A Light Weight Traffic Source for Web Traffic Simulation

Sireen Malik, Ulrich Killat
Digital Communication Systems
Technical University of Hamburg-Harburg, Germany
E-mail:{s.malik, killat}@ tuhh.de

***Abstract*— The inclusion of Transmission Control Protocol (TCP) is necessary for realistic simulations of data networks. However, the operation of HTTP-TCP sources puts a heavy tax on the simulations leading to two major scalability problems: the resources required by each source put a limit on the total number of sources that can be simulated, and the number of events generated by every simulated connection leads to long simulation times. The scalability problems are major obstacles for realistic simulations of optical networks. In this paper we present a new type of traffic source that generates traffic statistically similar to the traffic produced by a number of HTTP-TCP sources. The source is based upon an integrated packet-session levels model that captures the web user-behaviour as well as the TCP characteristics. To reduce the number of events we replace TCP's packet-feedback loop with a map-feedback loop. The simple solution eradicates the need for the acknowledgment-traffic flowing in the reverse direction of the data-traffic. To compare, we first generate aggregate traffic from realistic HTTP-TCP sources and then match the traffic generated by the Light Weight Traffic Source (LWTS). We show by comparing key traffic statistics that the source qualifies as a good candidate for generating network traffic.**

I. Introduction

Computer simulations attempt at modelling reality, however, even under the best conditions the implemented models are approximations. It is always arguable what level of detail is important for the specific characteristics under study? The same is true for network simulation. The findings [1,2] in the 1990s that the Internet traffic is Long Range Dependent (LRD) in its nature has not only had a deep impact on the issues of traffic engineering and network dimensioning but also on network simulations' methodology. From this perspective it becomes important to generate LRD traffic. LRD is attributed mainly to session characteristics, or user-behaviour. From the mathematical point of view it means that the traffic shows correlations at all time scales. From the performance point of view the most important repercussion is that the queuing behaviour is hugely different from the classical behaviour resulting from Poisson, or memory-less, process. Various methods have been proposed and a summary is presented in [3]. A sophisticated approach is the generation of traffic by super-position of on-off sources [4]. If either, or both, the on- and off-times are heavy-tailed distributed then the resulting traffic is LRD in nature.

The protocol-specific mechanisms and congestion control algorithms operating on small time scales give rise to complex structural properties which are different from large-time scaling behaviour. To complete the total picture, the inclusion of TCP is a first step towards understanding the complex phenomenon of multifractality which was discovered recently and is mainly attributed to the transport protocol [5,6].

In [7,8,9] the authors introduced the HTTP-TCP source which includes a full implementation of the Transmission Control Protocol (TCP) to transport the session-level data. It is a closed-loop source as it integrates TCP. The HTTP-TCP source encompasses both aspects of the scaling behaviour. A typical user is mimicked by an on-off behaviour at the session level. During the on-phase TCP transports the heavy-tailed web-pages from the server to the client. While the super-position of on-off sources leads to LRD traffic, the inclusion of TCP brings in the small-time scale behaviour thus leading to very realistic simulations.

Large capacity networks carry large volumes of traffic. A large number of traffic sources would be required to simulate a high speed network. However, each source puts a demand on the computer resources (CPU and memory). Therefore, the finite available computing resources bound the number of sources that can be simulated. Secondly, each source generates a number of events during the course of simulation. Understandably, a higher number of sources leads to longer simulation times. The scalability argument is true for every type of traffic source. However, it is even more serious for the HTTP-TCP source as it involves a client-server pair: both occupy memory-space and produce traffic. Events are generated for the data-traffic flowing from the server to the client as well as for the acknowledgment-traffic in the reverse direction. Undoubtedly, the realistic simulations come at the expense of longer simulation times and larger computing resources. Given that, the realistic simulation of optical networks at the packet level appears as an impossible task.

To address these scalability problems a solution is to use a few HTTP-TCP sources, or pilot sources, and replace the rest of the traffic with "replacement-traffic". Ideally, the key aggregate traffic characteristics should stay the same despite the replacement. Such a swap would still allow to observe the performance of any protocol or algorithm through the pilot sources. The most important step for this strategy to work is to have a replacement traffic source which would not only solve the above mentioned scalability issues but also match the important traffic statistics with high fidelity.

We introduce a new type of traffic source which generates traffic statistically similar to the traffic generated by a number of HTTP-TCP sources. We call it the Light Weight Traffic Source (LWTS). The LWTS is an on-off source. At the session level, it has exactly the same structure as that of an HTTP-TCP source. We will elaborate on this point in the coming sections. The difference between the two sources comes in the way the two

The research reported in this paper falls under the MultiTeraNet (PLANIP) project funded by the Ministry of Education and Research, Germany.

sources transport the web-pages. To address the second scalability problem (number of events) we use an approximate TCP protocol which we name as Pseudo-TCP (P-TCP). This modified transport protocol includes the dominant characteristics of real TCP, for example the Slow-Start (SS) behaviour, Congestion-Avoidance (CA), Fast-Retransmit and Recovery (FRR), and an approximate Exponential Back-off (Exp-BO) behaviour also referred to as Karn's algorithm. It also incorporates a mechanism to estimate the RTT distribution which plays an important role in the traffic characteristics.

