Comparison of various TCP Variant In Mobile Ad- Hoc Networks

Deepika Bhenia , Student,Electronics and communication Department, Sushila Devi Bansal College of Technology, Indore,India.

Vinod sonkar , Asst.professor,Electronics and communication Department, Sushila Devi Bansal College of Technology, Indore,India.

Abstract

Transmission control protocol (TCP) has undergone several transformations. Several proposals have been put forward to change the mechanisms of TCP congestion control to improve its performance. A line of research tends to reduce speed in the face of congestion thereby penalizing itself. In this group are the window based congestion control algorithms that use the size of congestion window to determine transmission speed. However, TCP has the problem that the performance deteriorates especially in large-bandwidth and long-delay networks. The two main algorithm of window based congestion control are the congestion avoidance and the slow start. The aim of this study is to survey the various modifications of window based congestion control. Much work has been done on congestion avoidance hence specific attention is placed on the slow start in order to motivate a new direction of research in network utility maximization. The well known problem of TCP in high bandwidth delay product networks is that the TCP Additive Increase probing mechanism is too slow in adapting the sending rate to the available bandwidth. Various TCP Variants have been suggested for this, such as TCP Vegas, Reno, NewReno, Sack. The performances of various variants are analyzed in bidirectional scenarios using throughput and congestion window plots .In this article we present the performance comparison of existing TCP variants: TCP TCP Vegas, Reno, NewReno, Sack. for mobile ad-hoc networks. The behavior of TCP was different depending on the type of TCP variants because of improper activation or missing of congestion control. This analysis and comparisons are necessary to be aware of which TCP implementation is better for a specific scenario.

Keywords- *Mobile ad-ho; Adaptive; TCP; Congestion control; Packet loss.*

1.INTRODUCTION

The transport layer protocol performs an end-to-end connection, end-to-end delivery of data packets, error detection, flow control, and congestion control for networks. The transmission control protocol (TCP) is the most usable transport layer protocol in the Internet today. It transports large number of the traffic on the Internet. Its reliability, end- to-end congestion control mechanism, byte stream transport mechanism, and its tactful and simple design have not only contributed to the success of the Internet, but also have made TCP an influencing protocol in the design of many of the other protocols and applications. Its adaptability to the congestion in the network has been an important feature leading to graceful degradation of the services offered by the network at times of extreme congestion. The TCP protocol has been extensively tuned to give good performance at the transport layer in the traditional wired network environment. However, TCP in its present form is not well-suited for mobile ad hoc networks (MANETs) where packet loss due to broken routes can result in the reverse invocation of TCP's congestion control mechanisms. The Transmission Control Protocol [1, 2] (TCP) is the most used transport protocol in the Internet today. It is a part of the TCP/IP protocol suite which allows computers, regardless of operating system and hardware, to communicate with each other. One of the major properties of TCP is that it is able to provide a connection-oriented data transfer service that is reliable to applications that require that no data is lost and/or damaged in the communication process. TCP is used in conjunction with the Internet Protocol [3] (IP) which only provides an unreliable connectionless data transfer service between different hosts. To be able to provide connection-oriented reliable communication, TCP needs to implement mechanisms on top of IP.TCP in its traditional form was designed and optimized only for wired networks. Extensions of TCP that provide improved performance across wired and single-hop wireless networks. Since TCP is widely used today and the efficient integration of an ad hoc wireless network with the Internet is paramount wherever possible, it is essential to have mechanisms that can improve TCP's performance in ad hoc wireless networks. This would enable the seamless operation of application-level protocols such as FTP, SMTP, and HTTP across the integrated ad hoc wireless networks and the Internet Although a number of studies have been conducted and protocol modifications suggested. The reason behind the variations of TCP is that each type possesses some special criteria, such as the traditional TCP has become known as TCP Tahoe. TCP Reno adds one new mechanism called Fast Recovery to TCP Tahoe [11]. TCP New Reno uses the newest retransmission mechanism of TCP Reno [8].

2. CONGESTION CONTROL MECHANISMS

This section describes the predominant example of end-to-end congestion control[2] in use today, that implemented by TCP. The essential strategy of TCP is to send packets into the network without a reservation and then to react to observable events that occur. TCP assumes only FIFO queuing in the network‛s routers, but also works with fair queuing.

