each other. In the end we shall do a head to head comparison to further bring into light the differences.

II. TCP PROTOCOLS

A. TCP RENO

This Reno retains the basic principle of Tahoe, such as slow starts and the coarse grain re-transmit timer. However it adds some intelligence over it so that lost packets are detected earlier and the pipeline is not emptied every time packet is lost. Reno requires that we receive immediate acknowledgement whenever segment is received. The logic behind this is that whenever we receive duplicate acknowledgment, then his duplicate acknowledgment could have been received if the next segment in sequence expected, has been delayed in the network and the segments reached there out of order or else that the packets lost.

B. TCP NEW-RENO

New RENO is a slight modification over TCP-RENO. It is able to detect multiple packet losses and thus is much more efficient that RENO in the event of multiple packet losses. Like Reno, New-Reno also enters into fast-retransmit when it receives multiple duplicate packets, however it differs from RENO in that it doesn't exit fast-recovery until all the data which washout standing at the time it entered fast-recovery is acknowledged. Thus it overcomes the problem faced by Reno of reducing the CWD multiples times. The fast-transmit phase is the same as in Reno. The difference in the fast-recovery phase which allows for multiple re-transmissions in new-Reno. Whenever new-Reno enters fast-recovery it notes the maximums segment which is outstanding. The fast-recovery phase proceeds as in Reno, however when a fresh ACK is received then there are two cases: If it ACK's all the segments which were outstanding when we entered fast-recovery then it exits fast recovery and sets CWD to thresh and continues congestion avoidance like Tahoe. If the ACK is a partial ACK then it deduces that the next segment in line was lost and it re-transmits that segment and sets the number of duplicate ACK Received to zero. It exits Fast recovery when all the data in the window is acknowledged [3].

C. TCP FACK

Improvement in SACK through Forward Acknowledgement is known as TCP FACK [12]. The use of FACK is just about same like TCP SACK but creates a little improvement estimated to it. This uses TCP SACK for efficiently evaluation the data's amount in transit [12]. Thus, TCP FACK announces an improved mode to split the window in case of congestion detection. When Contention Window is split instantly then sender pause sending for some time and then starts again when sufficient data has gone from the network. Here one RTT can be ignored when window is diminished step by step [12].Once there happens congestion then window would be split according to the multiplicative reduction of the accurate Contention Window. In the meantime the sender detects jamming, after it at least one RTT occurred. In between that RTT if it was in slow start manner then present Contention Window almost will be double than the Contention Window when congestion happened. So, in this situation, Contention Window is first split to evaluate accurate Contention Window which further ought to be reduced.

III. SIMULATION RESULTS

A. *Varying Link capacity*

The Link capacity is varied between 1 Mbps to 1000 Mbps.

