International Journal of Computer Architecture and Mobility
(ISSN 2319-9229) Volume 1-Issue 8, June 2013

New TCP Corruption Control Mechanism For Wireless Network

Nitin K. Mishra[#1], Gaurav Shrivastava[*2]

nmishra1708@gmail.com, gashr83@gmail.com

***Abstract*— TCP Congestion control is end to end control mechanism. Which is implemented through controlling some important parameter change comparing to wire networks, there are many different characteristics in wireless environments. In this paper new mechanism for TCP corruption control is presented. It considers the influences to TCP sender's packet sending rate not only by the congestion but also by the corruption. The sending window will be calculated after each transmission, according to number of corrupted packet. So, there is less packet drop occurring in Transmission. The comparative study of New TCP (NTCP) with other TCP variants is also presented with variation in speed of node, pause time and number of nodes in network. The New Mechanism can be easily implemented with fewer overheads and can effectively improve reliability with small variances of throughput and delay. Implementation and Simulation for this mechanism is performed in QualNet 5.0.1 Simulator.**

***Keywords*- Wireless Network, Congestion Control, Corruption Control, TCP**

I. Introduction

Transmission Control Protocol (TCP)[1,2], is the predominant Internet protocol and carries approximately 90% of Internet traffic in today's heterogeneous wireless and wired networks. TCP is widely used as a connection oriented transport layer protocol that provides reliable packet delivery over unreliable links. TCP does not depend on the underlying network layers and, hence, design of various TCP variants is based on the properties of wired networks. However, TCP congestion control algorithms may not perform well in heterogeneous networks. Wireless networks have higher bit error rates due to weather conditions, obstacles, and multipath Interferences, mobility Page Layout of wireless end-devices, and signal attenuation and fading, which may lead to packet loss [13]. Various TCP algorithms [14] and techniques have been proposed to improve congestion and reduce the non-congestion related packet loss. TCP Tahoe, TCP Reno, TCP New Reno with Selective Acknowledgement (SACK), TCP New Reno, TCP Vegas, and TCP Binary Increase Congestion (BIC) are examples of proposed end to end solutions. Snoop-TCP is a link layer control approach while N-TCP split connection approaches. They are all proposed to improve network Performance. The end to end techniques are the most promising because they require changes only to the end systems rather than to the intermediate nodes. These end-to-end control approaches are used in today's deployed networks.

II.Description of TCP variants

Four kinds of core algorithms of TCP congestion control algorithms[14] are executed at the source end and they belong to a source algorithms in terms of realization of the position .These algorithms are to achieve the purpose of congestion control through using and adjusting some parameter s ,of which the main parameters are :congestion(CWND),window slow start threshold (ssthresh),delay(RTT),and over time retransmission counter(RTO),etc.

II.1 Improved TCP Congestion Control Algorithm

II.I.I TCP Reno

This Reno [10] 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 receive 3 duplicate ACK's we take it as a sign that the segment was lost, so we retransmitted segment without waiting for timeout.
The basic algorithm is presented as under:

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

Problems

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 (Round-Trip Times). Another problem is that if the widow 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.

II.I.II TCP Lite

TCP Lite[8] is a service that provides a transport method that interrupts TCP in order to reduce the overhead involved in session management in which no data is transmitted or received. TCP Lite reduces or eliminates pure TCP protocol data units used in the setup and ACK while maintaining order, integrity, reliability and security of traditional TCP. TCP lite uses large window and protection against wrapped sequence numbers.

Problems

Lite performs over TCP same as Reno. But when window increases it have some problems to maintain them.

II.1.III TCP New Reno

New RENO[8] 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. Thus it overcomes the problem faced by Reno of reducing the CWND multiples times. The fast-transmit phase is the same as in Reno. The difference in the fast recovery phase which allows for multiple re-transmissions in new-Reno. Whenever new-Reno enters fast recovery it notes the maximums segment which is outstanding. 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 CWND to ssthresh and continues congestion avoidance like Tahoe. If the ACK is a partial ACK then it deduces that the next segment in line was lost and it retransmits 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.

Problems

New-Reno 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 reduce which other segment was lost. Round-trip time until all of the lost packets from that window has been retransmitted.

