PERFORMANCE OF TRANSMISSION CONTROL PROTOCOL ON SELF-SIMILAR TRAFFIC

Mazleena Salleh[1]
Ahmad Zaki Abu Bakar[2]

Faculty of Computer Science and Information System
University Technology Malaysia

[1]mazleena@fsksm.utm.my
[2]zaki@ fsksm.utm.my

ABSTRACT

Internet traffic will continue to expand in terms of both volume and users, and transmission congestion protocol (TCP) is accounting for more than 90% of the total traffic volume. It is well known that network traffic exhibit self-similar behavior and cannot be described by traditional Markovian models such as the Poisson process. The burstiness that occurs in self-similar traffic needs to be managed so as to enable optimal planning and to achieve higher quality of service. In relation to this, there must be a full understanding of the global network traffic especially on the impact of self-similar traffic on network protocol.

In this paper we focus on experimental quantitative comparison of four TCP variants namely Tahoe, NewReno, Vegas, and SACK running over self-similar traffic. With the aid of network simulator, NS2, the impact of TCP microflow is observed, and data analysis such as packets dropped and retransmission, throughput, and behavior of the data received over time is analyzed. We employed optimization method to measure Hurst parameter, the degree of self-similarity. Our study shows that at low and highly congested traffic, network traffic does not exhibit self-similarity. In addition TCP microflow exhibit self-similar behavior when traffic load is moderately congested and our result revealed that Vegas are the best performance TCP variant in self-similar traffic.

Keywords: Self-Similar, Hurst Parameter, Transmission Control Protocol, Autocorrelation.

1.0 INTRODUCTION

In the last decade, computer networks have experienced tremendous growth. More and more computers are getting connected to both private and public networks where most of traffic is generated by 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 (Rao and Chua, 2002).

In addition to the protocol usage, real-life traffic presents a self-similar behavior whereby changing timescale does not change the behavior of the traffic pattern (Leland *et. al.*, 1994). Empirical evidence cited by Peha (2002) 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.

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: We introduce the concept self-similar in Section 2 and in Section 3 we explain briefly the implementation of TCP variants. The research methodology is described in Section 4 while Section 5 discussed the results of the experiment. Finally we conclude our paper in Section 6.

2.0 SELF-SIMILAR TRAFFIC

Previously voice is the only kind of communication traffic and it had, and has, well-known characteristics, namely Poisson interarrival rates and exponential call length distribution (Fowler, 1999). Measurement of critical values of important parameter was easily done and queueing theory permitted design of voice networks to meet any desired performance characteristics. As communication technology evolves and new standards were developed, network traffic became more complex than ever. There are many varieties of network connectivity, architecture, and equipment, and subsequently, resulted in different type of traffic flows. In the early 1990's an astonishing discovery was unveiled by researchers at Bell Core regarding with the network traffic. Unlike voice network, packet networks exhibit burstiness on multiple time scales and do not follow the formal models for packet traffic such as Poisson, packet train models or fluid flow models (Leland *et. al.*, 1994). Later this phenomenon was also found to exist in wide area networks (Paxson and Floyd, 1995 and Feldmann *et. al.*, 1998). The occurrences of burstiness on multiple time scales follow the behavior of self-similarity where changing timescale does not change the behavior of the traffic pattern.

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, which are distributions whose tails decline via a power law with small exponent, less than 2. The presence of heavy tails in lengths of individual flows can be shown to induce self-similarity in network traffic. Heavy-tailed properties have been found in file sizes and user thinking time, flow (session) duration, as well as packet inter-arrival time distributions in the Internet (Park, 1996).

Local area network traffic as well as Internet is self-similar traffic where peak behavior shows over all timescales. Figure 1(a) and Figure 1(b) are two samples of traffic traces: Ethernet traffic traces taken in 1992 at Bellcore Morris Research Engineering Center and traces from Defense Advanced Research Projects Agency (DARPA) recorded in 1999 respectively. Each of the diagrams was recorded at different time scale of 0.1, 1.0 and 10 seconds. Even though the traffic traces were taken several years apart, the behavior of self-similarity still exists in these traffic traces.

