A Tool for RApid Model Parameterization and its Applications

Kun-chan Lan *
USC/ISI
4676 Admiralty Way
Marina Del Rey, CA 90292
kclan@isi.edu

John Heidemann †
USC/ISI
4676 Admiralty Way
Marina Del Rey, CA 90292
johnh@isi.edu

ABSTRACT

The utility of simulations and analysis heavily relies on good models of network traffic. However, it is difficult to model and simulate the Internet traffic because of the network's great heterogeneity and rapid change. The statistical properties of Internet traffic not only constantly change over time but also vary in other dimensions such as locations and directions. Previously we have developed a tool *RAMP* that supports rapid parameterization of traffic models from live network measurements. In this paper, we first extend RAMP to support near-real-time trace-driven simulation. Next, we demonstrate the applications of RAMP via three case studies: generation of realistic traffic model for simulation, generation of high bandwidth synthetic network traces, and analysis and modeling of malicious traffic. Finally, we discuss some lessons we learned from using RAMP for traffic modeling.

1. INTRODUCTION

It is difficult to simulate and model the Internet due to its scale, heterogeneity and dynamics [1]. The volume and statistical properties of Internet traffic not only constantly change over time but also vary in other dimensions such as locations and directions [2]. Considering the Internet's great technical and administrative diversity and immense variations over time regarding how applications are used, it is not obvious that one can model *his* traffic accurately based on the models derived from measurements taken previously from other parts of the network.

Motivated by the challenge and difficulty of modeling highly-diverse Internet traffic, previously we have developed a tool *RAMP* [2] that allow users to quickly parameterize traffic models based on the measurements and generate realistic *current* traffic in their simulations. Our approach does not make any underlying assumption of traffic properties (for example, heavy-tailed distribution for file size/transmission time) and hence is more applicable than existing approaches in coping with the heterogeneous nature of the Internet traffic. Opposed to traditional trace-replay time-series analysis techniques which typically ignore the fact that traffic frequently reacts to the network's current properties, our approaches focus on characterizing source-level pattern in which the data is sent. Our trace-driven application-level modeling approach employs a trace-analysis tool that infers traffic and topology characteristics, and a CDF-based traffic model generating synthetic traffic.

In this work, we have extended RAMP to support near-real-time simulation. It is useful for network operators to be able to simulate and visualize the *current* traffic on the operational network for various traffic engineering tasks such as QoS control, anomaly detection etc. Initially RAMP was designed to process traces and generate simulation models off-line. To model and simulate the traffic on-line[1], we have modified RAMP so that several copies of RAMP can be run in a pipeline fashion and the results of simulation can be visualized in near real-time.

A second contribution of this paper is to evaluate use of RAMP in three applications. First, we show the use of RAMP to generate realistic workload for simulation. Second, we utilize RAMP to generate high bandwidth network traffic traces. Lastly, we demonstrate the use of RAMP as a trace-driven analysis and modeling tool to study malicious traffic such as DDoS and worm traffic.

In this paper, we first describe the design and implementation of our tool RAMP (Section 3). Next, we show an extension of RAMP for supporting near-real-time trace-driven simulation (Section 4) Finally, we demonstrate the utilities of RAMP via three immediate applications (Section 5). We

*Kun-chan Lan and John Heidemann are partially supported in this work as part of the SAMAN project, funded by DARPA and the Space and Naval Warfare Systems Center San Diego (SPAWAR) under Contract No. N66001-00-C-8066. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of DARPA or SPAWAR.
†John Heidemann is also partially supported as part of the CONSER project, funded by NSF as grant number ANI-9986208.

ACM SIGCOMM 2003 Workshops August 25 and 27, 2003, Karlsruhe, Germany

[1]We do not mean "hard real-time". Our focus is to simulate the traffic that happened not long ago (eg. the traffic ten minutes ago).

also discuss some insight we learned from using RAMP for modeling traffic.

2. BACKGROUND

In this section we first describe the dataset we used to develop our tool and two statistical techniques, including wavelet scaling plot and Kolmogorov-Smirnov goodness-of-fit test, that help us validate the results of RAMP. We then summarize some related work.

2.1 Traces

The data used in our study are from two sources. One was collected on the web server of Internet Traffic Archive [3] (this set of trace will be referred to as "ITA" in this paper). The other was recorded at a 100Mbps Ethernet link connecting the Information Science Institute to the rest of the Internet (referred to as "ISI"). To design and validate our tool, we utilized two sets of one-hour long ISI data which were collected at different times of the same day. One was recorded starting at 2:00 pm (referred to as ISI-1) and the other was recorded starting at 7:00 pm 2001 (referred to as ISI-2). Intuitively, one captures the traffic during a normal business hour and the other shows traffic during after-hours. The details of the traces are described in [2].

2.2 Statistical Tests

To compare the scaling property of the traces and the generated synthetic traffic, we use wavelet scaling plot, a wavelet-based time series analysis [4]. In which the statistics of the time series is viewed at each resolution level or scale, taken as a function of scale. More details of this technique are described in [5, 6].

We use Kolmogorov-Smirnov goodness of fit test [7] to formally determine if two sets of traffic data are significantly different from each other, in addition to visually examining their CDF plots. More details of this technique are described in [2].

2.3 Related Work

Our work builds on prior work in traffic modeling, trace compaction and workload generation.

Our methodology is based on structural modeling approach [8] which emphasizes on characterizing source-level pattern in which data is sent. For most applications, the application-level pattern (such as request/reply patterns in web traffic) in which data is sent, does not react to the network dynamics. The structure we choose to model user behaviors of web traffic is similar to previous work of Mah [9] and Crovella et al. [10, 11]. We also adopt Mah's approach in terms of describing traffic based on CDF of real data, which has the advantage of being able to represent arbitrary distribution.

Trace compaction generally refers to the techniques used to retrieve "relevant characteristics" from the trace. The methodology we adopt to construct web model is closer to the work of Smith et al. [12]. Additionally, we also model path characteristics and provide more comprehensive validation mechanism. Furthermore, in addition to web traffic which previous work has focused on, we also include FTP traffic in our study, and automate the whole process from trace analysis to model validation.

Previously several studies have developed workload generators for web traffic including SURGE [10], IPB [13] and work at RPI [14]. In our work, we focus on parameterizing the traces and generating simulation models in a timely fashion to allow the users to study their *current* traffic. In addition to modeling user/application behavior, we also manages to estimate path characteristics which are important parameters to drive simulation.

Finally, while the studies described above were mainly based on traffic collected at the edges of the Internet, significant amount of efforts have also been put in the study and modeling of backbone-level traffic cite.

