Performance Analyses of TCP Westwood

[1]Vasudev I Kanani, [2]Mr.Krunal J Panchal

Department Of Computer Engineering
L.J. Institute of Engineering & Technology, Ahmedabad-382210, Gujarat, India
vasudev.kanani@gmail.com

Abstract **- Transmission Control Protocol congestion control is backbone of the Internet stability. TCP Westwood is sender-side modification of the congestion window algorithm that improves upon the performance of Reno in wired as well as wireless networks. Westwood continuously measure the bandwidth at the TCP sender side via monitoring the rate of returning ACKs. This estimate is then used to compute congestion window and slow start threshold when congestion occurs, after three duplicate acknowledgments or after a timeout. In contrast with TCP Reno which "blindly" halves the congestion window after three duplicate ACKs, TCP Westwood attempts to select a slow start threshold and a congestion window which are consistent with the effective bandwidth used at the time congestion is experienced. We call this mechanism faster recovery. The proposed mechanism is particularly effective over wireless links where sporadic losses due to radio channel problems are often misinterpreted as a symptom of congestion by Current TCP schemes and thus lead to an unnecessary window reduction. During the last decade, several congestion control mechanisms have been proposed to improve TCP congestion control. Compared to other algorithms shows that TCP westwood is better than other algorithms.**

Keyword **- TCP, Congestion Control Mechanisms**

I. INTRODUCTION

Since Internet is widely used, TCP is the most important protocol in the open system interconnection (OSI) architecture. TCP is a fourth layer transport protocol of OSI that uses the basic IP services to provide applications with an end-to-end communication and connection oriented protocol that ensures the reliable and ordered delivery of data. TCP was primarily designed for only wired networks, so TCP should be enhanced for wireless network, especially the congestion control. TCP Reno is most widely adopted TCP algorithm. It has good throughput in wired networks, but it is worse one in wireless networks - with TCP Reno, the wireless loss is processed as network congestion. The issue of TCP in wireless is how to distinguish the network congestion and wireless loss because wireless loss is not network congestion, so TCP doesn't have to process wireless loss as network congestion. To enhance TCP for wireless networks, TCP should need the clear distinction between network congestion and wireless loss. However, for distinction wireless loss and network congestion, there are many methods to be implemented to TCP, but it can't be deployed; only TCP mechanism can't distinguish wireless loss and network congestion. The way to control network congestion and wireless loss is to use TCP Westwood. We talk about TCP Westwood and performance of tcp westwood for wired and wireless network.

II. BACKGROUND AND MOTIVATION

TCP and IP are most important protocol in OSI refrence model. IP is connectionless network layer protocol which doesn't guarantee reliable and inorder delivery of packets. TCP is transport layer protocol that uses basic IP service to provide applications with end to end and connection-oriented mechanism that ensures reliable and ordered delivery of data.

Network congestion occurs when many hosts try to send data at high rate. And due to queue delay occurs in network and sender retransmit data in order to compenstate lost packet. So, average transmission capacity of routers are reduced. TCP controls its transmission rate by reducing its unacknowledged segments called TCP windows size. TCP congestion control mechanism is based on dynamic window adjustment. In a TCP congestion control mechanism connection is starts with slow start phase and congestion window size is doubled every round trip time until window size reaches slow start threshold. After threshold window size is increased by one packet per round trip time. If packet loss occur then threshold size is set to half of current size and window size is set to one segment. congestion avoidance mechanism maintains low delay and high throughput at sender side in network. TCP reno is most adopted TCP schemes and it has good throughput in wired network but performance of reno is worse in wireless network. TCP reno doesn't distinguish network congestion and wireless loss. Wireless loss is not a network congestion so, TCP does not have to process wireless loss as network congestion. Only TCP mechanism can not distinguish network congestion and wireless loss. So, for this TCP westwood is deployed which controls network congestion and wireless loss effectively. In this paper we see mechanism of TCP Westwood and performance of westwood compared to other congestion control algorithms.

2.1. A Transmission Control Protocol and Congestion Control

TCP congestion control mechanism was introduced by Van Jacobson in late 1980's. The internet was suffering from congestion collapse b'coz sender send their packets into internet as fast as advertised window allow and congestion occur at

router cause packet drop. So, sender would time out and retransmit packet, this gives more congestion in internet. TCP uses acknowledgment for transmission of new packet.