To reduce the number of events, we eliminate the acknowledgment-traffic by introducing a simple solution based on memory-maps. We use two memory-maps as databases. One database keeps track of the packet-losses and the other keeps track of the End-to-End (E2E) delay of each packet. The P-TCP "reads" these two maps and reacts accordingly. We will explain its complete behaviour and significant differences with the real TCP in the following sections.

To compare the two sources we first generate traffic from a number of HTTP-TCP sources and measure key traffic statistics such as the throughput (TP), the coefficient of variation (CV), the auto-covariance function (Z) and the Hurst parameter (H). The study in itself yields interesting results which are in agreement with the results measured from real traces. Then we use LWTS's to produce the aggregate traffic. We demonstrate, by matching the above stated statistics that the LWTS faithfully matches the traffic from HTTP-TCP sources. The simulations, however, are faster and lighter.

In section II we present the user-behaviour model for a HTTP-TCP traffic source. In section III we explain the working of the LWTS and how we tackle the above-mentioned scalability issues. In the later sections we discuss the traffic characterization parameters, the simulation setup and then the results.

II. HTTP-TCP SOURCE

We first briefly discuss the details of HTTP-TCP source. Let ist, depicts the time between the arrivals of consecutive user-requests for web-pages at the web-server. On getting a user-request the HTTP protocol at the session level fetches the requested web-page and passes it to the TCP. Let V be the average file-size which the web-server generates at rate r. Let Z be the average number of objects per web-page. If F denotes the average web-page size then $F = V * Z$. TCP transports the web-page from the web-server to the web-user. The transmission of an average web-page at the session level is done in average T_{on} time. After the download of a web-page, the user remains inactive for an average T_{off} time, or the think-time, before making the next request. Each web-user cycles through this on and off behaviour.

HTTP-1.1 is considered at the session layer because of its prevalence. It sends the web-pages through a persistent connection. This means that a single connection is used to transport all the files in the web-page. The usage of parallel connections is not modelled here. At the transport layer, TCP first establishes a connection between the web-server and the web-client with a three-way handshake and then transmits the actual data. Starting from 1 packet, TCP keeps doubling the number of packets in a window until it reaches its slow-start threshold or there is a packet loss in the former case it switches to the congestion-avoidance phase. For TCP the time between the transmission of two consecutive windows is the RTT. Given that there are no packet losses the connection will open to the maximum congestion window ($MaxCWND$) and then maintain this window till the whole file is transmitted. If there are losses TCP will switch to other phases. For a complete discussion on the operation of TCP we refer to [10].

Given that the most dominant component of Internet traffic is mainly composed of short TCP transfers [11], or "mice" it is fair to assume that only the slow-start phase of the TCP is enough to carry "most" of the web-traffic. This is the basis of the HTTP-TCP model. It seems counter-intuitive to talk of mice flows and heavy-tailed distribution in one breath! The term "mice flows" is used to indicate small flows on an average, and the heavy-tailed distribution indicates that some of the flows are really large. It may be argued that how much these large flows contribute to the traffic, and does the simulated traffic reflect reality? Clearly, it is an approximation which would not hold true for "elephant" flows that come, for example, from P2P applications. The model will have to be adapted to simulate reality even closer. However, the results of simulations indicate that the approximation is not actually that bad and that most of our findings conform to the statistics available from actual Internet traffic traces.

Assuming that in an appropriately dimensioned network there are negligible packet-losses in the slow-start phase, in [7,8,9] the authors introduced the following expression for the throughput (TP) of the HTTP-TCP traffic.

$$TP = \frac{F}{N * RTT + T_{off}}. \tag{1}$$

Where N is the average number of $RTTs$ required to transport an average web-page of size F. Clearly the source on-time, $T_{on} = N * RTT$.

III. LIGHT WEIGHT TRAFFIC SOURCE

This work is an extension of [24] in which we introduced the first version of this source. It was modelled as an open-loop source. That work is extended in this paper by integration of a feedback loop into the model.

A. User-behaviour

To keep as close as possible to the HTTP-TCP source we use exactly the same user-behaviour at the session level - there are no differences. The two sources differ in the way the web-pages are transported. Earlier P-TCP was introduced, in the following sub-section its details are given.

B. Reduction of number of events through Pseudo-TCP

To reduce the number of events, we look inside the working of TCP. As explained earlier, each user-request results in generation of a web-page which is transported by TCP. TCP is ACK-clocked, i.e. new packets are sent only in response to acknowledgments from the receiver. Each packet causes an acknowledgment packet in the reverse direction. This is one of the major functions of TCP's feedback loop. The other major function of the feedback loop is to provide an estimate of the RTT. Both functionalities depend on network conditions. The protocol adjusts its rate based on the network conditions, for example, in case of losses TCP will generate packets at a lesser rate and vice-versa. This variable rate brings the multifractal structure to the traffic at the small time scales.

