An Empirical Workload Model for Driving Wide-Area TCP/IP Network Simulations

Peter B. Danzig* Sugih Jamin* Ramón Cáceres† Danny J. Mitzel* Deborah Estrin*

*Computer Science Department, University of Southern California,
Los Angeles, California 90089-0781

†Computer Science Division, University of California,
Berkeley, California 94720

traffic@excalibur.usc.edu

Abstract

We present an artificial workload model of wide-area internetwork traffic. The model can be used to drive simulation experiments of communication protocols and flow and congestion control experiments. The model is based on analysis of wide-area TCP/IP traffic collected from one industrial and two academic networks. The artificial workload model uses both detailed knowledge and measured characteristics of the user application programs responsible for the traffic. Observations drawn from our measurements contradict some commonly held beliefs regarding wide-area TCP/IP network traffic.

The simulation techniques presented here will be useful in studying congestion control, routing algorithms, and other resource management schemes, for existing and future networks.

Keywords: Wide-Area Networks, Artificial Workload Models, Trace Analysis, Simulation, TCP/IP Internetworks

1. Introduction

When simulating new congestion control, flow control, and adaptive routing algorithms one needs to model the overall pattern of traffic flowing through the network, from distribution of packet sizes and interarrival times, to characteristics such as distribution of host reference patterns and direction of traffic flow. This paper presents an artificial workload model of wide-area network traffic based upon application-level analysis of wide-area TCP/IP [1] traces collected on two campus networks, the University of Southern California (USC) and the University of California at Berkeley (UCB), and one industrial research site, Bellcore.

This research was supported by an equipment grant from the Charles Lee Powell Foundation. Ramón Cáceres was supported by the NSF and DARPA under Cooperative Agreement NCR-8919038 with CNRI, by AT&T Bell Laboratories, Hitachi, a University of California MICRO grant, and ICSI. Danny Mitzel was supported by a fellowship from Hughes Aircraft Company.

The artificial workload model consists of a set of stub-network-specific arrival processes for new conversations between application programs, and stub-network-independent artificial workload model of each of these applications. The model is based on detailed knowledge and certain measured characteristics of those applications responsible for a significant percentage of wide-area traffic on the Internet. Since TCP packets constituted roughly 80% of the wide-area network traffic that we collected,[1] we restrict our model to TCP-based applications. Nevertheless, the model is not TCP-specific. We eliminated dependencies on the transport protocol so that the model could be used to study new transport protocols.

We selected to work on Internet traffic because it is the most successful example of a large and rapidly growing *internetwork*. To appreciate the size of the Internet, consider that more than 500,000 hosts are registered in the Internet domain name system [2] and that the NSFnet backbone connects more than 2,300 university, industry, and government networks [3], To this vast internetwork, large number of subnetworks are being attached almost on a daily basis: more than 720 new subnetworks joined the NSFnet backbone in the eighteen months between January, 1990, and June, 1991, and 38 subnetworks joined in the second week of June, 1991, alone [4, 22].

Future broadband wide-area networks will probably transfer larger amounts of data and carry a mix of traffic currently not found on the Internet. We believe this does not trivialize our present study, for several reasons. First, it will be several years before the current traffic mix changes appreciably. Second, as it changes, it will not obviate the existence of traditional traffic. Third, the workload model and many of the observations that we made while constructing it will be useful in evaluating new algorithms for internetworks in general; and the modeling approach we used can be applied to other types of internetwork. We encourage fellow researchers to construct and publish artificial workload models of their networks.

[1]For the UCB data, UDP packets make up 16% of all network traffic, while ICMP packets account for only 1% of all traffic. Of all UDP packets, 63.63% belongs to DNS, 15.82% to ROUTE, and 10.51% to NTP.

Previous traffic studies of TCP/IP have examined the statistics of the aggregated packet arrival process on local area networks [9, 10, 11], at border routers [12], and inside a wide-area backbone [13]. These studies have shown that packet interarrival times are not Poisson, but rather follow a *packet-train* model. The *packet-train* model has proven valuable in the study of packet routers design [14, 15].

The study presented in this paper is different from all the studies mentioned above. Instead of confining ourselves to the network and transport layers, we studied the characteristics of key applications responsible for 96% of wide area TCP traffic. We believe these applications are representative of applications currently running on wide-area networks other than the Internet. Two related studies, one at University College London and the other at Lawrence Berkeley Laboratory, identified a similar set of applications as being responsible for most of their wide–area TCP traffic [16, 17].

