Misbah Siddiqui, Saba Khalid,Prof Rizwan Beg/ International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 www.ijera.com
Vol. 2, Issue 5, September- October 2012, pp.008-013

THE ANALYSIS OF VARIOUS TCP SUB-VERSIONS AND MECHANISM FOR CONGESTION AVOIDENCE

Misbah Siddiquie
Student Btech. CSE
Integral University
Lucknow, India

Saba Khalid
Lecturer CSE dept.
Integral University
Lucknow, India

Prof Rizwan Beg
Professor and HOD(CSE/IT)
Integral University
Lucknow, India

ABSTRACT

TCP, the most widely used protocol on Internet, has a major problem in that its congestion control does not allow flows to obtain full bandwidth on fast-long distance links. A Performance analysis of TCP-controlled long file transfers in a WLAN in infrastructure mode also with Comparison and Analysis of Congestion Window for HS-TCP, Full-TCP and TCP-Linux in Long Term Evolution System Model is available in the literature with one of the main assumptions being equal window size for all TCP connections. In this paper, we extend the analysis to TCP-controlled long file uploads and downloads with different TCP windows. Our approach is based on the semiMarkov process considered in [1] and [2], but with arbitrary window sizes. We present simulation results to show the accuracy of the analytical model.

KEYWORDS:-WLAN, ACCESS POINTS, INFRASTRUCTURE MODE, UPLOADING AND DOWNLOADING, TCP WINDOW FULLTCP, HSTCP, TCP-LINUX, CWND, LTE, NS-2.

1. INTRODUCTION

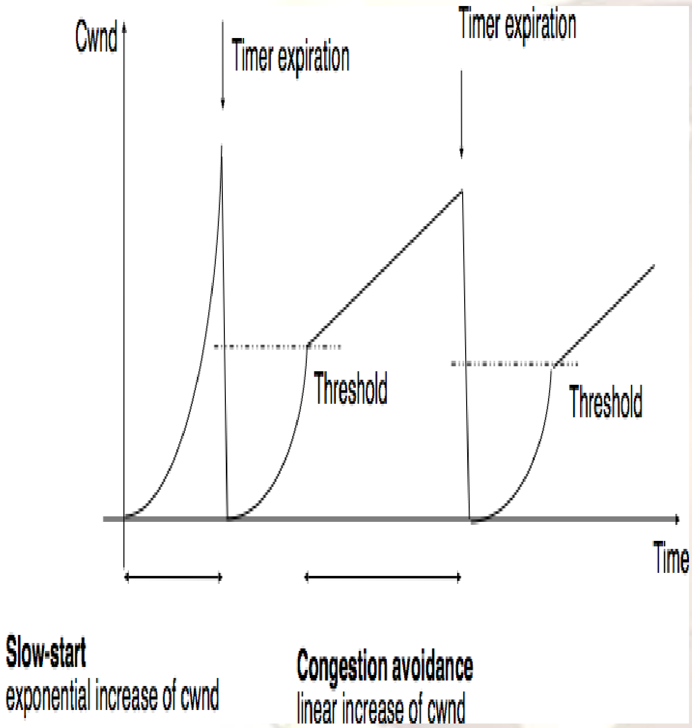
In the previous papers a performance analysis of TCP-controlled long file transfers in a WLAN in infrastructure mode along with comparison and analysis of congestion window for HS-TCP, Full-TCP and TCP-Linux in Long Term Evolution System Model are given, we have propose both performance analysis in terms of Aggregate throughput and congestion window. This paper is concerned with infrastructure mode WLANs that use the IEEE 802.11 DCF mechanism. We are interested in analytical models for evaluating the performance of TCP controlled simultaneous uploads and downloads where each connection has arbitrary TCP window size. A detailed analysis of the aggregate TCP download throughput in a WLAN for a single rate Access Point (AP) is given in [1] and[3] where it is assumed that all TCP window sizes are equal. Similarly, aggregate TCP throughput is evaluated for the multi rate case are in [4] and [5]. However, these works also consider only download or upload with constant windows.

The basic approach is to model the number of STAs with ACKs and data packets in their MAC queues as an embedded Discrete Time Markov Chain (DTMC) and aggregate TCP throughput in the Frame work of Renewal Reward theory(RRT) given in[1] and [4].

