Comparison of TCP Variants Over Self-Similar Traffic

Mazleena Salleh
Department of Computer System and Comm.
Faculty of Computer Science and Information System
University Technology Malaysia

mazleena@fsksm.utm.my

Ahmad Zaki Abu Bakar
Department of Information System
Faculty of Computer Science and Information System
University Technology Malaysia

zaki@ fsksm.utm.my

ABSTRACT

Network traffic of Internet will continue to expand in terms of both volume and users, and transmission congestion protocol (TCP) is accounting for more than 90% the Internet traffic. In addition to this, it is well known that Internet traffic exhibits self-similarity, which cannot be described by traditional Markovian models such as the Poisson process. In this paper we focus on experimental quantitative comparisons of four TCP variants namely Tahoe, NewReno, Vegas, and SACK running over self-similar traffic. With the aid of network simulator, NS2, aggregated network traffic of self-similarity behavior is simulated and will act as the background traffic. The impact of one TCP microflow will be observed, and data analysis such as congestion window, round trip time, throughput, and the behavior of the data received over time is analyzed. In measuring the degree of self-similarity we employed optimization method to determine Hurst parameter. From our analysis, TCP variants tend to demonstrate self-similar traffic flow even though initially the departure rate of TCP packets is constant With respect to performance, TCP-NewReno outperform other TCP variants with higher throughput and efficacy. In addition, we have also investigate the LAN traffic behavior of FSKSM, UTM, and proved that Internet Protocol (IP) dominated the total traffic where 90% is TCP and has a Hurst value of 0.88.

KEYWORDS

Self-Similar, Hurst Parameter, Congestion Control.

1. Introduction

In the last decade, computer networks have experienced tremendous growth. More and more computers get connected to both private and public networks where most of traffic is generated by traditional data transfer applications such as HTTP, NNTP or FTP. This consequently made TCP/IP the most widely used protocol for computer network and accounts for vast majority of the traffic over wide area network particularly the Internet [1].

In addition to the protocol usage, real-life traffic presents a self-similar behavior whereby changing the timescale does not change the behavior of the traffic pattern [2]. Empirical evidence cited in [3] have also shown that traffic carried on local-area and wide-area packet-switched networks could be better represented with self-similar models that incorporate long-range dependence (LRD), rather than the traditional Markovian models. Given the significance of this phenomenon, it is important that we understand the impact of self-similar behavior on the network protocol.

The issue of self-similarity has also been addressed in various studies from many aspects including its impact on network performance, modeling techniques, and causes of the appearance of self-similarity [4]. As for TCP and TCP variants, several comparison studies has been done regarding with the analysis of implementation [5], latency and throughput [6] and performance [7]. On self-similarity and TCP, workssuch as generation of pseudo-self-similarity [8] and self-similar by TCP [9, 10] have been put forward.

On the other hand, our research work investigates the performance of four TCP variants namely Tahoe, NewReno, Vegas, and SACK running over self-similar traffic. Our objective is to have a better understanding on some of the characteristics of TCP variants, and results obtained can contribute in providing a higher end-to-end throughput, better quality-of-service for applications and fast response from the network by utilizing the best-performed TCP variant protocol.

The outline of the paper is as follow: First we introduce the concept self-similar in Section 2. In Section 3 we present the theoretical comparison of TCP variants. The research methodology is described in Section 4 while Section 5 discusses the results of the experiment. Finally we conclude our paper in Section 6.

2. Self-Similar Traffic

Self-similar is the property that is often associated with fractals whereby objects appears the same regardless of the scale at which it is viewed. It is closely associated with the phenomenon of heavy-tailed distributions. In a large number of Internet traffic measurements several authors have detected self-similarity [2, 11] and the attributed of heavy-tailed properties are due to file sizes and user thinking time, flow (session) duration, as well as packet inter-arrival time distributions in the Internet [8, 12].

