ANALYSIS OF PASSIVE END-TO-END NETWORK PERFORMANCE MEASUREMENTS

A Dissertation Proposal
Presented to
The Academic Faculty

By

Charles Robert Simpson, Jr.

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
in
Electrical and Computer Engineering

School of Electrical and Computer Engineering
Georgia Institute of Technology
December 2006

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SECTION 1

INTRODUCTION

The Internet has become an increasingly essential part of modern life. To accommodate this growth and increased interest, much research needs to be completed on the behavior of the Internet and Internet users as well as the overall performance of this vast global network. In response to this need, network measurements have become a hot topic in Internet research.

The area of network measurements has recently become a major focus for those who wish to have an understanding of the traffic patterns of the Internet. The analysis of these measurements has led to better models for traffic flows, file sizes, burst sizes, and many other complex characteristics of the Internet. Such measurements have implications for the performance of networking protocols and operating system protocol stacks, the study of malicious network traffic, and the modeling and simulation of networks. Most of the sampling techniques for this data have come either from active measurements (`ping`[1]) or from localized passive measurements (`tcpdump`[2]). It has been documented that active measurements introduce bias into these measurements, and many claim that this bias is so great that some measurements collected are not representative of actual Internet traffic [3, 4]. As for the passive measurements that have been conducted, they are only able to analyze a small percentage of the Internet and cannot give a good representation of the end-user experience as they are only collected from a small number of hosts. It is important to be able to collect measurements from the perspective of the end-user because such a perspective gives an excellent insight into the "real" use of the network. Thus, the need for the study of large-scale, end-to-end, passive network measurements.

1.1 NETI@home

We introduced the NETI@home (NETwork Intelligence) software package, a distributed network monitoring infrastructure whose aim is to passively capture measurements from

end–hosts on the Internet, to address this need[5, 6]. Capturing measurements from end–hosts gives us a unique perspective on the behavior of both the network and network users. NETI@home is designed to run on end–user computers with a variety of operating systems and to require little or no intervention by the user. It collects an assortment of network measurements from these machines and then sends the results to Georgia Tech for analysis.

Another major issue for the large-scale collection of passive end-to-end network measurements is the privacy of the end users. Any collection done on an end-user's system must not invade their privacy whatsoever. Should their privacy be infringed upon, at the very least it would spell the end for such a large-scale collection system.

For more information on the NETI@home project, please see my Master's Thesis[7] as well as several NETI@home related publications including [5, 8, 9].

With thousands of users strewn across the globe, we believe we can leverage the volume and quality of the NETI@home data to provide a better understanding of the state of the global Internet. The research proposed utilizes data collected by NETI@home to examine aspects of network security, user behavior, and network behavior.

1.2 Network Security

As with any new technology, the Internet has grown from its infancy to a stage where security concerns become a considerable problem. Today's Internet is plagued with a plethora of worms, viruses, malware, spam, and other malicious traffic. Utilizing networking measurements enables researchers to study the growth and spread of this malicious activity, as well as investigate the origins of these problems.

1.3 User Behavior

The simulation of computer networks has become a popular method to evaluate characteristics of these networks across a wide range of topics, including protocol analysis, routing stability, and topological dependencies, to name a few. However, for these simulations to

yield meaningful results, they must incorporate accurate models of their simulated components.

One such component is end–user traffic generation. This component should be network–independent so that it can be used in a wide variety of simulation configurations without dependency on the simulated environment. These traffic models should be updated frequently, using recent measurements, to accurately reflect the changing nature and uses of the Internet. Further, such measurements should represent the heterogeneous connection methods and diverse locations of Internet users.

1.4 Network Behavior

The end–hosts which make up the Internet run a variety of operating systems, each with its own somewhat unique network protocol stack. The differences in these protocol stacks can have a significant impact on the interoperability of Internet end–hosts. Further, many protocol stacks lag far behind the research community and the IETF when implementing improvements to networking protocols. Finally, as with any complex piece of software, operating system network protocol stacks are plagued with bugs. By utilizing the observations and analysis of network measurements, these issues can be identified and hopefully corrected.

SECTION 2

ORIGIN AND HISTORY OF THE PROBLEM

2.1 Measurement Infrastructures

There are several projects whose aim is to collect and distribute network measurements. In this section we will attempt to categorize these projects. However, due to the sheer number of such projects, a complete and updated list cannot be attempted here.

One key distinction between network measurements is whether they are active or passive measurements. Currently, the NETI@home project collects only passive measurements, that is, measurements that do not inject network traffic and try to minimize the impact of their actions on the measured phenomenon. Active measurements [10, 11], on the other hand, directly interact with the system they are attempting to measure. These measurements typically inject network traffic, such as probe packets. Passive measurements [12, 13, 14] do not inject any traffic into the network; they merely monitor traffic on the network and infer measurements from the observed traffic. Many studies have compared active and passive measurements [3, 4] with both methods having advantages and disadvantages.

Active measurements have been used since the early days of the Internet. Some popular active measurement tools are `ping`[1] and `traceroute`[15]. While active measurements provide meaningful data in some cases, there are many measurements that cannot be feasibly made using active techniques, such as accurate statistics on packet loss [3]. Active measurements can also introduce a large amount of bias into the measured system, since they actually inject, and thus interact with, the measured traffic.

Passive measurements, on the other hand, have the goal of minimally affecting the measured network. The most popular passive measurement tools are sniffer based tools such as `tcpdump`[2] and Ethereal[16], which actually sample every packet that is seen on the link (in promiscuous mode) or every packet that is sent from or received by the host that

is sniffing (without promiscuous mode). Using passive measurements, one is able to see the actual users' experiences, if the passive measurements are made at the end host. Other forms of passive measurements observe the network at a point that is between the two end hosts. Two such proposed systems are OC3MON [12] and IPMON [13]. In addition, many studies have analyzed Internet traffic from a variety of points, studying different metrics such as round-trip times (RTTs), available bandwidth, packet loss, and various aspects of protocols such as TCP (receiver window sizes, throughput, time to live values, etc.). Internet Service Providers (ISPs) also collect many network measurements that would be useful to the research community for analysis [12, 13]. However, most ISPs are reluctant to release such information since it could potentially expose problems in their own networks [17].

In addition to being active or passive, many measurement infrastructures specialize in the types of measurements that they collect. We have identified three basic types of measurements infrastructures: general purpose monitoring infrastructures, security–related monitoring infrastructures, and network measurement monitoring infrastructures.

The first class, general purpose monitoring infrastructures, collect measurements which can be used for both security–related research and network measurement related research. The most basic type of general purpose measurements are passive network traces, such as those collected using `pcap`[18]. Several institutions, researchers, and network administrators collect traces. These traces are usually limited to the collector's own network. One of the earliest network monitoring infrastructures, which falls into this class of general purpose monitoring infrastructures, is Vern Paxson's National Internet Measurement Infrastructure, NIMI[10]. NIMI utilizes an active approach, taking measurements between specialized nodes placed strategically throughout the Internet. However, if this infrastructure can ever be put in place, it will still be unable to give an entirely accurate representation of the end–user experience, and it suffers from the unwillingness of Internet service

providers (ISPs) to install such measurement devices on their networks. Other infrastructures of note are SATURNE[11] (active), OC3MON[12] (passive), IPMON[13] (passive), and CAIDA's CoralReef[14] (passive). This class also includes the NETI@home project.

The second class of monitoring infrastructures, those related to security, collect measurements used to identify and track malicious activity on the Internet. One basic approach is the use of Honeynets[19, 20], machines placed on the Internet which have no other purpose than to record the traffic they receive, which largely consists of malicious activity. A different approach is CAIDA's Network Telescope project[21], which monitors an unused portion of the IP address space. This traffic, much like the traffic to honeynets, largely consists of malicious activity. One of the largest and most distributed security–related monitoring infrastructures is the Internet Storm Center[22]. The Internet Storm Center consists of a large number of independent monitoring systems such as honeynets, which aggregate their data, allowing a much larger picture of the malicious activity on the Internet.

Finally, the remaining class of monitoring infrastructures, those specializing in network measurements, collect measurements to further research into network design, operation, and performance. Two recent projects with competing aims have been developed to actively map the Internet with traceroute–like activity, the DIMES Project[23] and Traceroute@home[24].

2.2 Network Security

Much research has been accomplished on studying Internet worms and their behavior. Work has been accomplished on characterizing and looking at the trends of various worms[25, 26]. Further, a detailed study of the spread time, algorithms, and damage caused by recent worms has been conducted. For example, Shannon et. al. give an in-depth look at the Witty worm in [27], and Moore et. al. give an in-depth look at the Slammer worm in [28]. Both of these worms have been observed in the NETI@home dataset as well as other datasets studied such as those from the Georgia Tech Honeynet[29]. Data shows that these worms'

lingering effects are still active. CAIDA uses their previously mentioned Network Telescope, which consists of a full /8 network in order to observe worms, DoS attacks, network scanning, and other malicious activity[21]. Finally, the also previously mentioned Internet Storm Center is used to provide users and organizations with warnings against possible new threats seen on the Internet[22].

2.3 User Behavior

Portions of this work are based on work presented in [30] and [31] and we have chosen to adopt much of their nomenclature. However, we have attempted to expand upon their work in several ways. First, the work in [30] is based on packet traces collected from a campus network. In an attempt to represent more typical end–users, we use data collected by the NETI@home project. Also, the studies conducted in [30, 31] were specific to TCP connections on port 80. We, however, model any given TCP or UDP port, as well as all TCP or UDP traffic aggregated.

The need for accurate simulation models was discussed in [32]. Several other studies have discussed modeling of either application–specific [33, 34, 35, 36, 31] or general [37, 38, 39, 40] end–user network traffic. Also, several studies have used network traffic models in simulation environments including [41, 42, 43, 44].

2.4 Network Behavior

The study of end–hosts, network protocols, and operating system behavior has received much attention in the research literature. Among some of the more famous studies are those related to the performance and modeling of protocols such as TCP[45, 46].

Among protocol flags and options that have been studied, one of the most studied is the IP fragmentation option. Several studies have recommended the almost total abandonment of packet fragmentation and the adoption of this idea including [47, 48].

