Discrete-Approximation of Measured Round Trip Time Distributions: A Model for Network Emulation

Jay Aikat*, Shaddi Hasan+, Kevin Jeffay*, and F. Donelson Smith*
*University of North Carolina at Chapel Hill
+University of California at Berkeley
Email: {aikat, jeffay, smithfd}@cs.unc.edu, shaddi@berkeley.edu

***Abstract* — Empirical evaluations to study network performance, whether in a laboratory setting or on GENI testbeds, rely heavily on measurement-based modeling of round trip times (RTTs) to emulate realistic end-to-end delays of local and metropolitan area networks. For generating realistic traffic, we studied several models to emulate RTTs. In this paper, we performed experiments on real testbeds using synthetic TCP traffic generated from measurement data from a large university campus. As a result of our study, we present the Discrete-Approximation model for RTT (DA-RTT) emulation. Using three different metrics for performance evaluation, which include queue length at routers, connection response times, and connection durations, we demonstrate that the simple DA-RTT model closely represents the per-connection RTTs in the original traffic. While these experiments were performed in our laboratory, and not using GENI infrastructure, we present this as a possible model for adoption on GENI testbeds to emulate Round Trip Time Distributions for GENI experiments.**

***Index Terms*— Discrete Approximation modeling, Network Performance Evaluation, Round Trip Time emulation, Traffic Generation, GENI experiments**

1. INTRODUCTION

In their seminal paper, Floyd and Paxson [5] articulated the challenges to performing realistic and meaningful experiments. The research community responded and over time, ever more sophisticated means of performing evaluations of proposed network improvements using simulation, emulation, and live experimentation have been developed. Software simulators (*e.g.*, ns-2, ns-3, and GTNets), laboratory testbeds (e.g., Emulab), and wide-area network testbeds (e.g. GENI) have evolved to provide capabilities to more faithfully reproduce protocol and network dynamics that occur on production networks.

Synthetic traffic is vital to performing reproducible, controlled experiments. Traffic generation itself contains several important components. Emulating round trip time (RTT) is a key factor in faithfully representing traffic conditions on production networks. State of the art traffic generation tools, such as Harpoon [10], Tmix [14], and Swing [13], all provide a means of turning measurements of production network links into "realistic" synthetic traffic. The traffic is realistic because it directly reproduces measurements of real traffic such as total number of flows present, total number of bytes transmitted, and emulates some model of measured RTT.

A study of the leading network performance evaluations reveals that there is no accepted best practice for RTT emulation in network experiments. Previous studies [14] have shown that using *per-connection RTTs* for traffic generation faithfully represents the realistic conditions captured on the production network. In this paper, we therefore use the per-connection method as the control set for comparison, and we set out to find another method of RTT emulation that is simple to implement while also being robust in representing realistic traffic conditions. As a result, we developed a simple empirically-based model that produces performance results closest to the complex real-world RTT measured in production network traffic.

To the best of our knowledge, ours is the first research that has developed this method of RTT emulation as a viable and robust alternative to emulating measured per-connection RTT for traffic generation. And we demonstrate that the result holds for three different performance metrics, except for queuing dynamics under extremely high (95%) congestion constraints. These three performance metrics are not specific to our experiments. They represent application performance at the ends, through measuring connection duration and response times, and network performance through measuring the queuing dynamics for a FIFO (first-in-first-out) queue. Thus, they can be applied for any GENI experiments.

2. RELATED WORK

Most of the early work in traffic generation for network experiments focus on one or a limited set of workloads and network emulation techniques. Of the leading traffic generation systems, Harpoon uses either no RTT delay or a single delay for all flows, which is the mean RTT of all the flows in the original traffic data. In contrast, the Tmix and Swing traffic generators start with packet header traces and

This work was supported in part by a grant from the National Science Foundation.

emulate the range of RTTs measured in the original connections – Swing using a parametric model while Tmix using a non-parametric model, emulation the RTT for every connection as measured in the original traffic. The latter, or the *per-connection-RTT* model, is what we use for comparison against the new Discrete Approximation method or *DA-RTT* emulation in this paper. It is important to note that it has been clearly shown [14] that the per-connection RTT model faithfully emulates the minimum originally measured RTT for every connection during traffic generation.