Network traffic has peak behavior over all timescales and thus it does not follow Poisson traffic behavior [1]. A key paper by Willinger *et al.* [13] compared actual measurements and a synthetic Poisson model and concluded that the Poisson model lacks the burstiness over large time scales that are present in actual traffic measurements. Table 1 shows the comparison of Poisson and self-similar traffic flow. This burstiness is of great importance when designing a network. However this does not mean that the Poisson model cannot be used in the simulation of modern data networks. It is not appropriate for modeling at the packet level, but it is still very useful to model for example application level events. FTP sessions and TELNET connections are statistically consistent with Poisson arrivals [14].

The degree of self-similar is defined by Hurst parameter. Commonly used measurement for burstiness such as index of dispersion (for counts), the peak to mean ratio, or the coefficient of variation (interarrival times) is no longer meaningful for self-similar traffic [15].

2.1 Problem Arises from Self-Similar Behavior

Self-similar traffic implies that there exists the presence of concentrated periods of congestion and concentrated periods of light network load at a wide range of time scales. This scale-invariant burstiness introduces new complexities into resource control and quality of service (QoS) provisioning. A number of performance studies have shown that self-similarity has a detrimental effect on network performance leading to increased packet loss rate, delay, and a degraded delay-throughput trade-off relation [16]

Table 1 Poisson and Self-Similar Traffic

	Poisson Traffic	**Self-Similar Traffic**
Peak Behavior	Exist at small time scale but decrease in larger time scale	Exist in all time scale
Multiplexing Traffic	Averages out the peak behavior	Does not smooth the peaks, in fact makes them worse

During congestion or the occurrences of burstiness packet loss is amplifies [17]. Packets arrive in over-utilized network will experience long delays and may be even dropped due to buffer overflow. This will consequently result in a decrease of network utilization due to higher number of packet retransmission. This predicament has an impact on the designing of real-time, interactive multimedia applications. When Hurst parameter is high, the combination of round-trip delay and loss rate may be too extreme for delay- and loss-sensitive applications, such as Internet telephony and video-conferencing [18].

Several solutions can counteract this problem such as efficient switches, larger queue size and higher bandwidth. However this implies massive over-building and increases cost in the implementation of the network. On the other hand, use of large buffers in switches and routers to handle peaks in lieu of massive overbuilding, means long time delays.

3. Transmission Control Protocol (TCP)

Transmission congestion protocol (TCP) traffic flow make up of 90% of the total Internet traffic. The algorithm that is introduced by Van Jacobson [19] is based on a complex interaction of algorithms and is used to guarantee both the delivery of all data as well as to enable fair sharing of the network. TCP is transport layer protocol and also an end-to-end protocol. As a transport layer network protocol, TCP offers a reliable, connection-oriented, byte-stream service. TCP supports flow control, which prevents the sender from overrunning the buffer capacity of the receiver. In addition, TCP implements congestion control, which prevents the sender from injecting too much traffic into the network. Being an end-to-end protocol, TCP turns a host-to-host packet delivery service, provided by IP, into a process-to-process communication channel [20]. As such TCP has provides stable and reliable transfer of packet data across the Internet.

TCP is designed to control the amount of outstanding data in network by using congestion window mechanism. Two main variables used in this mechanism are, *cwdn* (congestion window) and *ssthresh* (slow start threshold value). There are two distinct phases that is the slow start and congestion avoidance phase. Figure 1 illustrates the overall operation of TCP [21].