SECTION 3

PRELIMINARY RESEARCH

3.1 NETI@home

NETI@home (NETwork Intelligence at home) is an open–source software package named after the popular SETI@home[49] software. The NETI@home client is available on the NETI@home website[6] and is designed to be run by any client machine connected to the Internet. When run on a client machine, the NETI@home software reports end–to–end flow summary statistics to a server at the Georgia Institute of Technology. The statistics collected and the functionality of the software are discussed in [5]. Since NETI@home is designed to run on end–user systems, it provides a unique perspective into the behavior of both end–users and their systems.

NETI@home is designed to run on any end–host machine connected to the Internet, to maximize the volume and variety of measurements gathered. As such, it has been written for and tested on the Windows, Solaris, Linux, and Mac OS X operating systems. Our basic approach is to sniff packets sent from and received by the monitored host and infer performance metrics based on these observed packets. NETI@home is built on top of the popular `pcap` software library[18], the defacto cross–platform packet sniffing library, and is written in C++. Further, we run the NETI@home software in non–promiscuous mode, which insures that we only observe and report on traffic specifically addresses to that end–system, to eliminate the possibility of duplicate measurements, to respect the rights and privacy of others, and to guarantee the collection of end–to–end measurements. One important requirement of the software is that it is to be unobtrusive and run quietly in the background with little or no intervention by the user, and using few resources.

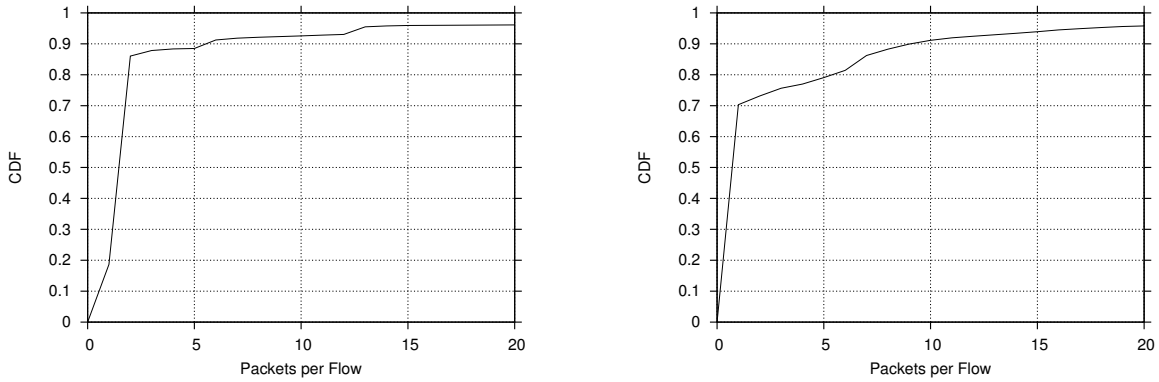
The complete list of statistics collected is too voluminous to list here; rather we give an overview of the types of statistics collected. Currently, all measurements made by NETI@home are passive. The NETI@home software collects end–to–end flow summary

statistics on the TCP, UDP, ICMP, and IGMP protocols, as well as their underlying protocols. In addition to these protocol–related measurements, the software also records information about the host on which it is running such as the operating system type and version as well as user–supplied geographical location information. For a more in–depth discussion of the statistics gathered see [5] and for more up–to–date information visit [6].

One key aspect of the NETI@home project is our commitment to user privacy. To aid in fulfilling this commitment, NETI@home users are able to select a privacy level that determines what types of data are gathered, and what is not reported. There are currently three privacy settings. Medium, the default setting, records only the network portion of the local, remote, and multicast IP addresses, as determined by the netmask. This allows for many interesting macroscopic studies of the Internet while not compromising the identity of the end–user. The high privacy setting does not record any IP addresses and the low privacy setting records the full local, remote, and multicast IP addresses observed. In addition to the privacy settings, each time a report is sent to the NETI@home server, an identical local copy is retained, so that users can view these contents and verify the operation of the software. Finally, the open–source nature of the software allows anyone to verify the functionality of the software.

Once NETI@home analyzes a specified number of flows or a specified amount of time has passed, the data is compressed using the `zlib` compression library[50]. NETI@home clients then report this data via TCP to a DNS name that resolves to a server that resides at the Georgia Institute of Technology. This server receives and collects all user reports. To ensure scalability, we plan to implement round–robin DNS for the server to allow multiple servers to simultaneously collect data.

For a project such as NETI@home to be useful there must be a large userbase. To this end, we have provided some incentive for participation as well as pursued various avenues of publicity. To encourage users to run the NETI@home client software we included a program called NETIMap. When run in conjunction with the NETI@home client software,



(a) Honeynet (b) NETI@home

Figure 1. CDF of the Number of Packets per TCP Flow

NETIMap plots the location of the remote host on a global map using CAIDA's NetGeo database[51]. In an effort to gain attention, we have pursued other avenues of publicity in addition to research literature. Examples of such publicity include Wired magazine[52] and the popular Slashdot website[53, 54]. Also, our software is available on the popular SourceForge website[55].

Since its debut on January 6, 2004, the NETI@home project has received reports from approximately 3500 unique IP addresses from about 40 nations, as of January 31, 2006.

The wealth of data collected during this time has led to many interesting observations and analysis, although there are still many more avenues of investigation to pursue.

The distributed approach to the collection of network measurements was inspired by the SETI@home project [49], NETI@home's namesake. Although SETI@home does not deal with network measurements (it actually looks for signs of intelligent life in the universe), it was one of the first programs to rely on regular users of the Internet to perform a data related function and then report back to a central server. Since its introduction, SETI@home has become extremely popular. NETI@home hopes to capitalize on this popularity and novel technique for the collection of network measurements.

3.2 Network Security

[8] presents a flow based comparison of the traffic seen on the Georgia Tech Honeynet, representing malicious traffic, to the data collected by NETI@home, representing typical end–user traffic. We first graphed the cumulative distribution function (CDF) of the number of packets for all TCP flows for each dataset. The results are shown in Figure 1. First, observe the Honeynet curve. One can see two distinct inflection points for packet counts equal to one and two. TCP flows which consist of just one packet most likely contain one SYN packet. It is possible to have a single packet flow that is not a SYN packet. For instance, a RST or SYN/ACK packet could be received from a host that received a spoofed connection attempt. Upon further investigation, we did not observe many flows of this nature.

TCP flows which consist of two packets most likely consist of one SYN and one RST packet or one SYN and one SYN/ACK packet with no final ACK to complete the three–way handshake. Again, there are other combinations of TCP flows consisting of just two packets, but we have not observed many of these combinations. Any TCP flow consisting of two or less packets is a failed connection. On a honeynet, we consider these failed connections to be malicious probes. Therefore on our honeynet dataset about 87% of all TCP flows can be considered to be probes.

We can contrast the NETI@home CDF with the honeynet CDF and see that about 73% of all TCP flows can be considered failed connections. In the NETI@home dataset, not all of these failed connections are necessarily malicious probe packets as they may be legitimately failed connections. However, it is interesting to note that in terms of number of packets per flow the majority of observed TCP flows for end–users are either probes or failed connections.

To better understand what ports and services malicious flows are targeting, we have generated a TCP Port Histogram over time for both the honeynet dataset as seen in Figure 2 and the NETI@home dataset as seen in Figure 3. Each row of points represents one day.