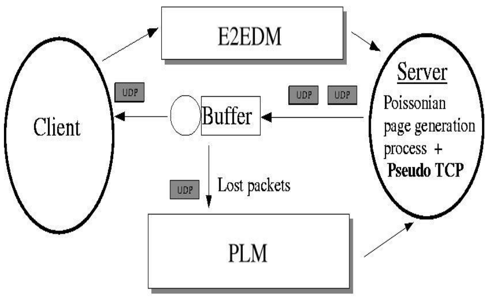

Fig. 1. Operation of LWTS

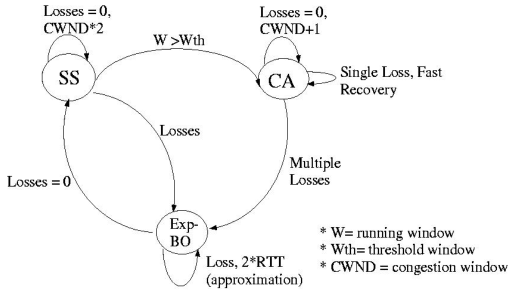

Fig. 2. Pseudo TCP phase transitions

The easiest way to reduce the number of events would be thus to eliminate the feedback loop reducing the number of events per source by half. However, to keep as close as possible to the reality, there must be a way to maintain the above-mentioned functionalities of TCP in the new source. We introduce a simple solution based on memory-maps. We use two memory-maps as databases. One database keeps track of the packet-losses and is called Packet Loss Map (PLM). The other keeps track of the E2E packet delay and is called E2E Delay Map (E2EDM). PLM is written directly by the buffers/queues whenever they lose a packet. The E2EDM is written by clients on reception of every packet. Every client calculates the E2E packet delay and writes it to the map. Fig. 1. gives a visual explanation of the idea. Because the need for real TCP is eliminated, the HTTP pages can be sent via UDP. In the following sub-sections we will point out the major commonalities and differences of LWTS with HTTP-TCP source.

1) Connection opening and closing: There are two aspects to TCP's connection opening phase via a 3-way hand-shake 1) 40Bytes signalling packets and 2) web-page requests from client to server. The first aspect is not incorporated at this stage. The second aspect is covered via the assumption that web-page request arrivals process is poissonian. We simply move this functionality to the server which picks a random off-time from a negative-exponential distribution between the transmission of consecutive web-pages. The mean value of the off-time distribution is set as the mean user-think time, or T_{off}. We do not model TCP connection closing.

2) Time-Out and Triple-Duplicates : There are no timers in P-TCP. This greatly simplifies the implementation of the protocol. The primary function of these timers is to estimate the Retransmission Time-Out (RTO), or the time after which the protocol will assume that a packet is lost and would take necessary steps to eradicate this situation. For P-TCP packet losses are written directly to the PLM and the information is read by the protocol. This also removes the need for triple-duplicates (TD) mechanism which also basically tells the protocol that a packet was lost and it has to be resent.

3) P-TCP phases: The P-TCP has SS, CA, FRR and Exp-BO phases in accordance with the spirit of TCP [RFC 2001 and 1122] and the original work by Jacobson [19]. Refer to Fig. 2. for more details. This figure is a state transition diagram explaining the functionality of P-TCP. Starting from the slow-start phase, the protocol starts sending data with a single packet in the first RTT (we will explain the estimation of RTT in the next sub-section). If there are no losses the protocol keeps doubling the Congestion Window $CWND$ for each RTT until the running window reaches a threshold $ssthresh$. This is the exponential window increase. The protocol then switches to CA phase in which the CWND is incremented by one segment after every successful, or lossless, transmission of a window of packets. This is the linear increase. This increase is followed till the protocol hits the Maximum Congestion Window $MaxCWND$, usually 65535Bytes. If there are no losses this window is maintained till the transmission of the web-page is finished.

If there are packet losses the buffer losing the packets writes the losses to the PLM. Before the protocol sends a new window of packets it reads the database for losses related to its specific connection. It does two operations based on this info 1) determines volume-to-send = volume-to-send + packet losses i.e., the web-page volume which was lost has to be retransmitted, and 2) decides on the next phase? Following possibilities arise:

* if there are a single or multiple losses in SS the next state is Exp-BO

* if there is a single loss in the CA reduce the W_{th} by half + 3 segments; set the next CWND equal to half of the running window (minimum 2); next state is CA. This is FRR.

* if there are multiple losses in CA the next state is Exp-BO.

It was decided to include Exp-BO phase because of its importance in inducing pseudo self-similarity under relatively high loss conditions [20]. However, our implementation is an approximation. Where as Karn's algorithm is based on the estimation of Retransmission Time Out (RTO), P-TCP only uses RTT estimates. The real TCP sends one packet and waits for an acknowledgment within its RTO. If the packet transmission is not successful, the RTO is doubled and a packet is sent again. The protocol keeps doubling the RTO till its 64 times the first RTO. This has an effect of reducing the packet sending rate by half which induces pseudo self-similarity. In Exp-BO of P-TCP the RTO is simply replaced by RTT. This mechanism approximates Karn's algorithm in that the rate is reduced by half in every RTT.

4) RTT estimation: When client receives a packet it calculates the E2E delay by simply noting the difference between the present system time and the time the packet was created, doubles this estimate so as to approximate the RTT and writes it to E2EDM. To keep close to the reality P-TCP uses the same Jacobson's estimator [19] as used by TCP of the HTTP-TCP source. P-TCP reads the database and adjusts the start of transmission of its next window. It is to be noted that both losses and RTT estimates are made at the end of the transmission of window and not on per packet basis as done by real TCP. This is an abstraction lower in granularity and slower in speed than real TCP but keeps close to the spirit of TCP.