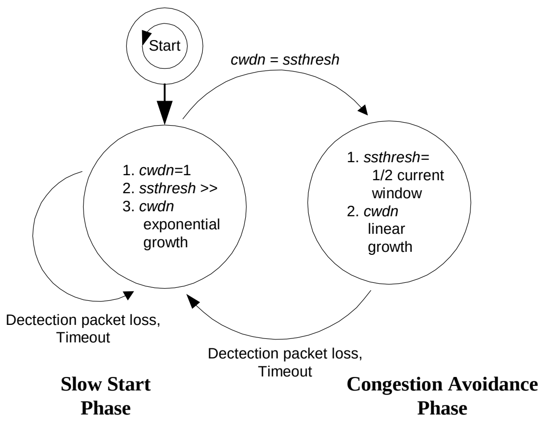
Start
cwdn = ssthresh
1. *cwdn*=1
2. *ssthresh* >>
3. *cwdn* exponential growth
1. *ssthresh*= 1/2 current window
2. *cwdn* linear growth
Dectection packet loss, Timeout
Dectection packet loss, Timeout
Slow Start Phase
Congestion Avoidance Phase

Figure 1 Transition Diagram of TCP

TCP-Tahoe was introduced by Jacobson in 1988 whereby fast retransmit stage is added to the basic TCP operation. Instead of waiting for timeout for retransmission, fast retransmit phase will retransmit packets after receiving three duplicated acknowledgements from the receiver. TCP-NewReno (1996) extends TCP-Tahoe with its fast recovery algorithm and partial acknowledgements as an indication of another packet lost. Sender comes out of fast recovery only after all outstanding packets are acknowledged. TCP-Selective Acknowledgement (SACK, 1996) enables receiver to give more information to sender about the received packets through extra information in each acknowledgment so that to inform the sender of which blocks of packets have successfully arrived. This solution allows sender to recover from multiple packets losses more efficiently since it can avoids false retransmits [22]. Unlike other TCP variants TCP-Vegas (1994) sender anticipates the onset of congestion by monitoring the difference between the rate it is expecting to see and the rate it is actually realizing. Vegas' strategy is to adjust the sender's *cwnd* in an attempt to keep a small number of packets buffered in the routers along the path [23].

4. Methodology

The aim of this research is to understand self-similar traffic and its effect on congestion control protocol namely TCP. To achieve this understanding two defined experiments has been done: (i) evaluation of FSKSM network traffic, and (ii) analysis on the performance of TCP variants on self-similar traffic.

TCPDump is used to capture FSKSM traffic traces. With the aid of SQL database, packet size is extracted together with its timestamp from the traffic traces. This will be the time series *f(t)* that will be analyzed.

Network Simulator (NS2) of VINT project [24] is used to simulate the network configuration as depicted in Figure 2. This network setup is for the analysis of Hurst parameter with respect to traffic load and the performance of TCP variants on self-similar traffic. The dumbbell topology consists of TCP senders and receivers and a pair of router that have a queue buffer size of 10. The small size buffer is chosen so as to simulate congested traffic. The bottleneck link has the capacity of 50Mbps, a propagation delay of 25ms and a drop tail queue [25]. Two kinds of traffic sources were used; self-similar traffic sources for generating background traffic, and a fixed-rate traffic source from which jitter measurements were determined. Self-similar traffic can be constructed by multiplexing a large number of ON/OFF sources that have ON and OFF period lengths and are heavy-tailed [26]. In this case, TCP packets are generated using the on-off Pareto model with the rate parameter of 1000k and shape of 1.2 at each of the sender nodes.

To illustrate the impact of self-similar traffic on TCP, a micro flow of TCP variants traffic is generated at node *x* running at the foreground of self-similar traffic. The Hurst value of 0.8 is chosen for the aggregated traffic based on the report done by Crovella *et al.* whereby from their measurement data of Web traffic, they estimated that the Hurst parameter to be around 0.8 [27]. Traffic traces of 100 seconds were collected and statistics such as packet arrival, packet drop, packet retransmitted and number of received acknowledgement are recorded. In addition, the throughput of the connection and the Hurst parameter based on byte received measured at fixed time granularity at the receiver is calculated.

