

PERFORMANCE ANALYSIS OF TCP VARIANTS PROTOCOLS FOR CONGESTION IN WIRELESS AD-HOC NETWORK

Author: Pooja Patel¹; Nirali Sarang²; Daizy Daruwala³

Affiliation: B.E.Computer Engg. Student¹; B.E.Computer Engg. Student; B.E.Computer Engg. Student³

E-mail: phpatel.4518@gmail.com¹; sarangnirali555@gmail.com²; modidaizy@yahoo.com³

ABSTRACT

TCP is the basic protocol for communication over the large area of network. The increasing demand and popularity of wireless communication system have led us to understand the better optimization of TCP. There are various TCP variants and each of them belongs to a different criteria. In this paper, we aim at studying TCP and its variants for congestion in wireless Ad-hoc network. This paper compares TCP variants specifically TCP Reno and TCP New Reno with original TCP in simulated network environment under DSDV routing protocol. We will consider different scenarios to carry out performance study of these protocols to ensure better data transfer speed, reliability and congestion control. Our analysis is based on three performance matrices which are throughput, end-to-end delay and packet delivery ratio. At the end we will classify which protocol performs better in different scenarios. The simulation work has been done in NS-2 environment.

Keywords: TCP, TCP Reno, TCP New Reno, congestion, NS-2, Ad-hoc Network, congestion.

1. INTRODUCTION

In this world, people require communication network every second in their daily life. Wireless Network fulfils the requirement to be connected all the time. TCP offers reliable and ordered delivery of a stream of bytes between applications running on hosts communicating over an IP network. Many Internet applications such as the WWW, Email, file transfer and remote administration rely on TCP. The Transmission Control Protocol provides a communication service at an intermediate level

between an application program and the Internet Protocol. At the Transport layer, it provides host-to-host connectivity.

TCP is the embodiment of reliable end-to-end transmission functionality in the overall Internet architecture [1]. TCP is a reliable stream delivery service that guarantees that all bytes received will be identical with bytes sent and in the correct order. Since packet transfer over many networks is not reliable, a technique known as positive acknowledgment with retransmission is used to guarantee reliability of packet transfers. This fundamental technique requires the receiver to respond with an acknowledgment message as it receives the data. The sender keeps record of each packet it sends. The sender also maintains a timer from when the packet was sent, and retransmits a packet if the timer expires before the message has been acknowledged. The timer is needed in case a packet gets lost or corrupted.

When there are several resources in a network that are shared by multiple competing senders it becomes difficult to manage the data rate used by each sender so that network need not be overloaded. The network congestion can cause severe degradation of throughput. If no proper approach is followed for controlling the congestion than it can even collapse the network.

2. CONGESTION IN TCP

In Internet congestion control is the responsibility of Transport Layer more precisely of the Transmission Control Protocol (TCP) [2]. TCP combines congestion control and reliability

mechanisms. To detect network congestion TCP simply observes occurring packet losses. Since on the Internet missing packets are almost always caused by congestion, a missing packet is interpreted as a sign for network congestion. TCP uses cumulative acknowledgments: a TCP receiver always acknowledges the end of the so-far correctly and completely received data when a new segment arrives. If segments are received out-of-order, i. e., some data is missing between the already known and the newly arriving data, the last acknowledgment is sent again (duplicate ACK). In TCP a window-based additive increase, multiplicative decrease mechanism is employed. The window size is increased by one segment in every round-trip time when no packet losses occur. In case of the reception of a duplicate acknowledgment a TCP sender will first assume that some packet reordering has occurred in the network. But upon reception of the fourth copy of an acknowledgment (Triple Duplicate ACK, TDACK) a congestion loss is assumed. In this case the missing segment is repeated and the window size is cut in half (multiplicative decrease).

Additionally, TCP uses a timeout that depends on the measured round-trip time of the connection. If this retransmission timeout (RTO) elapses without an acknowledgment TCP concludes severe congestion. Then the window size is reduced to one and the unacknowledged segment is sent again. The timeout until the next retransmission attempt if still no acknowledgment arrives is doubled. Thus this timeout grows exponentially. During the first phase of a connection and after a timeout a mechanism named slow start is employed. It allows for a faster convergence to the correct window size. While slow start is active, the window size is not increased by one segment size for every round-trip time, but instead for every received acknowledgment. This means that during this phase the window size grows exponentially.