The concepts of the mechanism of TCP congestion control depending dynamically on organize the size of window according to the network path state. TCP has critical design and the congestion control permits to TCP to adjust the network end-to-end connection by control the data rate. According to the available bandwidth. However, TCP give poor performance in high bandwidth channels due to the slow response with large congestion window [1]. If TCP sender not receives Acknowledgment (ACK) for TCP receiver after transmission segment, the connection goes to timeout where the first phase of start transmission called slow-start. In slowstart phase the packet sending and the window increasing exponentially in parallel with increasing the round trip time(RTT).The window increment not continues without limits, because that will cause packet losses when the packets injected in network connection larger than the available space in network bottleneck and receiver buffers. To overcome this defect of TCP slow start, the researchers have proposed many improved methods. Fig. 1 shows the phases of standard cwnd. The basic congestion control mechanism supported by TCP include following phases, Slow-Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery.

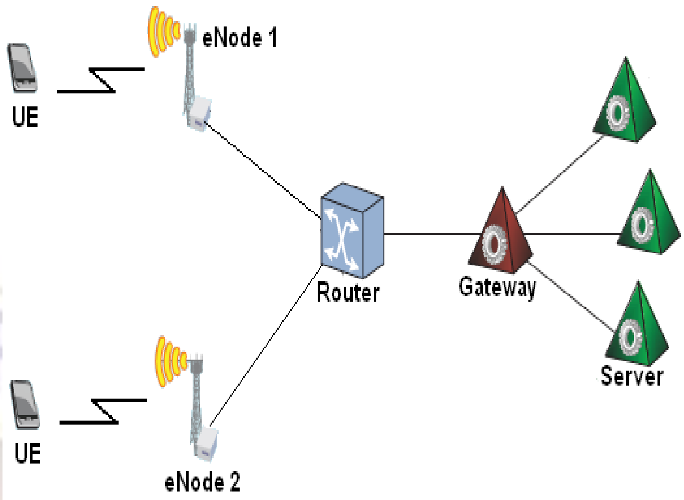
TCP congestion control includes two main phases, slow start phase and congestion avoidance phase, and these two phases used by TCPO sender to adjust the amount of data flow through network. TCP sender starts by sending packets inform of segment and waiting to receive the ACK to proving the sent packets if it's received or lost. If segment acknowledged, the sender sends two segments instead of one. After other ACK, the increment in congestion window duplicates to four. Actually, there is no exponential incrementing congestion window due to sometimes TCP sender receives delayed ACK and the sender transmits two segments every ACK.

The possibility of congestion to occur in network connections, when the network resources supply a large amount of data exceed the available bandwidth and these problem represents a big challenge in modern networks due to its cause up normal transferring in packets, like delayed packets and lost packets, where that will force the sender toresend these packets and the performance of network willsuffering from degradation [2]. Each transmission period starts with slow start phase andthe segment transfer from TCP sender to its peer until thecongestion window become equal to the present threshold(ssthresh) of the network pipeline. At this point, the network enters to congestion avoidance period and the window starts increasing linearly.



In other words, the window enlarges by one segment per RTT. [3]. There are two other phases which support some other TCP variant these are fast retransmit and fast recoveries . Fast retransmit mechanism allows to TCP sender to retransmit a segment after specific timeout period and no need to wait ACK from receiver.

This mechanism considers the packet lost if ACK delay [4].In other hand, when fast retransmit detects three duplicate ACKs, the fast recovery process starts from congestion avoidance region and use ACKs in the pipe to transmitting packets. This article provides comparison and investigation to the behavior of congestion window for three TCP versions; HSTCP, FullTCP, and TCP-Linux with analyzing the congestion window clocking and timing when these three TCP's experimented over a traffic model of LTE network.