Poisson traffic peak does exit at a small timescale, but unlike self-similar traffic, the peak decreases rapidly when moving to larger time intervals. This is proved by Willinger *et al.*,1997 in his report where he compared actual measurements with synthetic Poisson model and concluded that the Poisson model lacks the burstiness over large time scales that are present in actual traffic measurements. 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 (Paxson and Floyd, 1995). Table 1 tabulates the comparison of Poissor with self-similar traffic flow.

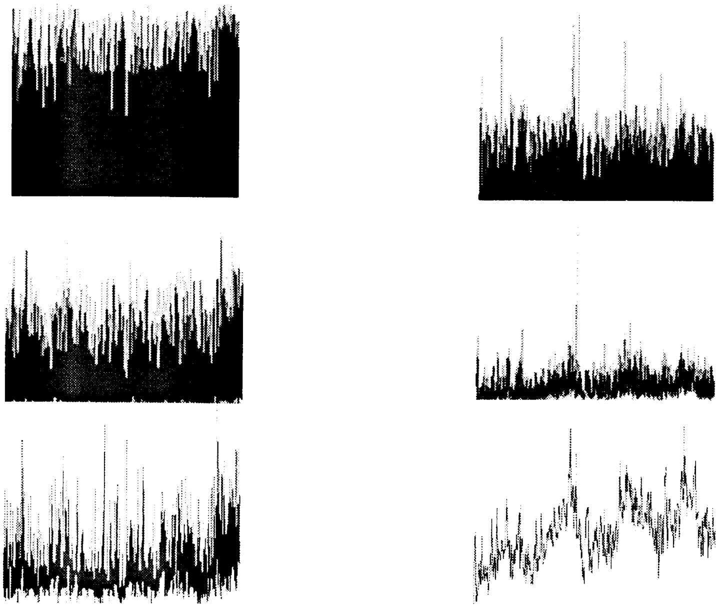

Figure 1(a) Ethernet Data from Bellcore

Figure 1(b) DARPA Data

Figure 1. Traffic Volume at Time Intervals of 0.1, 1 and 10 Seconds

Table 1. Comparison of 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

Burstiness in network traffic is of great importance when designing a network and the issue of self-similarity has also been addressed in various studies from many aspects. Since Leland *et al.*'s seminal paper (1994), various contributions have shown self-similarity to be ubiquitous characteristic of data network traffic. The study on TCP traffic has shown that TELNET packet arrivals and FTP data connection in WAN traffic is also self-similar (Wischik, 2001). Other studies include self-similarity impact on network performance (Park, Kim and Crovella, 1997), modeling techniques (Crovella and Bestavros, 1995) and causes of the appearance of self-similarity (Stallings, 1998) have been put forward.

2.1 Hurst Parameter

Self-similarity manifests itself in several ways: (i) slowly decaying variance that is variance decreases very slowly, even when the sample size grows quite large, (ii) Long-range dependence (LRD), (iii) non-degenerate autocorrelations, and (iv) Hurst effect. In this study we focus on the latter manifestation that is Hurst effect or Hurst parameter, H, that is the key measurement degree of self-similarity.

A statistical measurement that is closely related to self-similar is autocorrelation function (ACF). Autocorrelation measures the relationship between a random variable and itself, at different time lags and ACF shows the value of the autocorrelation coefficient for different time lags. For most processes such as Poisson, ACF drops to zero very quickly, immediately, or exponentially fast. For self-similar processes, the autocorrelation function for the aggregated process is indistinguishable from that of the original process. In other words a self-similar time series has the property that when aggregated, leading to a shorter time series in which each point is the sum of multiple points, the new series has the same ACF as the original. If a self-similar process has observable bursts on all time scales, it is said to exhibit LRD; values at any instant are typically correlated with value at all future instants. Given a time series $X = (X_t$: $t = 0, 1, 2,\dots)$ and defined m-aggregated series as