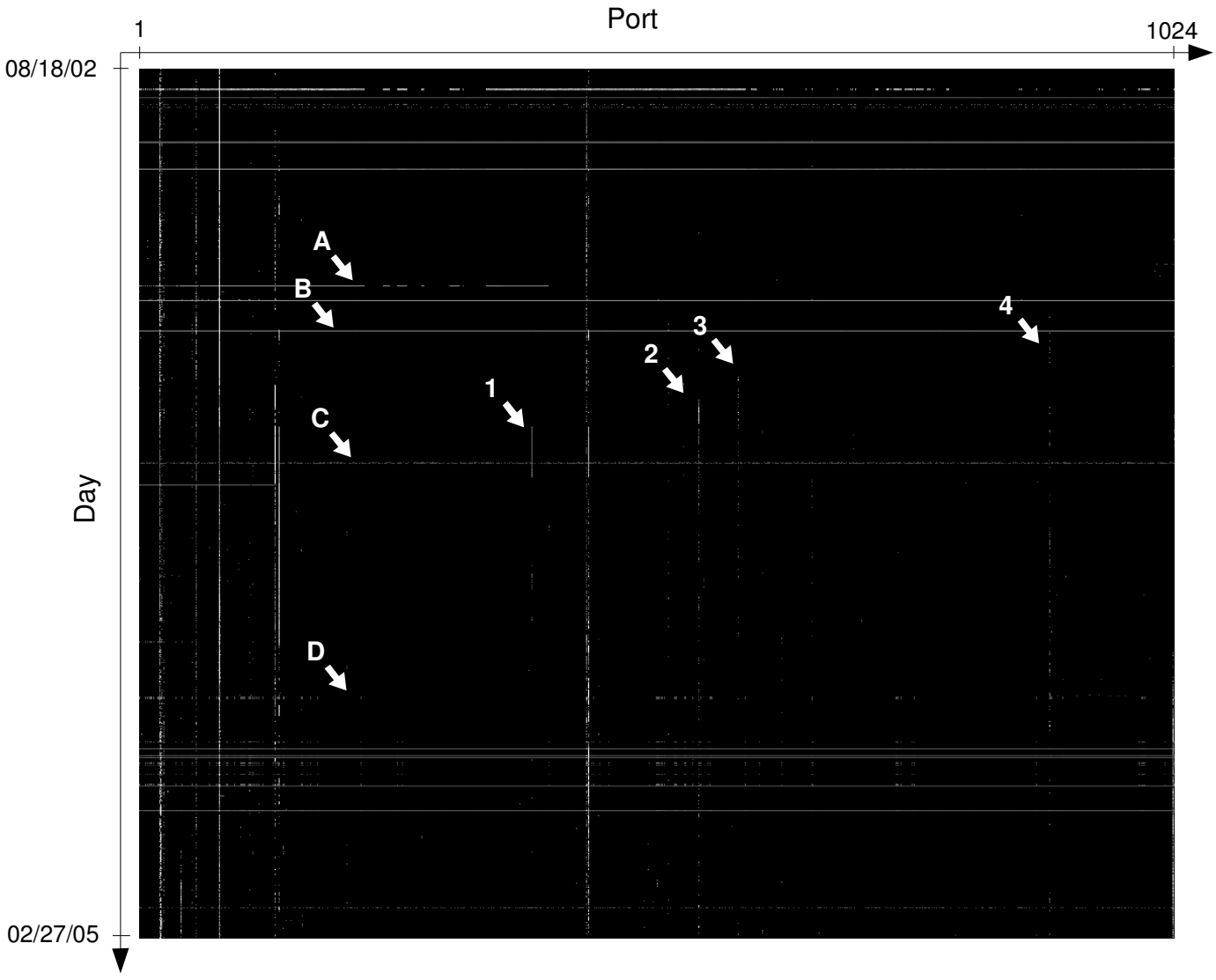


Figure 2. Honeynet TCP Port Histogram

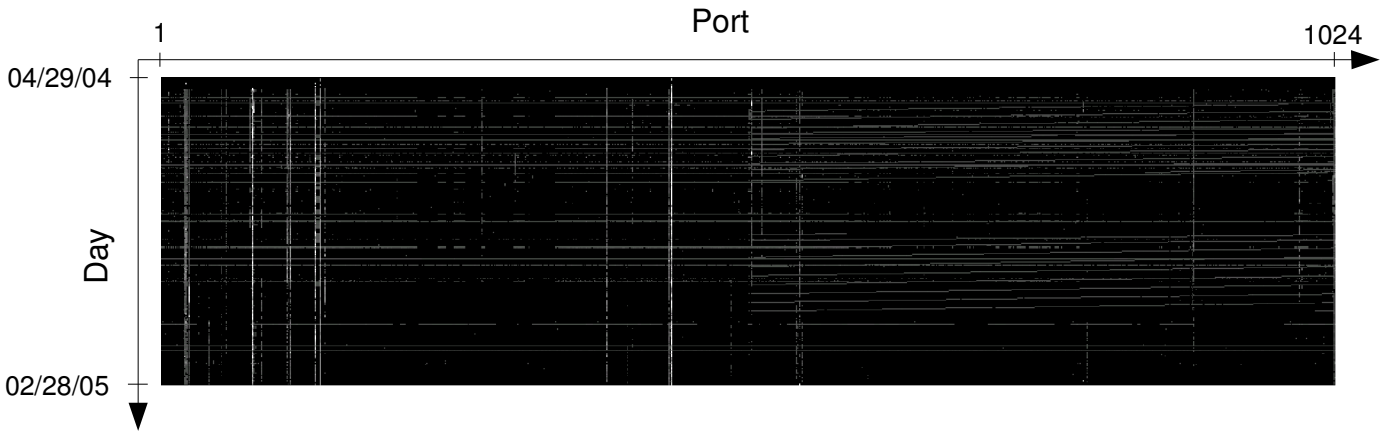


Figure 3. NETI@home TCP Port Histogram

The width of the rows span the local TCP ports from 0 to 1024, which are the well known ports[56]. The following formula was used to create the graphs, where i is the intensity value for a given point in a given row:

$$i = \begin{cases} 0 & \text{if } c = 0 \\ 0.75 \cdot \left(\frac{c}{c_{\max}}\right)^{0.45} + 0.25 & \text{otherwise} \end{cases} \quad (1)$$

The maximum number of packets destined to a certain port on one day (i.e. one row in the figure) is denoted $c_{\max}$. A port with a packet count of c is then visualized with intensity i according to the above formula. If c is zero, the intensity is also set to zero (black). Otherwise, the intensity is chosen to be a value between 25% gray ($i = 0.25$) to white ($i = 1.0$, for the port where $c = c_{\max}$). The exponent is used to boost dark pixels to make them more visible. We choose to represent no activity with dark regions because it provides better contrast for the faint areas of activity.

There are a number of observations to be made from these graphs. Two important characteristics of the figures to observe are the horizontal lines and the vertical lines. First, the horizontal lines represent port scans. Port scans are often malicious in nature as an attacker will generally use a port scan against a target in order to determine that target's weaknesses. In the honeynet data, a number of port scans can be seen over time, but the

NETI@home dataset shows a significantly denser number of port scans seen over time. This appears to be intuitive as there are an order of magnitude more NETI@home users, which are distributed across the Internet both topologically and geographically, than there are honeypots in our dataset. Some factors that would decrease the number of port scans seen by NETI@home end–users include firewalls, NATs, or other similar configurations. Even with these factors, some NETI@home users are seeing similar port scans as seen on our honeynet.

Another interesting observation is that there are a number of different types of scans seen. At least four different port scans are easily distinguished visually in the honeynet data as denoted by the letters $A - D$, and similar scans are observed in the NETI@home data. The most naive port scan will scan all ports (B). The more sophisticated port scans will skip ports that are of little interest (A, C, and D). There are a number of widely available port scanning tools, which offer various options for the scanning algorithm [57, 58].

One interesting difference seen in the horizontal lines in the NETI@home dataset are the stair step lines from approximately port 512 through 1024. Since the user that reported these flows was within the Georgia Tech network and used a low privacy level, we were able to determine what caused the stair step lines. An administrative machine within the Georgia Tech network was scanning ports 512 through 1024 over the course of several days. The algorithm consists of dividing the ports into a number of ranges and scanning one range each day. The source of the scanning was a machine used to help secure the network and so was altruistic. Therefore, we do not consider these scans to be malicious in nature.

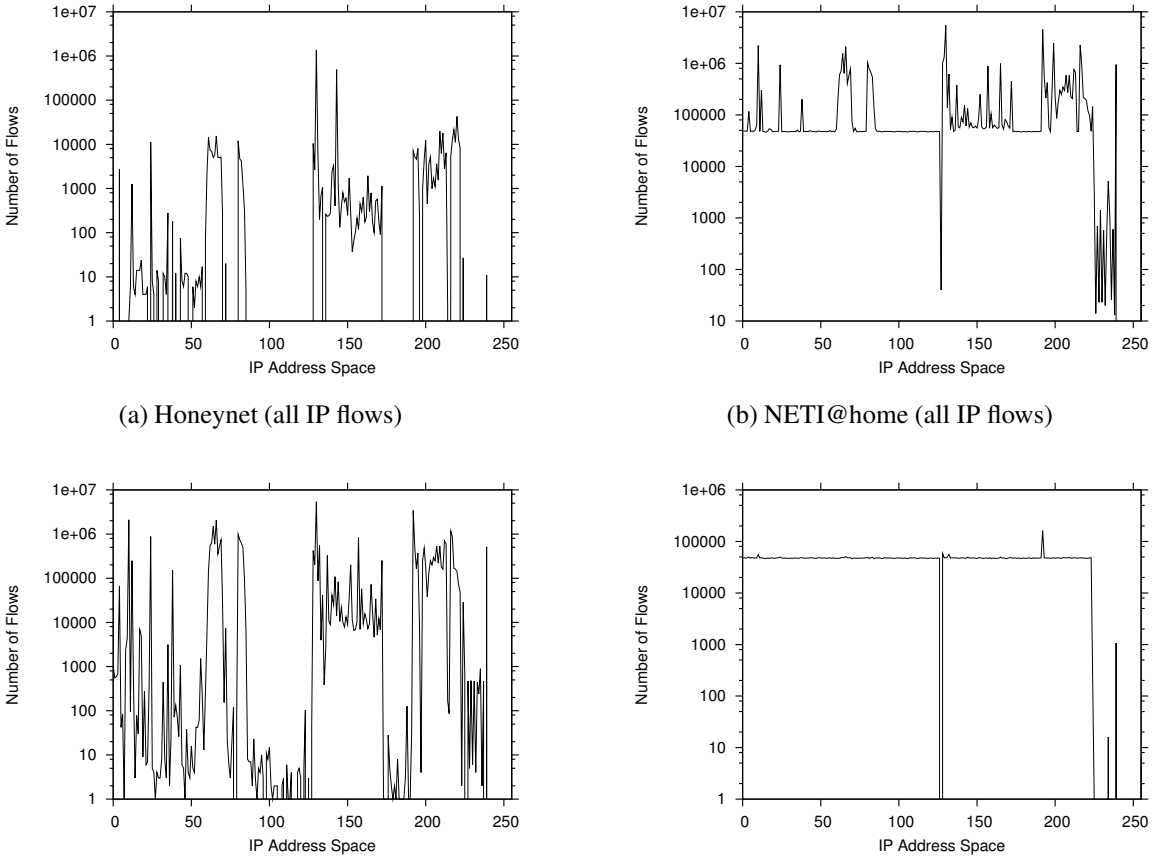
The second interesting aspect to observe in these graphs are the vertical lines. The vertical lines represent ports that have continual traffic over large time scales. Looking at the honeynet graph from left to right, the most prominent TCP ports with continual traffic are 22 (ssh), 80 (www), 135 (Microsoft Windows Service), 139 (Microsoft Windows Service), and 445 (Microsoft Windows Service). Most of these ports have been a target of

one or more worms in the past in addition to legitimate traffic.

There are a number of other vertical lines that are not as prominent in the honeynet dataset as seen in Figure 2. The vertical line denoted by '1' is LDAP traffic and was only seen for a short period of time. The line denoted by '2' represents traffic seen from the real time service protocol worm. The traffic at '2' is particularly interesting in the honeynet dataset. One can notice a bright burst of traffic starting on the worm release date that continues with intensity over the course of the next several days. After a number of days, the worm traffic slowly fades out as the infected machines are repaired. However, trailing effects of the worm can be seen from the point of release until the end of the dataset, which is over the course of more than a year. Therefore, we see lingering worm traffic exists on the Internet for long periods of time after the initial release date.

The line denoted by '3' represents traffic seen from the blaster worm as seen in Figure 2. This line also continues on for a long period of time, although its characteristics are not as distinguishable as the real time service protocol worm. In the honeynet data, it is not clear why traffic is seen at the line denoted by '4' at port 901. This may be traffic targeting an old Trojan port, RealSecure's management port, or Samba/SWAT on RedHat Linux based boxes. It is interesting to note that these trends seen in the honeynet data are repeated in the NETI@home data in addition to the legitimate traffic as seen in Figure 3. Although, it is difficult to distinguish between legitimate traffic and worm traffic in the NETI@home dataset.