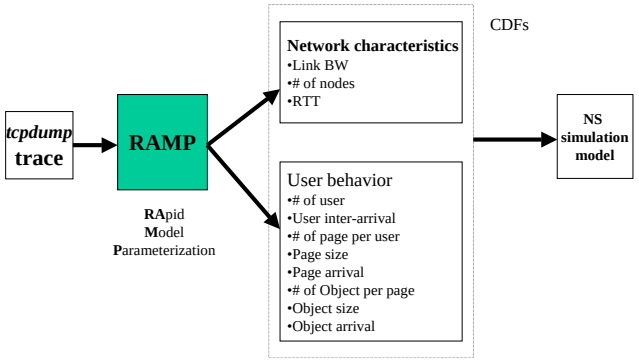


Figure 1: Data flow of RAMP

3. RAMP: RAPID MODEL PARAMETERIZATION

In this section, we summarize some important results in our previous work [2]. We first describe the design and implementation of RAMP. Next, we compare RAMP with an existing traffic generator. Finally, we discuss the effect of detail in a structured model. More details can be found in [2].

3.1 Design of RAMP

Motivated by the need that it is important to quickly parameterize models from new data to account for the diversity of the traffic, we previously designed a tool called *RAMP*. RAMP can convert live measurements into simulation models which then be used to generate realistic synthetic traffic. In this section we describe our approaches from analyzing the trace to finally generating the simulation model.

Our approach is to automatically generate statistics that model user behaviors and network path characteristics by analyzing TCP/IP header information captured in the measurements. The resulting model will then be built into the widely-used NS network simulator [15] and validated against the original trace via wavelet-based analysis and first order statistical comparison.

The input of RAMP is a tcpdump-format trace. The output of RAMP is a set of CDF (Cumulative Distribution Function) files that model the corresponding traffic, as

shown in Figure 1.

Specifically, the CDF files consist of two types of data. One set of CDF files model user/application level statistics of the traffic, such as user session arrival, page/file size etc. Based on the structural modeling approach [8], we design a three-level simulation model to characterize web traffic and two-level model to characterize FTP traffic as shown in Figure 2 and 3. The other set of CDFs model path characteristics of the network. In particular, we focus on characterizing RTT and bottleneck bandwidth of the measured traffic since they are important parameters for driving network simulation.

To determine the RTT of each TCP connection, for outbound traffic, we compute the difference of timestamp between data packet and the first ACK packet which has the same sequence number. For inbound traffic, we compute the RTT by taking the timestamp difference between the SYN packet and its corresponding ACK. For each connection we take the minimum of RTT samples as an approximation of propagation delay of the path (after dividing the RTT by 2) and consider the deviations from the minimum RTT as variances caused by queuing delay and transmission delay. We use this approximation to drive our simulation.

To compute the bottleneck bandwidth, for outbound traffic, we use Sender Based Packet Pair (SBPP) [16]. SBPP estimates the spacing between a pair of back-to-back TCP packets after passing the bottleneck link between local servers and remote clients by examining the arrival times of their corresponding ACKs. For inbound traffic, we rely on Receiver Only Packet Pair (ROPP) [17] which uses the arrival times of two consecutive full-size packets at the receiver to estimate the bottleneck bandwidth between remote servers and the local clients. We also apply similar techniques to filter noise such as density estimation as described in [18].

To reconstruct the data exchanges in the HTTP connections, we adopt a similar approach and heuristics from previous work [12]. One observation in their study is that when the server receives a HTTP request it will send TCP acknowledgments (ACKs) indicating the in-order byte sequence it has received, and all of the request messages will be ACKed before the corresponding HTTP response data is sent (note that here we assume there is no pipelining in use). Hence by looking at the uni-directional flow sent from server to the client, we can infer the size of request by the amount of ACK value advances and the size of response by the amount of data sequence number advances.

Note that, while we can estimate HTTP request/response and bottleneck bandwidth of inbound traffic using only one direction of the traffic (i.e. from servers to clients), we need to have traces from both directions to be able to infer RTT and the bottleneck bandwidth of outbound traffic.

3.2 Comparison with existing tool

To understand if RAMP can perform as well as existing work in terms of generating realistic synthetic workload, we have previously compared RAMP with SURGE [10], a popular web traffic workload generator. We demonstrated that our model parameterization tool is capable of achieving the same functionality of SURGE (i.e. generating similar traffic workload like SURGE) without suffering its limitation.

To validate RAMP against SURGE, we performed an experiment by running SURGE for 30 minutes and recording the traffic via *tcpdump*. We then fed the SURGE trace into

User behavior

1. User arrival is modeled as a Poisson process with a certain rate.
2. The number of pages per user session is randomly picked from the CDF(Cumulative Distribution Function) of trace.
3. the source of the page is chosen from a CDF that matches the popularity of servers
4. Each page is sequentially requested by the users as described below.

Page

1. Page size is chosen from a CDF
2. The inter-arrival time of page is picked from a CDF
3. The number of objects within one page is picked from a CDF
4. The size of request to a page is picked from a CDF
5. User decides a TCP connection is used for multiple request/response exchanges or a single request/response exchange based on the probability of persistent connection (HTTP1.1) versus non-persistent connection (HTTP1.0) computed from the trace. In persistent connection mode, all objects within the same page are sent via the same TCP connection.

Object

1. The inter-arrival time of object is picked from a CDF
2. The size of object is picked from a CDF
3. The TCP window size for both servers and clients are also randomly chosen from a CDF

Figure 2: Structural model of web traffic

RAMP and verified that if the output of ns-2 simulation model from RAMP agreed with SURGE trace. We look at the packet inter-arrival time and wavelet scaling plot of the outputs of SURGE and our model respectively. All the statistics match closely, as shown in Figure 4 and Figure 5.

One limitation of SURGE is that it attempts to fit the models into some widely-used analytic functions (such as using Pareto to describe the distributions of file sizes and off time). However, it is not universally true that all web traffic follow these assumptions. For example, these assumptions would break for a trace distribution site like ITA. We have observed that distribution of page size in ITA traffic (which is mainly made up by simple plain HTML files that describe traces and collection/analysis software) is not heavy-tailed, and hence can not be modeled by SURGE.

To demonstrate this aspect, we simulate the web traffic in ITA data with a SURGE-like analytic model in ns-2, similar to models used in previous studies [5, 6]. We show that this SURGE-like workload model does not accurately reproduce the ITA web traffic. As the wavelet plots shown in Figure 6, the traffic generated by the analytic model does not capture the scaling features of ITA traffic at both small and large time scales.

On the other hand, our tool is based on empirical distributions of traffic and does not have any implicit assumption about the distribution of the traffic, and hence is more flex-

User behavior

1. User arrival is modeled as a Poisson process with certain rate.
2. The number of files transmitted per user session is randomly picked from the CDF(Cumulative Distribution Function) of trace.
3. the source of the file is chosen from a CDF that matches the popularity of servers
4. User starts a new TCP connection for each new file which is sequentially transmitted as described below.

File

1. file size is chosen from a CDF
2. The inter-arrival time of file is picked from a CDF
3. The TCP window size for both servers and clients are also randomly chosen from a CDF

Figure 3: Structural model of FTP traffic

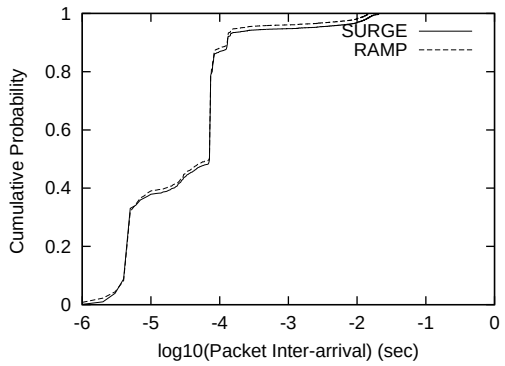


Figure 4: Comparison of packet inter-arrival time between SURGE and RAMP

ible to cope with the diversity of the traffic. As the wavelet plot shown in Figure 7, the ITA model generated by RAMP does capture the important features of ITA traffic (such as a dip at 500ms and similar energy levels).