$$X^{(m)} = (X_k^{(m)} : \ k = 1, 2, 3, \dots)$$

ACF can be determined by

$$\rho(k) = E[(X_t - \mu)(X_{t+k} - \mu)]$$

If X is self-similar, then all series of $X^{(m)}$ will have same ACF for all m that is

$$\rho_{x^{(m)}}(k) = \rho_x(k)$$

As $k \rightarrow \infty$,

$$\rho(k) \cong k^{-\beta} \quad \text{where } 0 < \beta < 1 \text{ and,}$$
$$H = 1 - \beta/2 \quad \text{with } 0.5 < H < 1$$

For stationary, ergodic process, $\beta = 1$ and the variance of the time average decays to zero at the rate of $1/m$. For a self-similar process, the variance of time average decays more slowly. For a perfectly self-similar process, $\beta = 0$ and the variance of the time average does not decay at all. When $H > 0.5$, this is called long-range dependence (Wischik, 2001). For Poisson traffic, $H \leq 0.5$ while ideal self-similar traffic, $H = 1$.

There are several Hurst estimators: time domain estimators, and frequency and wavelet domain estimators. Time domain estimators investigate the power-law relationship between a specific statistical property in a time series and the time-aggregation block size *m*. LRD exists if the specific property versus *m* is a straight line when plotted in log-log scale The slope of the line is an estimate of the Hurst exponent. These estimators use several methodologies: rescaled range statistic (R/S), absolute value, variance, and variance of residuals. Frequency and wavelet domain estimators operate in the frequency or wavelet domain. Similarly to the time domain methodologies, the investigation are based if a time series' spectrum or energy follows power-law behavior. These estimators include the Periodogram, the Whittle, and the wavelet Abry-Veitch (AV) estimators (Karagiannis *et. al.*, 2004).

Optimization method is another Hurst estimator that is based on model-testing paradigm that exploits the correlation structure of a time series (Kettani, and Gubner, 2002). Comparing with variance time plot or R/S plot, the computational time of optimization method is faster and in addition yields better confidence intervals as compare to wavelet.

2.2 Problem Arises from Self-Similar Behavior

The performance of network connections can be characterized by two main elements: bandwidth and delay. The higher the bandwidth is, the higher the end-to-end throughput and the better quality-of-service for the applications will be. Delay, on the other hand, is important in order to guarantee fast response from the network. In self-similar traffic, presence of concentrated periods of congestion and concentrated periods of light network load at a wide range of time scales introduces new complexities in the management of network performance which effect 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 (Park, 1997).

During congestion or the occurrences of burstiness packet loss is amplifies (Fowler, 1999). 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 H 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 (Hagiwara *et. al.*, 2001). 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.

3.0 TRANSMISSION CONGESTION PROTOCOL (TCP)

TCP traffic flow make up of 90% of the total Internet traffic. Van Jacobson, 1988 introduces TCP and it is based on a complex interaction of several algorithms 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 (Pentikousis, 2000). 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 mechanism. Two main variables used in this mechanism are, *cwdn* (congestion window) and (slow start threshold value). There are two distinct phases that is the slow start and congestion avoidance phase. Figure 2 illustrates the overall operation of TCP (Salleh, 2004). 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 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) 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 (Low, 2001). Unlike other TCP variants TCP-Vegas 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 (Low, 2002).