The available bandwidth changes over time so windows size must be changed over time. The algorithms used by TCP and problems. Basic algorithms are slow start algorithm, congestion avoidance, fast retransmit and fast recovery.

2.2. Slow Start Algorithm

In a slow start algorithm operates by observing that new packets should be injected in network based on rate of returned acknowledgement. Slow start uses congestion window. When new connection is established congestion window size is set to one segment. Congestion window is increased by one when acknowledgement is received. The sender starts to send by transmitting one segment and then wait for its ACK. When that ACK is received, the congestion window is incremented to two, and two segments can be sent. When two segments are acknowledged, then congestion window is increased to four. This gives an exponential growth, although it is not exactly exponential because the receiver may delay its ACKs, typically sending one ACK for every two segments that it receives. And when capacity of internet is reached then intermediate router discards packets. This tells sender that congestion window size is to large.

Slow start algorithm:
Initialize: cwnd = 1
For (each segment ACKed)
cwnd ++;
Until (congestion event or cwnd >ssthresh)[1]

Congestion Avoidance (CA)

Loss of packet by noise is very small, therefore loss of packet is somewhere in the network between source and destination. For packet loss thereare two indication one is receipt of duplicate acknowledgement and second is timeout occurring. When congestion occur in network then TCP must reduce transmission rate of packets in to network and invoke slow start mechanism. For congestion avoidance and slowstart congestion window and slow start threshold are required. Congestion Avoidance (CA) handles congestion by lowering the congestion window size increase to only 1 packet per RTT(Round Trip Time), giving cwnd a lower and linear growth. If Retransmit Time Out (RTO) occurs, CA will consider this as a loss of packet. CA will then set ssthresh to half the current cwnd and after this resets cwnd to one and initiate a SS(slow start).

/* slowstart is over */
/* if cwnd > ssthresh */
every new ACK:
cwnd += 1/cwnd
Until (timeout) /* loss event */ [1]

2.3 Fast Retransmit (TCP Tahoe)

Fast retransmit algorithm treating three duplicate acknowledgement as packet loss. Purpose of three duplicate acknowledgement is to tell sender that segment was received out of order and tell what sequence number is actually required. TCP doesn't know about that duplicate acknowledgement is caused by just reordering of segment so, it wait for small no. of duplicate aknowledgements. If three or more acknowledgements are received than that is indication of packet loss so retransmission of segment is occur without waiting for timeout. When three DUPK are occur than slowstartthreshold is set to ½ of congestion window and enter into slowstart.

If receiving 3DUPACK or RTO
Retransmit the packet
ssthresh = cwnd /2
cwnd = 1
perform slowstart [1].

2.4 Fast Recovery (TCP Reno)

After fast retransmit, fast recovery algorithm is immediately occur and sends missing segment without perform slowstart. This algorithm allow high throughput under moderate congestion. In this not performing slowstart is that after receiving duplicate ACKs tells TCP that segment is just lost and there is no congestion. Receiver only generate duplicate ACK when another segment is received, that segment left network and is in buffer. Data is still flowing between sender and receiver so, TCP doesn't reduce flow of window by going into slowstart.

If receiving 3DUPACK or RTO Retransmit the packet.

After retransmission do not enter fast recovery
Fast recovery algorithm (Reno)
ssthresh = cwnd/2 ;
cwnd = ssthresh + 3 ;
Each duplicate ACK received
cwnd+ + ;
If new ACK cwnd=threshold ;

return to congestion avoidance

3. TCP WESTWOOD MECHANISM

TCP Westwood congestion control algorithm use a bandwidth estimation, it executed at sender side of a TCP connection. The congestion window dynamics during slow start and congestion avoidance are unchanged. The general idea is to use the bandwidth estimate BWE to set the congestion window (cwin) and the slow start threshold (ssthresh) after a congestion episode. In TCP Westwood the sender continuously computes the connection BWE which is defined as the share bottleneck used by the connection. Thus, BWE is equal to the rate at which data is delivered to the TCP receiver. The estimate is based on the rate at which ACKs are received and on their payload. After a packet loss, the sender resets the congestion window and the slow start. Threshold based on BWE. The packet loss is suspected with a reception of three duplicates ACKs or timeout expiration. Another important element of this procedure is the RTT estimation. That is because the congestion window is set precisely to BWE * RTT after indication of packet loss.