3.3 Effect of detail in a structured model

Previously, we have developed a three-level model, as shown in Figure 2, to capture the characteristics of web traffic. To understand the importance of capturing the details of application-level structure in order to correctly model the traffic, we compare the results of using a simplified flow-based model (where the hierarchical relationship between page and object has been omitted). As shown in Figure 8, although the plots look similar at smaller time scales, the traffic generated by the simplified two-level web traffic model becomes less bursty at larger time scales (larger than 16 seconds). This example shows that it is important to capture the details of application-level structure (a three-level model rather than a two-level model in this case) in order to accurately reproduce the traffic. Furthermore, using empirical distributions from real traces alone provides no guarantee of model accuracy; differences in application structure also have an important effect on simulation accuracy.

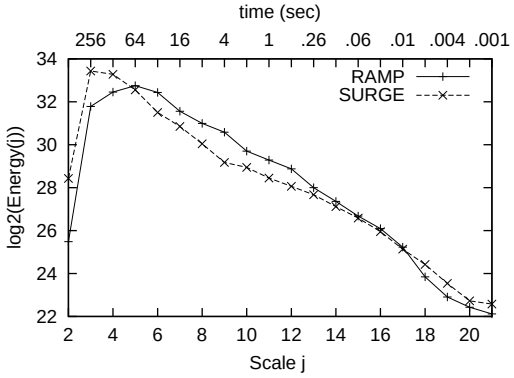


Figure 5: Comparison of wavelet scaling plots between SURGE and RAMP

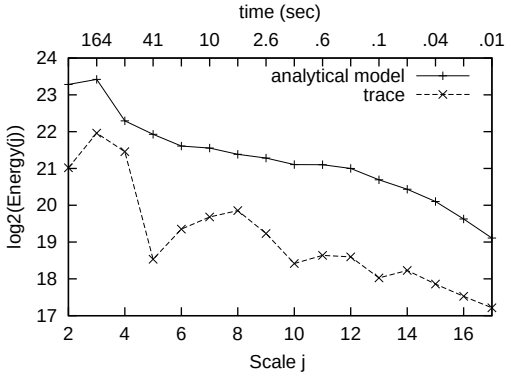


Figure 6: Comparison of wavelet scaling plots between an SURGE-like analytic model and trace for ITA web traffic

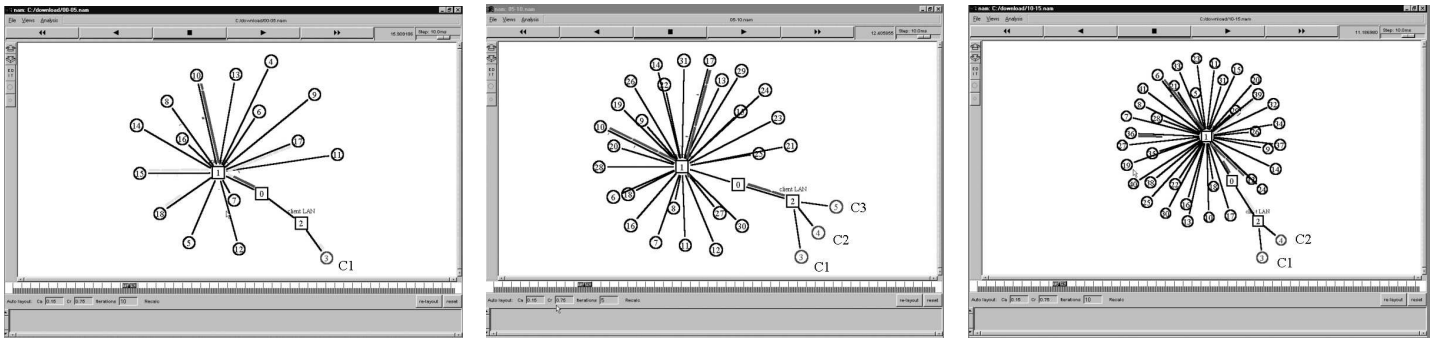
4. NEAR-REAL-TIME TRACE-DRIVEN SIMULATION

One of the design goals of RAMP is to *rapidly* parameterize traffic models. The time required for RAMP from analyzing the traces to finally generating the simulation models typically takes tens of minutes for trace with a size of several hundred megabytes. When running on a 1.7G Hz Pentium IV box with 1G memory, as shown in Table 1, we can see the process time of RAMP is approximately proportional to the number of packets in the trace (and hence also proportional to the file size).

On-line simulation is important for various traffic engineering tasks which have the nature of short turn-around

Trace	ISI-1	ISI-2	ITA
file size (MB)	614	561	203
no. of bytes (GB)	1.0	7.3	2.4
no. of packets (M)	9.2	8.4	2.5
no. of flows (K)	506	398	1.3
process time (min)	25	21	8
speed (thousand packets/sec)	6.1	6.3	5.7

Table 1: Process time of RAMP for different traces



(a) Snapshot of 1st 5-minute traffic (b) Snapshot of 2nd 5-minute traffic (c) Snapshot of 3rd 5-minute traffic

Figure 10: Visualization of ISI network traffic

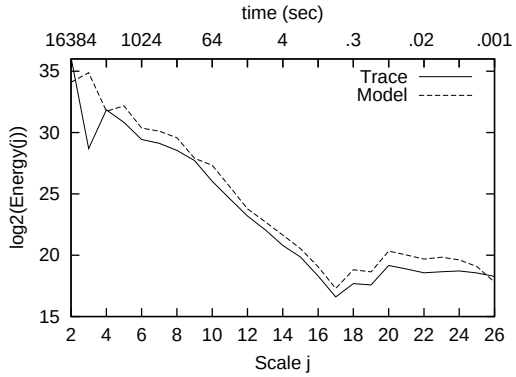


Figure 7: Comparison of wavelet scaling plots between model and trace for ITA outbound traffic

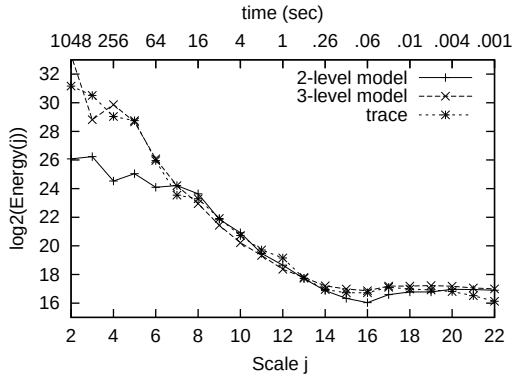


Figure 8: Comparison of wavelet scalin g plots between 2-level and 3-level web models