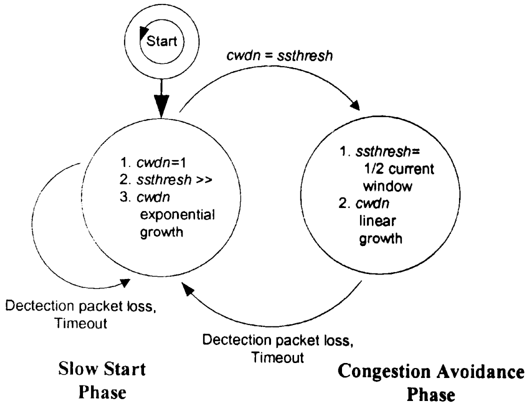

Figure 2. TCP Transition Diagram

4. METHODOLOGY

In this study we have conducted two types of experiments: (i) investigation of self-similarity in real network traffic and, (ii) analysis of TCP variants performance on self-similar traffic. In the first experiment the LAN network traffic traces of Faculty of Computer Science and Information System (FSKSM), UTM was captured using TCPDump, a UNIX packet-sniffing tool. TCPDump gather data from a network and display them in a semi-coherent fashion and by specifying a filter expression, only packets of interest will be captured and analyzed (Fuentes and Kar, 2005). With the aid of SQL database, the packets size is extracted together with its timestamp from the traces. This will be the time series $f(t)$ that will be analyzed.

For the second experiment, Network Simulator (NS2) of VINT project (Breslau *et. al.*, 2000) is used to simulate the network configuration as depicted in Figure 3. The dumbbell topology consists of TCP senders and receivers and a pair of router that have a queue buffer size of 50. The bottleneck link has the capacity of 50Mbps, a propagation delay of 25ms and a drop tail queue (He *et.al.*, 2004). Two kinds of traffic sources are generated: self-similar traffic sources for the background aggregated traffic (node 1 to n), and a microflow of TCP variant (node x) from which measurements will be taken.

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 (Crovella and Bestavros, 1995). In this case, TCP packets are generated using the on-off Pareto model with the rate parameter of 1M and shape value of 1.4 at each of the sender nodes. Figure 4 shows an example of the self-similar aggregated traffic traces captured at the bottleneck link. Traffic traces of 100 seconds were collected from the path x sender to x receiver and statistics such as packet arrival, packet drop, and packet retransmitted was recorded for different *H* of the aggregated traffic. In addition, the throughput connection and *H* parameter based on acknowledgement received at fixed time granularity is calculated. Throughout the experiments, we employed optimization method to calculate *H* parameter.

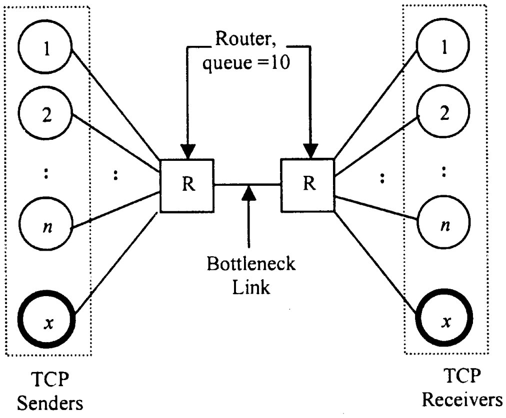

Figure 3. Network Configuration Setup

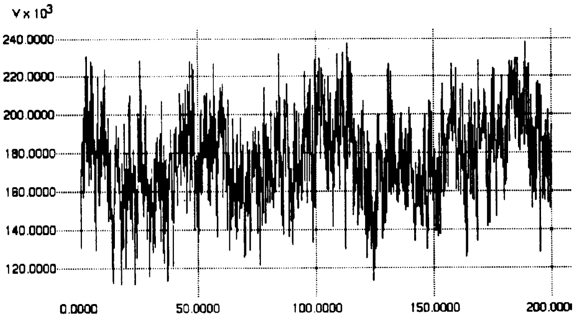

Figure 4. Traces of Synthesized Aggregated Traffic

5.0 RESULTS AND DISCUSSION

LAN traffic traces of FSKSM were captured on April 28, 2005 and during the interval of five minutes, 4880 packets (3,018,692 bytes) were collected. The traces are as shown in Figure 5 and the packets usage is tabulated in Table 2. From this result it clearly proves that TCP dominate Internet traffic as stated by Crovella and Bestavros (1996). The calculated Hurst parameter, *H* is 0.88, which is second order self-similar and does exhibit LRD.