The graphs in Figure 4 show where the traffic is coming from or going to within the entire IP address space. The IP address is divided into 256 buckets based on the first byte of the IP address. Figure 4(a) shows the honeynet graph. It is clear that certain portions of the address space have seen zero activity on the honeynet. These portions correspond with unallocated addresses as listed in the whois database. Given that there are no flows from most of these spaces to the honeynet, we conclude that there are not many spoofed IP packets coming from unallocated IPs to our honeynet. Further, either the number of



(a) Honeynet (all IP flows)

(b) NETI@home (all IP flows)

(c) NETI@home (no TCP port 445 flows)

(d) NETI@home (only TCP port 445 flows)

Figure 4. IP Address Space Distribution by Number of Flows

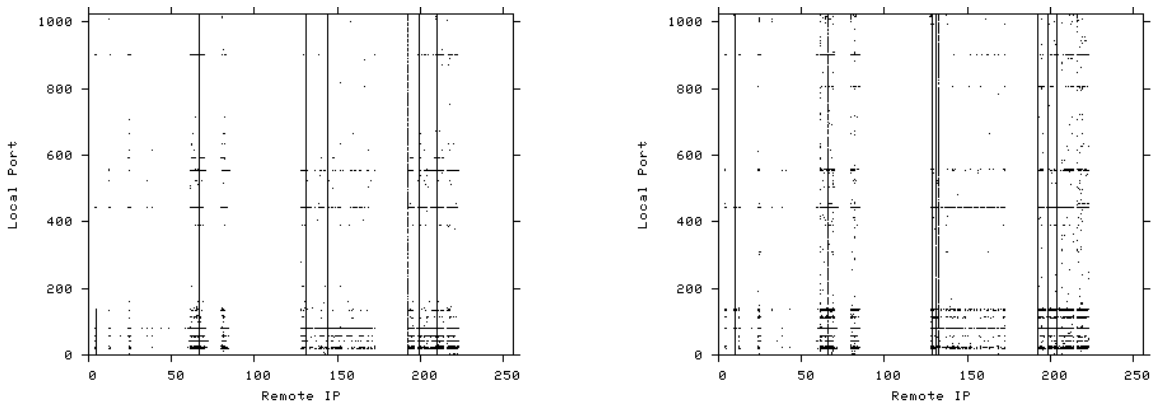
packets with spoofed IP addresses coming to our honeynet is low or they are intelligently designed.

The NETI@home dataset has an additional baseline of traffic seen across most of the address range as seen in Figure 4(b). Further investigation found that this baseline is caused by one or more NETI@home users sending out a large number of TCP flows to TCP port 445 over a short period of time. We are unsure how many users were reporting these results due to privacy settings. Figure 4(d) shows the number of flows to TCP port 445 versus the IP address space. There is clearly a horizontal line across the majority of the IP address space, which suggests that the NETI@home user or users were randomly scanning the IP address space on TCP port 445. The nature of this scanning may have be malicious in nature. For example, the user may have been infected with a worm as there have been worms that target TCP port 445. However, we cannot conclude for certain that the traffic was malicious in nature.

In Figure 4(d), there is a small increase in traffic at bucket number 10. This is probably due to local 445 traffic on private 10.0.0.0/8 networks. Similarly, there is an increase in traffic at bucket number 192. This increase would be due to local 445 traffic on private 192.168.0.0/16 networks. The sharp drop in traffic at bucket number 127 is due to the fact that the 127.0.0.0/8 network is the dedicated localhost network. Finally, the upper ranges of the IP address space did not see any scans. These ranges contain multicast, experimental, and other types of allocations.

To better compare the NETI@home data with the honeynet data, we graphed the NETI@home dataset filtering out traffic to TCP port 445 as seen in Figure 4(c). Comparing Figures 4(a) and 4(c), one can notice a striking similarity between the NETI@home data and the honeynet data. Some differences in the NETI@home data include traffic to the multicast range and some traffic in the unallocated ranges. However, visually the two graphs have notably similar shapes.

Based on our observations of the IP traffic seen relative to IP address space, we note



(a) Honeynet (b) NETI@home

Figure 5. Remote IP Address and Contacted Local TCP Port

a possible algorithm for detecting suspicious machines. In previous work, it was shown that a honeynet can be used to find compromised machines on large enterprise networks by marking any machine on the enterprise that attempts to connect to the honeynet as suspicious [59]. An extension that we draw from these graphs is that any machine attempting to connect to an unallocated IP address should be considered suspicious and may be compromised.

A graph of the remote IP versus local port for both datasets can be seen in Figure 5. Again, we only plot the well known TCP ports. In these graphs, one can see that remote IPs that appear in the flows are spread across the allocated IP spectrum, and again there is little traffic in the unallocated ranges, even in the NETI@home data. Based on these graphs, we observe that scans come from across the entire allocated IP address space.

3.3 User Behavior

The work presented in [9] presents our attempt to analyze network–independent user behavior using the NETI@home dataset. In this paper, we developed network–independent traffic models for network users.

Several characteristics of TCP and UDP flows were chosen to reflect network–independent

behavior and to wholly represent network client behavior.

Two categories of models were created in this study. The first is specific to a TCP or UDP port, that is we create a model of client behavior for a given TCP or UDP port. We use the model created for TCP port 80, the most common port used by World Wide Web servers, as an example. The second category of model created is an aggregate of all port–specific models. This model can be likened to a TCP or UDP client model. Such a model may prove useful for studies that are more generic and are not attempting to study a particular type of network traffic. All of these models incorporate empirical distributions directly interpreted from the NETI@home dataset.

The dataset used in this study consists of NETI@home data collected over a one year period from October 1, 2004 to September 30, 2005. This dataset includes over 36 million TCP flows and 93 million UDP flows, which form the basis of this work, as well as various other flow types and information about their corresponding hosts. Although an exact calculation is not possible due to privacy settings and dynamically assigned IP addresses, we estimate that this data was collected by approximately 1700 users. These users represent a heterogeneous sampling of Internet users running some 8 different operating systems and reporting from approximately 28 nations and 43 US ZIP Codes.

The first two aspects we model are empirical distributions of *bytes sent* and *bytes received*. These values are based only on the payload of the packets and thus do not represent the sizes of the TCP or UDP headers and their underlying headers or TCP's flow control and congestion control algorithms, merely transferred application information. This allows our models to be used in simulations where variations of TCP or UDP are employed.

The next aspect modeled is *user think time*. User think time is the term we use for the amount of time a client waits before initiating another flow. For this aspect, we developed two empirical distributions. One distribution describes the user think time when consecutively accessing a specific destination and the other describes the user think time when contacting a new destination.

Another aspect modeled is *consecutive contacts*. Consecutive contacts is the term we use for the probability that a client will choose to initiate another flow with the last destination contacted, or the client will choose to initiate a flow with a new destination. For this aspect, we developed a single empirical distribution.

Finally, the last aspect modeled is *contact selection*. Contact selection is the term we use for the frequency distribution of contacting specific destinations. This distribution can be thought of as modeling the popularity of a destination. For this aspect, we developed a single empirical distribution.

One other aspect that we believe to be worth modeling is related to *idle time*. For applications such as World Wide Web transfers, this aspect has little meaning, as web pages are simply requested and served. However, for interactive applications such as SSH or telnet, there are periods of time, *during* the flow, when there is no data transferred. However, using the NETI@home data, it is difficult to differentiate between network–dependent flow time and network–independent flow time. We are aware of work [38, 39] that attempts to capture this behavior and are considering implementing a similar technique into the NETI@home client software so that future models can incorporate this aspect of user behavior.

From the analysis of the NETI@home dataset described previously, we were able to generate a set of empirical distributions for each component of our models.

3.3.1 Bytes sent

The amount of bytes sent varies dependent on the port modeled. However, upon investigation of each modeled port, our findings seem intuitive.

Figure 6 depicts the cumulative distribution function of bytes sent for TCP port 80. Compared with previous studies [30], these results contain many more flows with zero bytes sent. However, upon investigation it does not appear that these results are due to a single NETI@home user or are anomalous. This difference in results is most likely due to the fact that [30] was based on data collected from a campus network, whereas NETI@home data contains users with less reliable network connections. The zero bytes

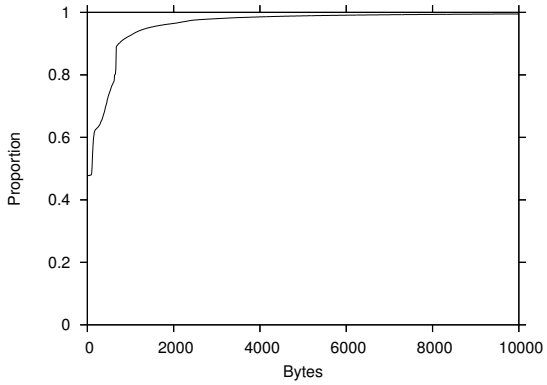


Figure 6. CDF of bytes sent for TCP port 80

sent flows typically represent flows in which the connection failed during the TCP three–way handshake. Although these flows do not generate much network traffic (usually no more than three packets), they are significant in terms of numbers of flows and most likely influence a user's behavior.

As can be seen in the figure, approximately 40 percent of flows to TCP port 80 send little or no data. There are several possible causes for the large number of flows sending little or no data. First, many of these flows are failed connection attempts. Many NETI@home users are utilizing less reliable network connections such as dial–up or wireless. Also, some of these flows may be to blocked sites. Many browsers and third–party software block advertisements and some organizations restrict the viewing of certain websites. Finally, a handful of NETI@home users periodically scan hosts on the Internet[8]. Considering that these users know that their network connections are monitored, it is unlikely that this scanning is intentional and may be the result of a virus or worm. While these results could be considered anomalous, we believe that this does indeed represent typical end–user behavior as seen on the Internet. Almost all remaining flows send no more than 10 KB of data to the server.