3. TRAFFIC GENERATION

3.1 Workload Generation

We use the Tmix traffic generation system for all our experiments. In Tmix each connection found in a trace of TCP/IP headers from a production network link is analyzed to produce a "connection vector." A connection vector includes the connection's start time relative to the beginning of the trace and a descriptor of each request-response exchange in the connection. A request-response exchange, called an "epoch," is described by a 4-tuple consisting of the request size (called the "*a*" unit size), the response size (called the "*b*" unit size) and two endpoint latency values (called the "*t*" values); one for the server-side "think time" between a request and its response and one for the user/client "think time" between successive requests. Unidirectional transfers have only an *a* or *b* value depending on the direction of transfer.

This is Tmix's "a-t-b-t" model for workload generation. The inclusion of these latencies represents all application-processing and/or user delays on both endpoints.

3.2 Round Trip Time Emulation

We experimented with several different methods for emulating RTTs. In this paper, we develop the new DA-RTT model and present the results for experiments using this model, comparing it with the results for experiments using Tmix's per-connection-RTT model, which emulates the specific minimum RTT found *for each connection* by analyzing the TCP/IP header traces.

We created the DA-RTT model from the empirical distribution of RTTs for the original trace. Our laboratory network has 30 pairs of traffic generators; hence we chose 30 values, thus creating 30 end-to-end paths in the network. The goal behind developing this model was to create as close an approximation of the empirical distribution of RTTs seen in the original trace as possible. For this we use the concept of a quantile function. A quantile function of a probability distribution is the inverse F^{-1} of its cumulative distribution function.

Our method of approximating the CDF of the RTTs was as follows: first we approximated the distribution such that we cut off the bottom 1% and top 1% of RTT values. These represented only 2% of connections but were skewing our overall approximations such that a very large portion of RTTs would be much larger than the median (or mean) RTTs. Now, with the remaining 98% of the distribution, we divided this distribution into 30 equal size bins, and then found the average RTT for each of these 30 bins in the distribution.

The resulting RTT values for our traffic formed this set: [8, 8, 10, 10, 12, 14, 14, 16, 18, 20, 22, 24, 26, 30, 34, 38, 42, 48, 52, 60, 74, 80, 82, 86, 92, 98, 124, 172, 258, 420] milliseconds (ms).

In addition, to put both these RTT emulation methods in perspective, we ran all these experiments using no RTT emulation – we call this the *nodelay-RTT* model. This is, admittedly, an extreme case where we emulate no connection RTTs at all. While this is not realistic, it serves a purpose here – to provide a quantitative assessment of the role of round trip times in traffic generation for network experiments by showing the effect of RTTs on performance metrics.

It is important to emphasize that all the experiments presented here emulate the same traffic workload generating 4.7 million connections in every experiment, regardless of the RTT model used in the experiment. Thus we can differentiate between the DA-RTT and per-connection RTT models as emulating the same traffic but simulating 30 end-to-end paths in the former and 4.7 million paths in the latter for experiments.

4. TRAFFIC CHARACTERISTICS

In all our experiments reported in this paper, we used traffic from the University of North Carolina (UNC), taken on the border link connecting the campus to the Internet service provider network. The UNC campus trace was a 1-hour trace on a weekday during the school year. It has almost 4.7 million connections with an offered load of 471 Mbps in one direction and 202 Mbps in the other. Figure 1(a) shows the link throughput for both directions.