II.II Corruption Loss Rate

The corruption loss[13] rate is the packet loss rate due to corruption. We don't adjust the sending rate when the corruption loss rate is low, but we need decrease the sending rate rapidly to improve the reliability when the corruption loss rate becomes higher. Otherwise, there will be more lost packets due to corruption and more packets will be retransmission. And the transmission reliability will be decreased; the energy consumption of mobile hosts and the system overheads will be increased. What degree does the corruption loss rate reach need to decrease the sending rate? We introduce the definition of corruption loss rate Pe . Definition of Pe : In a period of time T, if TCP sender send n packets and m packets of them be discard because one or more bits error caused by wireless link corruption, the corruption loss rate define as

$$Pe = 2(m/n).$$

The corruption loss rate decides by the Bit Error Rate (BER) of wireless link layer and the length (Length) of data frame:

$$Pe = 1 - (1 - BER)\ Length.$$

If the corruption loss rate Pe is higher than the certain lower limit Pemin, the sending rate will be decreased. Many factors decide the value of corruption loss rate lower limit Pemin, mainly include: the kind of application; the length of data frame; the bit error rate of wireless link layer; the bandwidth and the transmission delay of wireless network etc. Normally, we choose Pemin=0.4.

III.Proposed Work

New TCP is the modification of TCP corruption control mechanism [13][14] for wireless network .NTCP not only considers the packet loss due to congestion but due to wireless link corruption. The minimum limit of corruption loss rate depends on various factors such as length of frame, bit error rate, type of application and transmission delay of wireless network. A minimum value of Pe is selected on the basis of these factor be Pemin=0.4.On the basis of comparison, packet sending rate is decreased by decreasing the current congestion window. After generating a new congestion window based on corruption loss rate and its old values, packet drop rate is calculated .if the packet drop rate exceeds a certain value ,the new congestion window is calculated halved.

A. Initial Window

The initial sending window is calculated by following formula:

Iw= min (4 * SMSS, Max (2 * SMSS, 4380 byte));

B. Slow-Start Algorithm

The slow start algorithm is used to start a connection of ETCP and the periods after the value of retransmission timer exceed the RTO (retransmission timeout). In the start of ETCP, the size of cwnd will be initialized to 1.

The slow start algorithm describes as below:

```
if (Receive ACKs && cwnd < ssthresh)
{
cwnd = cwnd++;
}
```

The slow start algorithm will be ended in two conditions. First, if the congestion window size reaches the slow start threshold size (ssthresh), the slow start will be ended and then congestion avoidance takes over. Second, if there lose any packet due to congestion or high packet loss rate due to corruption, the slow start also will be ended and then fast recovery takes over.

C. Congestion Avoidance Algorithm

If the congestion window size (cwnd) is less than or equal to the slow start threshold size (ssthresh), DW-TCP is in slow start; otherwise ETCP is performing congestion avoidance.

The congestion avoidance algorithm describes as below:

```
if (Receive ACKs || (Receive Explicit Corruption Loss
Notification && Corruption Loss Rate Pe<Pemin))
{
if (cwnd > ssthresh)
cwnd = 1+1/cwnd
else
cwnd++;
if (Receive Explicit Corruption Loss Notification)
Pe=2 (m/n);
If (Pe < Pemin)
Pemin = Pe ;
temp=cwnd* Pemin;
cwnd=cwnd – temp;
}
```

In the algorithm, cwnd denotes the congestion window size; m denotes the total number of sending packets; n denotes the number of lost packets due to wireless link corruption; Pe denotes the corruption loss rate.

D. Fast Retransmission and Fast Recovery

If the network congestion or heavy corruption, the fast recovery algorithm will be taken. When the network congestion, set ssthresh to one-half the flight size or double of SMSS (maximum segment size) window.

```
if (Congestion || Heavy Corruption)
{
if (Receive Same ACK 3 Times || Retransmission
Timer Overtime) /* Congestion */
{
Ssthresh = max(flightsize/2 , 2*SMSS);
// Flightsize are those data which have no acknowledged.
if (Retransmission Timer Overtime)
{
cwnd = 1; Exit and call slow-start;}
else /* Receive Same ACK 3 Time */
cwnd = ssthresh;
}
else if (Receive Explicit Corruption Loss Notification[8]
&& Corruption Loss Rate Pe>=Pemin)
{
temp=cwnd* Pemin;
cwnd=cwnd+ temp;
};
if (Receive Explicit Loss Corruption Notification)
Pe= 2(m/n);
If (Pe < Pemin)
Pemin = Pe ;
temp=cwnd* Pemin;
cwnd=cwnd – temp;
}
```