3.3.2 Bytes received

The amount of bytes received by the client is also dependent on the port modeled. Figure 7 depicts the cumulative distribution function of bytes received for TCP port 80. Compared

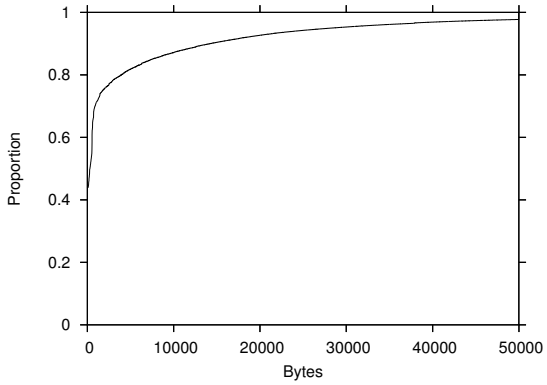


Figure 7. CDF of bytes received for TCP port 80

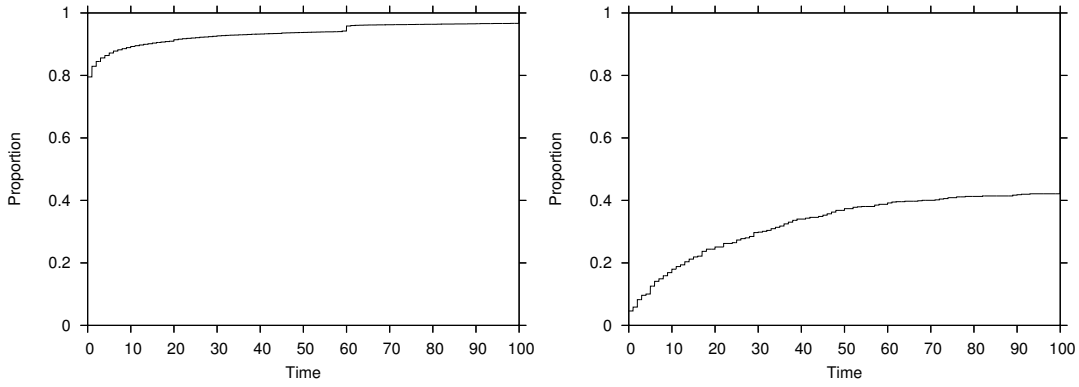
with [30], we also find that there are many more flows with zero bytes received. As with our findings for bytes sent, this is most likely due to failed connection attempts.

The distribution for bytes received has a much longer tail than that for the bytes sent. Approximately 40 percent of flows with a remote TCP port of 80 receive little or no data. However, more than 10 percent of these flows receive greater than 10KB of data.

3.3.3 User think time

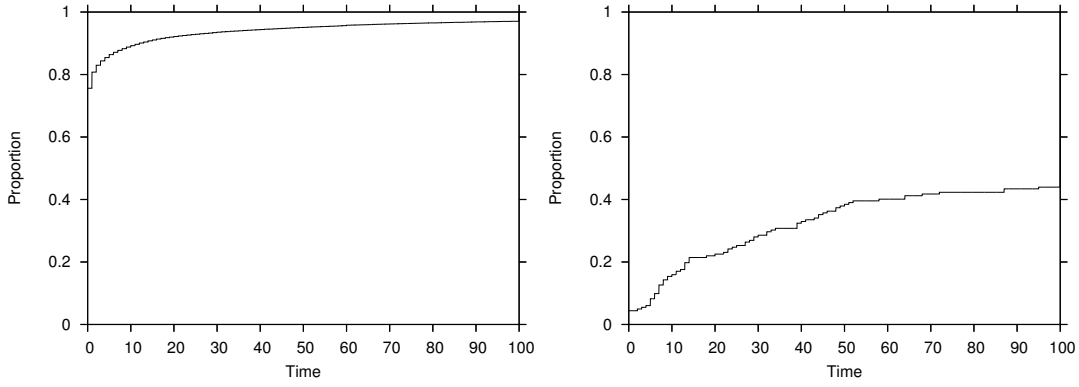
The cumulative distribution function for user think time to the same destination is given in Figure 8 and to differing destinations is given in Figure 9 for TCP ports 23 and 80. These findings show a tendency towards shorter user think times than was found in [30] for TCP Port 80. We can think of several reasons for this shortened user think time. First, the World Wide Web has become much more popular since the time of [30]'s publication. Also, it is likely that NETI@home captures data from users who are active more often than it does for inactive users as many users would simply turn off their machines while not using them, thus disabling NETI@home's monitoring. This would artificially inflate our numbers to show users that appear to be more active and is a source of bias.

We chose to model the user think time to the same destination separately from the user think time to a different destination. Figures 8(a) and 9(a) appear to be similar however. We believe that it is still appropriate to model these think times separately as these distributions can differ greatly for other TCP or UDP ports as is shown in Figures 8(b) and 9(b). These



(a) TCP port 80 (b) TCP port 23

Figure 8. CDF of User Think Time to Same IPs



(a) TCP port 80 (b) TCP port 23

Figure 9. CDF of User Think Time to Differing IPs

figures show the distributions for think times for TCP Port 23, the port commonly used for telnet.

For connections to TCP port 80, the majority of user think times tends to be less than 1 second. However, for connections to TCP port 23 (telnet), the user think times have a much heavier tail, with only approximately 40 percent of flows having think times less than 100 seconds.

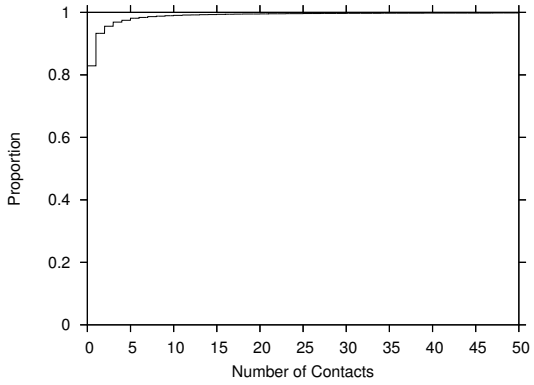


Figure 10. CDF of number of times an IP is contacted consecutively for TCP port 80

3.3.4 Consecutive contacts

In Figure 10, we present the cumulative distribution function for consecutive contacts for TCP port 80. These results also show a tendency towards a lower number of consecutive contacts than was found in [30]. However, this is intuitive considering the number of "failed" connection attempts observed previously.

Approximately 80 percent of the flows to TCP port 80 are not consecutive, that is the destination is contacted only once in a row. Further, over 99 percent of visits to a specific destination on TCP port 80 lasted for 10 or less flows in a row. Therefore, it appears that users tend to switch web destinations fairly often as was noted in [30].

3.3.5 Contact selection

Unlike [30], which used a Zipf distribution, we were able to construct a cumulative distribution function for contact selection due to the wide sampling offered by the NETI@home dataset. Figure 11 presents this CDF for TCP port 80. One possible source of inaccuracy for this aspect is the fact that we are unable to determine if a specific destination uses multiple IP addresses, thus reducing the frequency of selection a given contact may appear to have.

As can be seen in the figure, for TCP port 80 servers the distribution of the overall number of visits by NETI@home users is quite varied and has a heavy tail. Many servers are only visited a handful of times, however many other servers tend to be contacted quite

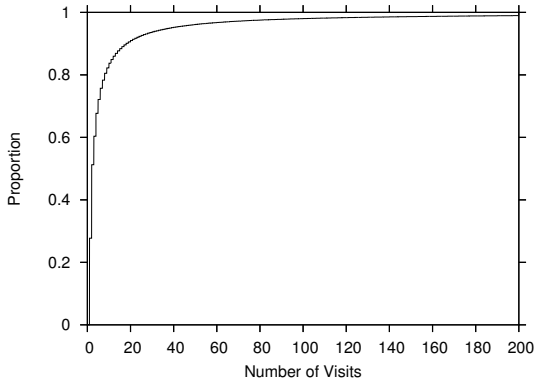


Figure 11. CDF of relative frequency of server visits for TCP port 80 over a one year period

often, with some servers receiving millions of visits over the year studied.

3.3.6 Simulation results

To judge the usefulness of our models, we have incorporated the above derived TCP traffic models into the GTNetS environment[60]. The GTNetS environment already has some HTTP traffic models as described in [30]. We incorporated the models derived from the analysis of the NETI@home datasets into GTNetS. We consider this approach to be a better one for traffic generation in network simulations, because NETI@home datasets are more current and continue to be so [5]. An analysis program generates these models automatically from the NETI@home datasets. The traffic distribution models can then be easily used by the application layer models which drive a network simulation. In our simulation experiments, we have concentrated on the World Wide Web traffic and the HTTP models. Our implementation samples the empirical distributions to determine the particular values used at a given time. This seems a logical choice since any single distribution doesn't seem to fit the complete dataset verifiably. We model the behavior of a web browser in GTNetS which sends a HTTP request to a designated webserver asking it to send a certain length of data that constitutes the response. When the simulation starts, the browser application chooses a server randomly from a list of target servers. It then chooses a *response size* that it wants to obtain from the webserver from the CDF that describes the *received bytes*. The size of the HTTP request packet is chosen from the *sent bytes* CDF plot. It may request

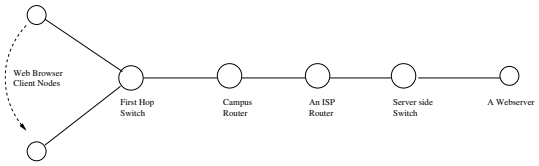


Figure 12. Network topology used for testing traffic models in simulation

one or more objects within the same TCP connection. Once the web browser application has received the appropriate response, it proceeds to select a different server or the same server for its next request and waits for an amount of time. This amount of time, which is obtained from the CDF that describes the *user think time*, depends on whether the same server is chosen or a different server is chosen.

The network topology for simulations is obtained from [41]. It consists of a large set of web browsers connected via a series of three routers to a webserver as shown in Figure 12. We have chosen this to be our baseline topology because we have earlier simulation experiments conducted using the models and datasets proposed in [41].

The simulation experiment is run using two HTTP traffic models. One of the traffic models is obtained from the datasets suggested in [30] and [41]. The other traffic model is one that is obtained from the NETI@home datasets. Intuitively, empirical traffic models should be more representative of a realistic dataset than statistical traffic generators, although the former cannot be subjected to extrapolations. All the measurements are the averages of three runs of a simulation at a given data point.

Table 1 shows the average and maximum response times for a given number of web clients when they request data from a webserver using the traffic models presented in this paper. Table 2 shows the average and maximum response times for the same number of web browsers when they request data from a webserver using the traffic models presented in [30].

It can be seen from the results in Table 1 and Table 2 that the maximum response time for the HTTP traffic model presented in [30] is substantially larger than the model that is derived from NETI@home dataset. A careful observation of the cumulative distribution

Table 1. Variation in Average and Maximum Response Times when using HTTP Traffic Model Presented in this Paper

Number of browsers	Average response time	Maximum response time
10	0.316738	0.738639
25	0.301151	0.740423
50	0.318433	0.738642
75	0.321075	0.743916
100	0.304644	0.745433
125	0.305372	0.751632
150	0.312204	0.839426

Table 2. Variation in Average and Maximum Response Times when using HTTP Traffic Model Presented in [30]

Number of browsers	Average response time	Maximum response time
10	0.461172	0.716738
25	0.339998	1.01388
50	0.344094	1.20155
75	0.375188	3.98217
100	0.380281	3.80786
125	0.332889	4.16023
150	0.405156	6.6588

functions of the two datasets shows that the NETI@home data has a larger proportion of flow sizes that are very small, most likely due to the inclusion of a large number of failed connections. This results in lower load on the webserver and consequently lower latencies. This is evident in the lower average and maximum response times as the traffic increases. On the other hand, the traffic model presented in [30] has a lesser number of flow sizes that are very small. This results in a larger load on the server and on the network as the number of web browsers increases. When the number of web browsers is fairly small, the difference is not appreciable because the flow size does not influence the network.