TCP uses a congestion window in the sender side to do congestion avoidance. The congestion window indicates the maximum amount of data that can be sent out on a connection without being acknowledged. TCP detects congestion when it fails to receive an acknowledgement for a packet within the estimated timeout. In such a situation, it decreases the congestion window to one maximum segment size (MSS), and under other cases it increases the congestion window by one MSS.

There also exists a congestion window threshold, which is set to half the congestion window size at the time when a re-transmit was required.

3. TCP VARIANTS PROTOCOLS

There are various TCP variants and each of them belongs to a different criteria. TCP uses different mechanisms for congestion avoidance and it has many variants for that such as TCP Tahoe, TCP Reno, TCP New Reno, TCP SACK etc. Here we have compared three protocols which are TCP, TCP Reno and TCP New Reno.

3.1 TCP RENO

TCP RENO retains the basic principle of Tahoe, such as slow starts and the coarse grain retransmit 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 [3]. 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 receive 3 duplicate ACK's we take it as a sign that the segment was lost, so we re-transmit the segment without waiting for timeout. Thus we manage to re-transmit the segment with the pipe almost full. Another modification that RENO makes is in that after a packet loss, it does not reduce the congestion window to 1. Since this empties the pipe. It enters into an algorithm which we call 'Fast-Re-Transmit'.

The problem with TCP RENO is that it 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. 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. Thus is cannot effectively detect multiple packet losses.

3.2 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 than RENO in the event of multiple packet losses. Like RENO, New RENO [3] 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 outstanding 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: 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. 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 [5].

The problem with New-Reno is that it suffers from the fact that it takes one RTT to detect each packet loss. When the ACK for the first retransmitted segment is received only then can we deduce which other segment was lost.

3.3 COMPARISON OF PROTOCOLS

There are many versions of TCP which have been modified time to time as per need. In the earliest TCP, there were limited facilities to minimize the network congestion. Implementation used cumulative positive acknowledgements and the expiry of a retransmit timer to provide reliability based on a simple go-back-n model. Several succeeding versions of TCP based on congestion control and avoidance mechanism have been developed since then.

Table 1. Comparison Table [4]

(N=Normal, E V=Enhanced Version, N M=New Mechanism)

TCP Variants	TCP Tahoe	TCP Reno	TCP New Reno
Congestion Avoidance	Yes	Yes	Yes
Fast Retransmission	Yes	Yes	Yes
Fast Recovery	No	Yes	EV
Retransmission Mechanism	N	N	N
Congestion Control	N	N	N
Slow Start	Yes	Yes	Yes

4. IMPLEMENTATION RESULTS

We begin the analysis of TCP Protocol and its variants. We check these protocols by three parameters such as end-to-end delay, packet delivery ratio and Throughput. The results obtained in the form of graphs, all the graphs are displayed and Result is discussed based on the figure.

4.1 SIMULATION PARAMETERS

In this we have taken following simulation parameters [3].

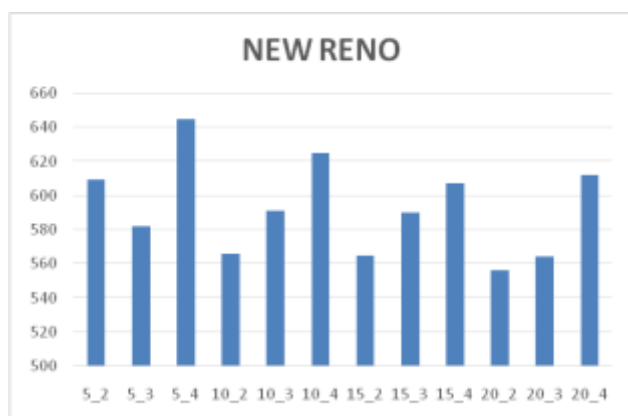
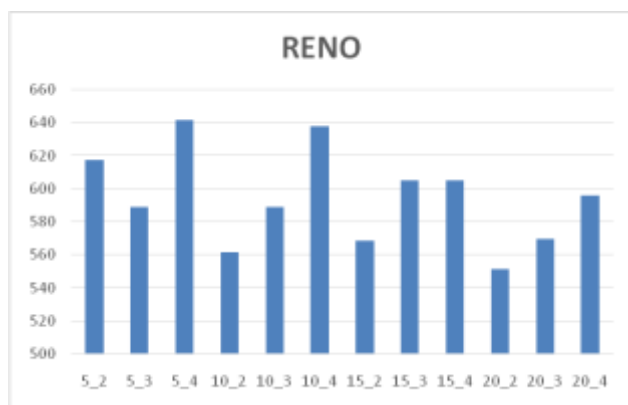
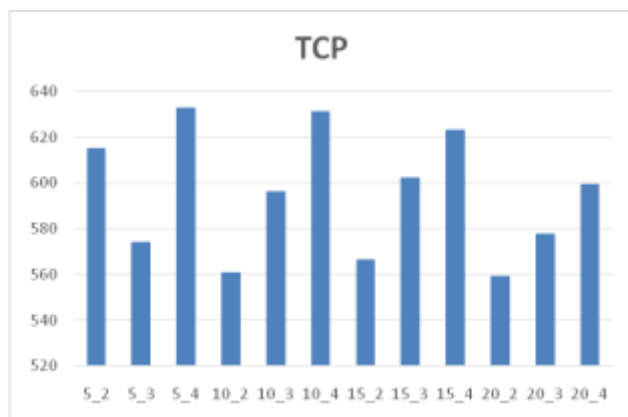
channel type	Channel/WirelessChannel
radio-propagation model	Propagation/TwoRayGround
network interface type	Phy/WirelessPhy
MAC type	Mac/802_11
interface queue type	Queue/DropTail/PriQueue
number of nodes	6