Simulation Time	100 s
No. of node	3
RTTP Time	20 ms

Table 1: Parameter changes with fix Link Capacity

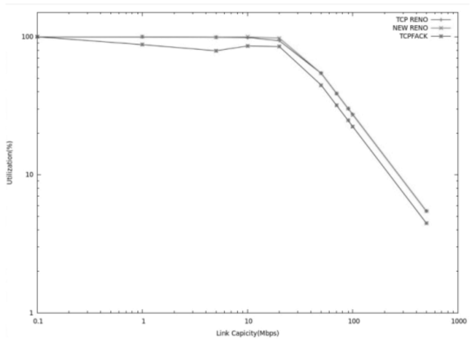

Fig. 1: Link capacity vs Link utilization

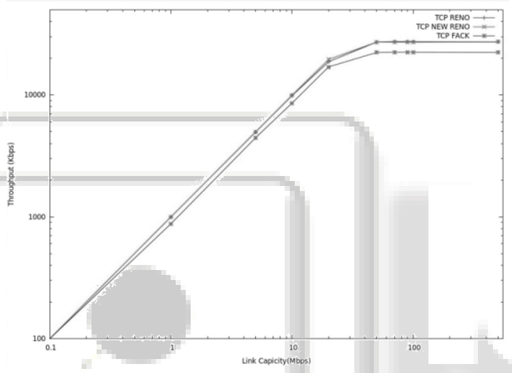

Fig. 2: Link capacity vs Throughput

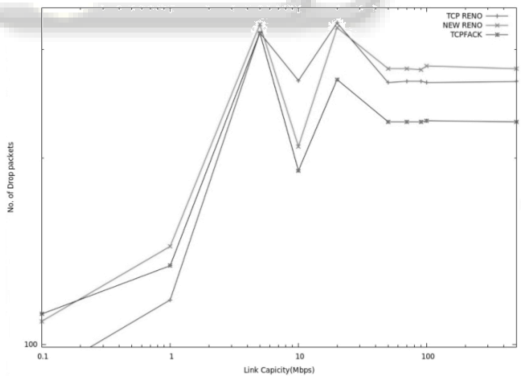

Fig. 3: Link capacity vs no. Of Drop packets

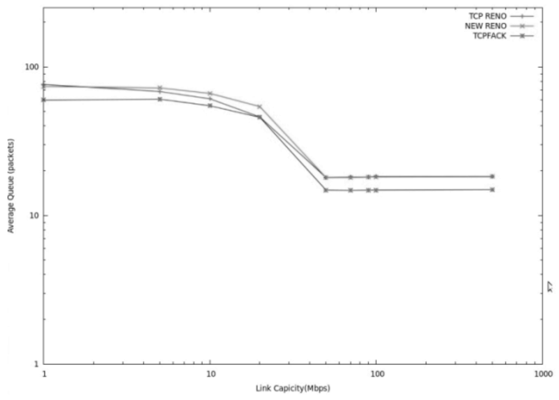

Fig. 4: Link capacity vs Avg Queue

B. *Varying RTTP Time*

The round-trip propagation time is varied between 1 ms to 1000 ms.

Simulation Time	100 s
No. of node	3
Link Capacity	10 Mbps

Table 2: Parameter changes with fix RTTP Time

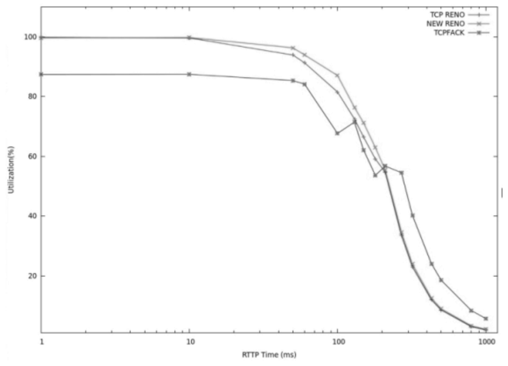

Fig. 5: RTTP Time vs Link utilization

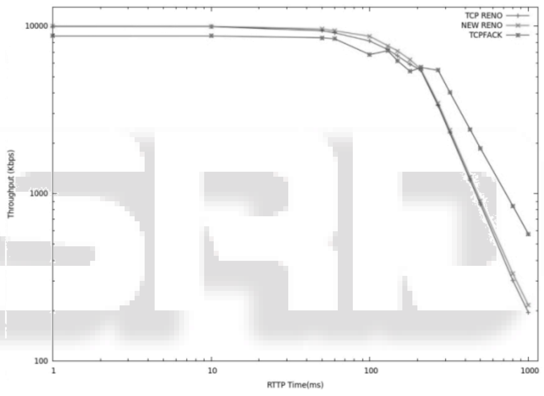

Fig. 6: RTTP Time vs Throughput

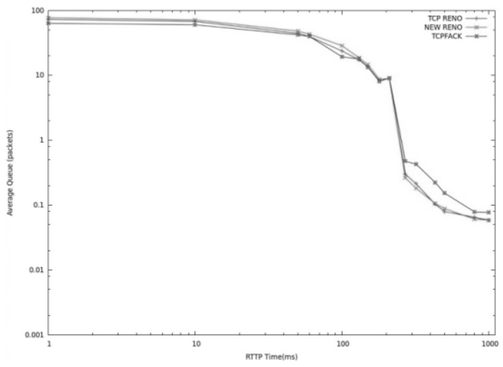

Fig. 7: RTTP Time vs Avg Queue

C. *Varying No. of Nodes*

The No. Of Nodes are varied between 3 to 1000.