It is important to consider that the purpose of this traffic source, as pointed earlier, is to replace a large chunk of traffic with high fidelity. We do not claim to have replaced the TCP protocol nor do we claim to have captured the whole essence of TCP's feedback loop. This source captures the most important characteristics of the user behaviour and integrates only those characteristics of the transport protocol which are relevant from the first and second order characteristics of the aggregate traffic. The more detailed TCP characteristics could be studied through the pilot HTTP-TCP sources.

C. LWTS throughput model

We model TCP also as an on-off source. It is an on-off structure within the on-time T_{on}. The protocol transmits an average window of W packets at rate r in TCP_{on} time, on an average. The protocol is inactive for TCP_{off} time, on an average. To come at a formula for the throughput of the LWTS we start with n, the average number of packets, required to transport an average web-page of F size. Let MSS be the maximum segment size MSS, then,

$$n = \left\lceil \frac{F}{MSS} \right\rceil . \tag{2}$$

Considering the assumption that only the slow-start phase is enough to transport an average web-page (as made for HTTP-TCP source),

$$N = \lfloor log_2 n + 1 \rfloor . \tag{3}$$

The average window-size W for an average web-page can be calculated by considering that $F = W * N * MSS$, which leads to

$$TCP_{on} = \frac{W * MSS.}{r} \tag{4}$$

Given than RTT is equal to the sum of average on- and off-times of the protocol, we can write, $RTT = TCP_{on} + TCP_{off}$. The LWTS has an on-off structure, at the transport layer, functioning within the on-phase of the on-off structure at the session layer. Intuitively,

$$T_{on} = N * RTT, \tag{5}$$

where N is the average number of RTTs to transport an average page. Now, the throughput for the LWTS can be be written as:

$$TP = r * \frac{T_{on}}{T_{on} + T_{off}} * \frac{TCP_{on}}{TCP_{on} + TCP_{off}}. \tag{6}$$

With a little manipulation, we get

$$TP = r * \frac{\frac{W*MSS}{r} * N}{N * RTT + T_{off}} = \frac{F}{N * RTT + T_{off}}. \tag{7}$$

This is an expression similar to (1). This is an integrated packet-session level model based on mean value analysis that ties the two layers in one neat formula. Throughput for M number of sources is then given as:

$$TP = M * \frac{F}{N * RTT + T_{off}} \tag{8}$$

IV. AGGREGATE TRAFFIC CHARACTERIZATION & SIMULATION SETUP

A. Traffic Characterization

Perhaps it is impossible to precisely characterize a stochastic process such as the Internet traffic. However, we make a judgment on the importance of some parameters over the others, and choose four parameters to characterize the web-traffic. For this study we used the packet inter-arrival-times process. The first parameter is the TP which is measured as average packet size over average packet inter-arrival time. The second parameter is the *coefficient of variation* (CV) and for a random variable is defined as the ratio of standard deviation to its mean. CV expresses the variations of the process normalized by the mean of the process. The information regarding the arrival of correlated arrivals is captured by the *auto-covariance function*. *Hurst parameter* (0.5<H<1) is specified as the fourth parameter. For a detailed introduction to the concepts of fractality or self-similarity, multi-fractality, LRD and heavy-tailed distributions [12,13,14] are recommended. Tersely, it is a measure of the *persistence* of a stochastic process. In other words, it measures the length of the LRD. As $H \to 1$, it indicates a greater degree of LRD. $H = 1$indicated a purely fractal process. $H = 0.5$ indicates the absence of LRD and the process is called short range dependent (SRD). Poisson process has H=0.5, and is SRD. There are many ways to measure the H parameter: V/T plot, R/S, etc. We use the the estimation technique developed in [15] based on the wavelet transform (WT) method.

B. Simulation setup for HTTP-TCP sources

For simulations we use the Ptolemy Simulator extended for network simulations at our department. Fig. 3 demonstrates the simulation setup for HTTP-TCP aggregate traffic trial. The core link has capacity C. The incoming links have capacity C'. S1, S2 and S3 depict the web-servers. C1, C2 and C3 are the clients. The delays for these links were kept fixed as 5msec for the core and 10msec for the incoming links. HTTP-1.1 was used at the session level. TCP Reno was used as the transport protocol for HTTP-TCP sources. From future optical networks point of view, we assume that the links would not be highly loaded given their large capacities. As a worst case scenario, because TCP suffers from congestion collapse for link loads greater than 75%, the utilization of the core link was kept around 65%-70%. The utilization of incoming edges was kept around 50%. The total number of sources to be simulated was divided equally among the three web-servers. Table 1. gives the details of the capacity setup for the expected TP on the core link. Buffer size was kept as $1000 Packets$. The packet inter-arrival times process was captured at the output queue of node R1. It should be noted that this queue passes the data-traffic from server to the client - the acknowledgment-traffic from the client to server does not pass through this queue.