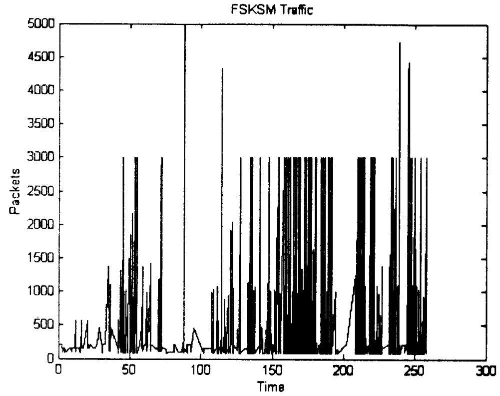

Figure 5. FSKSM Traffic Traces

Table 2. Packets Usage of FSKSM Traffic

Internet Traffic:		96.54%
TCP	90.7%	
UDP	5.84%	
Logical Link Control		2.5%
Address Resolution		0.94%

In the simulation experiment, we vary the number of nodes from 10 to 60 so as to investigate the *H* parameter in the aggregated traffic traces. The result is as shown in Figure 6 where *H* increases from 0.77 to 0.83. Beyond 60 connections, the behavior of the aggregated traffic does not exhibit self-similarity behavior and the number of packets drops from the

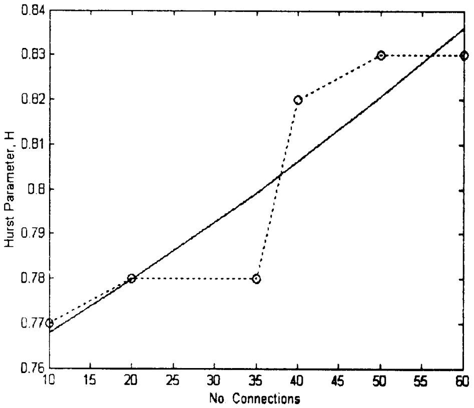

Figure 6. *H* vs Number of Connections

bottleneck line increases tremendously. This behavior reflects the congestiveness of the link where the bandwidth usage is at the maximum capacity. From this result we can categorize the traffic load as low, congestive and highly congestive traffic with respect to *H* as depicted in Table 3.

Table 3. Categorization of Traffic Load

Traffic	**No. Connections**	**Pkts Dropped**	***H***
Low	< 30	None	< 0.8
Congestive	40 - 60	< 20	$0.8 \leq H \leq 0.83$
Highly Congestive	> 60	> 20	$H > 0.83$ or not SS

For each different level of traffic load we inspect the performance of TCP variants. Figure 7 demonstrate the throughput of four TCP variants under study: Tahoe, NewReno, Vegas and SACK, while Figure 8 and Figure 9 shows the number of packets dropped and packets retransmitted respectively. At low traffic load, the throughput of all the four TCP variants increases rapidly until it reaches its optimum throughput at approximately 725 Mb where it starts to increase gradually. Initially all TCP variants will start with the slow start phase where *cwnd* grows exponentially. This causes the rapid increase in throughput. Once the size of *cwnd* reaches *ssthresh*, the protocol will enter the congestion avoidance phase and *cwnd* will grow linearly and hence the slow increase in throughput. At this traffic level Vegas throughput is out performed by other TCP variants as shown in Figure 7(a).