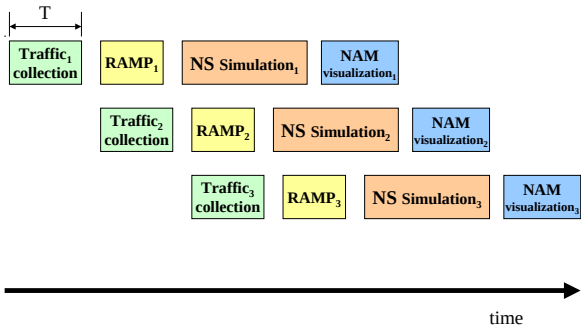


Figure 9: pipelined RAMP

time, such as feedback-based traffic management, adaptive parameter tunning for network control algorithm, short-term traffic prediction etc. RAMP by default takes a trace file as input and processes the traffic off-line. Although, as shown in Table 1, RAMP processing is slightly slower than real time (the traffic rate of ISI is around seven thousand packets per second), with minor software changes RAMP could parameterize the model in near real-time. The primary change to RAMP would be to incrementally update the output CDFs as each new flow arrives, instead of computing all flows at once.

To model and simulate traffic in near real-time, we have modified RAMP so that it can be run in a pipeline fashion as shown in Figure 9. By running several copies of RAMP in parallel with each copy only processing one subset of the traffic and updating the CDFs incrementally, we can have RAMP process the traffic and visualize the results via nam in near real-time.

Note that there are trade-offs in choosing the duration of traffic for each copy of RAMP to process (the time period T in Figure 9). The shorter the duration is, the shorter is the time lag from the time traffic is first recorded to the final visualization of the traffic via nam (and hence closer to "real-time"). On the other hand, while any trace is already bound to have incomplete flows due to its finite-duration nature, when the chosen T is shorter, the chances that one long flow is separated into two or more shorter incomplete flows will increase. There are two aspects to consider in this case. For

flows that start in current measurement period T and end in future time, we keep the states of these flows but ignore them from the computation of RAMP for the current T, which resultedly introduces some inaccuracy into the generated model. For flows start in previous measurement period and end in current T, we use previous stored states to include those incomplete flows into current period. However, the smaller T is, more states of these incomplete flows need to be kept (some flows might last across several T periods). Finally, when the measurement period T is small, it tends to reduce the usable samples (eg. number of back-to-back data packets) for the estimation of topology information and hence the results are more prone to the effect of the noise in the data.

Previous work [19] has shown 98% of Interent flows are shorter than 15 minutes. In our traces, 90% of flows have duration less than 2 minutes and 95% of flows are less than 20 minutes. Intuitively, one can choose the length of duration based on the nature of the traffic. For example, we can pick a shorter duration if the majority of flows are web traffic, and pick a longer duration if there is a significant amount of FTP traffic in the trace.

We have tested this extended version of RAMP on the ISI network. In our experiment, we set the trace collection period T to five minutes. We artificially generated FTP and web traffic and varied the number of clients and servers in each five-minute period (fifteen servers and one client in the first five-minute, twenty-four servers and three clients in the second five-minute, thirty-seven servers and two clients in the third five-minute). For a five-minute trace, it takes around one minute for RAMP to process the traffic and two minutes to run NS simulation using the generated CDFs on a Pentium III machine (obviously these numbers might change depending on the nature of the traffic and the speed of the hardware). Figure 10 shows the visualization of traffic on nam for a duration of fifteen minutes (each nam window models a different five-minute snapshot of the traffic.)

As shown in Figure 1, there are three stages in the data flow of RAMP: collecting traffic, generating models based on previously collected traffic and running ns simulation using the output models. For brevity, in the rest of this section we refer T_{sample} as the traffic collection period, T_{model} as the period of "observation" time that RAMP uses to generate the models and T_{sim} as the time period that simulation runs. In our current implementation, as implied in Figure 9, $T_{sample} = T_{model} = T_{sim}$. Furthermore, the input T_{model} to different RAMP instances are not overlapping (eg. $RAMP_2$ and $RAMP_3$ use disjoint time periods $Traffic_2$ and $Traffic_3$ respectively as inputs). One possible improvement for the pipelined RAMP is to make the choice of T_{model} independent of both T_{sample} and T_{sim}. In other words, the length of T_{model} could be different from the length of T_{sample} and T_{sim}. Additionally, one could allow the choices of T_{model} to be overlapping for different RAMP instances. For example, in Figure 9, instead of using the same duration as the most-recent sampling period (i.e. $Traffic_2$), $RAMP_2$ can use $Traffic_1 + Traffic_2$ as input to generate ns model. Similarly, $RAMP_3$ can generate model based on a longer period of traffic (such as $Traffic_2 + Traffic_3$ or $Traffic_1 + Traffic_2 + Traffic_3$) instead of $Traffic_3$. There are two aspects for allowing users to adjust the length of T_{model}. By setting T_{model} to a longer period, one can reduce the possibility of occurrences of incomplete flows while still model traffic in near "real-time". By setting T_{model} to a shorter period, one can more realistically capture the characteristics of the most recent traffic in the models.

5. APPLICATIONS OF RAMP

In this section, we demonstrate the utilities of RAMP via several immediate applications. First, we utilize RAMP to generate realistic synthetic traffic [2]. Next we demonstrate the use of RAMP to generate synthetic high bandwidth network traces [20]. We then present the results of using RAMP to analyze and model DoS attack traffic [21]. Finally, we discuss some lessons we learned from using RAMP.

5.1 Generation of Realistic Traffic Model for Simulation

Another design goal of RAMP is to generate *realistic* synthetic workload for simulation. To understand if RAMP can accurately reproduce the traffic under study, we use ISI-1 data to evaluate its accuracy. We first feed ISI-1 trace into RAMP, and then incorporate its output into ns-2 simulator and compare the result of the simulation against the original trace. The result shows the output of simulation match the original traces closely. Note that because currently our tool only supports web and FTP traffic, we first filter the traces so that they only contain web and FTP data before being compared against the simulation result (together web and FTP traffic account for 83.7% of the total traffic in terms of the number of bytes, and 48% in terms of the number of packets in the ISI-1 trace).

The statistics we use here for validation including the distributions of flow arrival, flow size, flow duration, packet inter-arrival time, wavelet scaling plot and the application-specific parameters, such as page size, page arrival, object size (for web traffic), file size, file arrival (for FTP traffic), user arrival and user duration. Here we only show outbound traffic and only CDF plots of flow statistics for simplicity. (although the graphs of inbound traffic are not shown here, they are consistent with the results of outbound traffic).

The CDF plots of flow statistics for the ISI-1 model are depicted in Figure 11, which shows that the model matches the trace closely. The Kolmogorov-Smirnov test D values for Figure 11(a), Figure 11(b) and Figure 11(c) are 0.0019, 0.0013, 0.0018 respectively. They all pass the K-S test given a critical value of 0.00874.