For HTTP-TCP settings, we followed [16] which gives the average file-size of $10Kbytes$ with shape-parameter $\alpha = 1.5$ for the heavy-tailed distribution. The maximum file-size is set as $100MBytes$. The average number of files per web-page was set as 6 from the geometric distribution. Therefore, the average web-page size is $60Kbytes$, or $480Kbits$. User think-time as approximately $40seconds$ and negative-exponentially distributed. Since $T_{off} \gg T_{on}$ from (1) we have the throughput of a typical web-user as $(480Kbits)/(40sec) = 12Kbps$ at the session layer and $(12Kbps * (1 + 40/1500)) = 12.3Kbps$ at the packet layer. For TCP, $MSS = 1460Bytes$ and $MaxCWND = 65535Bytes$ were set.

The heavy-tailed distribution used in our simulations is a Truncated Power Tail (TPT) distribution [17] with shape parameter of 1.5. The TPT distribution requires a large number of samples to converge, for example, for a shape parameter of

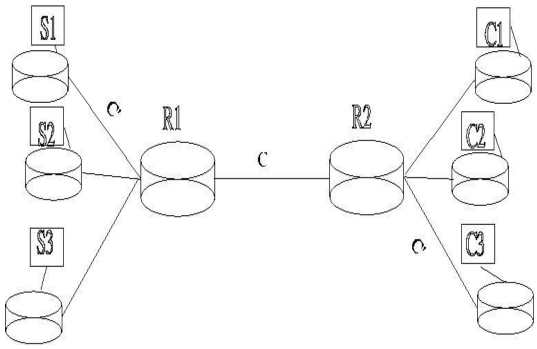

Fig. 3. Simulation setup. Capacities: the core link has C and the incoming links $C^{'}$. Propagation delays: 5msec of the core link and 10msec for the edge links.

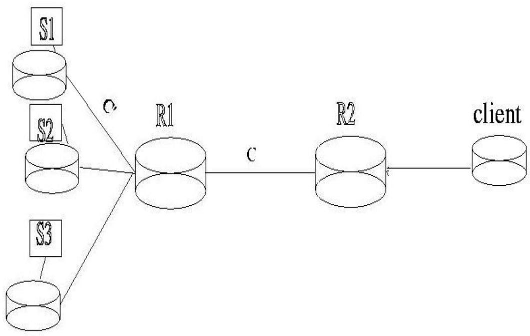

Fig. 4. Simulation setup. Capacities: the core link has C and the incoming links $C^{'}$. Propagation delays: 5msec of the core link and 10msec for the edge links.

TABLE I

CAPACITIES C AND $C^{'}$ FOR THE CORE AND THE INCOMING LINKS FOR THE SIMULATION SETUP.

Expected TP (Mbps)	C (Mbps)	$C^{'}$ (Mbps)
7.2	10	5
14.4	20	10
28.8	40	20
72	100	50
144	200	100
288	400	200

$\alpha = 1.5$, at least 10^6 number of samples are needed [18]. All our simulations were run long enough to allow sampling of at least 10^6 files.

C. Simulation setup for LWTS's

Fig. 4. gives the same setup for the LWTS traffic which is similar to Fig. 3. but with two modifications: only one client is present which is receiving traffic from all sources and secondly P-TCP is transporting the data instead of TCP, and uses UDP packets. The user-behaviour at the session level is implemented exactly similar to the set-up for HTTP-TCP sources. For P-TCP also $MSS = 1460Bytes$ and $MaxCWND = 65535Bytes$ were set.

We need one more equation to resolve r. We took advantage of [21] which gives:

$$Var = M * r^2 * (\frac{TCP_{on} * N}{N * RTT + T_{off}})(1 - \frac{TCP_{on} * N}{N * RTT + T_{off}}) \tag{9}$$

and

$$\frac{Var}{TP} = r * (1 - \frac{TCP_{on} * N}{N * RTT + T_{off}}) \tag{10}$$

for $T_{off} \gg T_{on}$ leads to the following expression for CV:

$$CV = \sqrt{\frac{r}{TP}} \tag{11}$$

To choose r, we first did a trial with HTTP-TCP sources as outlined in section V. We chose an arbitrary point where $CV \approx 1$ and noted that $TP \approx 72Mbps$, however, the sources are actually connected to S1, S2 and S3 and the three streams multiplex at $R1$. For each of these streams $TP = 24Mbps$, therefore, the source rate should be adjusted according to this value. A setting of $r = 24Mbps$ led to $CV = 1.042$ at node $R1$ for the aggregate traffic from LWTS's. This rate was kept fixed for all simulations.

The time TCP_{off} can be calculated by $TCP_{off} = RTT - TCP_{on}$. It is assumed that transport protocol's off-time has a negative exponential distribution. As an approximation we choose RTT as the sum of two way propogation delay. It is to be noted that as a result of this approximation, the RTT will not become fixed instead the sum of the means of the TCP_{on} and TCP_{off} distributions will determine the mean round trip time. Moreover, the RTT distribution will be the shaped by the two distributions.

V. SIMULATION RESULTS

A. Throughput

Fig. 5. shows the TP comparison between LWTS and HTTP-TCP traffic. The TP followed a linear growth with the number of sources i.e. for N number of sources, the total TP $\approx 12.3Kbps * M$. The LWTS shows an excellent match, following a similar linear increase.