When traffic load is congested all four TCP variants experiences packets dropped. Depending on the protocol of the TCP variants, each of them will enter their individual phases of recovery. The oscillation from the recovery phase to normal sending mode causes the throughput to fluctuation as illustrate in Figure 7(b). However Vegas throughput is more stabilized compare to the other three TCP variants. For highly congested traffic load, TCP variants experience an increase in packets dropped and as the result the throughput fluctuate vigorously with Vegas being the best performed TCP variants as depicted in Figure 7(c). This is due to congestion feedback mechanism of Vegas that facilitate in reducing packet losses. This can be further explained by inspecting the number of packets dropped and packets retransmitted. Initially Vegas experienced higher packets dropped compared to other TCP variants but the losses decreases as traffic load increases. On the other hand Tahoe, NewReno and SACK faced much higher packets dropped as traffic load increase and the redundant retransmission of these packets degrade the overall throughput. Unlike Tahoe, NewReno and SACK successfully retransmitted the exact packets that have been dropped. However, Tahoe implemented with only fast retransmit algorithm have retransmitted more than the number of packets dropped packet. This means that it might have retransmit packets that already successfully delivered. The impact from this flaw causes Tahoe to become the least performed TCP variants under highly congestive self-similar traffic load.

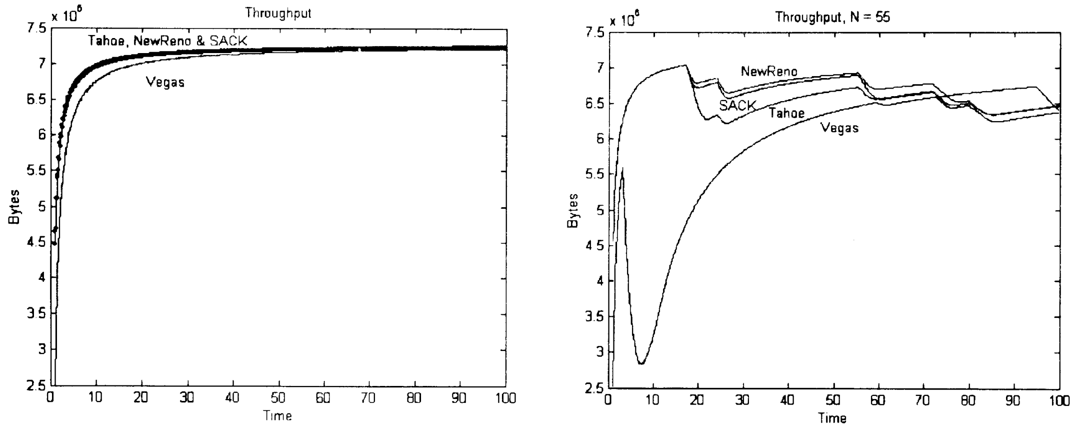

Figure 7(a). 20 Connections: Low

Figure 7(b). 55 Connections: Congested

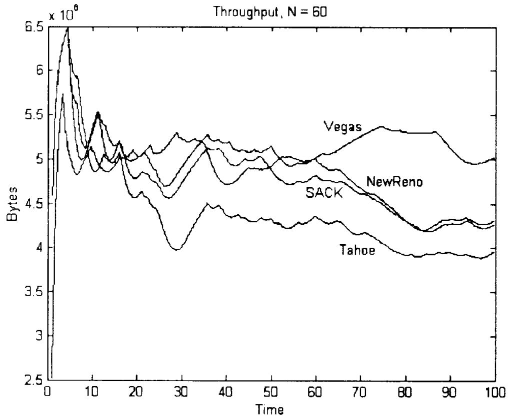

Figure 7(c). 60 Connections: Highly Congested

Figure 7 Throughput vs Number of Connections

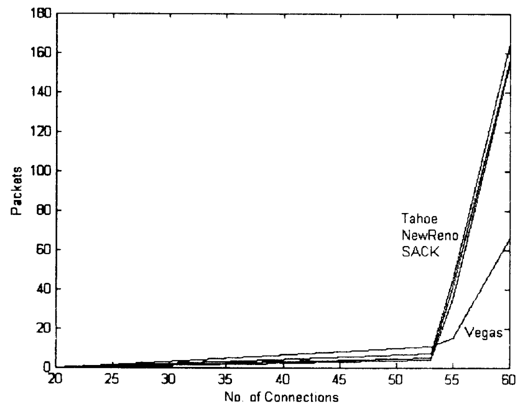

Figure 8. Number of Packets Dropped

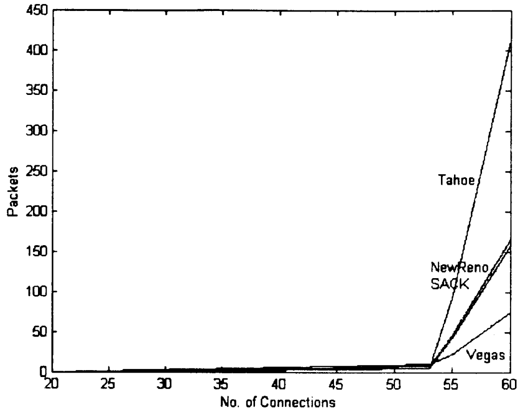

Figure 9. Number of Packets Retransmitted

NewReno and SACK act in a very similar behavior and thus their performance does not differ much from each other in all the cases of traffic load. The deployment of partial and selective acknowledgement allows the TCP protocol to behave intelligently when responding to packet losses and thus improves the retransmission control of packets when traffic is congested. However the reactive congestion control strategy in NewReno, SACK (and hence also Tahoe) caused them to be out performed by Vegas. In other words, the algorithm repeatedly increased the load in an effort to find the point at which congestion occurs, and then back off. On the other hand, the proactive approach of Vegas whereby it attempts to reduce its sending rate before packets start being discarded by the link manage to reduce packets dropped and thus improves the throughput performance.