2. RELATED WORK

The analytical work in this area has considered saturated and unidirectional traffic, i.e, either uplink or downlink; see, for example, [5], [6] and [7]. All the above papers assume that all the STAs are saturated, in other words they have packets to send to the AP at all time. In contrast, we consider TCP controlled transfers, where the "saturated nodes assumption "does not apply. All the related work that we are aware of assumes homogeneous TCP connections in the sense that the maximum window size is the same for all connections. [1] and [8] propose a model for a single rate AP-STA WLAN, assuming the same maximum size of TCP window for all TCP connections. An extension of this work in [2] considers two rates of association with long file uploads from STAs to a local server; the multi rate case is considered in [3]. [9] and [10] present analysis of TCP-controlled uploads and downloads with UDPtraffic in a single cell infrastructure WLAN. They assume equal TCP maximum window size for all connections, that TCP receivers use undelayed ACKs, and show that the total
TCP throughput is independent of the number of STAs in the system. Also, upload and download transfers obtain equal shares of the total throughput. The letter [11]gives an average value analysis of TCP performance with upload and download traffic. First, the authors provide an expression for the average number of active TCP stations. In [12], a finite buffer AP with TCP traffic in both upload and download direction is analysed with delayed and undelayed ACK cases.[13] provides an analysis for a given number of STAs and maximum TCP receive window size by using the well known p-persistent model and in system model proposed in [1]&[7]. However, [13] considers only download traffic or upload traffic, not both together. HTTP traffic is analyzed in [14]. A queueing model is proposed to compute the mean session delay in the presence of short-lived TCP flows. The impact of TCP maximum congestion

window size on this delay is studied. The analysisalso extended to consider the delayed ACK technique.

Our contribution: We provide a simple approach to model the aggregate throughput of long-lived TCP downloads and uploads also with Comparison and Analysis of Congestion Window for HS-TCP, Full-TCP and TCP-Linux in Long Term Evolution System Model given in [1]and[2].

2.1. BRIEF DESCRIPTION OF TCP'S AND LTE SYSTEMS

HSTCP introduce a new mechanism based on enhancement the time of loss recovery of classic TCP by variation the standard algorithm of TCP's Additive

Increase Multiplicative Decrease (AIMD). The developed mechanism of HSTCP works effectively with large congestion windows. That mean, when the congestion window less than the threshold, HSTC Puses AIMD algorithm to control congestion in network connections, but if congestion window high its use the algorithm of high speed AIMD [5]. Actually, designing of

HSTCP is depending on the response variation in environments of the low congestion occurring in bottleneck and to the response activation of standard TCP in environments of packet loss rates [6].FullTCP can only apply with the congestion control of TCPReno. However, FullTCP must be provided with fully algorithms of congestion control such as Tahoe, Vegas, and Sack [7]. In TCP-Linux, it's introduced three main characteristics to developing the performance of TCP-Linux.
These three elements are [8]:

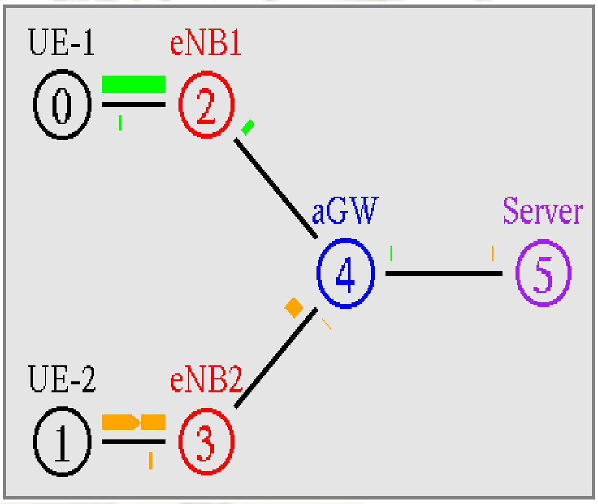
- Standard interface for congestion control algorithms.
- Redesigned loss detection module.
- New event queue scheduler that increases the test speed.

As we mentioned before, this study and the experimental test depended on analysis the behaviour of three TCP's over LTE network topology. LTE is the 3GPP specification for the fourth generation of mobile networks and referred to as Evolved UMTS Terrestrial Radio Access (E-UTRA). LTE systems aim to provide a step forward in wireless and mobile systems by providing high speed data transmission to users by providing low latency links an improved spectral efficiency. Comparing with Wideband Code Division Multiple Access
(WCDMA) and HSPA, LTE provides higher rate of data with downlink reaches more than 100Mbps uplink of 50 Mbps. Infact, LTE systems released wide achievements in telecommunication networks by providing lower user costs than other systems, better spectrum efficiency, and a very small latency [9, 10].
LTE systems is under developments now, and the TCP protocol is the most widely used protocol for wired and wireless systems, although TCP was not originally designed for real time applications and not for wireless networks then, we need to develop a new TCP mechanisms, or at least choose
a suitable TCP variant for each new network, to be more efficient and more reliability with this network.