B. Coefficient of variation

Fig. 6. shows the relationship between the CV and the throughput. There are two things to be noticed. Firstly, as the GCLT indicates convergence of CV at $\approx 1/M^{1-\frac{1}{\alpha}}$, it is interesting to note that the curve for CV for HTTP-TCP sources shows an approximate trend $\approx 18.5/M^{0.33}$ which is expected for $\alpha = 1.5$. Here 18.5 is the normalizing constant. Second thing to note is that the LWTS shows a very good match of CV, and follows the same trend as demonstrated by the HTTP-TCP traffic. The relationship between CV and number of users has not been directly presented in any previous work but is a natural result directly given the observations of LRD in traffic.

C. Hurst parameter

In Fig. 7. we compare the Hurst parameter estimates again showing a good match (except for 288Mbps for which we presently do not have a good explanation). First note that the measured values of H against the number of HTTP-TCP sources which shows that H starts going down until it stabilizes around 0.65 from 2338 sources, or from 28Mbps, onwards. For high aggregation the packet inter-arrival times have tighter space for variations resulting in smoother traffic. This is an interesting result which has been also reported from analysis of the Internet traffic traces [23].

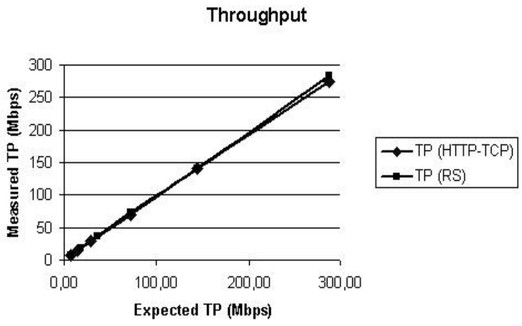

Fig. 5. Expected against measured TP comparison

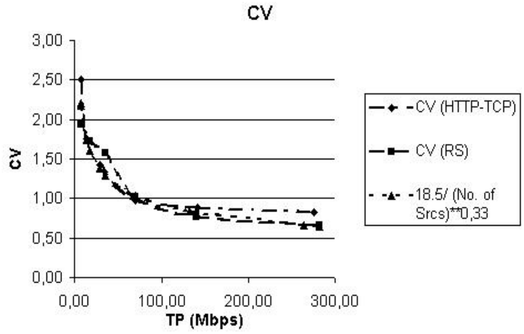

Fig. 6. CV comparison against TP

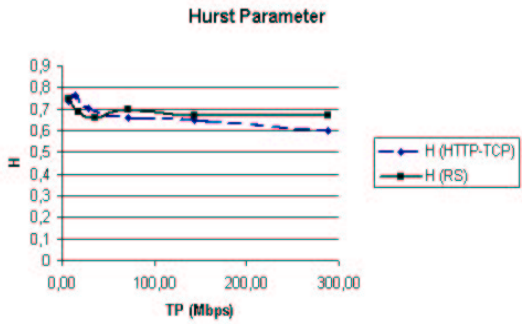

Fig. 7. Hurst parameter estimates against TP

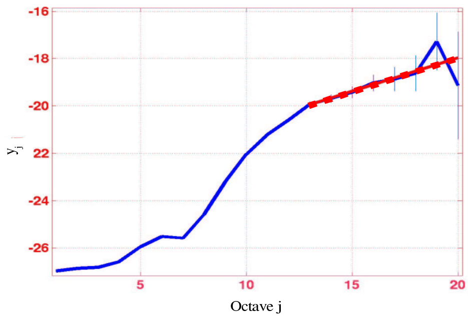

Fig. 8. Hurst parameter estimation with wavelet transform method for aggregate traffic from 5682 HTTP-TCP sources.

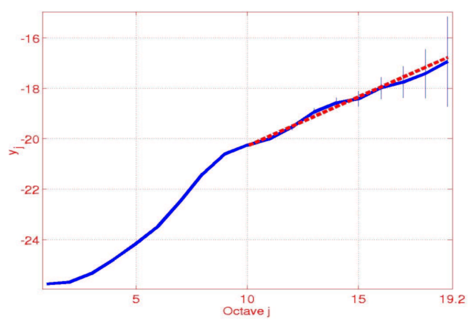

Fig. 9. Hurst parameter estimation with wavelet transform method for aggregate traffic from 5682 LWTS's.

We refer to the WT plot for HTTP-TCP sources in Fig. 8. for traffic at 72Mbps. There are clearly two slopes in the WT plot suggesting completely different structures in the large and small time scales. The WT plot estimates, in the range of octave 14-20, H≈0.66. However, even a visual inspection of the curve shows that the curve has a stronger slope in the region of octave 5-14. What does this mean? Consider the notch around octave 6 in the WT curve which is a rough indicator of RTT. This is an indicator of the scales where TCP is operational. A higher slope in the smaller time scales means that the TCP is adding to the fractality of the traffic. The most plausible explanation is a heavy-tailed RTT distribution which has been dealt by other authors [5]. The HTTP-TCP source captures both aspects (user and protocol) of the Internet traffic.