The code used for these simulations, as well as the empirical models, are available in the latest official distribution of the GTNetS environment.

3.4 Network Behavior

Since its debut on January 6, 2004, the NETI@home project has received reports from approximately 3500 unique IP addresses from about 40 nations, as of January 31, 2006.

The wealth of data collected during this time has led to many interesting observations and analysis, although there are still many more avenues of investigation to pursue. This section highlights some of the more interesting observations and analysis that have been made using the NETI@home dataset.

First, general observations have been made about the NETI@home user population and their use of the Internet. Figure 13 shows the distribution of operating systems run by NETI@home users. Figure 14 shows the proportion of users from different geographical locations, determined by both reverse DNS lookups on IP addresses, using the top level domain, as well as user–reported geographical location. Finally, Table 3 and Table 4 show some of the more popular TCP and UDP ports by flow and the applications most commonly associated with these ports. Table 5 and Table 6 show the popular ports in terms of bytes transferred. In the future, we hope to determine long–term trends of application popularity.

In this section, we focus on observations related to the networking performance of the

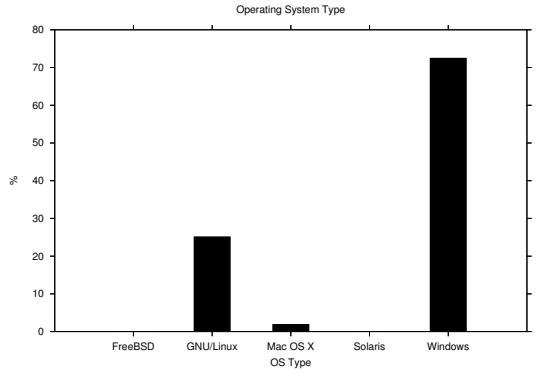


Figure 13. Percentage of Users by Operating System

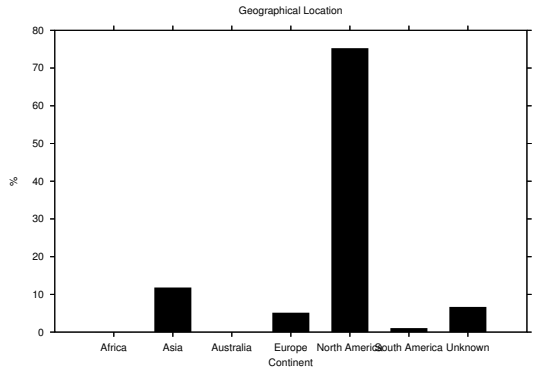


Figure 14. Percentage of Users by Geographical Region

Table 3. Popular TCP Ports (by Flows)

TCP Port Number	Common Use	Percentage of TCP Flows
80	HTTP (web)	45.00
445	Win2k+ Server Message Block	26.64
4662	edonkey, emule (P2P)	5.26
6881	bittorrent	3.93
443	HTTPS	1.60
6346	gnutella	1.28
110	POP3	1.17
5678	rrac	1.12
557	NETI@home	0.87
139	netbios and trojans	0.86

Table 4. Popular UDP Ports (by Flows)

UDP Port Number	Common Use	Percentage of UDP Flows
53	DNS	36.37
162	snmptrap	29.66
137	netbios	12.96
138	netbios	5.05
6881	bittorrent	4.89
161	snmp	4.12
4672	xmule, rfa	3.39
9646	*Unknown*	2.86
9313	*Unknown*	2.07
69	tftp	1.59

Table 5. Popular TCP Ports (by Bytes)

TCP Port Number	Common Use	Percentage of TCP Bytes
80	HTTP (web)	47.59
6881	bittorrent	14.16
4662	edonkey, emule (P2P)	10.36
3128	squid-http and trojans	1.56
443	HTTPS	1.48
139	netbios and trojans	1.04
22	SSH	0.98
10500	*Unknown*	0.92
8000	*Unknown*	0.86
8090	*Unknown*	0.70

Table 6. Popular UDP Ports (by Bytes)

UDP Port Number	Common Use	Percentage of UDP Bytes
1900	SSDP	22.09
162	snmptrap	19.44
53	DNS	16.07
138	netbios	7.75
137	netbios	5.95
6881	bittorrent	5.01
32770	Filenet NCH	3.39
1234	*Unknown*	3.39
67	Bootstrap Protocol Server	3.10
68	Bootstrap Protocol Client	3.10

NETI@home users.

3.4.1 TTL Analysis

For most flows, NETI@home records the minimum and maximum TTL values observed in both directions of a bidirectional flow. Using these values, and the assumption that most initial TTL values are either 255, 128, 64, or 32, we are able to infer hop count statistics for these flows. This section discusses the results of such an analysis.

There were a few notable anomalies found while analyzing the NETI@home data using this method. First, we found an unusually high number of UDP flows (77%) with a hop count of 105. Upon further investigation we found that all of these flows were communications between hosts on a local area network. This led us to believe that the hop counts for such flows should be very small and we found that several routers actually use an initial TTL value of 150. We then modified our analysis to account for this finding.

Another anomaly found in the NETI@home data was the existence of several flows (3% of TCP flows) that crossed the initial TTL boundaries. The vast majority of these flows, when calculating hop counts, showed that their actual hop counts were not varying, merely their initial TTL values. Upon further investigation, we found that 79% of the anomolous TCP flows were HTTP flows, with the remainder consisting of other high traffic applications such as HTTPS and POP. Selecting several of these websites and conducting a manual analysis with Ethereal[16] confirmed that this behavior is indeed occurring in individual flows. No immediate explanation was found however, as some flows exhibited varying initial TTLs between TCP connection establishment packets (SYN / ACK) and all other packets while other flows exhibited differing behavior on duplicate acknowledgment packets only. Further, some flows exhibited this differing behavior between HTTP data and all other packets and finally the remaining flows exhibited the behavior between seemingly randomly selected packets. One possible explanation for such behavior is that the user or an application is modifying the initial TTL value during the flow. To investigate this possibility we searched throughout the source code of the popular open–source Apache

Table 7. Initial TTL Values

Protocol	255	150	128	64	32
TCP	4.54%	0.07%	31.36%	63.88%	0.15%
UDP	1.33%	76.90%	17.97%	3.19%	0.61%
ICMP	64.89%	0.03%	13.77%	21.06%	0.25%
IGMP	0.01%	0.00%	0.00%	0.03%	99.96%

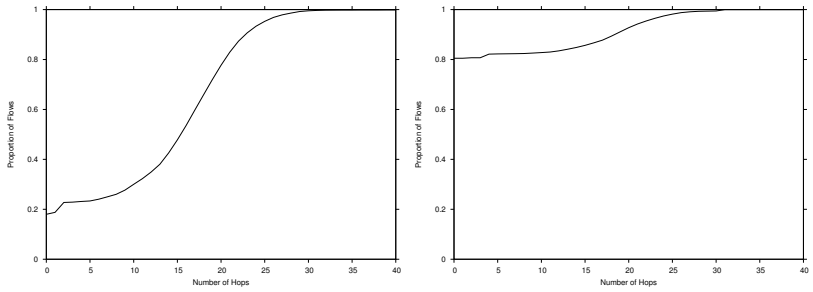
web server[61] and failed to find any code to modify the initial TTL. While having no confirmable explanation, we believe that such behavior may be the result of load–balancing by servers, in spite of the fact that it occurs within a single flow. Thus, one source of bias in our hop count study, in addition to our assumed initial TTL values, is the fact that we only record the maximum and minimum TTL value observed. In cases where the initial TTL value is varied, the maximum and minimum hop count may go unrecorded.

After identifying and analyzing these anomalies in our dataset, we calculated hop counts for all of the observed flows. We found that the average hop counts vary dependent upon the protocol, most likely due to the fact that some protocols are used more often in a local network setting. Table 7 summarizes the assumed initial TTL values and their frequency and Figure 15 plots the cumulative distribution functions of the average hop counts for TCP, UDP, and ICMP.

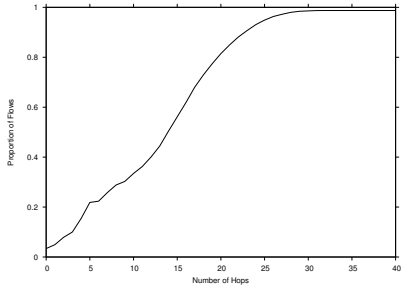
In addition to calculating the average hop counts observed, we analyzed the variation of the hop counts for the flows based on the maximum and minimum TTL values observed. Variations in the hop counts are most likely a result of routing changes over the flow's lifetime. Table 8 summarizes the average variations we observed for TCP, UDP, and ICMP flows as well as the percentages of these flows for which hop count changes occur.

3.4.2 Frequency and Use of Network Address Translation and Private IP Addresses

One of the greatest strengths of the NETI@home project is that all connections are observed from the viewpoint of the end–user. Such a vantage point gives us the unique ability to observe local network traffic as well as the use of network address translation (NAT), which



(a) TCP (b) UDP



(c) ICMP

Figure 15. CDF of Average Hop Counts

Table 8. Hop Count Variation

Protocol	Average Hop Count Variation	Flows with Variation
TCP	0.15	3.74%
UDP	0.00	0.03%
ICMP	0.02	0.82%

would otherwise be difficult to measure. Exercising this ability, we determine the number of NETI@home users that utilize NAT. Further, we observe the number of NETI@home users who utilize the private IP address space[62] for their local network connection.

When a NETI@home user participates in a network connection we are able to observe their local IP address, provided their privacy setting allows such monitoring. Further, when a NETI@home user reports their data to the NETI@home server, we are able to determine their "external" IP address. Should these IP addresses differ, we determine that NAT is being utilized. We find that this occurs for 81.15% of our users.

Investigating the local IP addresses further, we calculated the percentage of NETI@home users with local IP addresses in the private IP address space. We have found that 77.30% of NETI@home users utilize an IP address reserved for private use. Thus, the majority of NETI@home users are members of local area networks, although these local networks may consist of one machine.