3. PROPOSED TOPOLOGY AND NETWORK SIMULATION

we used NS-2 network simulator to modulate the proposed topology of LTE and also the experiments to monitor the behaviour of TCP versions used in these article performed according to NS-2, where NS-2 is a better simulation deals with TCP's protocols. NS-2 not just a simulator, but it's a discrete event aimed to support the research and studies that deals with communications and networks analysis. Also, NS-2 provides environments to simulate and modeling multicast protocols, networks traffics, handovers, and other networks resources and conditions for wireless and wired channels [12]. In our research, we used NS-2 version 2.32 , and this version installed over Windows

XP or using Cygwin, where Cygwin provides a Linux-like environment under Windows, because NS-2 already supported a Linux operating system only, then we need to get a virtual environment



SIMULATION PARAMETERS

Parameter	**Value**
TCP protocols	
HSTCP, FullTCP, TCPLinux	
Propagation Delay of all links	2 msec
Bandwidth of a GW link	10 Mbps
Bandwidth of Server	100 Mbps
Bandwidth of UE link	1Mbps
Packet Size	1500 Bytes
Window size	100 Kbytes
Simulation Time	50 sec

As shown in Fig. 3, we proposed six nodes elements. Two nodes as mobile nodes (UE's), two base stations (eNB's), one gateway router, and one FTP server. The proposed LTE model is assumed a bottleneck connection with bandwidth of 100Mbps and delay of 2 msec. The parameters of modeling and simulation presented in Table 1, where all links are set to unified propagation delay of 2 msec. The maximum packet size of TCP was set to 1500 Bytes, with minimum window size of 100 Kbytes. NS-2 representation of links depend on bandwidth of the link, type of link (duplex or simplex) and the propagation delay of packets over this link. In LTE networks simulation, we have many bandwidths and two types of links but only one delay for all links.

As we mentioned before, NS-2 provides the agents of most of TCP's including the three TCP's studied here. TCP-Linux not provided directly in NS-2.32, but it need to install it by using a compatible patch and process some modification in NS-2 environments. In TCP-Linux, TCP agent name change from (Agent/TCP/Sack1) to (Agent/TCP/Linux), and the TCP destination set to Sack1.

However, to implementing TCP-Linux in TCL script (TCL is the language used in programming inside NS-2), we must choose one of two available agents form, either using the agent in the form of: (Agent/TCPSink/Sack1) or using the agent in form: (Agent/TCPSink/DelAck). As shown in script below:

```
/* TCP-Linux agent */
set tcp [new Agent/TCP/Linux]
set tcpsink [new Agent/TCPSink]
```

In HSTCP, the structure of agents differs from that used in TCP-Linux. The implementation of HSTCP is controlled by the manner to adjusting the congestion control. This TCP variant depends on modifying the response function and only determines the effect when the congestion window is high. In NS-2, implementing of HSTCP to test the congestion window behavior or for any other application also needs to use TCP source and destination with different agents. When HSTCP is applied in NS-2, must use the form shown below:

```
/* HSTCP agent */
set tcp [new Agent/TCP/Sack1]
set tcpsink [new Agent/TCPSink/Sack1]
```

The implementation of FullTCP in NS-2 is still under development because of its adding newly to the TCP suite of the agents which supported by NS-2 configuration. FullTCP agent is differ from the other agents used under NS-2 due to Full TCP agent is classified as bidirectional protocol. That means, FullTCP is a two way TCP where its provide bidirectional data transmission. In addition, FullTCP support a new data structure, where it uses the bytes instead of packets in representing the sequence numbers. FullTCP can implementing in TCL script in the form shown below:

```
/* FULLTCP agent */
set tcp [new Agent/TCP/FullTCP]
set tcpsink [new Agent/TCP/FullTCP]
```

4. RESULTS ANALYSIS AND DISCUSSIONS

The results obtained from experiments divided into four parts, three parts to analyze the slow-start and congestion avoidance phases of each TCP, and the last compares the full period of cwnd of three TCP's in one experiment.