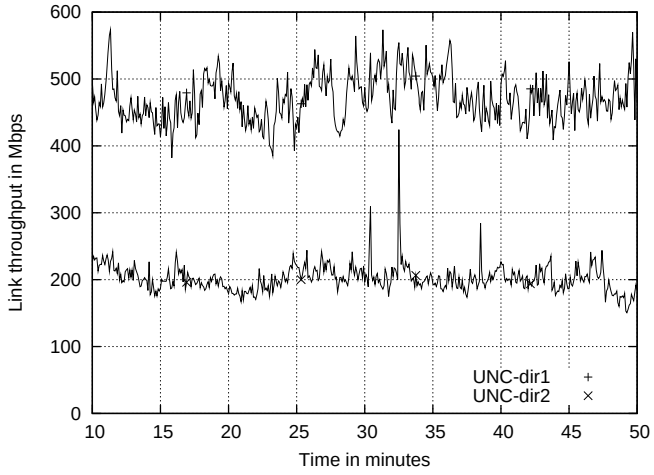

Figure 1: (a) Offered load

The cumulative distribution of round trip times for the original trace is shown in figure 1(b). The mean RTT for connections in the UNC trace was 80 ms while the median RTT was 36 ms. By design, we maintain this median value for the DA-RTT paths.

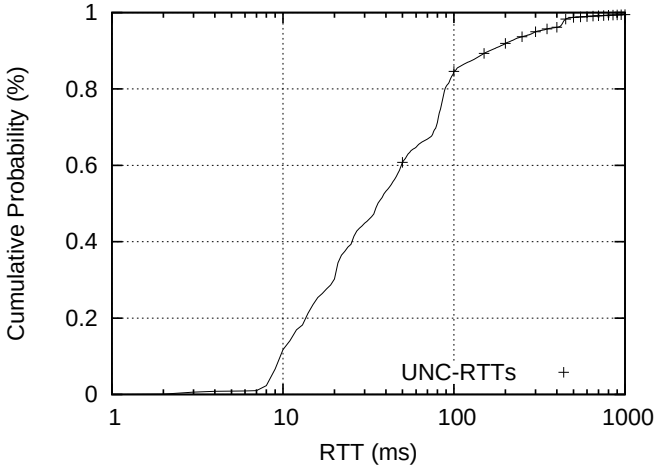

Figure 1: (b) RTT of original traffic

5. EXPERIMENTAL METHODOLOGY

The laboratory network used for our experiments is shown in Figure 2. At each edge of the network are 30 workload generator systems. We used the Tmix traffic generation system (that can be obtained from the contact given in [11]) on all 60 machines to create workloads based on the a-t-b-t model as described in section 3.1, and to emulate both the *per-connection-RTT* and the *DA-RTT* methods (described in section 3.2). For all the experiments discussed in this paper, we assigned to each side of every TCP connection the maximum receiver window size exactly as was determined from analysis of the original trace.

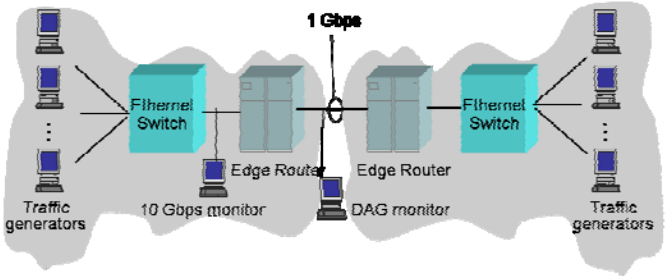

Figure 2: Experimental Network Setup

The core of this network consists of two 10 Gbps Ethernet switches that aggregate the packet traffic from the generator machines and two software router machines connected by a 1 Gbps Ethernet fiber in between. For experiments with an uncongested network ("*unconstrained mode*") the 1 Gbps link is used directly since its capacity is significantly greater than the load generated from either trace.

For experiments with the link operating under a load that is near saturation (95% of link capacity), the dummynet bandwidth emulation function was used to constrain the link bandwidth ("*constrained mode*") with the capacity of the router-to-router link set to 496 Mbps for the UNC trace. In all cases the router queues were set to a very large size (65,000 packets) which was determined to be sufficient to avoid any packet drops at the queue so loss rates were not a factor in any of the results, even in constrained mode.