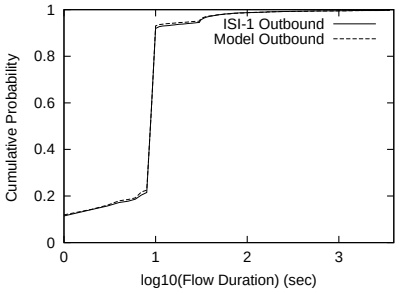
The corresponding wavelet scaling plot for the ISI-1 model, as depicted in Figure 12, also shows large degree of resemblance between trace and model, such as similar energy value (the model has slightly lower energy though) and a dip around 128ms (which reflects the RTT of the underlying traffic).

The CDF plots of model parameters such as page/file size, user arrival etc. also match closely (are not shown here), though it is not surprising though since the model is directly driven by those parameters.

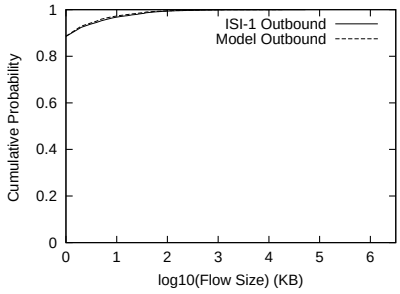
All the statistical comparisons show RAMP is able to accurately reproduce the original traffic. More details can be found in [2].

5.2 Generation of High Bandwidth Network Traffic Traces

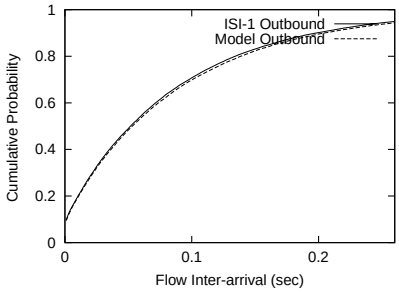
Most real world network traces that are publicly available are low bandwidth traces from OC3c, OC12c links or



(a) Comparison of flow duration between model and ISI-1 trace



(b) Comparison of flow size between model and ISI-1 trace



(c) Comparison of flow inter-arrival time between model and ISI-trace

Figure 11: Comparison of flow statistics for model and ISI-1 outbound traffic

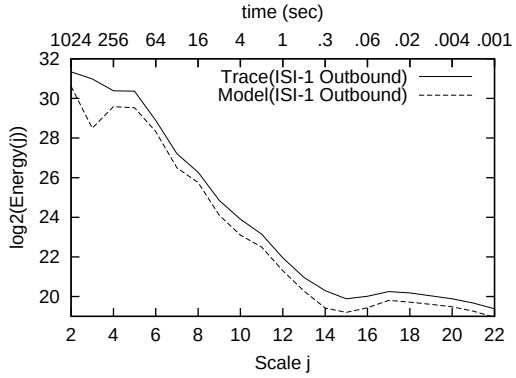


Figure 12: Comparison of wavelet scaling plots between model and trace for ISI-1 o utbound traffic

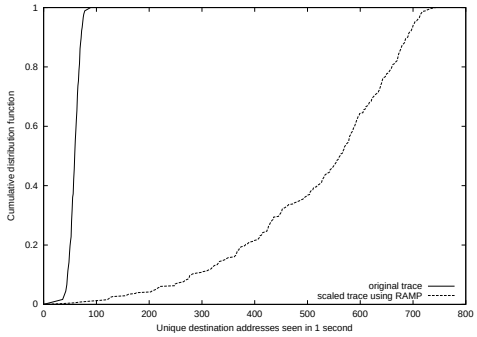


Figure 13: Distributions of unique destination addresses in an interval between original trace and scaled high bandwidth trace using RAMP

FDDI or Ethernet traces. In contrast to the wide availability of low bandwidth traces, traces from high bandwidth links (OC48, OC192 links, such as in the core of the Internet) are not widely available. The administrative constraints and difficulty of implementing a mechanism for collecting traces at high speeds [22] are factors that contribute towards the lack of public availability of such traces. As link speeds increase in the future, the difficulties involved in obtaining high bandwidth traces will increase.

In the absence of real high bandwidth traces, one option is to attempt to generate the traces that are likely to resemble real traces. Applications of such traces include studies of the behavior of routers, switches and network protocols on high speed links.

To utilize RAMP to generate high bandwidth traces, we first use RAMP to extract the statistics of user and application characteristics and topology information such as link latencies and bottleneck link bandwidth from low bandwidth traces. We then feed the output of RAMP into a ns simulation model.

To obtain higher bandwidth traces from the simulation, we first scale the number of nodes based on a previous study [23], which has shown that the host pairs increases as the square root of the bit rate. To simulate 1Gbps link with 10Mbps trace, the number of servers and clients was chosen to be approximately 10 times the number of server and client IP addresses found in the trace. The link bandwidths were increased linearly by a factor of 100 and the user inter-arrival rates were decreased linearly by a factor of 100.

Simulating a backbone network topology is a difficult task because simulating the entire topology along with traffic sources at each node strains the available computing resources [1]. The topology chosen should be such that it should be easy to increase the amount of traffic on the link on which the traffic trace is recorded. Hence, we have simplified the backbone topology to a dumbbell topology with clients and servers on either side of the bottleneck link. A packet from a client to a server traverses four router nodes. In the simulation, the traffic traces are collected on the bottleneck link.

We evaluate the quality of generated traces using several metrics including distributions of packet inter-arrival time, the number of unique destination address, flow duration and flow size etc. For brevity, here we only show the comparison of the number of unique destination addresses seen in an interval between the original traffic and the generated high bandwidth trace.

It is important to properly scale the unique destination address in the generated synthetic high bandwidth traces. The

destination address determines where the packet needs to be sent and the distribution of the number of unique addresses seen in an interval affects mechanisms such as routing table caches used in a switch or router. As shown in Figure 13, the unique destination addresses in the generated high bandwidth trace has been scaled by a factor of around 10, which correctly reflects the scaling in the number of destination hosts in the simulation.

More details about using RAMP to generate high bandwidth synthetic traces are described in [20].

5.3 Analysis and modeling of malicious traffic

The sudden and explosive growth of the Internet has witnessed a very complex evolution of network traffic patterns giving rise to *normal* and *malicious* traffic. Normal traffic includes traffic generated by well-known services and applications: web, ftp, nntp, and smtp, for example. On the other hand, malicious traffic is generated by non compliant applications and causes disruptive effects on the network, such as DDoS attacks and worm traffic.

During the last few years, network traffic has increasingly witnessed a surge in *malicious traffic*. One question that occurs naturally is how this malicious traffic affects the *normal* background traffic. To answer this question, we used an extended version of RAMP as a trace-driven analysis and modeling tool to study the characteristics of network traffic during phases dominated by malicious behavior of DDoS attacks and worm propagation, and compare it with phases when such activity is negligible.

We looked at several metrics to understand the impact of malicious traffic such as DDoS and worm on the network. Our study concentrates on short-lived flows such as web mice and DNS traffic which are more delay-sensitive than long-lived flows like FTP. We investigate the impact of malicious traffic on the DNS lookup latency and web latency. DNS lookup latency is defined as the time lapse between the client sending out a request to the DNS sever and the client finally receiving an answer from a DNS server that terminates the lookup, by returning either the requested name-to-IP mapping or an error indication. To extract the statistics about lookup latency, we adopt an approach that is similar to that used in previous study [24]. We define web latency as the time lapse between the issue of HTTP request to the receiving of response data.