In Fig. 9. we show the WT plot for the aggregate traffic from LWTS's. This curve is similar to the one shown by the HTTP-TCP in Fig. 8. and demonstrates that the LWTS has captured the fractal character at most of the scales- specially at small and large time scales. The source matches the reality closely by reproducing the two distinct characteristic behaviours (protocol and user-behaviour) of the Internet traffic.

D. Autocovariance

The choice of analysing the correlations of the trace for 5682 sources producing approx. 72Mbps was made as for this trace the measured CV was approximately equal to one. A $CV = 1$ misleads one into thinking that the process might have become poissonian and does not have LRD! However, this is not the case as the process does not become poissonian rather the evidence of LRD is shown in Fig. 10. In the estimation of auto-covariance, thresholds of $\pm 0.2\%$ were set above and below the zero line. Points falling between these thresholds indicated absence of auto-covariance. The estimation showed that at a lag of 16000, covariance died out. The process showed dependence over four orders of magnitude despite the fact that $CV \approx 1$. This is another interesting result of our experiment.

Fig. 11. shows the same plot for 72Mbps aggregate traffic from LWTS's. Except for the small variations in the initial lags which are attributed to the approximations of P-TCP, the new sources produces the same LRD behaviour as produced by the HTTP-TCP sources - it also lasted for 4 orders of magnitude. The LWTS produces a good match of the LRD behaviour.

While in one way or another the above discussed results are already known from the measurements done on the Internet traffic traces it is, however, to be noted that we are reproducing these results through computer simulations which match the reality very closely.

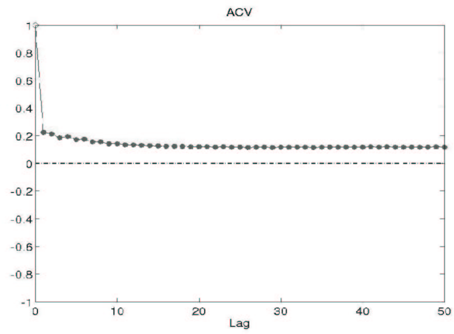

Fig. 10. Autocovariance of the traffic trace for 5682 sources

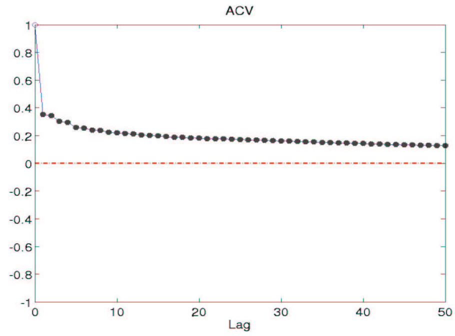

Fig. 11. Autocovariance of the traffic trace for 5682 LWTS at 72Mbps aggregate traffic

At the end of this section we only briefly mention that we compared the mean packet End to End (E2E) delay for both scenarios and found a good match for this parameter, too. We are not reporting the results.

E. Lighter and faster simulations

The main objective of the LWTS is to solve the scalability problems resulting from the closed loop operation of HTTP-TCP source. Table IV gives an idea of the speed. The simulations were run for 350 seconds for a target of 72Mbps. The LWTS led to faster simulation. It showed a speed gain of 4 times. The removal of the acknowledgment-traffic in opposite direction leads to faster simulation times.

There is a dramatic impact on the memory usage by LWTS. This is achieved by the elimination of client objects and by using a lighter protocol. Fig. 12. gives an idea of the memory requirements of the two sources. The HTTP-TCP sources follow a linear curve approximated by $M_{requirement}[MBytes] \approx 0.02 * N_{clients}$, where $N_{clients} \approx TP/12.3Kbps$. Linux OS presently allows a memory usage of 4GBytes out of which 1GBytes is used by the kernel and related programs, leaving 3GBytes for the simulation. Therefore maximum 150,000 HTTP-TCP sources can be simulated. However, in comparison, the memory requirement by the new source is much smaller, for example, the same number of sources requires only 60MBytes of memory. It is to be noted that beyond this point it is not possible to simulate HTTP-TCP sources which is now possible with the new source.

VI. Conclusion

In this paper we have presented a new type of traffic source which not only captures the important aspects of the Internet traffic with high fidelity but also solves the scalability problems associated with TCP based traffic sources. The simulation results are in accordance with the results reported from the analysis of real Internet traffic traces, and thus confirm that our simulations get very close to the reality. By matching its important traffic characteristics with the statistics of traffic from realistic HTTP-TCP sources, we have shown that the new source qualifies as a replacement for the realistic HTTP-TCP source.

We believe the idea of P-TCP suitably captures the essence of TCP while allowing lighter and faster simulations. The simplicity of this idea and easy implementation should allow room for more research in the high-speed networking area.

Under the assumption of negligible losses, we kept our focus around the slow-start phase of TCP and the resulting RTT distribution. We showed that TCP introduces significant fractality in the small time scales and is important from source modelling point of view. In this paper, we have not considered large, or "elephant", flows for which the model will have to be modified. This forms the scope of our future research.

TABLE IV

SPEED AND MEMORY GAIN

Sources	Simulation Time
5682-HTTP-TCP Sources	≈27Min.
5682-Light Weight Traffic Sources	≈7Min.