In all the experiments above, Hurst parameter of the time series is calculated by employing optimization method. Optimization method is based on the exaction form of covariance function where exploits the covariance function that is specified by the network traffic. Comparing with variance time plot or R/S plot, optimization method process much faster and yield better

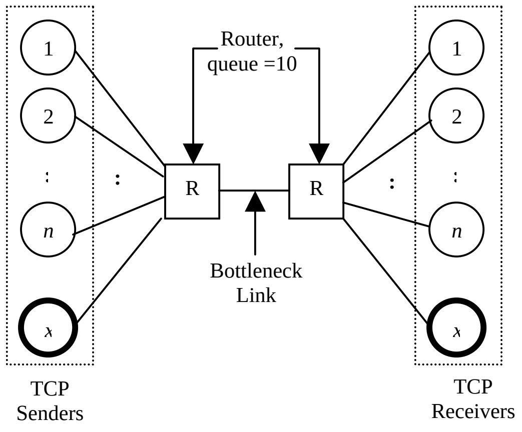

Figure 2 Network Configuration Setup

confidence intervals as compare with wavelet [28].

5.0 Results and Discussion

During the interval of five minutes, 4880 packets (3,018,692 bytes) of FSKSM traffic traces were collected. The packets is utilized for address resolution protocol (0.94%), Internet protocol (96.54%) and logical-link control (2.5%). Out of the total number of Internet traffic, 90.7% is TCP and only 5.84% is user datagram protocol (UDP) and this clearly proves that TCP dominate Internet traffic as stated by [27]. Figure 3 shows *f(t)* and the calculated Hurst parameter, *H* is 0.88, which is second order self-similar and does exhibit LRD.

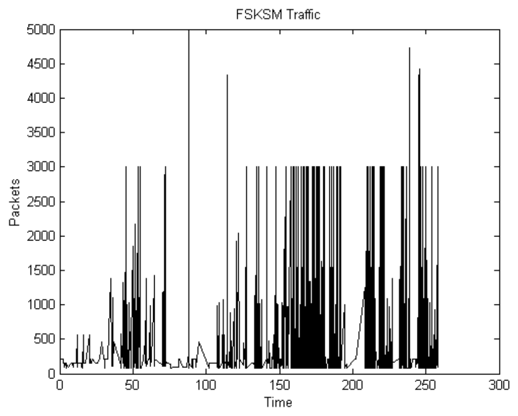

Figure 3 FSKSM Traffic Traces

The statistical data for each of the TCP variants microflow is tabulated in Table 2. Here we defined efficiency of traffic flow as the ratio of number received packets against the total number of send packets. The total number of send packets is the summation of total received packets, total dropped packets and total packets retransmitted. Based on the statistics obtained, TCP-NewReno and TCP-SACK have a higher number of packets arrived and both of these TCP variants have the same efficiency value. On the other hand TCP-Vegas has the least number of packets transfer and the lowest efficiency value.

Table 2 Statistic Analysis of TCP Microflow

	Tahoe	**NewReno**	**Vegas**	**SACK**
No. Packet Arrive	70070	77312	61853	75673
No. Packet Drops	43	35	39	33
No. Packet Retransmit	45	35	45	33
No. Acknowledgement	69960	77276	61860	75665
Efficiency	0.9987	0.9991	0.9986	0.9991
Hurst Value (byte received)	0.84	0.82	0.9	0.84

Another factor that should be noted here is that even though the microflow packets is generated at a constant rate, the effect of self-similar traffic has an impact on the behavior of these packet flow in the link. This is shown by the value of the Hurst parameter based on *f*(*t*) of the microflow and it is approximately 0.85, near to the Hurst parameter of the background traffic. This illustrates that when background traffic exhibits long-range dependence and self-similarity, TCP sustains these properties too. The basic retransmission mechanism and the slow start could be the reason why TCP react as such.