The traces we utilize in this study are from two different locations: one at Los Nettos [25], a regional area network in Los Angeles, and the other at the Internet2 [26] peering link at USC. Los Nettos has peering relationships with Verio, Cogent, Genuity, and the LA-Metropolitan Area Exchange, and serves a diverse clientele that includes academic institutes and corporations around the Los Angeles area. We monitor the Verio and Cogent peering links that experience an average utilization of 11% at 110Mbps and 38Kpps (packets-per-second) The USC trace machine monitors the Internet2 traffic to and from USC. The average utilization of link monitored by the trace machine is 6% at 60Mbps and 25Kpps.

5.3.1 Analysis of DDoS traffic

We used RAMP to analyze the distributions of DNS latency and web latency from 12 DDoS attack traces. We found that the average DNS latency can increase to as high as 230% and the average web latency can increase by 30% upon interaction with DDoS traffic. Figure 14 shows the change in latency at Los Nettos during one particular ping reflection attack [27]. This attack employs 145 distinct reflectors located in different countries such as Brazil, Japan, Korea, Singapore, and United States, generating attack rates of 4300pps. During the attack, we observed a 230% increase in the mean latency for DNS lookup, from 0.13s (with median latency 11ms) to 0.44s (with median latency 23ms). We believe the sudden increase of traffic during an attack leads to higher average buffer occupancies at the router, resulting in increased queuing delays. We also looked at the effect of DDoS attack on web mice since such flows are more sensitive to the delay. As shown in Figure 14(b), the mean latency of web flows has increased from 9s (with median latency 1.2s) to 11.9s (with median latency 2.9s), resulting in a 30% increase during the attack. Note that the DNS and web latencies increase even when the link is still under-utilized (11%).

5.3.2 Modeling DDoS attack generated by worm-infected hosts

Worm infection is on the rise. Worms like Code Red and Nimda can infect thousands of hosts within short periods of time and generate significant network traffic [28]. We have observed the propagation of the Apache mod_ssl worm (aka the Slapper worm) in our traces. The Slapper worm propagation did not generate disruptive amounts of traffic at our data collection point. However, if all the infected machines launched a coordinated DDoS attack, it would have had a disastrous effect. To understand its effect on the network when all worm-infected hosts launch a coordinated DDoS attack, we used RAMP to derive the distributions of RTT of the worm-infected network, and the flow rates and packet size of DDoS traffic from the traces. The derived empirical distributions were then input into ns simulation. We use a simple dumbbell topology to model the network and constant bit rate sources to model DDoS attackers.

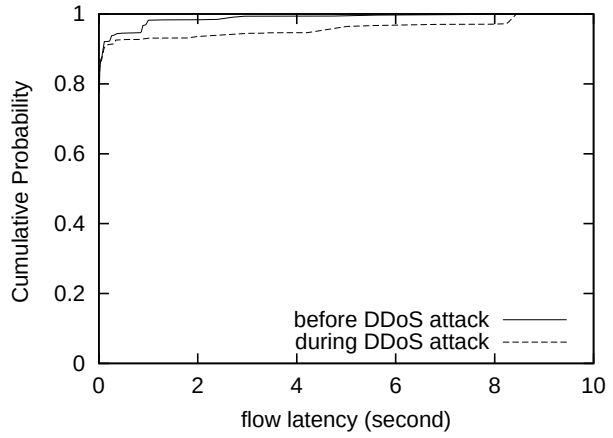
Figure 15 shows the attack intensity when generated by worm-infected hosts. We observed that the different RTT distributions of the attackers cause distinctively different transient ramp-up behavior before the steady state attack rate is achieved. Also, when all the worm-infected hosts launch a DDoS attack, the average traffic generated due to the attack during the steady state is fifty times larger than that generated by the DDoS attack that we traced.

The above results show that although short duration DDoS attacks might not be disruptive in terms of causing network failures and reducing aggregate throughput, delay-sensitive traffic such as DNS and small/medium web transaction will still be affected by these attacks. Over-provisioning the links alone does not provide the complete solution, since the short burst of DDos traffic can result in increases in latency without affecting the throughput. We feel that the above observations can be used as hints to design better AQM mechanisms to provide differential services in order to protect short-lived traffic.

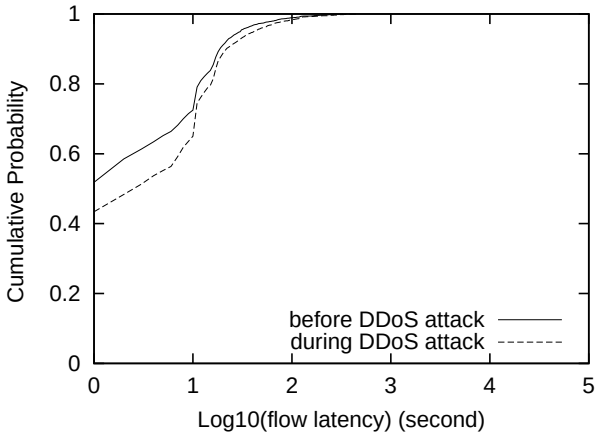
More details about using RAMP to study the impact of malicious traffic on the network are described in [21].

5.4 Discussion

In this section, we describe some experiences we learned from using RAMP in these applications: the use of trace-driven application-level model.



(a) DNS lookup latency increases by 230% during attack



(b) Latency experienced by web flows increases by 30% during attack

Figure 14: Increase in DNS and web latency during DDoS attack at Los Nettos

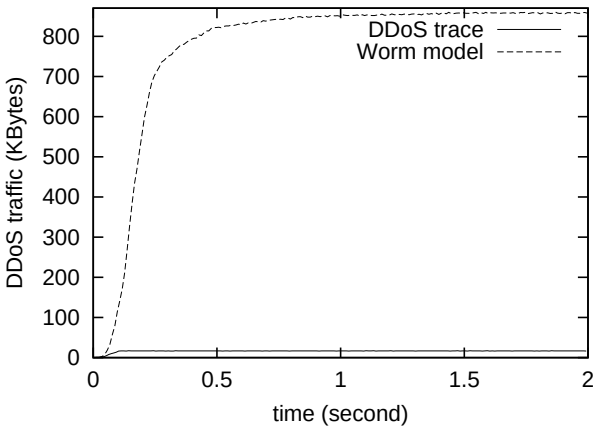


Figure 15: Comparison of DDoS attack intensities; the DDoS attack and when an attack is launched by worm-infected hosts

The design of RAMP is based on the structural modeling approach [8] which implicitly takes into account the hierarchical structure of application and intertwined networking mechanisms to reproduce the traffic. Using a trace-driven application-level model provides a short turn-around time from analyzing to modeling new types of traffic. Considering the constantly-changing nature of the Internet traffic, it is very important to be able to quickly reproduce *real world* traffic from measurement for the simulation study before the data becomes obsolete or irrelevant. By modeling DDoS attackers as CBR sources (Section 5.3), it only took us a few hours to extend RAMP to support DDoS traffic. On the other hand, if we had chosen to model DDoS traffic using traditional time-series analysis approaches, the effort for distribution estimation and parameters fitting alone would have taken days if not weeks.