As shown in Fig. 4, the slow-start phase of TCP-Linux shows that it reaches to congestion within 1.25 sec, and the maximum window reach to 100 Kbyte. It have a congestion avoidance after 1.5 sec and then it start a new slow-start phase with new ssthresh at 1.5 sec to a new congestion point at 50 Kbytes. That's mean ssthresh=cwnd/2, since TCP-Linux implementation evaluates the initial ssthresh from the cwnd size of the previous TCP connection.

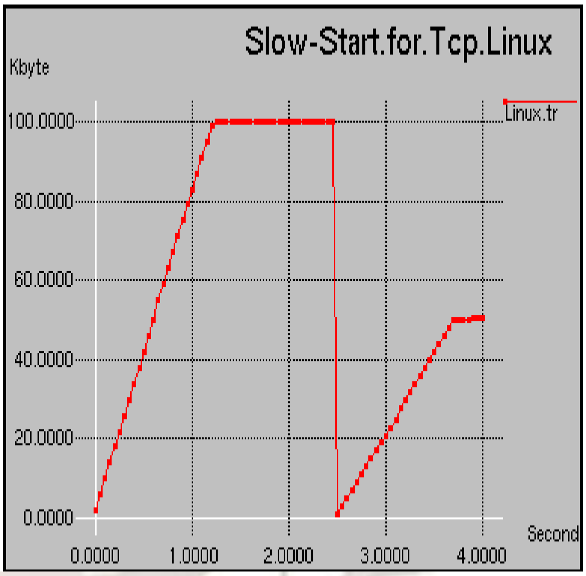


Fig. 4 Slow start for TCP Linux

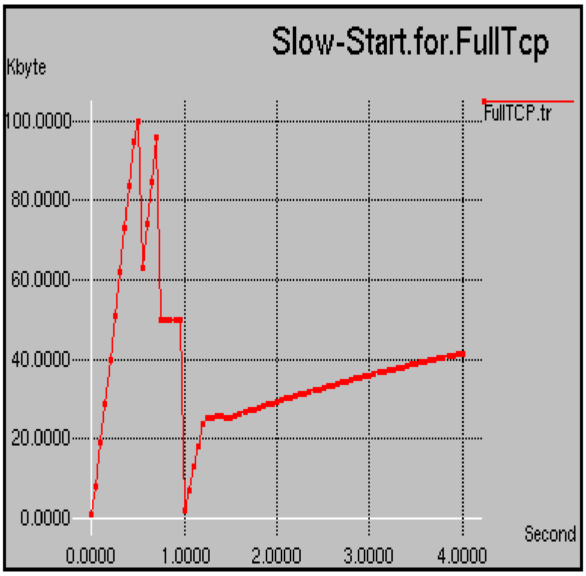


Fig. 5 Slow start for Full TCP

In this Fig. 5, the slow-start phase of FullTCP needs only to 0.5 sec to reach window of 100 Kbytes, and no clearance congestion avoidance here, but we can note some fast recovery with some packets lost. The new slow-start phase starts after 1 sec to reach a new congestion point at 60 Kbytes. Slow-start of FullTCP, starts unstable, but after first 1 sec did well and provided a stable behaviour of cwnd.

From Fig. 6 we illustrate the slow-start of HSTCP, where it reached within 1 sec, to 80 Kbytes as ssthresh, without congestion avoidance. So it needs long slow-start time for new session, reached to 10 seconds. Whatever, it provided a regular cwnd but it had a small window compared with other two variants.

