RESEARCH ARTICLE OPEN ACCESS

TCP Congestion Control Mechanisms evaluation over Wired / Wireless Environments: A Survey

K. Vasudha Rani [1], Prof.P. Chenna Reddy [2]
Research Scholar, RAYALASEEMA UNIVERSITY, Kurnool
Director, Academic Audit, JNTU- Ananthapuramu, Ananthapuramu

ABSTRACT
This paper explores the attributes of TCP agents, namely “Tahoe, Reno, New Reno, Vegas, and Sack”, and computed their throughputs in the simulated environment utilizing the NS-2 test system by shifting different exhibition of parameters, for example, data transmission, simulation time and number of activity sources with a specific end goal to figure out which one of them is the best for which scenario. Simulation was performed to concentrate the execution of the Protocols as far as Bandwidth, Simulation time and number of activity sources. After the effects of the reproduction, demonstrated that Reno is great when bundle of the packets losses are little. New Reno is observed to be valuable when there are more packet losses of information. Sack is found to very poor at the point when different packet losses happen in a same congestion window, while Vegas Is the best while fluctuating data transmission inferable from the utilization of time based data transfer capacity estimation plans to control its congestion window.

Keywords:- TCP, NS-2

I. INTRODUCTION

The Transmission Control Protocol (TCP) is one of the superior transport Protocol under the transport layer of OSI, which support the World Wide Web, email and records exchange is a basic functionality over the internet. TCP provides better performance in terms of reliability, control flow of information, fast error recovery. TCP mainly uses to communicate host-to-host in the network. Once the host-to-host connection established exchange the data between the segments happen due to segments lost will happen by congestion control. The TCP functionality well defined and implemented in the RFC 793 [1]. Congestion control is a well-recognized problem in the complex computer networks. Congestion control causes two basic problems namely call small-packets, source-quench problems explained under RFC 896[2]. The small packet problem well fixed in the RFC like if any data input from the user and which not acknowledged previous packet in the queue, a new tcp segment data transmitted. Congestion control algorithms (“congestion avoidance, Slow start, Fast retransmit and recovery”) are well explained under RFC 5681 [3]. TCP have lot of issues when network running on high overload. In this paper we discussed congestion control between different TCP variants, such as “Tahoe, Reno, New Reno, Sack, Vegas, Westwood, Fack and Veno” and compared the performance in two stages.in the first stage we compared the performance evaluation between TCP Reno and RQB.in the second stage we assess the execution of four protocols of TCP that is Reno, New Reno, Sack and Vegas.

II. TCP/IP Overview

This paper will explore the execution correlations of these previously mentioned adaptations of TCP and discover which one is better in which cases. Generally, TCP/IP are the two different layer protocols which works seamlessly. TCP is the upper layer of which responsible for data transfer of a packet whereas IP is more responsible for logical addressing explained in the RFC 791. TCP standardization approved by IETF as RFC 793 [4] in 1981 and have wide range of features like “Interoperability, multi-vendor support, logical addressing, name resolution, routing, multiplexing/DE multiplexing, Flow control and error control”. IP standardized with RFC791 [5] in 1981. IP is belonging to network layer and have unique features, namely “unreliable, support for connectionless, data deliver without Acknowledgments”.

2.1. TCP Congestion Control

Congestion control balance the network traffic in the computer networks in order to eliminate the congestion during over huge incoming data. This is gradually decreases throughput of the network. In initial days the theory of the concept defined by the author “Frank Kelly”. Congestion control algorithms are classify based on type, deployment, capability and fairness.TCP congestion control have lot of flavors which are listed in the below diagram

III. TCP VARIANTS

TCP have lot of congestion control avoidance strategies, which are illustrated in the below diagram

S.No	TCP flavor	Feedback	Modification at	advantages	performance
1	BIC	Loss	Sender	High bandwidth	
2	New Reno	Loss			Delay
3	Vegas	Delay	Sender	Less Loss	Proportional
4	CUBIC	Loss	Sender	High bandwidth	
5	RED	Loss	Router	Smaller delay	

Table.3.1.TCP variants

3.1 TCP Reno

TCP Reno handles duplicate ACKs (Fast recovery and fast retransmit). Fast recovery avoids slow start. when time out happen it performs retransmit and slow start. TCP Reno is the enhanced version of TCP Tahoe during one packet is dropped in a single round-trip time.