For the majority of NETI@home users utilizing NAT, the IP address conversion is from an IP address in the private range to an IP address in the public range. However, we found that 17.01% of NETI@home users actually convert from one public IP address to another, in effect utilizing two public IP addresses. While some of these users may have been reassigned a new public IP address before reporting to the NETI@home server, there is evidence to suggest that this is not always the case. As there are a limited number of publicly available IP addresses, one role of NAT is to allow multiple machines to utilize a single public IP address. As is evidenced by these numbers, it appears that NAT is in fact helping to preserve public IP addresses for the time being. However, it also appears that NAT is being used when a public IP address has already been allocated to the offending host.

3.4.3 Adoption and Use of Selected Protocol Flags and Options

NETI@home records several statistics on various flags and options for TCP, UDP, ICMP, and IGMP as well as their underlying IP headers. In this section we discuss a selection of

Table 9. TCP Option Capability

Protocol Option	Neither Host Capable	One Host Capable	Both Hosts Capable
TCP SACK	60.28%	20.92%	18.80%
TCP Window Scaling	96.62%	3.03%	0.35%

these flags and options, namely TCP selective acknowledgment (SACK) capability, TCP window scaling capability, the specified TCP maximum segment size (MSS), the TCP urgent flag, the TCP push flag, and the IP don't fragment flag.

TCP SACK and TCP window scaling require capability on both sides of the network connection for functionality[63, 64, 65]. We found that TCP SACK capability is fairly common in the observed flows, with a significant amount of flows having only one end–host capable. On the other hand, we found that TCP window scaling capability is not very common, although this option has been described for many years and has become a rate limiting factor among TCP flows[66]. It is intuitive that the bandwidth–delay products of Internet connections will increase over time, thus increasing the need for the adoption of TCP window scaling. Table 9 summarizes these findings.

We also studied the MSS specified by the observed end–hosts of the TCP flows. We found that the average MSS specified was 467 and the median MSS was 1460, which corresponds to the Ethernet MTU. We also observed a minimum MSS of 24 and a maximum of 16856 bytes. Further, 63.87% of TCP end–hosts either did not explicitly specify a MSS, thus according to [67] they adopt the default MSS of 536, or their MSS declaration was unobserved.

Finally, a count of the number of TCP flows utilizing the push flag and the urgent flag as well as flows utilizing the don't fragment flag was made. We found that the push flag is actually quite common, appearing at least once in 62.59% of the observed TCP flows and in 20.75% of observed TCP packets. On the other hand, we found the urgent flag to be extremely rare, only appearing in less than 0.01% of TCP flows. The don't fragment flag

also appeared quite often, in 33.91% of observed flows. The high occurence of the don't fragment flag is most likely an attempt to avoid the performance penalties of fragmentation.

SECTION 4

PROPOSED RESEARCH

The object of the proposed research is to analyze passive end–to–end network performance measurements to obtain a better understanding of network activity. Specifically, there are three key areas that are to be investigated: network security, user behavior, and network behavior. Each of these three characteristics are vital to understanding the nature of the Internet.

To complete this investigation, the NETI@home dataset will be heavily analyzed, as will other datasets when necessary. The NETI@home infrastructure's unique end–user perspective will aide in all three of these areas and provide an insight which would be otherwise much more difficult, if not impossible.

4.1 Network Security

Utilizing the NETI@home dataset, we plan to analyze aspects related to network security. Effort will be made to distinguish the various types of security related activity seen and their frequency. By better understanding what a "typical" end–user observes, we will provide researchers with the data needed to combat and mitigate this malicious activity.

Particularly, we want to compare malicious Internet activity to "typical" end–user activity to understand the types and volumes of malicious activity seen by "typical" end–users and what can be done to mitigate this problem.

4.2 User Behavior

One of the strengths of NETI@home is its ability to observe network activity from the perspective of "typical" end–users. This perspective not only gives us insight into what the end–user sees, but how the end–user behaves. This perspective will be used to understand end–user behavior, in a network–independent fashion. It is difficult to conduct such studies without an infrastructure such as NETI@home in place. Other approaches that could be

used would be to measure from a midpoint in the network, from the server–side, or from a gateway such as those at the edges of campus networks. However, each of these approaches have problems that are addressed by NETI@home. From the midpoint of the network, it is difficult to be certain of one's sample size. Also, from this perspective end–to–end measurements are lost. From the server–side, many end–to–end measurements can be made. However, the server–side perspective depends on the popularity and audience of that particular server. Further, it is difficult to increase the sample size to many servers as many server administrators would be reluctant to give up such sensitive information. Finally, measurements made from campus gateways are frequently the most studied perspective. This perspective also has its drawbacks. For instance, campus network users are can not be considered "typical" end–users. Also, from the perspective of the gateway, information about local area network traffic is also lost. Thus, NETI@home is in the unique position to provide detailed analysis of end–user habits and behavior.

We plan to develop network–independent models of end–user traffic suitable for use in network simulation environments.

4.3 Network Behavior

Finally, we plan to study the behavior of end–host machines and protocols. As the Internet consists of a variety of machine types and operating systems, the NETI@home dataset, with its variety of users and operating systems, will be useful. Effort will be made to determine what effect the differences in machine configurations has on the overall interactions on the Internet. Further, we will study networking protocols to determine how well they perform on the Internet and how well they are implemented by each operating system. As each operating system tends to have its own custom network stack, there should be some differences in the way protocols behave from operating system to operating system. Also, several protocols provide options that may or may not be implemented and used by the various operating systems.

Further, we plan to analyze trends in network activity, particularly those related to geographical location and end–host operating system.

SECTION 5

WORK REMAINING TO BE DONE

The work to be accomplished consists of in–depth studies of all three previously mentioned areas of network measurements. Other goals that are to be met include research collaboration, trend analysis, and various other studies of interest.

5.1 Research Collaboration

It has been noted that the area of network measurements suffers from the unwillingness of researchers to share traces amongst themselves, usually due to privacy concerns. Such reluctance hampers this field as results are often not reproducible and conclusions, while sometimes questioned, often can not be authenticated. [68, 69] To address this need we plan to set up a collaboration environment around NETI@home.

First, the NETI@home dataset will be further anonymized to protect the privacy of its users. Once this anonymization is complete, we will publicly share the data so that researchers around the world can use the dataset as a basis for studies. We will also publicly release our analysis tools so that other researchers can use them and improve them. We would like to provide a framework under which the data can be analyzed. This will allow individual experiments and analysis to be small pieces of code that can then be uploaded onto our servers so that other researchers can look over the methods and results, fulfilling the need for reproducibility and authentication.

5.2 Trend Analysis

Another remaining task is the identification and analysis of various trends in the NETI@home dataset. For instance, we will investigate trends related to the geographical locations of NETI@home users. Figure 14 shows the distribution of NETI@home users by continent. Utilizing this variety, we suspect that trends will appear dependent on the user's geographical location such as the locality of contacted servers and the times of activity.

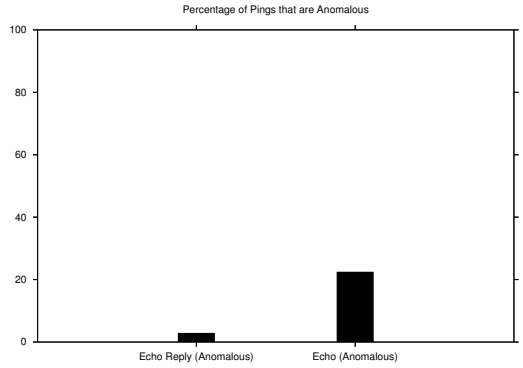


Figure 16. Anomalous Echo by Percentage

One other aspect we will analyze are trends related to end–host operating systems. Figure 13 shows the diversity of operating systems on which NETI@home is running and gathering data. Different operating systems will most likely have dissimilar implementations of networking protocols which may affect the performance of these systems.

5.3 Other Directions

Other directions of study include the DNS infrastructure and suspected botnet behavior. The DNS infrastructure has been heavily studied, however very little research has been performed from the perspective of the end–user. Upon initial investigation, we noticed that end–hosts frequently (approximately 6% of TCP DNS flows) contact the root DNS servers directly. This unusual and potentially dangerous behavior should be investigated further as well as its causes and trends.

We have previously noted[8] that several NETI@home users appear to be infected with malware. Further, we have noticed suspicious behavior while studying ICMP traffic. While not confirmed just yet, we believe that this behavior may be related to a new type of botnet that uses ICMP as a covert channel of communication. Specifically, we have noticed that several ICMP packets contain invalid type and/or code values. Figure 16 shows the percentage of observed Echo and Echo Reply ICMP packets (used for `ping`[1]) which contain anomalous code fields.

SECTION 6

FACILITIES AND EQUIPMENT NEEDED

To conduct the proposed research, one x86 development machine and the NETI@home infrastructure, including the NETI@home server, are needed. All software and development tools are open source and freely available. The x86 development machine is available and ready. The NETI@home infrastructure is available and operational.

REFERENCES

[1] M. Muuss, "ping." Software on-line: http://ftp.arl.mil/ mike/ping.html, 1983. Ballistic Research Lab.

[2] V. Jacobson, C. Leres, and S. McCanne, "tcpdump." Software on-line: `http://www.tcpdump.org`, June 1989. Lawrence Berkeley Laboratory.

[3] P. Barford and J. Sommers, "A comparison of active and passive methods for measuring packet loss," October 2002. University of Wisconsin Technical Report.

[4] K. Mochalski and K. Irmscher, "On the use of passive network measurements for modeling the Internet," in *KiVS*, 2003.

[5] C. R. Simpson, Jr. and G. F. Riley, "NETI@home: A distributed approach to collecting end-to-end network performance measurements," in *PAM2004 - A workshop on Passive and Active Measurements*, April 2004.

[6] C. R. Simpson, Jr., "NETI@home." Software on-line: `http://neti.gatech.edu`, 2003. Georgia Institute of Technology.

[7] C. R. Simpson, Jr., "A distributed approach to passively gathering end–to–end network performance measurements," Master's thesis, Georgia Institute of Technology, May 2004.

[8] J. B. Grizzard, C. R. Simpson, Jr., S. Krasser, H. L. Owen, and G. F. Riley, "Flow based observations from NETI@home and honeynet data," in *Proceedings from the sixth IEEE Systems, Man and Cybernetics Information Assurance Workshop*, pp. 244–251, June 2005.