2.1 Additive Increase/Multiplicative Decrease

TCP maintains a new state variable for each connection, called Congestion Window [3], which is used by the source to limit how much data it is allowed to have in transit at a given time. The congestion window is congestion control‛s counterpart to flow control‛s advertised window. TCP is modified such that the maximum number of bytes of unacknowledged data allowed is now the minimum of the congestion window and the advertised window.

MaxWindow = MIN (CongestionWindow, AdvertisedWindow)

EffectiveWindow = MaxWindow − (LastByteSent −LastByteAcked).

That is, MaxWindow replaces Advertised Window in the calculation of Effective Window. Thus, a TCP source is allowed to send no faster than the slowest component—the network or the destination host can accommodate .The problem, of course, is how TCP comes to learn an appropriate value for Congestion Window. Unlike the Advertised Window, sent by receiving side of the connection, there is no one to send a suitable CongestionWindow to the sending side of TCP. TCP does not wait for an entire window‛s worth of ACKs to add one packet‛s worth to the congestion window, but instead increments CongestionWindow by a little for each ACK that arrives. Specifically, the congestion window is incremented as follows each time an ACK arrives:

Increment = MSS × (MSS/Congestion Window)

Congestion Window + = Increment

That is, rather than incrementing Congestion Window by an entire MSS bytes each RTT, we increment it by a fraction of MSS every time an ACK is received. The important concept to understand about AIMD is that the source is willing to reduce its congestion window at a much faster rate than it is willing to increase itscongestion window.

2.2 Fast Retransmit and Fast Recovery

The mechanisms described so far were part of the original proposal to add congestion control to TCP. It was soon discovered, however, that the coarse-grained implementation of TCP timeouts led to long periods of time during which the connection went dead while waiting for a timer to expire. Because of this, a new mechanism called *fast re- transmit* was added to TCP. Fast retransmit is a heuristic that sometimes triggers the retransmission of a dropped packet sooner than the regular timeout mechanism.

3. VARIANTS OF TCP

This section describes the basic functions of TCP variants, which can clearly substitute the TCP implementations such as TCP Reno, TCP NewReno and TCPSack ,TCP Vegas.

A. *TCP Reno*

TCP Reno [1] 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 a packet is lost. Reno requires that we receive immediate acknowledgement whenever a segment is received. The logic behind this is that whenever we receive a 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 packet is lost. If we receive a number of duplicate acknowledgements then that means that sufficient time have passed and even if the segment had taken a longer path, it should have gotten to the receiver by now. There is a very high probability that it was lost. So Reno suggests an algorithm called 'Fast Re- Transmit'. Whenever we receive3 duplicate ACK's we take it as a sign that the segment was lost, so we re-transmit the segment without waiting for timeout. The basic algorithm is presented as under:

1.Each time we receive 3 duplicate ACK's we take that to mean that the segment was lost and we re-transmit the segment immediately and enter 'Fast- Recovery'.

2.Sets ssthresh to half the current window size and also set CWND to the same value.

3.For each duplicate ACK receive increase CWND by one. If the increase CWND is greater than the amount of data in the path then transmit a new segment else wait.

Reno performs very well over TCP when the packet losses are small. But when we have multiple packet losses in one window then RENO doesn't perform too well .The reason is that it can only detect single packet losses. If there is multiple packet drops then the first info about the packet loss comes when we receive the duplicate ACK's. But the information about the second packet which was lost will come only after the ACK for the retransmitted first segment reaches the sender after one RTT. Another problem is that if the window is very small when the loss occurs then we would never receive enough duplicate acknowledgements for a fast retransmit and we would have to wait for a coarse grained timeout. .Reno's Fast Recovery algorithm is optimized for the case when a single packet is dropped from a window of data.

B. TCP-NewReno.

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 was out standing at the time it entered fast recovery is acknowledged. The fast-recovery phase proceeds as in Reno, however when a fresh ACK is received then there are two cases:

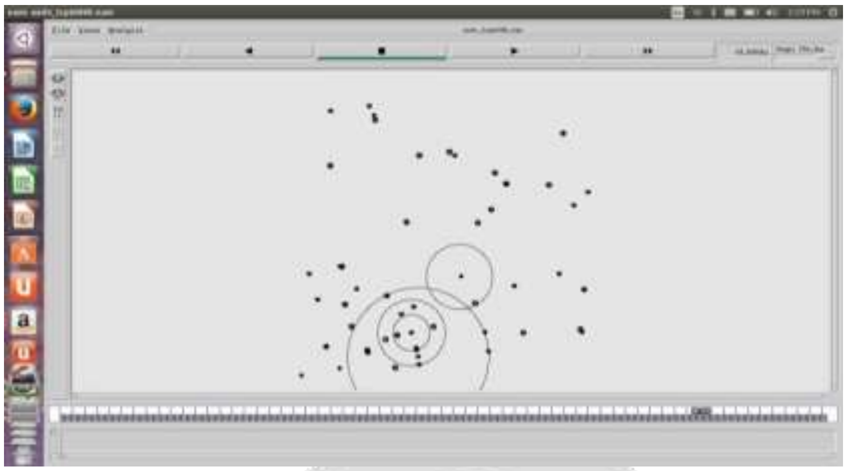
1. If it ACK's all the segments which were outstanding when we entered fast recovery then it exits fast recovery and sets CWD to threshold value and continues congestion avoidance like Tahoe.

2.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 ACKS received to zero. It exits Fast recovery when all the data in the window is acknowledged.

C. TCP Sack
TCP Sack adds a number of Sack blocks in the TCP packet, where each Sack block acknowledges a non-contiguous set of data has been received. The main difference between SACK TCP and Reno TCP implementations is in the behavior when multiple packets are dropped from one window of data. SACK sender maintains the information which packets is missed at receiver and only retransmits these packets. When all the outstanding packets at the start of fast recovery are acknowledged, SACK exits fast recovery and enters congestion avoidance [17].

D. TCP Vegas
TCP Vegas is a TCP congestion avoidance algorithm that emphasizes packet delay, rather than packet loss, as a signal to help determine the rate at which to send packets[14][15].TCP Vegas detects congestion at an incipient stage based on increasing Round-Trip Time (RTT) values of the packets in the connection unlike other flavors like Reno, NewReno, etc., which detect congestion only after it has actually happened via packet drops. The algorithm depends heavily on accurate calculation of the Base RTT value. If it is too small then throughput of the connection will be less than the bandwidth available while if the value is too large then it will overrun the connection. A lot of research is going on regarding the fairness provided by the linear increase/decrease mechanism for congestion control in Vegas. One interesting caveat is when Vegas is inter-operated with other versions like Reno. In this case, performance of Vegas degrades because

4.Simulation Environment and Results Analysis:
In this simulation there is scenario in which we use 60 nodes in wireless networks. In this investigation we use routing protocol AODV (Adhoc on demand distance vector) with different types of TCP variants Reno,New Reno,Vegas,Sack based on ad-hoc wireless network of 70 nodes. The investigation involves the measurement of different parameters like Throughput, Delivery Ratio, Number of packet send, Number of packet dropped ,Average delay ,Average jitter of the network in each of the TCP variants. Finally the result achieved AODV routing protocol with TCP variants no of nodes in the network will be accessed.We discussed the results of simulated scenario of TCP variants Reno, New Reno, Vegas, Sack with different parameters using Ns-2 simulator .



Throughput: Throughput is defined as the total amount of data received by destination node from the source node divided by the total time it takes from the destination to get the last packet and it measures is bits per second (bit/s or bps).

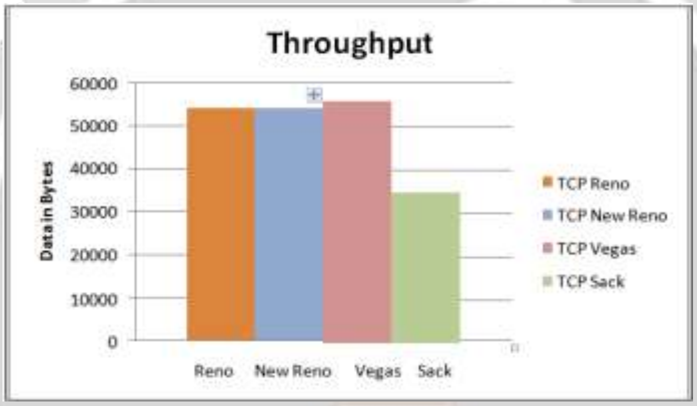


Fig.1 Throughput

Fig 1.shows the throughput in terms data in bytes. This shows that Vegas has better throughput as compared to others has low throughput value.

Number of packet send: Rate of transmission of packets.