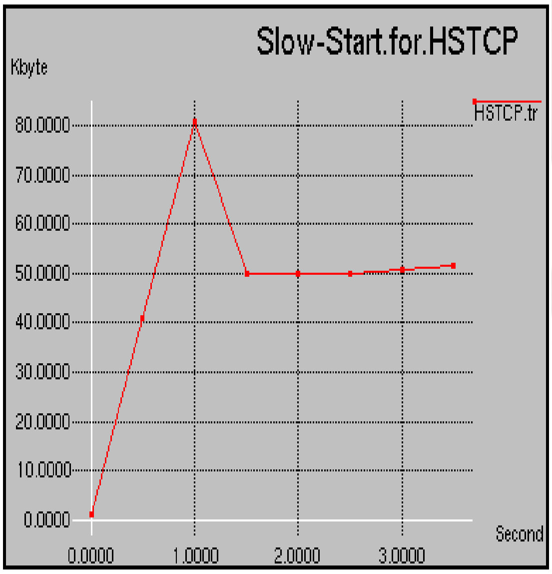


Fig. 6 Slow-Start phase of HSTCP

The three TCP's can provide a good and regular cwnd, inspite of the differences is slow-start phase behavior. As shown in Fig. 7, the maximum slow-start founded in TCP-Linux where it exceeded 1 sec, with ssthresh of 50 Kbytes with small congestion avoidance. But cwnd of FullTCP performed a better ssthresh of 60Kbytes. A zooming graph of one complete period of cwnd of three TCP's, shown in Fig. 8, where the maximum congestion point of network got with FullTCP, in other side, the minimum point appeared with HSTCP. The maximum slow-start phase occurred with TCP-Linux with maximum congestion avoidance of 1 Kbyte, where, lager congestion avoidance increment will improve performance.

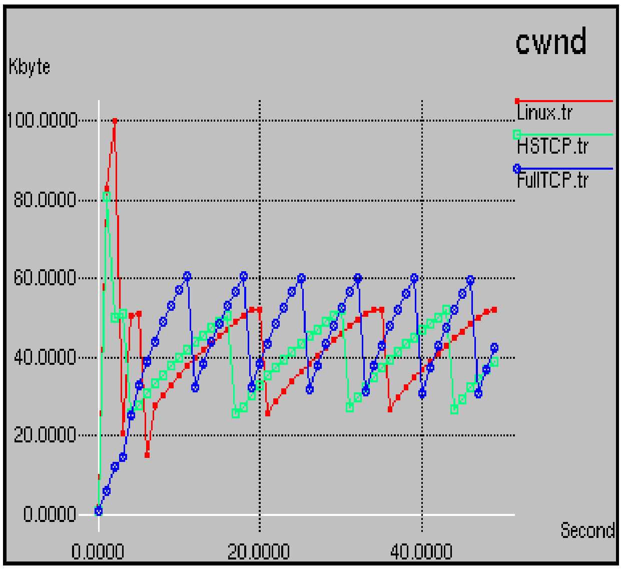


Fig. 7 cwnd of HSTCP, TCP-Linux, and FullTCP

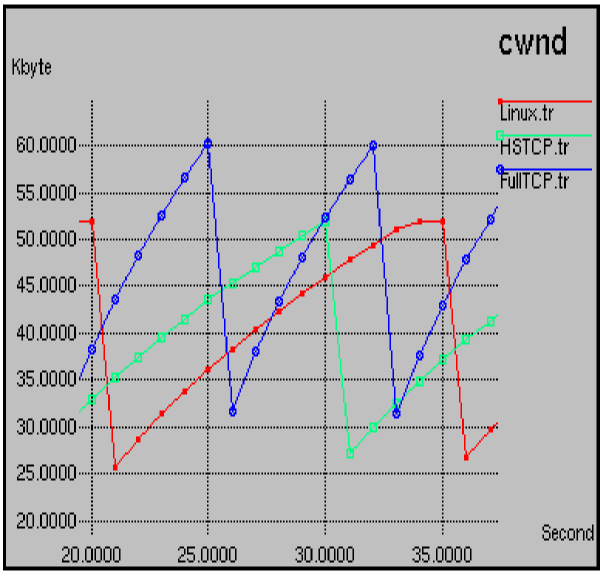


Fig. 8 Enlarged plot of Slow-Start and Congestion Avoidance

When cwnd can reach maximum value, the losses in packets may occur and performance decrease. In the plot here at Fig. 8, the cwnd can be larger than the bandwidth delay product, we only get 60 Kbytes in FullTCP session as a throughput. The losses occur when the window size becomes larger than the point of congestion window of the network, or we can say the packets lost because the network has not enough capacity for storing all the sent packets, so, if the window size is too large, some packets are dropped and performance decreases, since when a loss occur, the cwnd is divide by two.