With respect to the self-similar behavior of TCP, Table 4 tabulated the calculated *H* based on the ACK received at the sender. At low traffic load, TCP does not demonstrate self-similar behavior but as the load increases and packets dropped occurs, the microflow traces exhibit self-similarity with *H* slightly higher than that of the aggregated traffic. Here we can say that the congestion control of TCP causes it to sustain the self-similarity in its operation. Based on the perspective of chaotic system, when a system is prone to chaotic behavior (e.g. long-range dependence and self-similarity), a slight change in initial conditions can impact performance into the distant future. Congestible systems such

Table 4. Hurst Parameter of TCP Microflow Traces

	< 50 connections		**55 connections**		**60 connections**	
	H	Pkt Dropped	*H*	Pkt Dropped	*H*	Pkt Dropped
Tahoe	Not SS	0	0.86	35	0.8	155
NewReno		0	0.83	41	0.84	157
SACK		0	0.82	45	0.84	165
Vegas		0	0.91	15	0.87	66

as TCP are a likely candidate for this phenomenon. In a congestible system, as input load increases from 0, the useful output first increases, then reaches a maximum, and finally falls. When useful output such as throughput decreases with respect to input load, the system is considered to be congested. When output is at its maximum, a slight perturbation can cause the system to operate in either the congested or uncongested region for extended periods (Peha, 1997).

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 with $H = 0.88$. This result conformed to the findings by Leland (1994) and, Crovella and Bestavros (1996). From our simulation experiments we have categorized traffic load based on the H parameter and based on this categorization of traffic load, we investigate the performance of TCP variants: Tahoe, NewReno, Vegas and SACK. At low load, TCP microflows do not exhibit self-similarity behavior. However at higher load, all TCP variants microflow show evidence of self-similarity in the traffic traces which is due to the control procedure of TCP. In term of performance, Vegas throughput exceeds the other three TCP variants under congested self-similar traffic. Therefore here we proposed TCP Vegas as the transport protocol in a self-similar traffic such as LAN and WEB so as to optimize network performance.

REFERENCES

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

Crovella, M.E., and Bestavros, A. (1996). *Self-similarity in World Wide Web Traffic: Evidence and Possible Causes.* ACM SIGMETRICS, pp.160-169, 1996.

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

Feldmann, A., Gilbert, A. C., Willinger, W., and Kurtz, T. G. (1998). *The Changing Nature of Network Traffic: Scaling Phenomena.* ACM Computer Communication Review, Vol. 28, pp. 5-29, Apr. 1998.