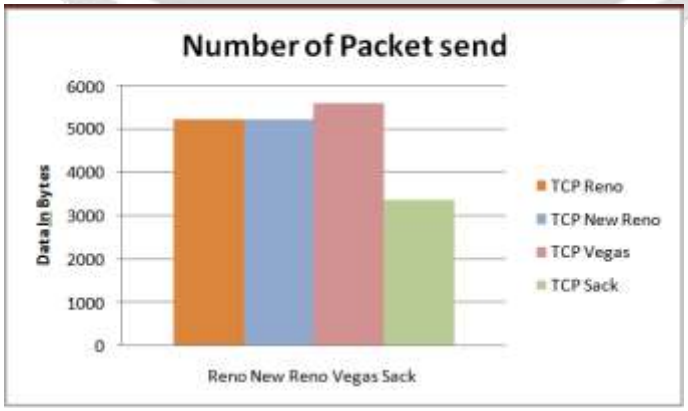


Fig.2 Number of packet send

Fig 2 shows the number of packet send in terms of bytes. This shows that data rate of Vegas is better than other variants.

Vegas is better than other variants.

Number of packet Dropped: Failure of one or more transmitted packets to arrive at their destination.

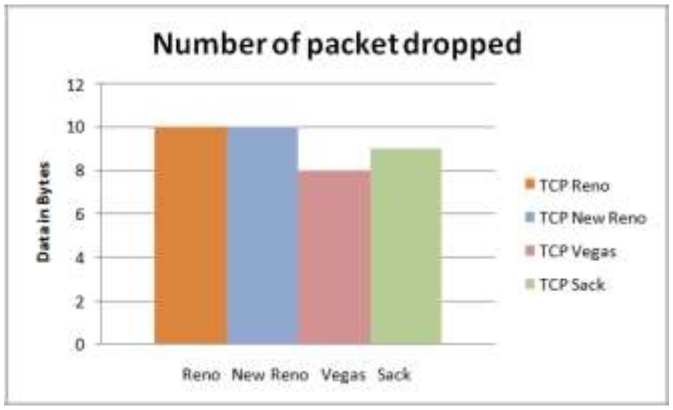


Fig.3 Number of packet dropped

Fig 3 shows that number of packet dropped in terms bytes. This shows that Vegas has less number of packet dropped as compared to other variants.

Delivery Ratio: Packet delivery ratio is the ratio of total packets sent by the source node to the successfully received packets by the destination node.

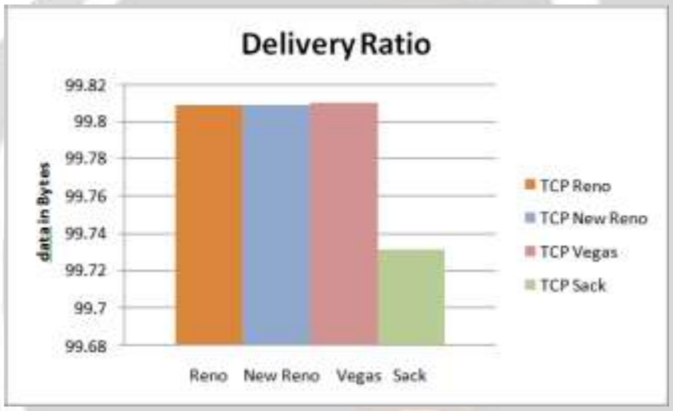


Fig.4 Delivery Ratio

Fig 4 shows the delivery ratio of TCP in terms of bytes. This shows that TCP Vegas has better delivery ratio than other variants.

Average Delay: Average end-to-end delay is the time interval when a data packet generated from source node is completely received to the destination node.

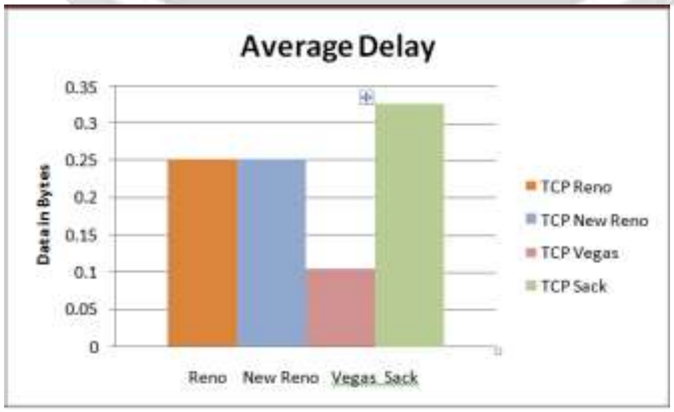


Fig.5 Delivery Ratio

Fig 5 shows the average delay of all the TCP variants.This shows that the Vegas has less delay as compared to other variants.