3.1 End-to-End Bandwidth measurement

The key idea of the TCP Westwood, presented before, is to continuously estimate, at the TCP sender, the packet of the connection. This is done by monitoring the ACK reception rate. The estimated connection rate is then used to improve the efficiency of slow start and congestion control algorithms. If an ACK is received at source at time t2, this implies that a corresponding amount of data d2 has been received by the TCP receiver. Therefore, we can measure the following sample of bandwidth used by that connection as: b2 = d2/(t2-t1), where t1 is the time of the previous ACK that was received. An average of the samples is calculated and used to calculate the estimation of the available bandwidth. This bandwidth estimation works in the following way:

Bandwidth estimation (BWE) algorithm BWE

$b_k = \alpha_k b_{k-1} + (1 - \alpha_k)[(b_k + b_{k-1})/2]$

Where b_k = sample bandwidth = $d_k / t_k - t_{k-1}$ where d_k = amount of bytes acknowledged by ACK k, t_k = arrival time of ACK k, $\alpha_k = [2\tau - \Delta(t_k - t_{k-1})] / [2\tau + \Delta(t_k - t_{k-1})]$ where τ : is the cut- off frequency of this Tustin filter

3.2 Round-Trip Time Estimation

When a host transmits a CP packet to its peer, it must wait a period of time for an acknowledgment. If the reply does not come within the expected period, the packet is assumed to have been lost and the data is retransmitted. The problem is the protocol does not define the length of the period to wait. All modern TCP implementations seek to find a proper waiting time by monitoring the normal exchange of data packets and developing an estimate of how long is "too long". This process is called Round-Trip Time (RTT) estimation. RTT estimates are one of the most important performance parameters in a TCP exchange, especially when you consider that on an indefinitely large transfer, all TCP variants eventually drop packets and retransmit them, no matter how good the quality of the link. RTT is a key component of TCP Westwood algorithm.

3.3 Setting cwnd and ssthresh in TCPW

Let us first assume that a sender has estimated BW, and let us describe in this subsection how is used to properly set cwnd and ssthresh after a packet loss indication. First, we note that in TCPW, congestion window increments during slow start and congestion avoidance remain the same as in Reno, i.e. exponential and linear, respectively. A packet loss is indicated by the follows: (a) the receipt of 3 DUPACKs, or (b) a coarse timeout expiration. In case the loss indication is 3 DUPACKs, TCPW sets cwnd and ssthresh as follows.

3.4 Algorithm after three duplicate ACK.

The pseudo code of the TCP Westwood algorithm after three duplicate acknowledgements is:

```
After 3 DUPACKS
If receiving 3 DUPACKS
Set ssthresh =(BWE*RTTmin) /seg_size;
and if cwnd > ssthresh then set cwnd = ssthresh ;
enter congestion.
```

In the pseudo-code, seg_size indicates the length of TCP segments in bits. During the congestion avoidance phase the sender is trying for extra available bandwidth. If three duplicate ACKs are received, the network capacity might have been reached or that in case of wireless links, one or more segments have were dropped due to sporadic losses.

3.5 Algorithm after timeout

The pseudo code of TCP Westwood algorithm after timeout is:

```
After Timeout
 If RTO then set
ssthresh = (BWE*RTTmin) /seg_size;
if (ssthresh < 2) ssthresh =2; end if ;
cwin = 1;
end if
enter slow start;
```

The rationale of the algorithm above is that after a timeout, cwin is set to equal one and ssthresh is set BWE. A speedy recovery is ensured by setting ssthresh to the bandwidth estimation at the time of timeout expiration.

Modified westwood[1]

This modification is done by Shimaa Hagag and Ayman El-Sayed in 2012. They changed congestion avoidance mechanism and Retransmission TimeOut mechanism of TCP Westwood algorithm.