5. PERFORMANCE EVALUTION

We begin the analysis of TCP Protocol and its variants. We check these protocols by three parameters such as end-to-end delay, packet delivery ratio and Throughput. The results obtained in the form of graphs, all the graphs are displayed and Result is discussed based on the figure.

5.1 Throughput

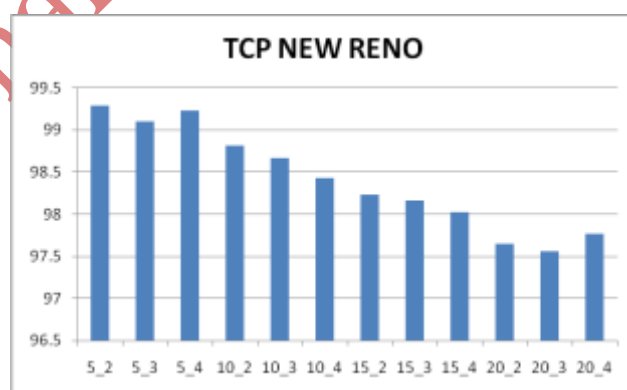
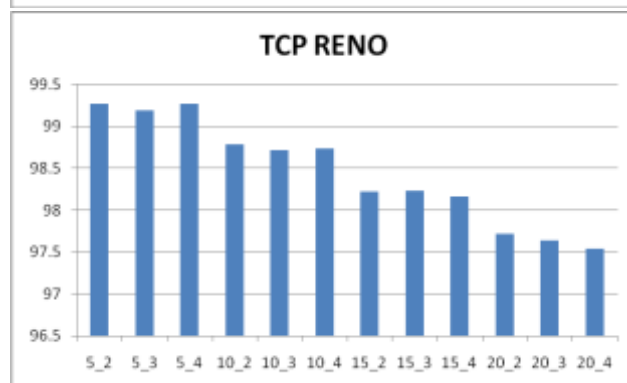
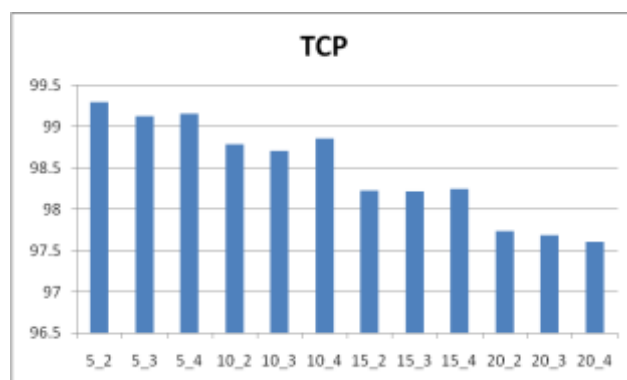
Throughput is the total amount of data received by destination nodes in a given time interval [6].