We verified empirically that no part of the configuration had constrained resources other than when the router-to-router path was intentionally constrained with dummynet. Monitors and instrumentation software were used to record: packets and bytes sent between the routers in one millisecond intervals; a log of the queue size (number of packets in the queue) sampled every 10 milliseconds; and connection durations and response times measured by the traffic generators. Each experiment was run for 60 minutes but data used in the results was collected only during the middle 40 minutes to eliminate startup and termination effects.

6. RESULTS

We begin with a review of the effects of RTT on application performance, or user visible measures of TCP performance, namely connection duration and response time in an unconstrained network. Connection duration is simply the time between the transmission of the first data byte of a connection and the receipt of the last data byte. Response time is the time between the transmission of the first data byte of a request and the receipt of the last data byte of its response. The queue length at the router is measured in this study as the number of packets in the queue at any given time.

a. Unconstrained Mode

In this section, we present the results for all three performance metrics using the *DA-RTT* model in the *unconstrained* mode. For comparison, we show results for the control or *per-connection-RTT* model and the *nodelay-RTT* models as well. All three experiments were run using the *a-t-b-t* workload model described in section 3.1.

In Figures 3 and 4, we show the Cumulative Distribution Functions (CDFs) for connection duration and response times for connections using these three RTT models. In Figure 5, we show the Complementary Cumulative Distribution Functions (CCDFs) for router queue lengths for the duration of the experiments using these three RTT models.

The distribution of connection duration for the *DA-RTT* model practically tracks that of the *per-connection-RTT* model for the body as well as the tail (not shown here) of the distributions – see Figure 3. When there is no RTT emulation, 60% of the connections complete in less than 127 ms, whereas with the per-connection-RTT model, 60% of connections take upto 370 ms to complete. The distribution of connection

duration for the *DA-RTT* model practically tracks that of the *per-connection-RTT* model for the body as well as the tail (not shown here) of the distributions.

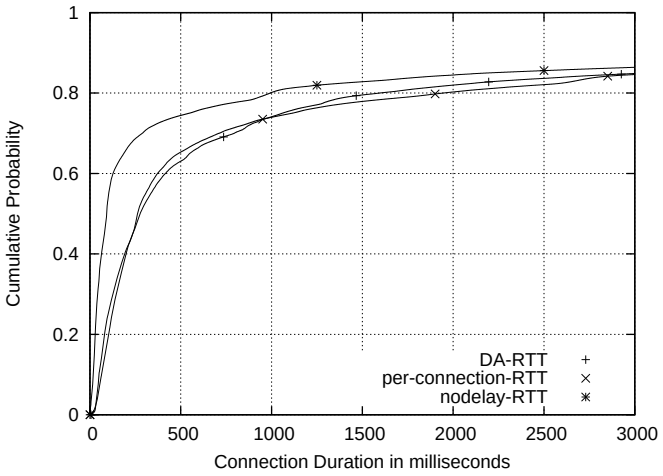

Figure 3: Connection duration – CDF

It is important to note here that the RTT is not the only delay that has an impact on the duration of connections. There are server-side processing times and user thinktimes that have been faithfully reproduced in all these experiments, regardless of the RTT emulation used. Hence, as the connection duration gets longer, RTT plays a smaller role than these other time components, especially user thinktime, that are part of the a-t-b-t workload generation.