Congestion avoidance
slow start is over */
/*cwnd > ssthresh */
Every Ack:
Estimate BWE
Set BWE = BWcurrent
BWratio = BWcurrent/BWprevious
If (1.5>BWratio >= 1)
cwnd = cwnd + 1/cwnd
If (BWratio >= 1.5)
cwnd = cwnd + 2/cwnd
Else if (BWratio < 1)
cwnd = cwnd + 0
Until (timeout or 3 DUPACKs)

Modified RTO Calculation Algorithm[1]
RTO new=(RTT new/RTT old)

where RTTnew is the new round trip time estimation after congestion recovery and RTTold the round trip time estimation before congestion.

4. PERFORMANCE

4.1 Simulation Results Comparison [1]

Throughput refers to the packets sent by the sender and correctly received by receiver. Delay, is the required time of two-way communication, it may range from a very few microseconds, it can be measured as per packet transfer times. Packet losses (packet drop rates), measured as the number of dropping packets per unit of time, it is also defined as the packets that are retransmitted again from the source. Congestion window is flow control imposed by the sender, while the advertised window is flow control imposed by the receiver.

4.2 Simulation setup

The simulations are often used for understanding and prediction of the behavior of protocols and data streams in networks. All simulation results in this paper are obtained from paper1. Figure 2 shows the network topology. The topology has five nodes connected to each other via four TCP connections; each link is labeled with its bandwidth capacity and its delay.

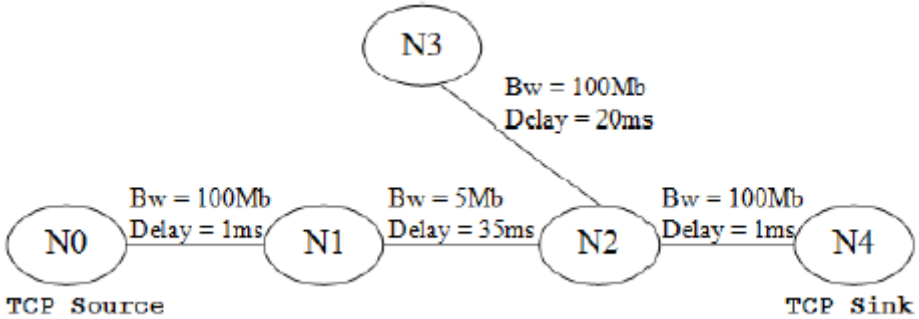

Fig 2: Network topology [1]

4.3 Simulation Results by Shimaa Hagag and Ayman El-Sayed

In Figure 3, the network throughput is presented for TCP Reno, Newreno, Sack, Vegas, Tahoe, Westwood, and Westwoodnew. It is noted that TCP westwoodnew modified algorithm by Shimaa Hagag and Ayman El-Sayed has highest throughput on the steady state time. It determines the network status by estimating the current BW without looking to the previous BW which determines the previous network status, therefore the rate of the sending packets in TCP Westwood and base on that TCP Westwood sending the same rate of packets whether the network status is heavy or not heavy by TCP WestwoodNew determine the rate of packets sending based on the network status if it not heavy TCP Westwood new increase the rate of the

sending packets which is increase the throughput performance in the network if network status is heavy TCP WestwoodNew remain the rate of sending packets constant.