Fast recovery will perform in several steps. After fast retransmit, fast recovery eliminates slow start.it performs intuition (“duplicate acknowledgements indicate that the data is getting through”). retransmit set to “packet lost” after three consecutive duplicate ack. Once the packet loss occurs cwnd, ssthresh values are updated. Ssthresh become cwnd/2 and cwnd becomes ssthresh to enter in the congestion avoidance.TCP Tahoe and Reno comparison for one pocket loss explained in the following fig.3.1.1 and fast recovery process illustrated in fig.3.1.2.

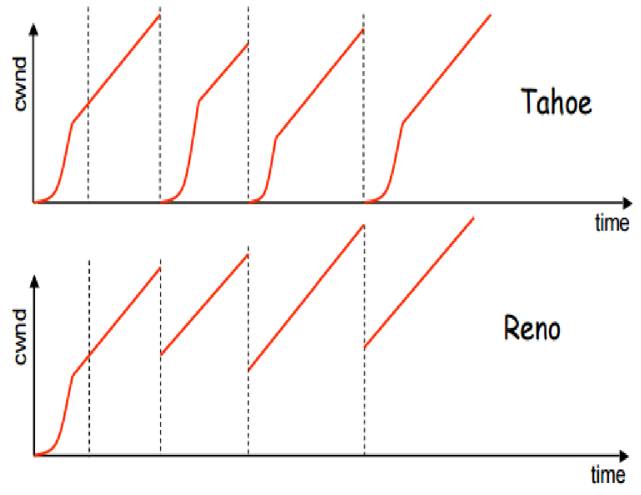

Fig.3.1.1 Comparison for one pocket loss

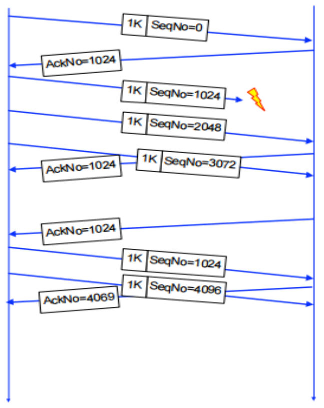

Fig.3.1.2. Sequence diagram for Fast Recovery

3.2 TCP New Reno

TCP New Reno is the enhancement for TCP fast recovery algorithm and which generally used in absence of SACK. The implementation of Fast recovery algorithm standardized in RFC 2581.The BSD Reno algorithm implemented in 1990 and referred as Reno algorithms in [FF96]. In Reno data retransmit only single data packet when retransmit timeout occurs. The problem arises only when the multiple data packets arise in a single window without SACK presence.

In absence of SACK TCP sender know the packet loss after three duplicate acknowledgement received and then retransmit the referred packet. The original fast transmit and recovery algorithms are standardized in RFC 2581.New Reno is the enhanced version of the original one modified in step 1 and 5 as follows

Step 1: set the ssthresh value based on the below formula (1) after third duplicate ACK received.

ssthresh=max(fsize /2, pow(2*MSS))----→ (1)

Step 2: retransmit last segment and make cwnd as ssthresh + (3*MSS)

Step 3: cwnd=cwnd+1 (for each redundant ACK)

Step 4: Advertising new window by receiver along with new value of cwnd, segment

Step 5: when ACK for all data set cwnd=min (ssthresh, fsize + MSS) or ssthresh (calculated in step1)

Step 6: Noted send high value after retransmit timeout.

New Reno have small variation with Reno in absence of SACK.

3.3 TCP Sack

Sack (selective acknowledgement) is standardized in RFC 2018 and is well extended in RFC 2883.Sack used at data receiver side, which acknowledge the non-contiguous segments of data. Sack does not receipt for duplicate segments of data where D-Sack [9] (Duplicate sack) notify the duplicate segments. TCP header has sack option which contains block of data. Each block of data receiver has two edges namely left and right. Left edge have initiated sequence number where as right edge has following next sequence number of the last edged-sack specify the duplicate packet sequence number which fires the acknowledgement.

D-SACK operational steps: -

1. for most recent packet-sack notes the contiguous data sequence number for receiver
2. each D-sack block must have one contiguous data packet (i.e. if duplicate it lodes in the subsequent blocks)
3. left edge have initial sequence number and followed number loaded by right edge.
4. if duplicate block of receiver data notify second edge data block should notify each duplicate segment noted in only one ack packet and no longer existed when the packet is dropped.

D-sack algorithm address various issues, which someone listed below

- duplicate block
- out-of-order block
- out-of-order and duplicate block
- duplicate of out-of-order block
- partial duplicate block
- only one duplicate block
- Two-duplicate blocks with cumulative ack.
- Two-duplicate blocks without cumulative ack.