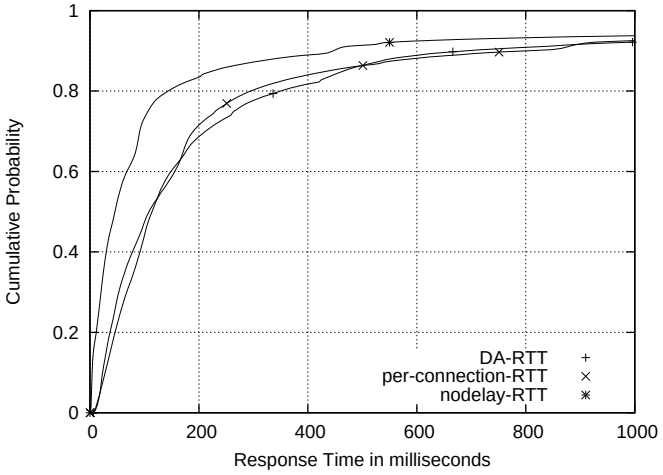

Figure 4: Response Time – CDF

Figure 4 shows that the distribution of response times for the *DA-RTT* model also closely tracks that of the *per-connection-RTT* model for the body of these distributions. 60% of the response times are 68ms or less when using the *nodelay-RTT* model as compared with 156ms or less for both the *DA-RTT* and *per-connection-RTT* models. Since the user thinktimes have no impact on response time (server-side processing times, however, still do), there is a greater impact of RTT on response time. And again, we find that the response times for the *DA-RTT* model practically tracks that of the *per-connection-RTT* model.

We studied several other models for RTT emulation (not shown here). For shorter connection durations and response times, we found there were significant differences among the different RTT models. Hence, it is noteworthy that of all the models we developed and tested, none of them matched the control *per-connection-RTT* model as closely as this *DA-RTT* model.

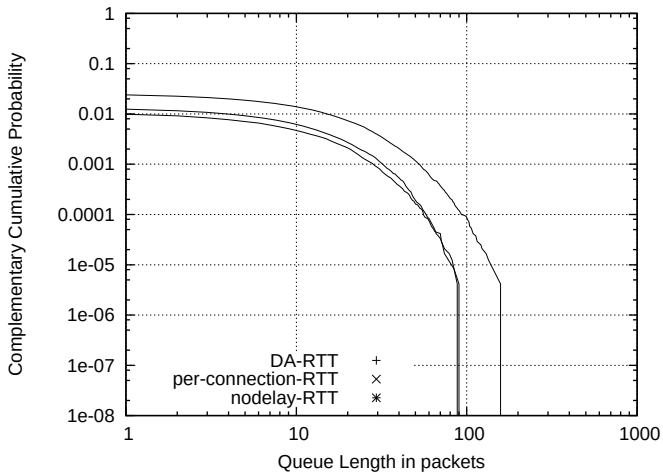

Figure 5: Queue Length – CCDF

In Figure 5, we show the results for the queue length distributions for the three experiments – using *nodelay-RTT*, *DA-RTT*, and *per-connection-RTT* models. We show only the tail of the queue length distributions because the body of these distributions was close to zero because there was little congestion except during brief spikes in the original traffic. Again, we note that the results are very similar when using the *DA-RTT* and *per-connection-RTT* models.

So, clearly, if these were the performance metrics of interest, then the *DA-RTT* model could work just as well for RTT emulation as the *per-connection-RTT* model.

b. Constrained Mode

In this section, we present the results for all three performance metrics using the *DA-RTT* model in the *constrained* mode. Figures 6, 7, and 8 show the CDFs for connection duration, response time, and queue length for the three RTT models.

Even in the *constrained* mode, the *DA* RTT model results in a distribution for connection durations that is comparable to that when using the *per-connection-RTT* model. For response times below 500ms, as shown in Figure 7, there is a small shift, with *per-connection-RTT* having faster response times. This is due to the fairly large difference in queue buildup for the *DA-RTT* model as compared with the *per-connection-RTT* model, as seen in Figure 8.