Furthermore, while the network infrastructure is continuously upgraded and network traffic is constantly changing, for most applications the application-level pattern (such as request/reply patterns in web traffic) in which data is sent does not react to the network dynamics. By capturing this application structure invariant, RAMP provides an efficient and reasonable way to predict future traffic (Section 5.2): by taking a sample of original traffic and feeding it into a scaled version of the system, we can predict the future traffic in a consistent manner (Although it is impossible to prove that RAMP will accurately predict future high-rate traffic until we have traces from such links, it is important to provide researchers a tool to study *plausible* traffic to evaluate future links). On the contrary, it will be a non-trivial task if we try to employ traditional time-series forecasting techniques to achieve similar results.

6. FUTURE WORK

Currently, except from keeping the history of incomplete flows, our simple implementation of pipelined RAMP does not manage to transit the states of network traffic from one instance of RAMP to the other. Hence, it might not be applicable for solving some traffic engineering problems that require the knowledges of long-term properties of network traffic. We plan to investigate this issue as future work.

Possible improvements to RAMP in the future would include a better queuing model, support of other types of important traffic, modeling of temporal relationship between different types of traffic, long-term traffic prediction and integration of distributed measurements.

We modeled queuing delay as an extra component of propagation delay instead of the end result of interaction between aggregation of flows and limited buffer size (which is hard to characterize just by only looking at TCP/IP header information). This approach was sufficient for our data sets which have low link utilization and zero packet drop. However, for sites that experience serious congestion (like flash crowd), our approximation might introduce some inaccuracy

into the result and require further study.

Our tool currently supports web, FTP and DoS traffic, which only accounts for a subset of real network traffic. To make the output of RAMP more representative, we would like to incorporate other types of important traffic such as DNS, multimedia traffic (such as Real Audio/Video), game traffic and increasingly popular peer-to-peer traffic into our tool.

To accurately model traffic, it is important to characterize the temporal relationship between different types of traffic. For example, DNS behavior is very likely linked closely to web traffic pattern since most of the web connections are usually preceded by DNS lookups. We plan to study this issue and understand how to orchestrate different traffic classes correctly in the model.

Currently our model is based traces recorded at a single tap point of network. However, distributed measurement is required in order to get a network-wide view of traffic and to correctly model the behavior of cross traffic, while keeping the size of collected data maintainable. Integrating distributed data together would require approaches for overlap detection and hole filling. To address this problem, we plan to explore and extend the techniques developed in previous work of distributed network monitoring such as SCAN [29] and recent work in network tomography [30, 31, 32, 33, 34], as well as to employ new algorithms and tools to merge distributed data into a coherent model.

Measurement study of Internet traces shows that the WAN performance is reasonably stable over terms of several minutes; meanwhile, nearby hosts experience similar or identical throughput performance within a time period measured in minutes [35, 16]. Our model parameterization tool outputs simulation model at the time scale of tens of minutes for hour-long traffic, which matches the level of stability reported in previous study and hence is applicable to simulate *present* traffic and predict short-term traffic trend. However, to simulate and predict long-term trend of traffic (for example, at the time scale of days), we need to understand how the traffic evolves and correlates in time.

7. CONCLUSION

Floyd and Paxson [1] characterized the problems, the constantly-changing and decentralized nature of the Internet, result in a poor understanding of traffic characteristics and make it difficult to define a typical configuration for simulating the Internet. Motivated by their observations, we develop a tool called *RAMP* that support rapid parameterization of live network traffic for generating realistic application-level simulation models. Our model is based on estimation of user/application behaviors and network conditions from captured tcpdump trace. In this paper, we first describe the design and implementation of RAMP. We then extend RAMP to support near-real-time trace-driven simulation. Finally, we demonstrate the utilities of RAMP via three immediate applications: generation of realistic synthetic traffic for simulation, generation of high bandwidth network traces, and analysis and modeling of malicious traffic.

Acknowledgements

We would like to acknowledge the many contributions of Polly Huang to this work. Her contributions include software for producing global scaling plots and more importantly some technical conversations about how to interpret the output of wavelet scaling analysis. We would also like to thank Walter Willinger for providing useful insight in interpreting the result of Kolmogorov-Smirnov test. For collection of traces we would like to thank the help of Vern Paxson of ICIR, and Alefiya Hussain and Jim Koda of ISI. We would like to thank F. Donelson Smith and Felix Hernandez Campos for providing part of their codes to us for reconstructing HTTP connection information [12]. We would also like to thank Purushotham Kamath, Alefiya Hussain and Debojyoti Dutta for providing helpful feedback from using RAMP in their work [20, 21], and Tim Buchheim for helping integrate RAMP with other research activities. Finally, we thank the anonymous reviewers for their valuable comments to help us improve the quality of this paper.

8. REFERENCES

[1] S. Floyd and V. Paxson, "Difficulties in simulating the Internet," *ACM/IEEE Transactions on Networking*, vol. 9, no. 4, pp. 392–403, Feb. 2001. [Online]. Available: http://www.icir.org/vern/papers.html

[2] K. Lan and J. Heidemann, "Rapid model parameterization from traffic measurement," *ACM Transactions on Modeling and Computer Simulations*, 2003. [Online]. Available: http://www.isi.edu/~kclan/research.html

[3] V. Paxson, "Internet Traffic Archive, http://www.acm.org/sigcomm/ita/," 2000. [Online]. Available: http://www.acm.org/sigcomm/ITA/

[4] P. Abry and D. Veitch, "Wavelet analysis of long-range-dependent traffic," *IEEE Transactions on Information Theory*, vol. 44, no. 1, pp. 2–15, 1998. [Online]. Available: citeseer.nj.nec.com/abry98wavelet.html

[5] A. Feldmann, A. C. Gilbert, P. Huang, and W. Willinger, "Dynamics of IP traffic: A study of the role of variability and the impact of control," in *Proceedings of the ACM SIGCOMM*. Cambridge, MA, USA: ACM, Aug. 1999, pp. 301–313. [Online]. Available: http://www.acm.org/sigcomm/sigcomm99/papers/session8-3.html

[6] P. Huang, A. Feldmann, and W. Willinger, "A non-intrusive, wavelet-based approach to detecting network performance problems," in *Proceeding of ACM SIGCOMM Internet Measurement Workshop 2001*, San Francisco Bay Area, Nov. 2001. [Online]. Available: http://www.tik.ee.ethz.ch/~huang/publication/

[7] F. J. Massey, "The kolmogorov-smirnov test of goodness of fit," *Journal of the American Statistical Association*, vol. 46, no. 253, pp. 68–78, Mar. 1951.

[8] W. Willinger, V. Paxson, and M. Taqqu, "Self-similarity and heavy-tails: Structural modeling of network traffic," in *Self-Similarity and Heavy-Tails: Structural Modeling of Network Traffic, in A Practical Guide To Heavy Tails: Statistical Techniques and Applications, R.J. Adler, R.E. Feldman and M.S. Taqqu, editors. ISBN 0-8176-3951-9. Birkh auser, Boston, 1998.*, 1998.

