Performance Analysis of TCP variants in Wired and Wireless scenario

Vivek kumar Majithiya[1] Priyang Bhatt [2]
[1, 2] Department of Information Technology
[1, 2] G.H.Patel College of Engineering & Technology, Vallabh Vidyanagar, Gujarat, India

Abstract---TCP is a transport layer protocol which provides reliable and connection oriented transmission. The main problem of TCP is in wireless networks. In wired network, packet loss is mostly due to network congestion, while in wireless network the packet loss may be due to network congestion or may be due to mobility of nodes. TCP starts its back off procedure when it detects packet loss, if the packet loss is due to node movement then the back off procedure will be unnecessary and it will create delay in the network. There are many TCP variants available. In this paper we have analyzed five variants of TCP, TCP Tahoe, TCP Reno, TCP NewReno, TCP Vegas, and TCP Westwood by calculating its throughput and delay in wired as well as wireless scenario.

Keywords: MANET, TCP, back off procedure

I. INTRODUCTION

Mobile ad hoc network is an autonomous and self-configuring network [1] composed of several mobile nodes i.e. mobile equipments like cell phones, PDAs, tablets etc. These mobile nodes communicate with each other through the wireless links established between nodes. Furthermore the communications between nodes occur without the support of any infrastructure or any centralized access point like a Base Station. Thus MANET never forms any fixed topology as it is dynamic in nature.

The Transmission Control Protocol (TCP) [2] is used by the vast majority of Internet applications. The TCP is a connection-oriented transport protocol that provides a reliable byte-stream data transfer service between pairs of processes. When two processes want to communicate, they must first establish a TCP connection (initialize the status information on each side). Since connections must be established between unreliable hosts and over the unreliable Internet communication system, a handshake mechanism with clock based sequence numbers is used. The TCP provides multiplexing facilities by using source and destination port numbers. These port numbers allow the TCP to set up virtual connections over a physical connection and multiplex the data streams through that connection. Each connection is uniquely specified by a pair of sockets1identifying its two sides and by specific parameters such as sequence numbers, window sizes, and status information (necessary for reliability and flow and congestion control mechanisms – see the following for more details).At the end of a communication, the connection is terminated or closed to free the resources for other uses. The TCP is able to transfer a continuous stream of bytes, in each direction, by packaging some number of bytes into segments (TCP data unit). The size of these segments and the timing at which they are sent is generally left to the TCP module. However, an application can force delivery of segments to the output stream using a push operation provided by the TCP to the application layer. A push causes theca to promptly forward and deliver data to the receiver. Apart from this basic data transfer, the TCP provides some more services. First of it is able to recover from data that are damaged, lost, duplicated, or delivered out of order by the Internet communication system (reliability).Missing or corrupted segments are detected and retransmitted assigning a sequence number to each transmitted segment, and requiring a positive acknowledgment(ACK) from the receiver. If the ACK is not received within a timeout interval, the data are retransmitted. At the receiver, the sequence numbers are used to correctly order segments that may be received out of order and to eliminate duplicates. Damage is handled by adding a checksum to each transmitted segment, checking it at the receiver, and discarding damaged segments.

The TCP also provides a means for the receiver to govern the amount of data sent by the sender (flow control). This is achieved by returning a ‘‘window’’, named advertised window, that indicates the allowed number of bytes that the sender may transmit before receiving further permission.

Moreover, TCP users may indicate the security and precedence of their communication. Provision is made for default values to be used when these features are not needed. And the most important feature provided bt TCP is TCP congestion control

II. TCPCONGESTION CONTROL ALGORITHMS

The four algorithms, Slow Start, Congestion Avoidance, Fast Retransmit and Fast Recovery [3] [4] are described below.

A. Slow Start

Slow Start, a requirement for TCP software implementations is a mechanism used by the sender to control the transmission rate, otherwise known as sender based flow control. This is accomplished through the return rate of acknowledgements from the receiver. In other words, the rate of acknowledgements returned by the receiver determines the rate at which the sender can transmit data.

When a TCP connection first begins, the Slow Start algorithm initializes a congestion window to one segment, which is the maximum segment size (MSS) initialized by the receiver during the connection establishment phase. When acknowledgements are returned by the receiver, the congestion window increases by one segment for each acknowledgement returned. Thus, the sender can transmit the minimum of the congestion window and the advertised window of the receiver, which is simply called the transmission window. At some point the congestion window may become too large for the network or network conditions may change such that packets may be dropped. Packets lost will trigger a timeout at the sender. When this happens, the sender goes into congestion avoidance mode as described in the next section.