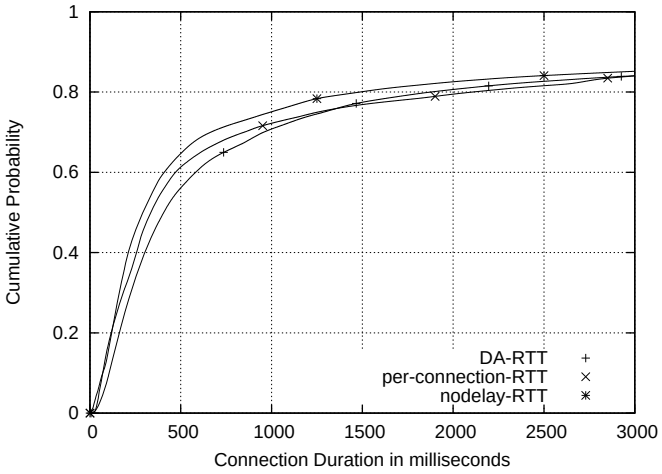

Figure 6: Connection duration – CDF

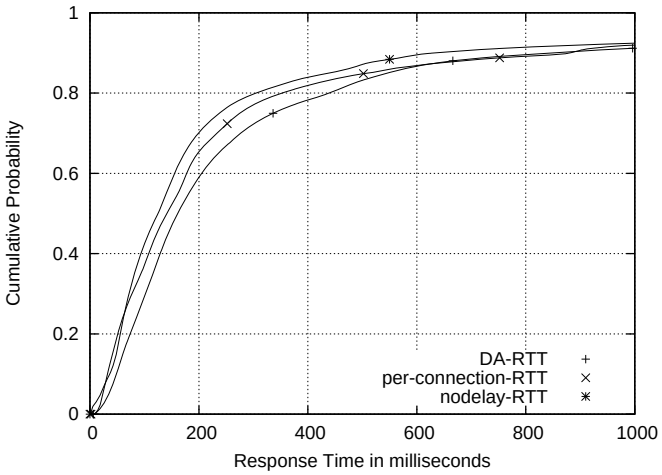

Figure 7: Response Time – CDF

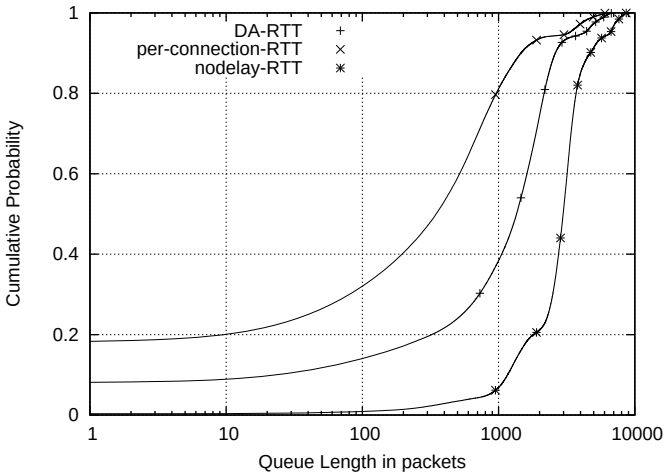

Figure 8: Queue Length – CDF

This buildup for the *DA-RTT* model is likely due to the fact that RTT emulation by paths (which is what the *DA-RTT* model emulates) would lead to many connections that originally had long RTTs now having very small RTTs (and vice versa). In such cases, if these connections also had a large amount of data to send, then that would directly and drastically affect the queue.

Even so, we find that the effect of this fairly significant difference in queuing dynamics, shown in Figure 8, is not as large on connection duration and response time as might be expected. The RTT emulation used in an experiment drastically affects the queue length distribution for that experiment. The *nodelay-RTT* produces the heaviest queue length distribution, with the queue having more than 1000 packets in it for 93% of the time. Compare this with the *DA-RTT* model resulting in that same level of queuing for 60% of the time, and the *per-connection-RTT* model having 1000 packets or more for only 20% of the time. We note that these conditions are at 95% constraint on the link in the middle of the network.

7. FUTURE WORK

We used 30 values to approximate the 4.7 million per-connection RTTs in our traffic. We conjecture that if more paths were used, for example if 150 paths were used, this *DA-RTT* model would produce results that would even more closely resemble the *per-connection* RTT model. The choice of the number of paths for this model was purely dependent on the topology and resources of the physical laboratory network in our study. While it may not be practical to expect such a large number of machines, it is definitely possible to emulate a large number of end-to-end virtual paths using a smaller number of physical machines. We plan to deploy and test both the DA-RTT and per-connection RTT models on GENI testbeds.

Our traffic workload consisted of only TCP connections. While this represents the vast majority of the traffic, we would like to, in future, include the UDP traffic as well. We are currently developing similar tools for modeling UDP traffic and then generating the same on GENI testbeds.