Figure 4 show the comparison of the flow throughput where TCP-NewReno outperforms the other three TCP variants. Even though initially TCP-Vegas starts with a higher throughput, but its performance slowly stabilized at time = 50, when packets start to drop while the other TCP variants continue increasing their individual throughput. Other behaviors of TCP such as congestion window (*cwnd*), round trip time (RTT), and packet retransmitted vs. time are recorded as shown in Figure 5, 6 and 7 respectively. Regarding with *cwnd* size, TCP-NewReno and TCP-SACK operate at a larger window size that is at approximately 200 packets, and never fall to less than 25 packets when congestion is detected and thus this explain the higher throughput that was achieved. However *cwnd* behavior of TCP-Tahoe and TCP-Vegas is more chaotic and frequently goes into the slow start phase where *cwnd* starts at 1. The RTT for TCP-Tahoe is constant at the rate of 0.075 sec throughout the simulation whereas for the other TCP variants, the RTT changes accordingly to the congestion of the network traffic. When comparing Figure 6 and 7, the sporadic spike in RTT occurs during the retransmission of packets.

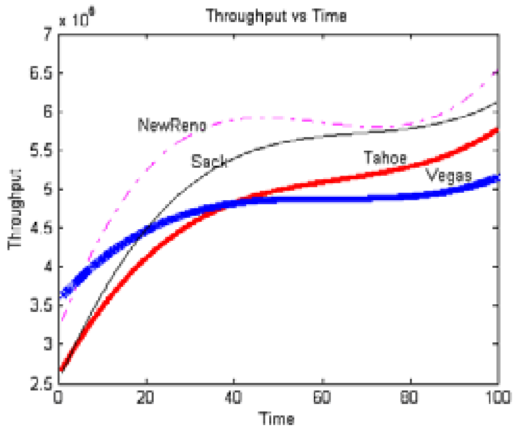

Figure 4 Throughput of TCP Variants

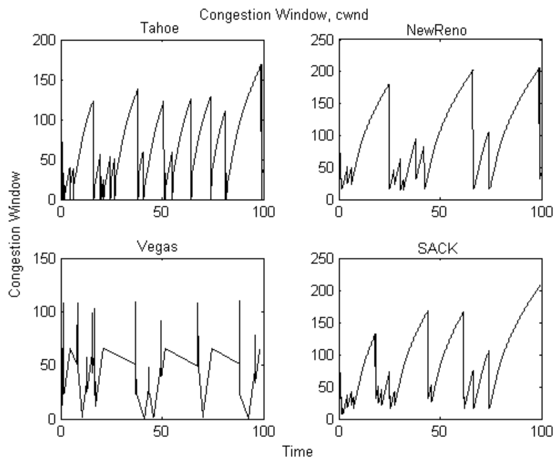

Figure 5 Congestion Window of TCP Variants

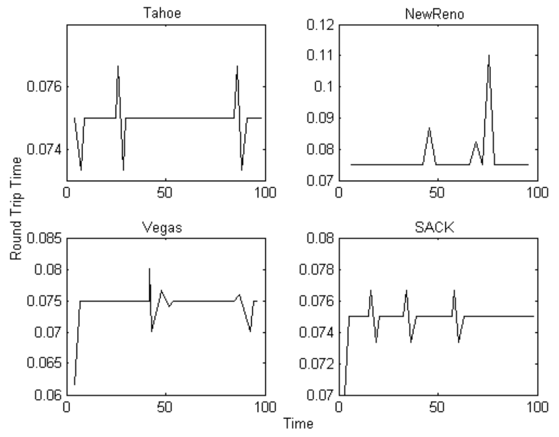

Figure 6 Round Trip Time of TCP Variants

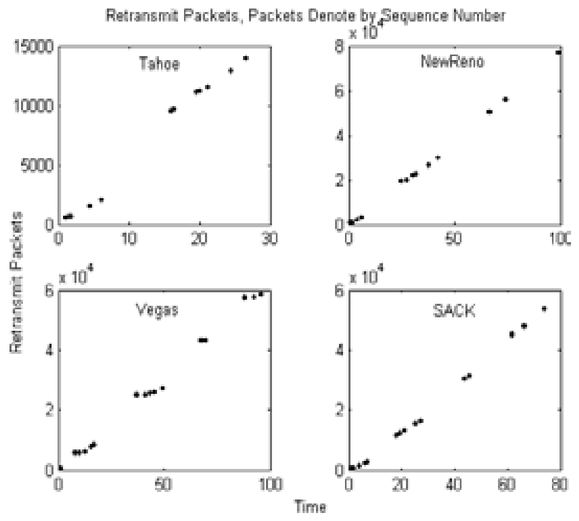

Figure 7 Occurrences of Retransmitted Packet

From the results obtained we can conclude that TCP-NewReno has a higher performance than other TCP variants. Its fast recovery from packet loss and slow start phase that does not reset to 1 during congestion proves to be favorable in self-similar traffic.

6. Conclusion

Self-similarity and scaling phenomena have dominated Internet traffic analysis for the past decade and self-similar traffic models may better describe traffic in many of today's computer networks comparing to traditional Markovian models. The causes of this apparent self-similar behavior must be identified so that network designers could respond with greater understanding. The burstiness of packet traffic, occurring as it does on multiple time scales, presents a dilemma to the network designer.