PAWS "Protection Against Wrapped Sequence Numbers" explained in RFC 1323[10]. It helps D-SACK to identify the duplicate of sequence numbers. The comparison of SACK with their protocols addresses in [11] and extension of the protocols expressed in [12].

3.4 TCP Vegas

TCP Vegas is the upgraded version and address specific issues of the TCP Reno. It is more suitable for proactive approaches not for reactive scenarios. It is differing in three cases where rest of the functionality similar to Reno. Vegas is a modified version of Reno in terms of re-transmission, congestion avoidance and enhanced flavor of slow-start scenario.

TCP Vegas have list of new algorithms which are listed below:

i. Detection of Congestion during slow-start.
ii. Detection of Congestion during avoidance of congestion.
iii. Quick transmit mechanism
iv. Acknowledgement for non-Ack packets by maintaining RTT time stamp.
v. Prevention of multiple reduction for congestion window.
vi. Reduction of 1/4th window size after recovery

IV. TOPOLOGY CONFIGURATION & SIMULATIONS RESULTS

To estimate the performance of the above stated protocols we configured topology in three various scenarios. The parameters for the three scenarios are configured same and changed in each scenario to get the optimal throughput of the network. All the three scenarios evaluate the performance in terms of throughput versus with bandwidth, simulation time, no. of the sources. The band width parameter specified in between 0.1Mb to 10Mb, Simulation time ranges between 5ms to 200ms, No. of traffic sources used between from 2 to 35.

In first scenario we compared TCP-RQB with TCP-Reno to get optimal throughput (versus with bit error rate, Bandwidth), Packet loss rate (versus with bit error rate, bandwidth). The simulation results are visualized in the following Fig.4.1, Fig.4.2, Fig.4.3, Fig.4.4.

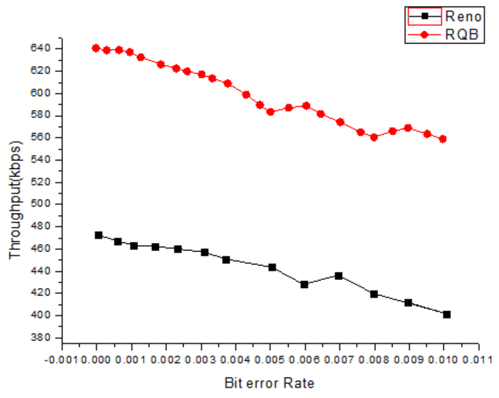

Fig.4.1. Throughput vs Bit error rate

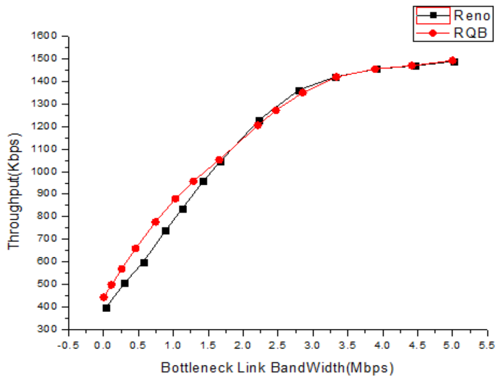

Fig.4.2 Throughput vs Bandwidth

When we changed bandwidth in between the two nodes Through put value is linearly increased and from the Fig.4.5 we analyzed that Vegas have better throughput other than three protocols.