5.CONCLUSION

The comparison on existing TCP variants based on throughput and packet loss analyzed from various experimental results obtained from ns2. and described about the protocol which one is better and suitable for packet and link utilization in the network congestion and link failure condition in Ad-hoc network environment because the traditional TCP treat all packet losses due to the congestion, it does not treat from the link failure.

This paper compares the TCP variants performance using AODV routing protocol on NS2 simulator with different parameters Throughput ,Number of packet send, Number of packet dropped ,Delivery Ratio, Average Delay, Average Jitter. Simulation results shown through graphs represent overall performance of TCP variants with AODV routing protocol .From the obtained results shown by graphs, we can say that TCP Vegas shows highest efficiency and performs best Although there are various protocols have been used, that can overcome the congested and unreliable nature of network. Though there are various schemas and mechanisms proposed, there is no single mechanism that can overcome the unreliable nature of network in a reliable way .Here each and every variant has its own advantages and disadvantages to solve the networks problems of TCP protocol.
In short, any protocol will be effective based on the parameters that are to be taken into consideration. To conclude this area is not completely explored to it maximum and still lot more research can be done towards establishing a basis for the development of new protocol.

6. FUTURE WORK

Our review suggests that in forthcoming efforts, analysis of TCP algorithm and comparisons of them. In future we explore the effect of congestion avoidance algorithm on all TCP variants. In order to accurately simulate the realistic congested network environment, there is a need to experiment with multiple TCP flows. Our future work is to propose a new algorithm for congestion avoidance in congested network to improve the TCP environment.

REFERENCES

[1] A.Gurtov and S. Floyd, "Modelling wireless links or transport Protocols,"ACM SIGCOMM, April 2004.

[2] K. Fall, S. Floyd **"**Simulation Based Comparison of Tahoe, Reno and SACK TCP**"**, 1998**.**

[3] M. Mathis, J. Mahdavi,"Forward Acknowledgement: Refining TCP Congestion Control" in Proceedings of ACM SIGCOMM, 1996.

[4] W. Stevens, "TCP Slow Start, Congestion Avoidance Fast Retransmit Algorithm"**,** IETF RFC 2001, January 1997.
[5] Van Jacobson," Congestion Avoidance and Control"

SIGCOMM Symposium on communications Architectures

and Protocols, rd Stevens: TCP/IP Illustrated, Volume 1: "The Protocols", Addison Wesley, 1994.

[6] Renaud Bruyeron, Bruno Hemon, Lixia Zhang: "Experimentations with TCP Selective Acknowledgment", ACM SIGCOMM Computer Communication Review, April 1988.

[7] S.Floyd, T.Henderson "The New-Reno Modification to TCP's Fast Recovery Algorithm" RFC 2582, Apr 1999.

[8] Ad Hoc Mobile Wireless Networks: Protocols and Systems, C.K. Toh, Springer Prentice Hall Publishers, ISBN 013 007 8174, 2001.

[9] Ahmad Al Hanbali, Eitan Altman, Philippe Nain "A Survey of TCP over Ad Hoc Networks "June, 2005.

[10] Yi-Cheng Chan, Chia-Liang Lin, and Fang-Chun Liu "A Competitive Delay-Based TCP", 2008 IEEE.

[11] Qualnet simulator version 3.10 user's manual by Scalable Network Technologies, Inc, 2000, 2001.

[12] Qualnet simulator Scalable Network Technology, "QualNet4.0 simulator" Tutorial -Qualnet forum- www.scalable-networks.com.

[13] Pawan Kumar Gupta “Throughput Enhancement of TCP over Wireless Links”January2002.”http://etd.ncsi.iisc.ernet.in/bitstream/2 005/48/1/Throughput_Enhancement_Of_TCP_Over_Wireless_Links_Te xt.pdf”.

[14] Jinwen Zhu, Tianrui Bai , Performance of Tahoe, Reno, and SACK TCP at Different Scenarios, 27-30 Nov. 2006