In this study we have shown that even in a small LAN such as at FSKSM, UTM, traffic flow still exhibit self-similar properties. We have also executed an analysis of the performance of TCP-Tahoe, TCP-NewReno, TCP-Vegas and TCP-SACK over self-similar traffic. From the experiment, TCP-NewReno outperforms other TCP variants with regard to efficiency and throughput. Also we have noted that control procedure of TCP tend to form a self-similar TCP traffic flow when running across an aggregated self-similar traffic.

Due to limited computer resources, the simulation time as well as traffic traces can be considered as short. But as an initial starting point, the work and findings have tremendously gives an insight of self-similar traffic and the understanding of TCP variants. For future works, we will consider varying load capacity of background traffic from low to high load while analyzing the performance of TCP variants as well as incorporating wireless environment.

7. Acknowledgement

We extend our acknowledgment to Mohd Yazid Idris and Mohd Foad Rohani in provisioning the required tools for the execution of the experiments.

References

[1] N. S. V. Rao and L. O. Chua. "On Dynamics of Network Transport Protocol". Proceedings of Workshop on Signal Processing, Communications, Chaos and Systems. 2002. pp. 39-53.
[2] W. Leland, M. Taqqu, W. Willinger, and D. Wilson. "On the Self- Similar Nature of Ethernet Traffic (Extended Version)". IEEE/ACM Transactions on Networking. February 1994. 2(1), pp. 1-15.

[3] J. M. Peha. “Protocols Can Make Traffic Appear Self-Similar”. Proceedings of IEEE/ACM/SCS Communication Networks and Distributed Systems Modeling and Simulation Conference. 1997. pp .47-52.

[4] K. Park, G. Kim, and M. Crovella. “On the Effect of Self-Similarity on Network Performance”. Proceedings of the SPIE International Conf. On Performance and Control of Network System. Nov 1997. pp. 296-310.

[5] B. Moraru, F. Copaciu, G. Lazar and V. Dobrota. “Practical Analysis of TCP Implementations: Tahoe, Reno, NewReno”. Proceedings of RoEduNet International Conference: Networking in Education and Research. 2003. pp. 125-130.

[6] B. Sikdar, S. Kalyanaraman and K. S. Vastola. “Analytic Models and Comparative Study of the Latency and Steady-State Throughput of TCP Tahoe, Reno and SACK”. IEEE/ACM Transactions on Networking. 2003.

[7] H. Koga, Y. Hor and Y. Oie. “Performance Comparison of TCP Implementations in QoS Provisioning Networks”. IEICE Trans. on Communications. June 2001.E84-B(6). pp. 1473-1479.

[8] L. Guo, M. Crovella, and I. Matta. “How does TCP generate Pseudo-self-similarity?” Proceedings of the Ninth International Symposium in Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS'01). 2001. pp. 215-223.