Simulation Time	100 s
Link Capacity	10 Mbps
RTTP Time	20 ms

Table 3: Parameter changes with fix No. Of node

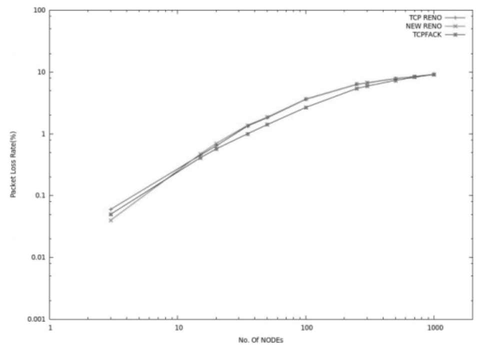

Fig. 8: No. Of Nodes vs Loss Rate

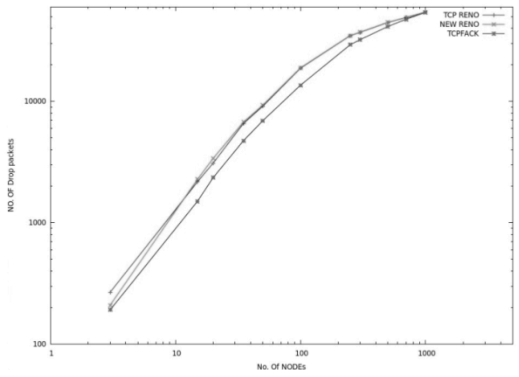

Fig. 9: No. of Nodes vs No. Of Drop packets

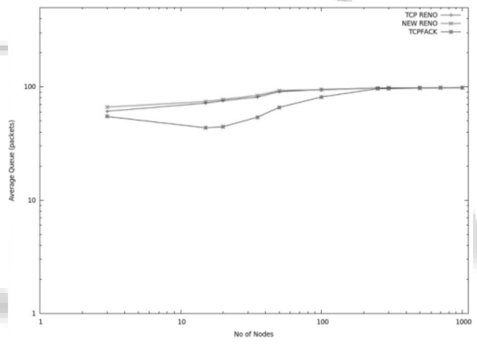

Fig. 10: No. Of Node vs Avg Queue

IV. CONCLUSION

This paper investigates the performance of difference TCP Protocols and compares them. Result show that TCP New Reno achieves high link utilization, negligible packet loss and high throughput and TCP New Reno has better performance than other TCP protocol.

ACKNOWLEDGMENT

I thankful to Mr. Ghadiya Hitesh , Mr. Bhavin Sorathiya and my friends for providing me guidance and technical support in conducting this work. I am also thankful to Network Simulator developers for providing such a smart platform for carrying this work.

REFERENCES

[1] V.Jacobson. “Congestion Avoidance and Control”. SIGCOMM Symposium no Communication Architecture and protocols.
[2] V.Jacobson “Modified TCP Congestion Control and Avoidance Algorithms”. Technical Report 30, Apr 1990.
[3] S.Floyd, T.Henderson “The New-Reno Modification to TCP’s Fast Recovery Algorithm” RFC 2582, Apr 1999.
[4] O. Ait-Hellal, E.Altman “Analysis of TCP Reno and TCP Vegas”.
[5] K.Fall, S.Floyd “Simulation Based Comparison of Tahoe and Reno”.
[6] L.S.Brakmo, L.L. Peterson, “TCP Vegas: End to End Congestion Avoidance on a Global Internet”, IEEE Journal on Selected Areas in Communication, vol. 13[1995],(1465-1490).
[7] Behrouz A. Forouzan, “TCP/IP Protocol Suite”, Third edition, published by TATA Mc Graw-Hill Publishing Company Limited
[8] H.Balakrishnan, V. Padmanabhan, S. Seshan and R. Katz, “A comparison of Mechanisms for improving TCP performance over wireless links,” Proceedings of ACM SIGCOMM'96, Aug. 1996
[9] V.Anantharaman, S.-J.Park, K. Sundaresan,and R. Sivakumar, “TCP Performance over Mobile Ad-hoc Networks: A Quantitative Study,” 'To appear in Wireless Communications and Mobile Computing Journal (WCMC), Special Issue on Performance Evaluation of Wireless Networks, 2003.
[10] R.Caceres & L. Iftode, “Improving the performance of reliable transport protocols in mobile computing environments,” IEEE JSAC. Vol.19 No.7, July 2001
[11] Network Simulator 2 (ns2), http://www.isi.edu/nsnam/ns/
[12] B. Qureshi, M. Othman, Member, IEEE, and. A. W. Hami “Progress in Various TCP Variants”, IEEE, February 2009