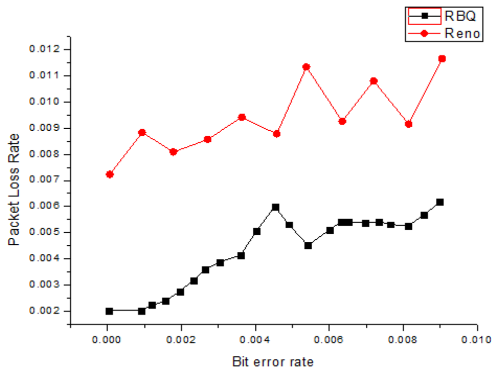

Fig.4.3. Packet Loss rate vs Bit error rate

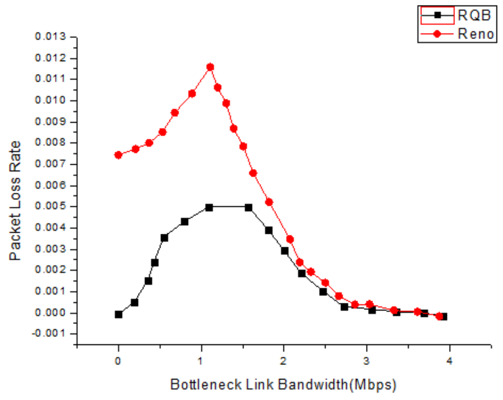

Fig.4.4 packet loss rate vs Bandwidth

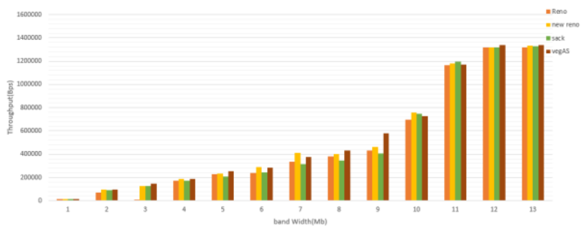

Fig.4.5. Throughput vs bandwidth for Reno, New Reno, SACK, Vegas

In terms of Simulation time all four protocol performance not satisfactory when simulation time is high, but Vegas performed good when the simulation time is less than the value 60ms.The performance illustrated in the below fig.4.6. In this scenario we analyzed the average Throughput value for all the protocols.

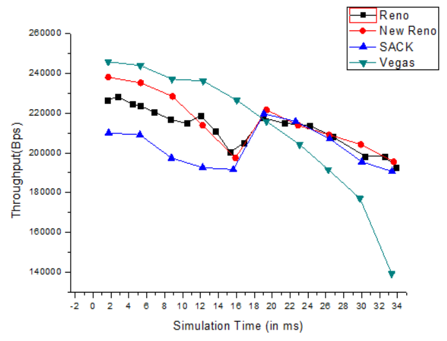

Fig.4.6. Throughput vs Simulation Time

Vegas performed well when No. of traffic resources increased. Vegas, Reno, New Reno constantly incremented when the no. of traffic sources gradually increased. Sack performed very poor due to high packet losses in the same window, which illustrated in the below Fig.4.7.

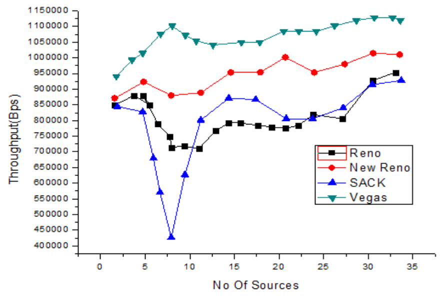

Fig.4.7. Throughput vs number of traffic sources

V. CONCLUSION

As a conclusion, TCP-RQB Agent Algorithm is must better than the customary TCP Reno Protocol. In New Reno, identify of various packet losses are accessible, so it is valuable when there are more packet losses of information. At the point when hearty is basic to various packet losses in one window of information, and loss of information is costly, Sack is reasonable from different Protocols. Vegas is prescribed in situations where substantial transmission capacity is required because of now is the ideal time based data transmission estimation calculation.

REFERENCES

[1] TCP and Its working functionality explained under
  a. https://www.rfc-editor.org/rfc/pdfrfc/rfc793.txt.pdf

[2] Congestion control over IP/TCP explained under
  a. https://www.rfc-editor.org/rfc/pdfrfc/rfc896.txt.pdf

[3] TCP Congestion control algorithms are explained under
  a. https://www.rfc-editor.org/rfc/pdfrfc/rfc5681.txt.pdf

[4] J. Postel, DOD standard transmission control protocol. Marina del Rey, Calif.: Information Sciences Institute, University of Southern California, 1980.

[5] J. Postel, DoD Standard Internet Protocol. Ft. Belvoir: Defense Technical Information Center, 1980.

[6] "Congestion avoidance and control", V. Jacobson, ACM SIGCOMM Computer Communication Review, no. 4, vol. 18, pp, 1988, 314-329.

[7] M. Welzl, Network congestion control. Chichester, West Sussex, England: J. Wiley, 2005.

[8] [S. Floyd, The New Reno Modification to TCP's Fast Recovery Algorithm. International Computer Science Institute, 2004, pp. 1-18.

[9] SACK explained in detail in the following RFC
  a. https://www.rfc-editor.org/rfc/pdfrfc/rfc2883.txt.pdf

[10] PAWS explained in detail in the following RFC
  a. https://www.rfc-editor.org/rfc/pdfrfc/rfc1323.txt.pdf

[11] "Simulation-based comparisons of Tahoe, Reno and SACK TCP", K. Fall and S. Floyd, ACM SIGCOMM Computer Communication Review, no. 3, vol. 26, pp, , 1996, 5-21.

[12] An Improved TCP Congestion Control Algorithm Over Mixed Wired/Wireless Networks,2009, IEEE.