[9] D. R. Figueiredo, B. Liu, A. Feldmann, V. Misra, D. Towsley, and W. Willinger. “On TCP and Self-Similar Traffic”. Performance Evaluation, 2004 (to appear).

[10] P. Haga, P. Pollner, G. Simon, I. Csabai, and G. Vattay. “Self-generated Self-similar Traffic”. Nonlinear Phenomena in Complex Systems. 6(4). 2003. pp. 814 – 823.

[11] Paxson, V., and S. Floyd, “Wide-Area Traffic: The Failure of Poisson Modeling”. IEEE/ACM Transactions on Networking 1995. 3(3). pp. 226-244.

[12] Kihong Park, Gi Tae Kim, and Mark E. Crovella. “On the Relationship Between File Sizes, Transport Protocols, and Self-Similar Network Traffic”**.** In Proceedings of the International Conference on Network Protocols. 1996. pp. 171-180.

[13] W Willinger, M. S. Taqqu, R. Sherman and D. V. Wilson. **“**Self-similarity Through High-Variability: Statistical Analysis of Ethernet LAN Traffic at the Source Level”, IEEE/ACM Transactions on Networking. 1997. 5(1). pp. 71 – 86.

[14] V. Paxson and S. Floyd, “Wide-Area Traffic: The Failure of Poisson Modeling”. IEEE/ACM Transactions on Networking. 1995. 3(3). pp. 226-244.

[15] W. Stallings. “High-Speed Networks: TCP/IP and ATM Design Principles”. Prentice Hall, New Jersey, 1998.

[16] K Park. “On the Effect and Control of Self-Similar Network Traffic: A simulation Perspective”. Proceedings of the 29th conference on Winter simulation, U.S.A. 1997. pp. 989 – 996.

[17] T.B. Fowler. “A Short Tutorial on Fractals and Internet Traffic”. Telecommunication Review. 1999. vol. 10. pp. 1-15,

[18] T. Hagiwara, H. Majima, T. Matsuda, and M. Yamamoto. “Impact of Round Trip Delay Self-Similarity on TCP Performance”. APCC2001. pp.149-152.

[19] V. Jacobson and M. Karels. “Congestion Avoidance and Control.” ACM SIGCOMM, 1988. pp. 273-288.

[20] K. Pentikousis. “Can TCP be the transport protocol of the 21st century?” ACM Crossroads, Special Issue: Mobile and Wireless Computing. 2000. 7(2).

[21] M. Salleh and A.Z. Abu Bakar. “Chaos Control in Wired Cum TCP Contextualization and Wireless Environment”. Proceeding ACIT 2004, Algeria.

[22] S. Low. “Equilibrium & Dynamics of TCP/AQM”. Sigcomm, August 200.

[23] S. Low, L. Peterson, and L. Wang. “Understanding TCP Vegas: A Duality Model”. Journal of the ACM (JACM). 2002. 49(2). pp. 207-235.

[24] L. Breslau, D. Estrin, K. Fall, S. Ford *et. al*. “Advances in Network Simulation.” IEEE Computer. May 2000. 33(N5). pp. 59-67.

[25] G. He, Y. Gao, J.C. Hou and K. Park. “A Case for Exploiting Self-Similarity of Network Traffic in TCP Congestion Control”. Computer Networks: The International Journal of Computer and Telecommunications Networking. 2004. 45(6).

[26] M.E. Crovella and A. Bestavros. Explaining World Wide Web Traffic Self-Similarity. Technical Report TR-95-015 Boston University, 1995.

[27] M. Crovella, and A. Bestavros, “Self-similarity in World Wide Web Traffic: Evidence and Possible Causes”. ACM SIGMETRICS. 1996. pp. 160-169.

[28] H. Kettani, and J.A. Gubner, “A Novel Approach to the Estimation of the Hurst parameter in Self-Similar Traffic”. Proceedings of Local Computer Networks, 27th Annual IEEE Conference. 2002. pp. 160 – 165.