3. CONCLUSIONS

This paper provides a mechanism to the behavior of congestion window for different TCP versions, HSTCP, FullTCP, and TCP-Linux, over a model of LTE network using NS-2 network simulator. The aim of this work was to establish a bound on the performance of cwnd of these TCP's in high bandwidth networks also with Performance analysis of TCP-controlled long file transfers. The investigation has been carried out by modelling all links and traffics of LTE networks with standard parameters, such as bandwidth and delay, to monitor the cwnd, slow-start, and congestion avoidance of these variants. Full TCP provided a well performance and a big widow size compared with other two variants, so HSTCP had a longest slow-start period with a congestion level near to that produced by TCP-Linux. Generally, the implementation of three TCP's shows the differences in behaviour of congestion window phases in spite of it is a little but when used over high bandwidth or any other high speed networks, we can see a large rate of packets losses, because of incompatibility between the congestion window parameters and the link capacity.

REFERENCES

[1] Ghassan A. Abed Mahamod Ismail Kasmiran Jumari “A Comparison and Analysis of Congestion Window for HS-TCP, Full-TCP,and TCP-Linux in Long Term Evolution System Model” in IEEE conference on open system(ICOS2011).

[2] Pradeepa BK and Joy Kuri ”Aggregate AP Throughputs for Long File Transfers in a WLAN controlled by Inhomogeneous TCP” Connections in proc. of IEEE 2011.

[3] A. Kumar, E. Altman, D. Miorandi, and M. Goyal, "New insights from a fixed point analysis of single cell IEEE 802.11 WLANs," IEEE ACM Transactions on Networking, Vol. 15, No 3, pp. 588-601, June 2007.

[4] F. Cali, M. Conti, and E. Gregori, "Dynamic Tuning of the IEEE 802.11 protocol to achieve a Theoretical Throughput Limit;' IEEElACM Transaction On Networking, vol. 8 no. 6 pp. 785-799, Dec 2000

[5] R. Bruno,M . Conti,a nd E. Gregori," Throughput Analysis of TCP clients in Wi-Fi Hot Spot Networks," Networking 2004, LNCS 3042, pages 626- 637,2004

[6] R. Bruno, M. Conti, and E. Gregori, "Throughput analysis and measurements in IEEE 802.11 WLANs with TCP and UDP traffic flows," IEEE Transactions on Mobile Computing, vol.7, no.2, pp. 171-186, Feb. 2008.

[7] R. Bruno , M. Conti, and E. Gregori, "An accurate closed-form formula for the throughput of long-lived TCP connections in IEEE 802.11 WLANs;' Computer Networks, vol. 52, pp. 199-212,2008

[8] R Bruno, M Conti, and E Gregori, "Average value analysis of 802.11 WLANs with persistent TCP flows," IEEE Communication Letters, vol.13, noA, Apr. 2009.

[9] O. Bhardwaj, G. V. V. Sharma, M. Panda, and A. Kumar, "Modeling finite buffer effects on tcp traffic over an ieee 802.11 infrastructure wlan," Computer Networks, vol. 53, pages. 28552869, Nov 2009

[10] J. Yu, S. Choi, "Modeling and analysis of TCP dynamics over IEEE 802.11 WLAN," Proceedings o/ the IFlP WONS" pp. 154-161 Obergurgl, Austria, 2007

[11] K. Fall, S. Floyd, and T. Henderson, “Ns simulator tests for reno fulltcp,” U.S. Department of Energy,1997.

[12] D. X. Wei and P. Cao, “NS-2 TCP-Linux: An NS-2 TCP Implementation with Congestion Control Algorithms from Linux,” Workshop on NS-2, October 2006.

[13] T. Mahmoodi,“ Transport Layer Performance Enhancements over Wireless Networks,” University of London, August 2009.

[14] Y. Tan,“Active Queue Management for LTE Uplink in eNodeB,” Master Thesis, Helsinki University of Technonlogy, February 2009.

[15] A. Racz, A. Temesvary, and N. Reider, “Handover Performance in 3GPP Long Term Evolution (LTE) Systems,” Mobile and Wireless Communications Summit, 2007. 16th IST, pp. 1–5, July 2007.

[16] The Network Simulator - NS-2. In: Information Sciences Institute, University of Southern California, USA, 2006.