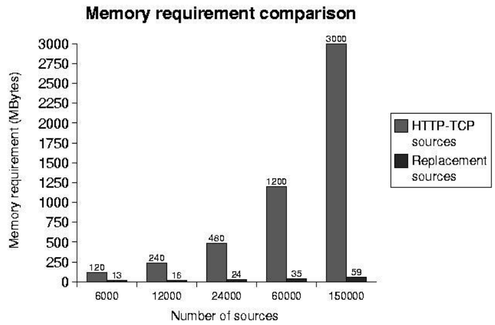

Fig. 12. Memory requirement against TP

References

[1] V.Paxon and S. Floyd, "Wide-Area Traffic: The Failure of Poisson Modeling", IEEE/ACM Transactions on Networking, 1995

[2] W. Leland, M.Taqqu, W.Willinger and D. Wilson, "On the Self-Similar Nature of Ethernet Traffic", IEEE/ACM Transactions on Networking, 1995

[3] Helmut Hlavacs, Gabriele Kotsis, Christine Steinkellner, "Traffic Source Modelling", Technical Report No. TR-99101, Institute of Applied Computer Science and Information Systems, University of Vienna, Austria.

[4] Mark Crovella and A. Bestavros, "Explaining world web web traffic self-similarity", Technical Report TR-95-015, Computer Science Department, Boston University, Oct 1995.

[5] A. Feldmann, A. C. Gilbert, P. Huang, W. Willinger, "Dynamics of IP traffic: A study of the role of variability and the impact of control", ACM/SIGCOMM'99, Boston, MA, 1999.

[6] A. Feldmann, A. Gilbert, W. Willinger, "Data Networks as cascades: Explaining the multifractal nature of Internet WAN traffic", ACM/SIGCOMM'99, Boston, MA, 1998.

[7] K. Below and U. Killat, "Reducing the Complexity of Realistic Large Scale Internet Simulations", IEEE Global Communications Conference (GLOBECOM), San Francisco, USA, 2003

[8] K. Below and U. Killat, "On the Configuration of Simulations of Large Network Models with HTTP/TCP Sources", 15^{th} ITC Specialist Seminar, Internet Traffic Engineering and Traffic Managment, Würzburg, 2002

[9] K. Below, U. Killat, "Allocation of HTTP/TCP Sources for Traffic Generation in Large Network Simulations", 2nd Polish-German Teletraffic Symposium (PGTS 2002), Gdansk, Poland, pp. 175–182, September 2002

[10] W. Stevens. TCP/IP Illustrated, Vol.1 The Protocols. Addison-Wesley, 1994

[11] The website of Cooperative Association for Internet Data Analysis (CAIDA): http://www.caida.org

[12] M.S. Taqqu, Walter Willinger and R. Sherman and D.V.Wilson, "Self-Similarity through High-Variability:Statistical Analysis of Ethernet LAN Traffic at the Source Level", Computer Communication Review Vol 25, 1995

[13] M.S. Taqqu, Walter Willinger and R. Sherman, "Proof of a fundamental result in self-similar traffic modelling", Computer Communication Review, 1997.

[14] Walter Willinger, V.Paxon, Rolf H. Reidi, Murad S. Taqqu, "Long-Range Dependence and Data Network Traffic", P.Doukhan, G. Oppenheim and M.S. Taqqu, eds. 2001

[15] D. Veitch, P.Abry, "A Wavelet Based Joint Estimator for the Parameters of Long-Range Dependence", IEEE Trans. Info, April 1999

[16] Hyoung-Kee Choi and John O. Limb, "A behavioural Model of Web Traffic", 7^{th} Annual International Conference on Network Protocols, Toronto, Canada, 1999

[17] Lester Lipsky, Michael Greiner, Hans Peter Schwefel, Helmut Gogl, "How to Model Telecommunication (and other) Systems Where "Self-Similar" (Power-Tail) behaviour is Observed", July 16, 1997

[18] Mark E. Crovella and Perter Lipsky, "Long Lasting Transient Conditions in Simulation with Heavy-Tailed Workloads", In proceedings of the 1997 Winter Simulation Conference (WSC97). Boston University, University of Connecticut, 1997

[19] V. Jacobson, "Congestion Avoidance and Control", ACM/SIGCOMM-88, August 1988

[20] Ling Guo, Mark Crovella and Ibrahim Matta, "How does TCP generate Pseudo-self-similarity", Computer Science Department, Boston University, Boston, MA 02215

[21] Jia-Shing Jou and John S. Baras, "A Multilevel On/OFF Model for Multifractal Internet Traffic", Technical Report CSHCN TR 2002-5 (ISR TR 2002-9), University of Maryland.

[22] Mark E. Crovella and Lester Lipsky, "Simulations with Heavy-Tailed Workloads", Self-Similar Network Traffic and Performance Evaluation, ISBN 0-471-31974-0, John-Wiley & Sons, INC., 2000

[23] C. Fraleigh, F. Tobagi, C. Diot, "Provisioning IP Backbone Networks to Support Latency Sensitive Traffic," in Proc. IEEE INFOCOM 2003, San Francisco CA, April 2003

[24] Sireen Malik, Ulrich Killat, "Replacing HTTP-TCP Traffic Sources with Open-Loop UDP Traffic Source", Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS), San-Jose, California, USA, July 2004