[9] B. Mah, "An empirical model of HTTP network traffic," in *Proceedings of the IEEE Infocom.* Kobe, Japan: IEEE, Apr. 1997, pp. 592–600. [Online]. Available: http://www.ca.sandia.gov/~bmah/Papers/Http-Infocom.ps

[10] P. Barford and M. Crovella, "Generating

representative web workloads for network and server performance evaluation," in *Proceedings of the ACM SIGMETRICS*. Madison WI: ACM, Nov. 1998, pp. 151–160. [Online]. Available: http://cs-www.bu.edu:80/faculty/crovella/paper-archive/sigm98-surge.ps

[11] M. E. Crovella and A. Bestavros, "Self-similarity in world wide web traffic: evidence and possible causes," in *Proceedings of the ACM SIGMETRICS*. Philadelphia, Pennsylvania: ACM, May 1996, pp. 160–169. [Online]. Available: http://www.cs.bu.edu/~best/res/papers/sigmetrics96.ps

[12] F. D. Smith, F. H. Campos, K. Jeffay, and D. Ott, "What TCP/IP protocol headers can tell us about the web," in *SIGMETRICS/Performance*, Cambridge, MA, June 2001, pp. 245–256. [Online]. Available: http://www.cs.unc.edu/~jeffay/networking.html

[13] B. Mah, P. Sholander, L. Martinez, and L. Tolendino, "Ipb; an internet protocol benchmark using simulated traffic," in *Proceedings of MASCOTS '98*. Montreal, Canada: IEEE, Aug. 1998. [Online]. Available: http://www.employees.org/~bmah/Papers/

[14] M. Yuksel, B. Sikdar, K. S. Vastola, and B. Szymanski, "Workload generation for ns simulations of wide area net works and the internet," in *Proceedings of Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS) part of Western Multi-Conference (WMC)*, San Diego, CA, 2000, pp. 93–98. [Online]. Available: http://www.cs.rpi.edu/~yuksem/

[15] L. Breslau, D. Estrin, K. Fall, S. Floyd, J. Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, Y. Xu, and H. Yu, "Advances in network simulation," *IEEE Computer*, vol. 33, no. 5, pp. 59–67, May 2000, expanded version available as USC TR 99-702b at http://www.isi.edu/~johnh/PAPERS/Bajaj99a.html. [Online]. Available: http://www.isi.edu/~johnh/PAPERS/Breslau00a.html

[16] V. Paxson, "Measurements and analysis of end-to-end internet dynamics," in *Ph.D. thesis, University of California, Berkeley*, Apr. 1997. [Online]. Available: http://www.icir.org/vern/papers.html

[17] K. Lai and M. Baker, "Measuring bandwidth," in *INFOCOM (1)*, 1999, pp. 235–245. [Online]. Available: http://gunpowder.stanford.edu/~laik/projects/nettimer/

[18] ——, "Nettimer: A tool for measuring bottleneck link bandwidth," in *Proceedings of the USENIX Symposium on Internet Technologies and Systems*, Mar. 2001. [Online]. Available: http://gunpowder.stanford.edu/~laik/projects/nettimer/

[19] N. Brownlee and kc claffy, "Understanding internet traffic streams: Dragonflies and tortoises," *IEEE Communication*, 2002. [Online]. Available: http://www.caida.org/outreach/papers/2002/Dragonflies/

[20] P. Kamath, K. Lan, J. Heidemann, J. Bannister, and J. Touch, "Generation of high bandwidth network traffic traces," in *Proceedings of MASCOTS '02*. Fort Worth, Texas, USA: IEEE, Oct. 2002, pp. 401–410. [Online]. Available: http://www.isi.edu/~kclan/paper/tg.ps

[21] K. Lan, A. Hussain, and D. Dutta, "Effect of malicious traffic on the network," in *PAM 2003*, San Diego, CA, April 2003. [Online]. Available: http://www.isi.edu/~kclan/research.html

[22] G. Iannaccone, C. Diot, I. Graham, and N. McKeown, "Monitoring very high speed links," in *Proceedings of the ACM SIGCOMM Internet Measurement Workshop '01*, San Francisco, California, November 2001. [Online]. Available: http://www.acm.org/sigcomm/imw2001/program.html

[23] K. Claffy, "Internet measurement: myths about internet data," in *Usenix LISA 2001*, San Diego, CA, Dec. 2001. [Online]. Available: http://www.caida.org/outreach/presentations/Myths2002/

[24] H. B. Jaeyeon Jung, Emil Sit and R. Morris, "Dns performance and the effectiveness of caching," in *Proceedings of the ACM SIGCOMM Internet Measurement Workshop '01*, San Francisco, California, November 2001. [Online]. Available: nms.lcs.mit.edu/papers/dns-imw2001.html

[25] L. N.-P. packets since 1988, http://www.ln.net.

[26] I. 2, http://www.internet2.edu.

[27] V. Paxson, "An analysis of using reflectors for distributed denial-of-service attacks," *ACM Computer Communications Review (CCR)*, vol. 31, no. 3, July 2001. [Online]. Available: http://www.icir.org/vern/papers/reflectors.CCR.01.ps.gz

[28] H. Wang, D. Zhang, and K. Shin, "Detecting syn flooding attacks," in *Proceedings of the IEEE Infocom*. New York, NY: IEEE, June 2002, pp. 000–001. [Online]. Available: citeseer.nj.nec.com/508971.html

[29] R. Govindan, C. Alaettinoğlu, and D. Estrin, "Self-configuring active network monitoring (SCAN)," Feb. 1997, white paper.

[30] CAIDA, "Internet measurement infrastructure," 2002, http://www.caida.org/analysis/performance/measinfra/.

[31] M. Mathis and J. Mahdavi, "Diagnosing internet congestion with a transport layer performance tool," in *Proceedings of INET '96*, Montreal, June 1996. [Online]. Available: http://www.psc.edu/~mathis/papers/

[32] Y. Vardi, "Network tomography : Estimating source-destination traffic intensities from link data," *The Journal of American Statistics Association*, vol. 91, no. 433, pp. 365–377, Mar. 1996.

[33] J. Cao, D. Davis, S. Wiel, and B. Yu, "Time-varying network tomography : Router link data," *The Journal of American Statistics Association*, vol. 95, no. 452, pp. 1063–1075, Feb. 2000. [Online]. Available: http://cm.bell-labs.com/cm/ms/departments/sia/cao/htmls/pub.html

[34] J. Cao, S. V. Wiel, B. Yu, and Z. Zhu, "A scalable method for estimating network traffic matrices," *Bell Labs Tech. Report*, 2000. [Online]. Available: http://cm.bell-labs.com/cm/ms/departments/sia/cao/htmls/pub.html

[35] H. Balakrishnan, M. Stemm, S. Seshan, and R. H. Katz, "Analyzing stability in wide-area network performance," in *SIGMETRICS/Performance*, 1997. [Online]. Available: www-2.cs.cmu.edu/~srini/Papers/publications/1997.sigmetric/sigmetrics97%.pdf