B. Congestion Avoidance

During the initial data transfer phase of a TCP connection the Slow Start algorithm is used. However, there may be a point during Slow Start that the network is forced to drop one or more packets due to overload or congestion. If this happens, Congestion Avoidance is used to slow the transmission rate. However, Slow Start is used in conjunction with Congestion Avoidance as the means to get the data transfer going again so it doesn't slow down and stay slow.

In the Congestion Avoidance algorithm a retransmission timer expiring or the reception of duplicate ACKs can implicitly signal the sender that a network congestion situation is occurring. The sender immediately sets its transmission window to one half of the current window size (the minimum of the congestion window and the receiver's advertised window size), but to at least two segments. If congestion was indicated by a timeout, the congestion window is reset to one segment, which automatically puts the sender into Slow Start mode. If congestion was indicated by duplicate ACKs, the Fast Retransmit and Fast Recovery algorithms are invoked.

As data is received during Congestion Avoidance, the congestion window is increased. However, Slow Start is only used up to the halfway point where congestion originally occurred. This halfway point was recorded earlier as the new transmission window. After this halfway point, the congestion window is increased by one segment for all segments in the transmission window that are acknowledged. This mechanism will force the sender to more slowly grow its transmission rate, as it will approach the point where congestion had previously been detected.

C. Fast Retransmit

When a duplicate ACK is received, the sender does not know if it is because a TCP segment was lost or simply that a segment was delayed and received out of order at the receiver. If the receiver can re-order segments, it should not be long before the receiver sends the latest expected acknowledgement. Typically no more than one or two duplicate ACKs should be received when simple out of order conditions exist. If however more than two duplicate ACKs are received by the sender, it is a strong indication that at least one segment has been lost. The TCP sender will assume enough time has lapsed for all segments to be properly re-ordered by the fact that the receiver had enough time to send three duplicate ACKs.

When three or more duplicate ACKs are received, the sender does not even wait for a retransmission timer to expire before retransmitting the segment.

D. Fast Recovery

Since the Fast Retransmit algorithm is used when duplicate ACKs are being received, the TCP sender has implicit knowledge that there is data still flowing to the receiver. The reason is because duplicate ACKs can only be generated when a segment is received. This is a strong indication that serious network congestion may not exist and that the lost segment was a rare event. So instead of reducing the flow of data abruptly by going all the way into Slow Start, the sender only enters Congestion Avoidance mode. Rather than start at window of one segment as in Slow Start mode, the sender resumes transmission with a larger window, incrementing as if in Congestion Avoidance mode. This allows for higher throughput under the condition of only moderate congestion [5].

III. TCP VARIANTS

A. TCP-Reno

TCP-Reno is an implementation of TCP used by most networks today [7]. It uses different congestion control algorithms. They include Congestion Avoidance mechanisms, Fast Recovery, Fast Retransmit and Slow Start. TCP-Reno exploits packet losses in the network to estimate the available bandwidth in the network. It activates Slow Start process in the start of a TCP connection as well as after timeouts during the connection. During this process it initially increases CWND exponentially but after Slow Start Threshold increases it linearly which is known as Congestion avoidance mechanism. Fast Retransmit and Fast Recovery mechanisms are initiated after receiving three duplicate Acknowledgements (ACKs) or when a timeout occur. These two mechanisms improve the performance of TCP-Reno which interprets timeouts an indication of serious congestion in the network [8].However Fast Recovery and Fast Retransmit mechanisms result in more efficient transfer of packets in the network.

B. TCP-New Reno

TCP-New Reno is a variant of Reno with an improved Fast Recovery (FR) algorithm in order to solve the timeout problem where multiple packets are lost from the same window. Congestion Control components of TCP-New Reno and TCP Reno are identical. TCP- New Reno distinguishes a Full ACK (FA) from a Partial ACK (PA) by modifying TCP-Reno's Fast Recovery behavior after it receives a non-duplicate ACK. FA acknowledges all the outstanding segments at the beginning of FR. However PA acknowledges only some of the outstanding data. TCP New Reno unlike Reno can recover from multiple segment losses by retransmitting only one lost segment in the same window per RTT and remains in Fast Recovery unless and until a full ACK is received [8].