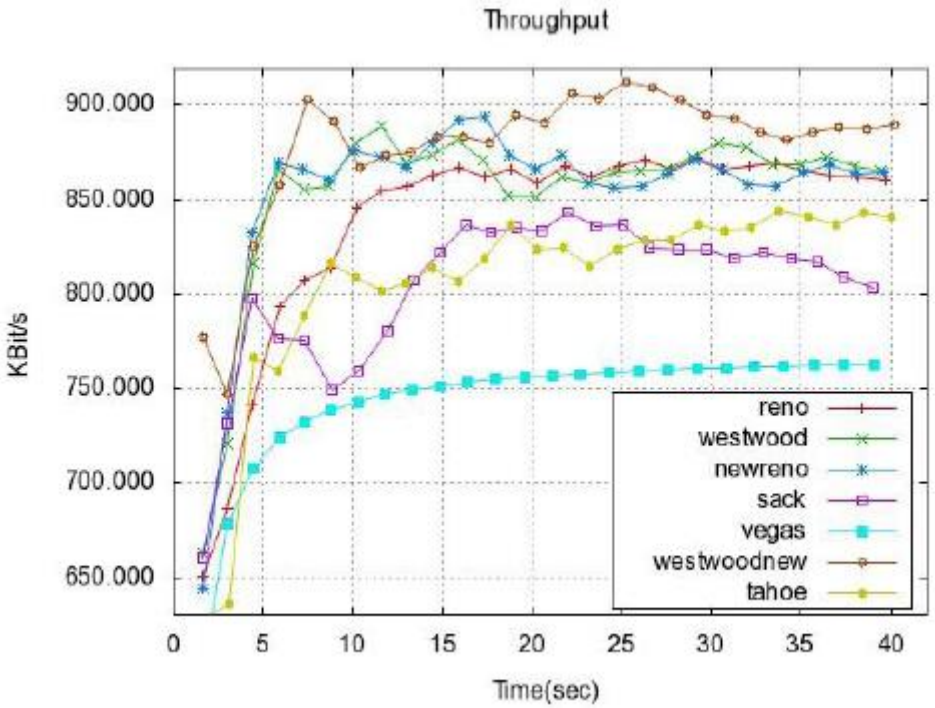

Fig 3 Network Throughput

5. CONCLUSION

From this analysis of TCPWestwood congestion control mechanism we conclude that Westwood performs wells over wireless network and give better throughput compare to other congestion control mechanisms. Enhanced TCPWestwood perform wells and give improved throughput than current TCPWestwood by modifying congestion avoidance algorithm. Performance of TCPWestwood can also improve by modifying slowstart, congestion avoidance algorithm, bandwidth estimation technique.

REFERENCES

[1] Shimaa Hagag and Ayman El-Sayed "Enhanced TCP Westwood Congestion Avoidance Mechanism (TCP WestwoodNew)" International Journal of Computer Applications (0975 – 8887) Volume 45– No.5, May 2012

[2] Ehab A. Khalil "SIMULATION-BASED COMPARISONS OF TCP CONGESTION CONTROL" International Journal of Advances in Engineering & Technology, Sept 2012

[3] M. Kalpana1 and T. Purusothaman, "Performance Evaluation of Exponential TCP/IP Congestion Control Algorithm", International Journal of Computer Science and Network Security (IJCSNS), VOL.9 No.3, March 2009.

[4] Ka-Cheong Leung, Victor O.K. Li, Daiqin Yang, "An Overview of Packet Reordering in Transmission Control Protocol (TCP): Problems, Solutions, and Challenges," IEEE Transactions on Parallel and Distributed Systems, pp. 522-535, April, 2007

[5] Inwhee Joe, Jaehyung LeeAn "Enhanced TCP Protocol for Wired/Wireless Networks" Fifth International Joint Conference on INC, IMS and IDC, 978-0-7695-3769-6/09 © 2009 IEEE DOI 10.1109/NCM.2009.253

[6] Mario Gerla, M. Y. Sanadidi, Ren Wang, and Andrea Zanella, Claudio Casetti, Saverio Mascolo "TCP Westwood: Congestion Window Control Using Bandwidth Estimation" 0-7803-7206-9/01/$17.00 © 2001 IEEE

[7] Claudio Casetti , Mario Gerla, Saverio Mascolo, M.Y. Sanadidi And Ren Wang "TCPWestwood: End-to-End Congestion Control for Wired/Wireless Networks Wireless Networks" Kluwer Academic Publishers. Manufactured in The Netherlands, 2002.

[8] P. Kuusela, P. Lassila, J. Virtamo and P. Key, "Modeling RED with Idealized TCP Sources", 9th IFIP Conference on Performance Modeling and evaluation of ATM & IP networks, 2001.

[9] S. Ryu, C. Rump, and C. Qiao, "Advances In Internet Congestion Control", IEEE Communications Surveys & Tutorials, Third Quarter, vol.5(1), 2003, pp:28-39.

[10] C. Wanxiang, S. Peixin, and L. Zhenming, "Network-assisted congestion control", Info-tech&Info-net International Conferences, vol.2, JUN 2001, pp:28-32.

[11] K Fang-Chun., and X. Fu, "Probe-Aided MulTCP: an aggregate congestion control mechanism", ACM SIGCOMM Computer Communication Review, Vol.38 (1), JAN 2008, PP: 17-28.