Fowler. T.B. (1999). *A Short Tutorial on Fractals and Internet Traffic.* Telecommunication Review. 1999. Vol. 10, pp. 1-15.

Fuentes, F. and Kar, D.C. (2005). *Ethereal Vs. TCPDump: A Comparative Study on Packet Sniffing Tools for Education Purpose.* JCSC 20(4), April, 2005.

Hagiwara, T., Majima, H., Matsuda, T., and Yamamoto, M. (2001). *Impact of Round Trip Delay Self-Similarity on TCP Performance.* Proceedings of Computer Communication and Networks Conference, pp. 166-171, Oct. 2001.

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

Jacobson, V., and Karels. M. (1998). *Congestion Avoidance and Control.* ACM SIGCOMM, 1988.

Karagiannis, T., Molle, M., and Faloutsos, M. (2004) *Long-Range Dependence Ten Years of Internet Traffic Modeling*, Internet Computing, IEEE, Vol 8(5), pp. 57 – 64, Sept.-Oct. 2004.

Kettani, H., and Gubner, J.A.(2002). *A Novel Approach to the Estimation of the Hurst Parameter in Self-Similar Traffic.* Proceedings of Local Computer Networks, 27th Annual IEEE Conference, pp.160 – 165, 2002.

Leland, W., Taqqu, M., Willinger, W., and Wilson, D. (1994). *On the Self- Similar Nature of Ethernet Traffic* (Extended Version). IEEE/ACM Transactions on Networking, 2(1), pp. 1-15, February 1994.

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

Low, S. (2001). *Equilibrium & Dynamics of TCP/AQM.* Sigcomm, August 2001.

Park, K. (1997). *On the Effect and Control of Self-Similar Network Traffic: A simulation Perspective.* Proceedings of the 29th conference on Winter simulation. Atlanta, Georgia, United States, pp. 989 – 996, 1997.

Park, K., Kim, G. and Crovella, M. (1997). *On the Effect of Self-Similarity on Network Performance*, Proceedings of the SPIE International Conference On Performance and Control of Network System, pp. 296-310, Nov 1997.

Park, K., Kim, G.T. and Crovella, M.E. (1996) *On the Relationship Between File Sizes, Transport Protocols, and Self-Similar Network Traffic.* Proceedings of the International Conference on Network Protocols, pp. 171-180, October, 1996.

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

Peha, J.M. (1997). *Protocols Can Make Traffic Appear Self-Similar.* Proceedings of IEEE/ACM/SCS Communication Networks and Distributed Systems Modeling and Simulation Conference, pp. 47-52, 1997.

Pentikousis, K. (2000). *Can TCP be the Transport Protocol of the 21st Century?* ACM Crossroads, 7(2), Special Issue "Mobile and Wireless Computing", Winter 2000.

Rao, N.S.V. and Chua, L.O. (2002). *On Dynamics of Network Transport Protocol.* Proceedings of Workshop on Signal Processing, Communications, Chaos and Systems, pp. 39-53, 2002.

Salleh, M. and Abu Bakar, A.Z. (2004). *TCP Contextualization and Chaos Control in Wired Cum Wireless Environmen.* Proceedings of ACIT 2004, Algeria.

Stallings, W. (1998) *High-Speed Networks: TCP/IP and ATM Design Principles*, Prentice Hall, New Jersey, 1998.

Willinger, W.; Taqqu, M.S.; Sherman, R.; Wilson, D.V. (1997). *Self-Similarity Through High-Variability: Statistical Analysis of Ethernet LAN Traffic at the Source Level.* IEEE/ACM Transactions on Networking, 5(1), Febuary, 1997.

Wischik, D. (2001). *Implications of Long-Range Dependence.* Statistical Laboratory, University College London. February 2001. Available at: http://www.cs.ucl.ac.uk/staff/D.Wischik/Interests/Topics/optnet-lrd.pdf