C. TCP-Vegas

Bandwidth Estimation scheme used by TCP Vegas is more efficient than other TCP variants. This scheme makes bandwidth estimation by using the difference between the expected flow rates and the actual flow rates. It extends TCP-Reno by modifying its Congestion Avoidance mechanism. Like TCP-Reno it uses Slow Start and Fast Retransmission. TCP-Vegas use its Congestion Avoidance mechanism in order to avoid packet loss by decreasing its CWND as soon as it detects congestion in the network. This is unlike TCP-Reno which initiates its Congestion Avoidance mechanism and increases its CWND until a packet loss is detected in the network. Also Retransmission mechanism used by TCP-Vegas is more efficient as compared to TCP-Reno as it retransmits the corresponding packet as soon as it receives a single duplicate ACK and does not wait for three ACKs. TCP-Vegas as compared to TCP-Reno is more accurate and is less aggressive, thus it does not reduce its CWND unnecessarily [9].

D. TCP-Westwood

The Fast Recovery algorithm from TCP New Reno has been modified. To help gain faster recovery bandwidth estimation (BWE) algorithm also has been added. This BWE function is what makes TCP Westwood standout. Influenced by TCP Vegas, BWE uses the RTT and the amount of data that has been sent during this interval to calculate an estimate of the currently successful transfer rate. The bandwidth estimate is then used when a loss is detected, setting cwnd and sssthresh at values near the estimation. The main purpose behind this is to improve the throughput in wireless links, where loss is more often caused by link failure than by congestion. There is also the general benefit that starting CA at higher values will lower the recovery time on most networks, thus lowering the transfer times.

E. TCP-Tahoe

TCP-Tahoe complements TCP with different mechanisms of Slow Start, Congestion Avoidance and Fast Recovery. However it does differ from other TCP variants by using a modified Round Trip Time Estimator. TCPTahoe sets Slow Start Threshold (ssthresh) to half of the CWND and sets the value of CWND to 1. Source enters in the Slow Start phase after an acknowledgement of the retransmitted packet has been received. If the size of the CWND is less than the new ssthresh, CWND size increases exponentially. After Slow Start sender enters in Congestion Avoidance phase and onwards the size of CWND increases by 1/CWND for each received acknowledgement [11].

IV. TCP IN WIRED NETWORK

We have analysed five variants of TCP in wired network and calculated its throughput and delays. The results for throughput are

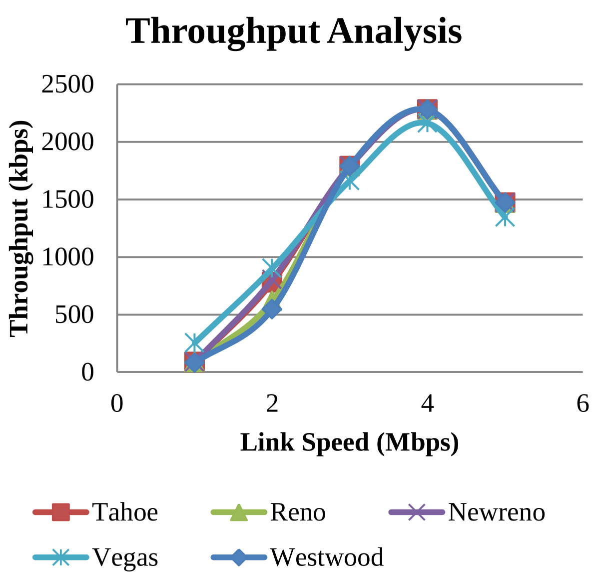

Fig. 1: Throughput Analysis of TCP Wired Network

The results for delay are

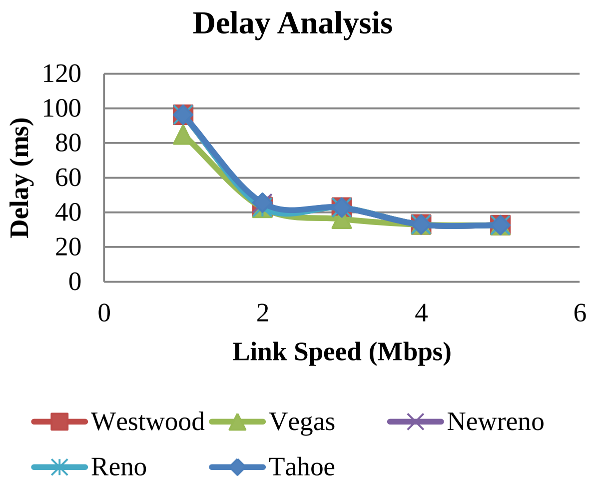

Fig. 2: Delay Analysis of TCP Wired Network

V. TCP IN WIRELESS ENVIRONMENT

The results for throughput and delay of different TCP variants on increasing number of nodes are