8. CONCLUSION

The discrete approximation model of RTT emulation may be used instead of the *per-connection* RTT model where a simpler yet empirically-based model is desired for traffic generation. If the experiment does not involve very heavily congested scenarios, the *DA-RTT* model produces results at the application-level and at the network-level that are very similar to that of the *per-connection* model. In the presence of heavily *constrained* links in the network, however, the *DA-RTT* model creates significantly longer queues, and hence this must be taken into account when using this model for traffic generation.

Why does this matter? Emulating *per-connection-RTT* involves measuring every connection RTT and assigning the original connection RTT to that exact connection at the time of generating it during the network experiment. The *DA-RTT* model is an approximation of the empirical model and is easier to implement because it only requires that we pick a discrete set of values that approximate the original minimum RTT distribution, and then assign these values to a small number of end-to-end paths in the experimental network, while still emulating the entire workload of TCP connections. The more the number of paths, the better will be the approximation. Hence, where appropriate, the *DA-RTT* model could be used for realistic RTT emulation for network experiments. We present this as a realistic solution for RTT emulation when running experiments on the GENI testbeds.

As GENI experimenters design and setup their experiments on the various testbeds and GENI clusters, they will need to emulate round trip times for generating traffic. We present this new DA-RTT model as an easy-to-implement, feasible and realistic alternative to emulating the specific measured RTT for each connection.

For GENI experimenters interested in using the traffic generation tools for workload generation and RTT emulation with their own traces, we are currently working on a GENI project to make these available to the GENI community in the near future.

REFERENCES

[1] L. Andrew, S. Floyd, and G. Wang, *Common TCP Evaluation Suite*, draft-irtf-tmrg-tests-02.txt, July 2009.

[2] P. Barford and M. E. Crovella, *Generating representative web workloads for network and server performance evaluation*, Proceedings of ACM SIGMETRICS, pages 151–160, 1998.

[3] B. Mah, An Empirical Model of HTTP Network Traffic, *Proc. IEEE INFOCOM '97*

[4] J. Cao, W.S. Cleveland, Y. Gao, K. Jeffay, F.D. Smith, and M.C. Weigle, "Stochastic Models for Generating Synthetic HTTP Source Traffic," *Proceedings of INFOCOM 2004*, pp. 1546-1557

[5] S. Floyd and V. Paxson, *Difficulties in simulating the internet*, IEEE/ACM Transactions on Networking, 9(4):392–403, August 2001.

[6] M. Garrett, and W. Willinger, Analysis, Modeling, and Generation of Self-Similar VBR Video Traffic, *Proc. ACM SIGCOMM '94*.

[7] D. Heyman, and T.V. Lakshman, Source Models for VBR Broadcast Video Traffic, In *IEEE/ACM ToN*, vol. 4, no 1, pp. 37–46, Feb. 1996

[8] L. Le, J. Aikat, K. Jeffay, and F. D. Smith, *The effects of active queue management and explicit congestion notification on web performance*, IEEE/ACM Transactions on Networking, 15(6):1217–1230, December 2007.

[9] Mena, and J. Heidemann, An Empirical Study of Real Audio Traffic, *Proc. IEEE INFOCOM 2000*

[10] J. Summers and P. Barford, *Self-configuring network traffic generation*, Proceedings of Internet Measurement Conference, 2004.

[11] Tmix, http://netlab.cs.unc.edu/Tmix

[12] V. Paxson. Empirically Derived Analytic Models of Wide-Area TCP Connections, *IEEE/ACM ToN*, 2 (4) 316-36, August 1994

[13] K. Vishwanath and A. Vahdat, *Swing: Realistic and responsive network traffic generation*, IEEE/ACM Transactions on Networking, August 2009.

[14] F. Hernandez-Campos, K. Jeffay, and F. D. Smith, *Modeling and Generation of TCP Application Workloads*, Proceedings of the Fourth IEEE International Conference on Broadband Communications Review, September 2007.