[9] C. R. Simpson, Jr., D. Reddy, and G. F. Riley, "Empirical models of TCP and UDP end–user network traffic from NETI@home data analysis," in *20th ACM/IEEE/SCS Workshop on Principles of Advanced and Distributed Simulation (PADS 2006) (to appear)*, May 2006.

[10] V. Paxson, J. Mahdavi, A. Adams, and M. Mathis, "An architecture for large-scale Internet measurement," *IEEE Communications*, vol. 36, pp. 48–54, August 1998.

[11] J. Corral, G. Texier, and L. Toutain, "End-to-end active measurement architecture in IP networks (SATURNE)," in *PAM2003 - A workshop on Passive and Active Measurements*, April 2003.

[12] K. Thompson, G. J. Miller, and R. Wilder, "Wide-area Internet traffic patterns and characteristics (extended version)," *IEEE Network*, 1997.

[13] C. Fraleigh, C. Diot, B. Lyles, S. Moon, P. Owezarski, D. Papagiannaki, and F. Tobagi, "Design and deployment of a passive monitoring infrastructure," *Lecture Notes in Computer Science*, vol. 2170, 2001.

[14] K. Keys, D. Moore, R. Koga, E. Lagache, M. Tesch, and k claffy, "The architecture of CoralReef: An Internet traffic monitoring software suite," in *PAM2001 - A workshop on Passive and Active Measurements*, CAIDA, April 2001. http://www.caida.org/tools/measurement/coralreef/.

[15] V. Jacobson, "traceroute." Software on-line: ftp://ftp.ee.lbl.gov, 1989. Lawrence Berkeley Laboratory.

[16] G. Combs and et al., "Ethereal: - a network protocol analyzer." Software on-line: http://www.ethereal.com, 2004.

[17] M. Murray and kc claffy, "Measuring the immeasurable: Global Internet measurement infrastructure," in *PAM2001 - A workshop on Passive and Active Measurements*, April 2001.

[18] V. Jacobson, C. Leres, and S. McCane, "libpcap." Software on-line: http://www.tcpdump.org, 1989. Lawrence Berkeley Laboratory.

[19] L. Spitzner, "Know your enemy: Honeynets." http://www.honeynet.org/papers/honeynet/. Honeynet Project.

[20] N. Provos, "A virtual honeypot framework," in *13th USENIX Security Symposium*, August 2004.

[21] D. Moore, "Network telescopes: Observing small or distant security events." http://www.caida.org/outreach/presentations/2002/usenix_sec/, August 2002. Invited Presentation at the 11th USENIX Security Symposium (SEC 02).

[22] "The sans internet storm center." http://isc.sans.org.

[23] "DIMES – distributed internet measurements and simulations." http://www.netdimes.org/, April 2006.

[24] "traceroute@home." http://www.tracerouteathome.net/, April 2006.

[25] D. M. Kienzle and M. C. Elder, "Recent worms: a survey and trends," in *WORM'03: Proceedings of the 2003 ACM workshop on Rapid Malcode*, pp. 1–10, ACM Press, 2003.

[26] N. Weaver, V. Paxson, S. Staniford, and R. Cunningham, "A taxonomy of computer worms," in *WORM'03: Proceedings of the 2003 ACM workshop on Rapid Malcode*, pp. 11–18, ACM Press, 2003.

[27] C. Shannon and D. Moore, "The spread of the witty worm," *Security & Privacy Magazine*, vol. 2, no. 4, pp. 46–50, 2004.

[28] D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, and N. Weaver, "Inside the slammer worm," *Security & Privacy Magazine*, vol. 1, no. 4, pp. 33–39, 2003.

[29] "Georgia Tech honeynet research project." `http://www.ece.gatech.edu/research/labs/nsa/honeynet.shtml`, March 2005.

[30] B. A. Mah, "An empirical model of HTTP network traffic," in *IEEE INFOCOMM*, April 1997.

[31] F. D. Smith, F. Hernandez-Campos, K. Jeffay, and D. Ott, "What TCP/IP protocol headers can tell us about the web," in *ACM SIGMETRICS*, pp. 245–256, 2001.

[32] S. Floyd and V. Paxson, "Difficulties in simulating the internet," *IEEE/ACM Transactions on Networking*, vol. 9, pp. 392–403, August 2001.

[33] P. Barford and M. Crovella, "Generating representative web workloads for network and server performance evaluation," in *ACM SIGMETRICS*, 1998.

[34] J. Cao, W. S. Cleveland, Y. Gao, K. Jeffay, F. D. Smith, and M. C. Weigle, "Stochastic models for generating synthetic HTTP source traffic," in *IEEE INFOCOMM*, March 2004.

[35] Y.-C. Cheng, U. Holzle, N. Cardwell, S. Savage, and G. M. Voelker, "Monkey see, monkey do: A tool for TCP tracing and replaying," in *Proceedings of USENIX Technical Conference*, June 2004.

[36] H.-K. Choi and J. O. Limb, "A behavioral model of web traffic," in *ICNP*, 1999.

[37] C. Barakat, P. Thiran, G. Iannaccone, C. Diot, and P. Owezarski, "Modeling internet backbone traffic at the flow level," *IEEE Transactions on Signal Processing – Special Issue on Networking*, vol. 51, August 2003.

[38] F. Hernandez-Campos, A. B. Nobel, F. D. Smith, and K. Jeffay, "Understanding patterns of TCP connection usage with statistical clustering," in *IEEE MASCOTS*, 2005.

[39] F. Hernandez-Campos, F. D. Smith, and K. Jeffay, "Generating realistic TCP workloads," in *Computer Measurement Group International Conference*, December 2004.

[40] J. Sommers, H. Kim, and P. Barford, "Harpoon: A flow–level traffic generator for router and network tests," in *ACM SIGMETRICS*, June 2004.

[41] M. Christiansen, K. Jeffay, D. Ott, and F. D. Smith, "Tuning RED for web traffic," *IEEE/ACM Transactions on Networking*, vol. 9, pp. 249–264, June 2001.

[42] L. Le, J. Aikat, K. Jeffay, and F. D. Smith, "The effects of active queue management on web performance," in *ACM SIGCOMM*, pp. 265–276, August 2003.

[43] M. Weigle, K. Jeffay, and F. D. Smith, "Delay–based early congestion detection and adaptation in TCP: Impact on web performance," *ACM Computer Communications Review*, vol. 28, pp. 837–850, May 2005.

[44] J. Xu and W. Lee, "Sustaining availability of web services under distributed denial of service attacks," *IEEE Transactions on Computers*, vol. 52, pp. 195–208, February 2003.

[45] M. Mathis, J. Semke, J. Mahdavi, and T. Ott, "The macroscopic behavior of the TCP congestion avoidance algorithm," *SIGCOMM Computer Communications Review*, vol. 27, no. 3, 1997.

[46] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP throughput: A simple model and its empirical validation," in *SIGCOMM '98: Proceedings of the ACM SIGCOMM '98 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, 1998.

[47] C. A. Kent and J. C. Mogul, "Fragmentation considered harmful," in *SIGCOMM '87: Proceedings of the ACM Workshop on Frontiers in Computer Communications Technology*, 1988.

[48] C. Shannon, D. Moore, and K. C. Claffy, "Beyond folklore: Observations on fragmented traffic," *IEEE/ACM Transactions on Networking*, vol. 10, December 2002.

[49] D. P. Anderson and et al., "SETI@home: Search for extraterrestrial intelligence at home." Software on-line: `http://setiathome.ssl.berkeley.edu`, 2003.

[50] J. loup Gailly and M. Adler, "zlib compression/decompression library." Software on-line: `http://www.gzip.org/zlib/`, 1995.

[51] D. Moore, R. Periakaruppan, J. Donohoe, and kc claffy, "Where in the world is netgeo.caida.org?," in *INET 2000*, June 2000.

[52] M. Delio, "NETI to examine net's strengths." On–line: `http://www.wired.com/news/technology/0,1282,63180,00.html?tw=wn\_techhe%ad\_2`, April 2004. Wired.

[53] CmdrTaco, "NETI@Home to examine net's strengths." On–line: `http://slashdot.org/article.pl?sid=04/04/27/1257211\&mode=thread\&tid=1%26\&tid=95`, April 2004. Slashdot.

[54] timothy, "NETI@home data analyzed." On–line: `http://it.slashdot.org/it/05/04/25/1710223.shtml?tid=172\&tid=95\&tid=2%18`, April 2005. Slashdot.

[55] "SourceForge." `http://sourceforge.net/`, April 2006.

[56] J. Reynolds and J. Postel, "Assigned numbers," October 1994. RFC 1700.

[57] "nmap." `http://www.insecure.org/nmap/`, March 2005.

[58] "nessus." `http://www.nessus.org/`, March 2005.

[59] J. Levine, R. LaBella, H. Owen, D. Contis, and B. Culver, "The use of honeynets to detect exploited systems across large enterprise networks," in *Proceedings of 4th IEEE Information Assurance Workshop*, (West Point, NY), June 2003.

[60] G. F. Riley, "The Georgia Tech Network Simulator," in *Proceedings of the ACM SIGCOMM workshop on Models, methods and tools for reproducible network research*, pp. 5–12, 2003.

[61] The Apache Software Foundation, "Apache." Software on-line: `http://www.apache.org`, 2005.

[62] Y. Rekhter, B. Moskowitz, D. Karrenberg, G. J. de Groot, and E. Lear, "Address allocation for private internets," February 1996. RFC 1918.

[63] V. Jacobson and R. Braden, "TCP extensions for long–delay paths," October 1988. RFC 1072.

[64] V. Jacobson, R. Braden, and D. Borman, "TCP extensions for high performance," May 1992. RFC 1323.

[65] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow, "TCP selective acknowledgment options," October 1996. RFC 2018.

[66] Y. Zhang, L. Breslau, V. Paxson, and S. Shenker, "On the characteristics and origins of internet flow rates," in *ACM SIGCOMM*, August 2002.

[67] J. Postel, "The TCP maximum segment size and related topics," November 1983. RFC 879.

[68] V. Paxson, "Strategies for sound internet measurement," in *IMC '04: Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement*, 2004.

[69] C. Shannon, D. Moore, K. Keys, M. Fomenkov, B. Huffaker, and k claffy, "The internet measurement data catalog," *SIGCOMM Computer Communications Review*, vol. 35, no. 5, 2005.