IV.Computer Simulation

In this paper all the simulation work is performed in QualNet wireless network simulator version 4.0 [15]. Initially number of nodes are 50, Simulation time was taken 200 seconds and seed as 1. All the scenarios have been designed in 1500m x 1500m area. Mobility model used is Random Way Point (RWP). In this model a mobile node is initially placed in a random location in the simulation area, and then moved in a randomly chosen direction between at a random speed between [SpeedMin, SpeedMax]. The movement proceeds for a specific amount of time or distance, and the process is repeated a predetermined number of times. We choose Min speed = 5 m/s, Max speed = 30m/s, and pause time = 5s to 30s. All the simulation work was carried out using TCP variants (Reno, Lite, Tahoe) with DSR routing protocol .Network traffic is provided by using File Transfer Protocol (FTP) application. File Transfer Protocol (FTP) represents the File Transfer Protocol server and client. In wireless network which we have used have following values for different parameter:

A. Mobility model Random Way Point

Minimum speed: 0 mps
Maximum speed: 5mps, 10 mps, 15 mps, 20mps, 25 mps, and 30 mps
Pause time: 5s, 10s, 15s, 20s, 25s, 30s.
Simulation Time: 200s

B. Terrain
Coordination: 1500 * 1500 m

C. Connection
FTP (File transfer protocol) : 41 (client) to 1(server)
Item size 512(byte)

D. Radio/physical layer parameters:
Radio type**:** 802.11b Radio
Data rate: 2Mbps
Packet reception model: Bit error rate (bpsk.ber)

E. MAC Protocol
802.11

F. Routing Protocol
DSR (Dynamic Source Routing)

G. Transport Protocol
TCP Tahoe, TCP Reno, TCP Lite, TCP ETCP.

H. Number of Node
50

I. Node Placement
Random

G. Seed
1

V. Simulation Methodology

Parameter	Scenario1	Scenario2	Scenario3
Simulation Time	Constant	constant	constant
Node	constant	constant	change
Area	constant	constant	constant
Pause Time	constant	change	constant
TCP Protocol	change	change	change
Routing Protocol	constant	constant	constant
Node Speed (m/sec)	change	constant	constant

A. Performance metrics used for this works are as follows:

1. *Throughput* is the measure of the number of packets successfully transmitted to their final destination per unit time. It is the ratio between the numbers of sent packets vs. received packets.

2. *Signal Received with error* is the measure of signal received, but they have error. The error may be occurring due to noise or due to heavy traffic.

3. *Bytes received* are the measure of total packet received by server. The packets may be drop due to heavy traffic. So received packets may be vary according to traffic conditions.

4. *Packet loss* is the measure of total discarded packet due to corruption or due to packet drop. We can calculate it by subtracting total received packets by server in total sent packet by client.

VI. Results and Analysis

In experimental process variation in one of the parameter including number of node pause time was done each time and their effect on performance of different TCP protocols was determined while rest of the parameters kept constant effect of simulation studies on performance of NTCP and other TCP protocols under experimental condition mentioned above are represented graphically as follows:

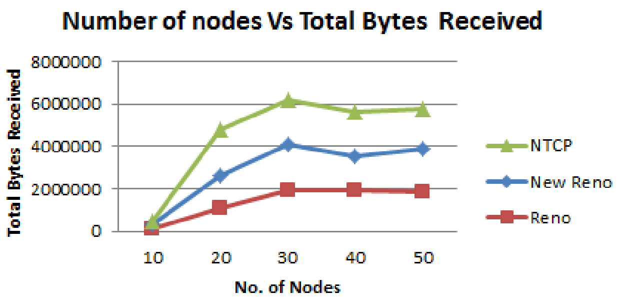

Fig 1

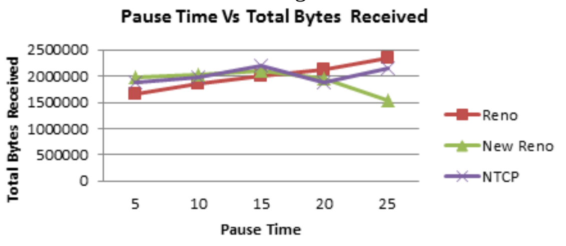

Fig 2

Fig 1 & 2 shows total bytes received of different variants with pause time and number of nodes in network. It is observed that NTCP has less loss than other TCP variants, because of optimal routing path between sender and receiver .Increment in then no. of nodes causes congestion consequently signal were distorted.