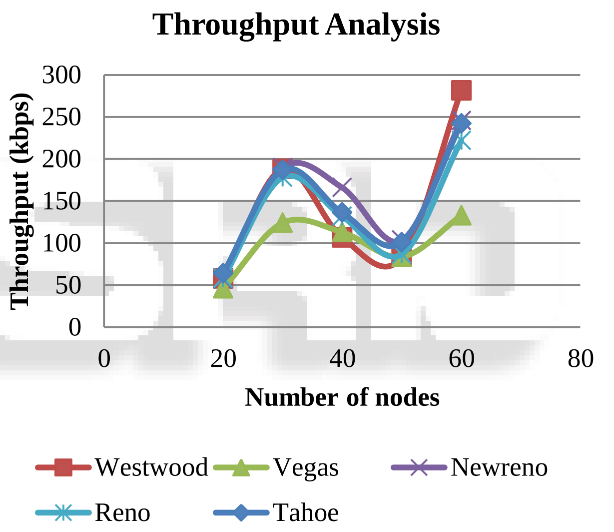

Fig. 3: Throughput Analysis of TCP Wireless Network

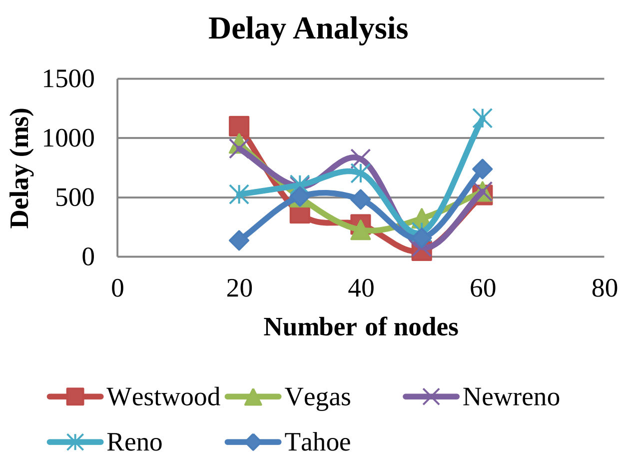

Fig. 4: Delay Analysis of TCP Wireless Network

VI. CONCLUSION

As we can see from the graphs that TCP Reno and TCP NewReno performs almost same, TCP Vegas performs better for small link speed for throughput but when the link speed is increased than the performance of Vegas degrades for throughput. TCP Vegas not performs well than all other variants in terms of delay.

TCP Westwood gives better result in terms of Throughput and delay in increasing number of nodes in wireless network.

ACKNOWLEDGMENT

My Sincere thanks to my guide Prof. Priyang Bhatt, for providing me an opportunity to do my research work. I express my thanks to my Institution namely G. H. Patel College of Engineering and Technology for providing me with a good environment and facilities like Internet, books, computers and all that as my source to complete this research work. My heart-felt thanks to my family, friends and colleagues who have helped me for the completion of this work.

REFERENCES

[1] C. E. Perkins, “Ad Hoc Networking”, Addison -Wesley, Pearson edu. Jan 2001.
[2] C. Callegari , S. Giordano, M. Pagano, T. Pepe, “Behaviour analysis of TCP linux variants”, ELSEVIER 2011
[3] K. Kathiravan, Dr. S. Thamarai Selvi, A. Selvam “Tcp Performance Analysis for mobile adhoc network using on demand routing protocols”
[4] JACOBSON, V. Congestion avoidance and control. In *Proceedings of SIGCOMM ’88* (Stanford, CA, Aug. 1988), ACM.
[5] W. Stevens. TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms, January 1997, RFC 2001.
[6] Suhas Waghmare, Aditya Parab, Pankaj Nikose, Prof. S J Bhosle, ”Comparative analysis of different TCP variants in a wireless environment
[7] C. Grimm and H. Schwier, *Empirical Analysis of TCPVariants and Their Impact on GridFTP Port requirements*,ICNS07, Athens, Greece: IEEE, 2007.[8] MO. Richard and J. La, *Analysis and Comparison of TCPReno and Vegas*, INFOCOM’99. Berkeley, California: IEEE, 1999.
[8] R. Awdeh, Evaluating Fairness in Heterogeneous Wireless Ad Hoc Networks, ICCCN’07. Sharjah, UAE: IEEE, 2007
[9] K. Fall and S. Floyd, Simulation-based comparisons of Tahoe, Reno, and SACK TCP, ACM SIGCOMM Computer Communication Review, Vol. 26, pp. 5-21, 1996.
[10] D. Kim and H. Bae., *Analysis of the Interaction between TCP Variants and Routing Protocols in MANETs*, ICPPW05,Oslo, Norway: IEEE, 2005.
[11] Shagufta Henna, “A Throughput Analysis of TCP Variants in Mobile Wireless Networks”, IEEE 2009