5.2 Packet Delivery Ratio

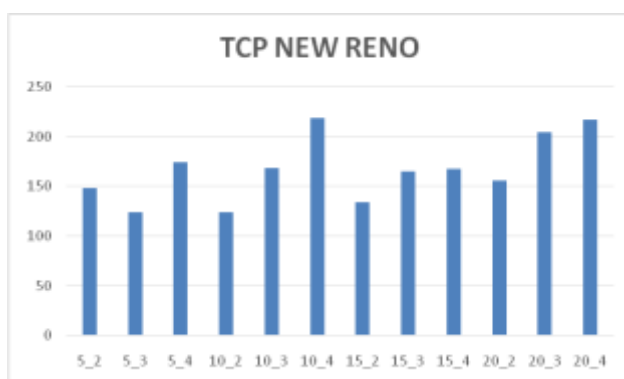
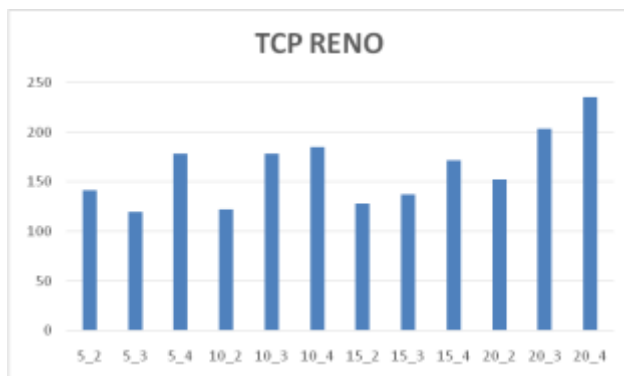
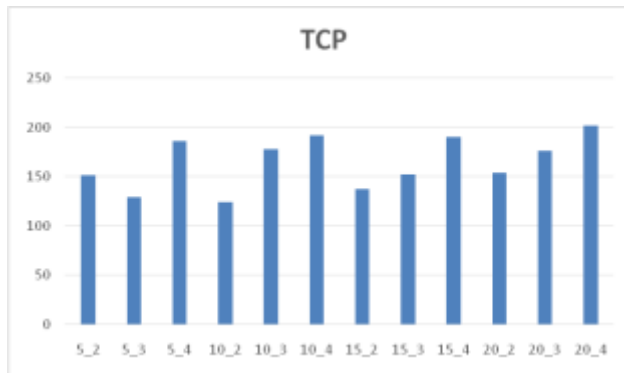
PDR is the ratio of data packets delivered to the destination to those generated by the sources [7]. It specifies the packet loss rate, which limits the maximum

throughput of the network.



5.3 End to End Delay

The end-to-end delay of a packet is defined as the time a packet takes to travel from the source to the destination. It includes all possible delay caused by buffering during route discovery latency, transmission delays at the MAC, queuing at interface queue, and propagation and transfer time. It is measured in seconds.



6. ACKNOWLEDGMENTS

Our thanks to the experts who have contributed towards development of the template.

7. CONCLUSION

Basically we are working for the TCP congestion control protocol in network. As per our scenarios, we conclude that as we change the number of nodes for different

protocols, the change reflects in our analysing parameters. As protocols varies due to the congestion control we analyse that throughput increases, packet delivery ratio decreases and end-to-end delay decreases that reflects in our results.

In future, we will examine the same scenarios for AODV protocol and also analyse TCP variant protocols (TCP Reno and TCP New Reno) for the same parameters.

8. REFERENCES

- [1]. R. Ole J. Jacobsen "The Internet Protocol Journal".
- [2]. Poonam Tomar, Prashant Panse "A Comprehensive Analysis and Comparison of TCP Tahoe, TCP Reno and TCP Lite" (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 2 (5). 2011, 2467-2471.
- [3]. Md. Monzur Morshed, Meftah Ur Rahman, Md. Habibur Rahman, Md. Rafiqul Islam "Performance Comparison of TCP variants over AODV, DSDV, DSR, OLSR in NS-2" IEEE/OSA/IAPR International Conference on Infonnatics, Electronics & Vision.
- [4]. David Espina, Dariusz Baha "The present and the future of TCP/IP".
- [5]. Yuvaraju B N, Dr. Niranjan N Chiplunkar "Scenario Based Performance Analysis of Variants of TCP Using NS2 - Simulator" International Journal of Computer Applications (0975 - 8887) Volume 4- No.9, August 2010.
- [6]. C. Ernesto Carrillo A., V'ictor M. Ramos R. "Performance evaluation of reactive and proactive routing schemes for infrastructure wireless mesh networks".
- [7]. Tariq A. Alahdal, Saida Mohammad "Performance of Standardized Routing Protocols in Ad- hoc Networks" IEEE INTERNATIONAL CONFERENCE ON COMPUTER, ELECTRICAL AND ELECTRONICS ENGINEERING (ICCEEE).