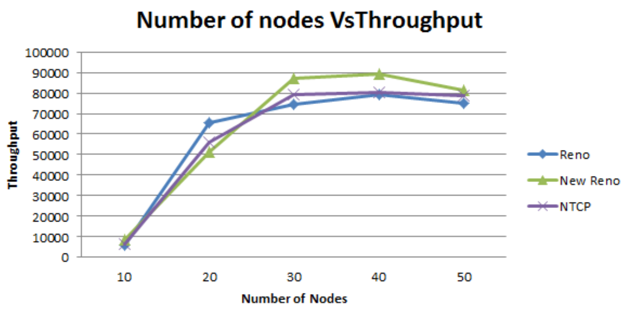

Fig 3

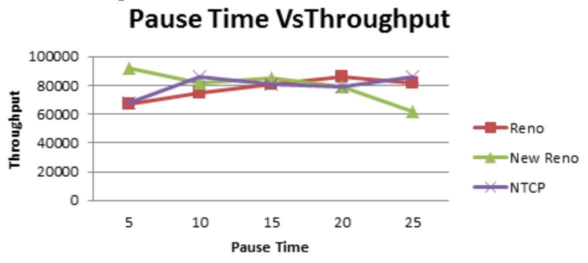

Fig 4

Fig 3 & 4 shows throughput of different TCP variants with pause time and number of nodes .Throughput is the ratio of number of sent packets with number of received packets. It is also observed that throughput of NTCP is better than other TCP variants .Here receiver receives maximum packets and only fewer packets were discarded ,so we can conclude that here in NTCP throughput is better than other TCP variants.

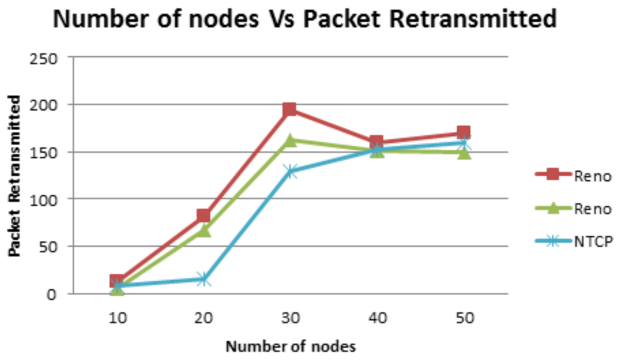

Fig 5

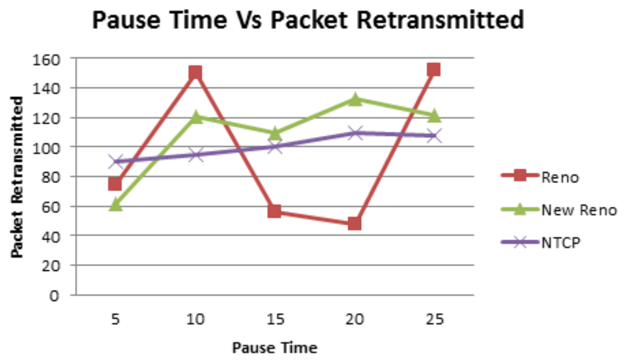

Fig 6

Fig 5 & 6 shows Packet Retransmitted of different variants with pause time and number of nodes in network. It is observed that NTCP has less retransmit than other variants because of due less packet loss.

VII. Conclusion

We proposed New TCP (NTCP) with a new calculation for sending window and implement ted it in a Mobile Ad-Hoc Network. Extensive simulation studies were undertaken to compare its performance with other standards TCP Reno, TCP New Reno and TCP Sack etc. over Wireless Network. The simulation results show that the performance of NTCP is better than other TCP variants. From result we conclude that the performance of NTCP is better in high density node because in this condition sender can get different paths through different nodes. Thus we conclude that, receiver can receive maximum data when we increase number of nodes in network. The existing TCP variants and its applicable algorithm are analyzed and describe about the protocol which one is better and suitable for packet and link utilization in the network congestion because the traditional TCP treat all packet losses due to the congestion, it does not treat from the link failure. The most of protocol shows better uses and many of them shows poor responsiveness to changing network conditions and network utilization.

References

[1] C. Perkins, Ad Hoc Networking, Addison-Wesley 2008, ISBN 0201309769

[2] C K Toh, Ad Hoc Mobile Wireless Networks, Prentice Hall Publishers ,2005.

[3] T.P. Pedersen, ―A Threshold Cryptosystem without a Trusted Party‖, In Proc. Of Eurocrypt'91, Lecture Notes in Computer Science, LNCS 547, Springer Verlag, pp.522-526, 1991

[4] Sanjeev Sharma ,Mukesh Kumar Dhariwal ―An Improved Mechanism for Congestion Control in TCP for Ad Hoc Network‖ International Journal of Computer Applications (0975 – 8887) Volume 20– No.2, April 2011.

[5] R. Vinod Kumar, Dr.R.S.D.Wahida Banu ―A New Approach For Load-balancing In EEAODV Protocol‖ IEEE-International Conference On Advances In Engineering, Science And Management (lCAESM -2012) March 30, 31, 2012

[6] Vasilios A. Siris_ and Despina Triantafyllidou ―Seamless Congestion Control over Wired and Wireless IEEE 802.11 Networks ―NETWORKING 2004, LNCS 3042, pp. 1470–1475, 2004.IFIP International Federation for Information Processing 2004.

[7] K.Srinivasa Rao, R.Sudhistna Kumar, P. Venkatesh, R.V.Sivaram Naidu,and A.Ramesh ―Development of Energy Efficient and Reliable Congestion Control Protocol for Multicasting in Mobile Adhoc Networks compare with AODV Based on Receivers‖ International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9522 Vol. 2, Issue 2,Mar-Apr 2012, pp.531-534.

[8] Ms.Harshada Pingale,Ms. Ashwini Rakshe,Mr. S.A.Jain, D. Mr. S.R.Kokate ―A STUDY OF CONGESTION AWARE ADAPTIVE ROUTING PROTOCOLS IN MANET‖ International Journal of Advanced Technology & Engineering Research (IJATER), VOLUME 2, ISSUE 2, MAY 2012.

[9] Shital kumar Jain, Shrikant Kokate, Pranita Thakur, Shubhangi Takalkar ―A Study of Congestion Aware Adaptive Routing Protocols in MANET‖ Computer Engineering and Intelligent Systems ISSN 2222-1719 (Paper) ISSN 2222-2853 Vol 3, No.4, 2012

[10] Mr.S.A.Jain , Ms Aruna.A, Kadam ―A Study of Congestion Control Mechanism Using Link Failure Detection in MANET‖ International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9522 Vol. 2, Issue 1,Jan-Feb 2012, pp.1009-1012.

[11] Vishal Sharma , Amitoj Soni , Sodhi Gunjit Singh , Takshi Gupta ―AN EFFICIENT APPROACH FOR ROUTING IN WIRELESS AD HOC NETWORK USING ARTIFICIAL INTELLIGENCE‖ International Journal on Computer Science and Engineering (IJCSE), ISSN : 0975-3397 Vol. 4 No. 02 February 2012.

[12] Soundararajan, R.S. Bhuvaneswaran ―Adaptive Multi-Path Routing for Load Balancing in Mobile Ad Hoc Networks‖, Journal of Computer Science 8 (5): 548-555, 2012 ISSN 1549-3535 © 2012 Science Publications.

[13] Xu Chang-Biao, Long Ke-Ping, Yang Shi-Zhong. Corruption-based TCP rate adjustment in wireless networks In: Chinese Journal of Computers, 2002, 25(4): pp.438-444.

[14] V. Jacobson "Modified TCP Congestion Control and Avoidance Algorithms". Technical Report 30, Apr 1990.

[15] Qualnet Developers Guide-5.0.1