The decision to characterize application traffic was supported by the following observations. Measured interarrival times alone are not adequate to characterize conversations for the purpose of driving flow and congestion control algorithm simulations; interarrival times are themselves a function of existing flow control mechanisms. Interarrival times may be useful in characterizing interactive traffic due to its low volume. However, bulk traffic must be characterized by the amount of data transferred; its duration mostly reflects network link speed and flow control algorithms. Furthermore, although interactive conversations are bidirectional, they send much more data in one direction than in the other; an accurate model must take this into account. Finally, some applications converse with more networks than do others (see Figure 8).

From these observations, we concluded that researchers would benefit from more realistic traffic models, particularly in studying switching and control mechanisms through simulation. When simulating wide-area networks, current practice ignores the distribution of number of bytes transmitted, the bidirectionality of bulk traffic sources, and the duration of interactive connections., instead idealized FTP [5] and TELNET [6] sources are used; the idealized FTP sources send huge quantities of data in one direction, while the idealized TELNET sources send a

Poisson stream of small packets in one or both directions [7, 8]. This paper makes the first step towards an internetwork source model. It outlines the necessary steps to describe and simulate a new conversation between two networks. However, it does not seriously investigate the question of when to establish a conversation between two networks; we are currently addressing this problem.

2. Measurement and Analysis Methodology

Below we describe the data collection methods, their associated loss rates, and our definition of a conversation.

2.1. Data Collection Sites

Wide-area traffic data was collected at two university campuses (UCB and USC) and one industrial research laboratory (Bellcore). The data collected at UCB traced all traffic between the campus and the Bay Area Regional Research Network (BARRNET); data collected at USC traced all traffic between the campus and Los Nettos; and data collected at Bellcore traced all traffic between their Morristown laboratory and the John von Neumann Center Network (JVNCNET).

2.2 Trace Contents

We analyzed a sample of 5,891,622 TCP packets from UCB, 5,221,036 from USC, and 1,703,269 from Bellcore. These packets correspond to one day of continuous trace collection at UCB and USC, and three days of continuous tracing at Bellcore. The collection started at 10:20 on Tuesday, October 31, 1989 at UCB, 14:24 on Tuesday, January 22, 1991 at USC, and 14:37 on Tuesday, October 10, 1989 at Bellcore.

Each trace record consists of a time stamp and the first 56 bytes of raw network data. The time stamp records the arrival time of the packet at the tracing apparatus. The 56 bytes of data

hold the packet headers from the datalink layer (Ethernet), the network layer (e.g. IP), and the transport layer (e.g. TCP and UDP).[2]

2.3. Tracing Instrumentation and Packet Loss Rate

The UCB data was collected with a Sun 3 workstation equipped with a microsecond timer [18]. The resulting time stamp resolution was 10 microseconds. The workstation ran a modified Unix kernel with a circular buffer big enough to hold 128 full-size Ethernet packets. A dedicated user program transferred trace records from this buffer to tape. No packet losses due to buffer overflows were detected during the UCB measurements. The packet loss rate induced by separate stress testing was less than 5% in the worst case.

The USC data was collected using the NNStat program suite [19] on a Sun SparcServer 4/490. The NNStat program uses the Sun *gettimeofday()* system call which has a 20-millisecond resolution. During similar measurements, we estimated the loss rate by sending a Poisson stream of *ping* packets. We observed that 0.6% of these packets were missing from the tape.

The Bellcore data was collected using a Sun 3 workstation augmented with a microsecond interval timer and a single-board computer dedicated to collecting and time-stamping trace packets. The time-stamps have a 10 microsecond resolution. A hierarchical system of double buffering carried the trace records from the single-board computer to tape. No packet loss was detected anywhere in the monitoring system during the Bellcore measurements [11].

2.4. Are the Traces Representative?

Both USC and UCB campuses use mostly UNIX and IBM computing systems. Bellcore uses mostly UNIX systems. We believe that the systems traced are representative of sites currently attached to the Internet, and that our analysis also applies to other sites. However, we recognize

[2]We did not encounter any packets with IP protocol options. We do take into account the length of any TCP option present when calculating the size of the data part of a packet. In the UCB trace, we found 0.02% of the IP packets carrying TCP data to be IP fragments. For USC, the number was 0.05%, and for Bellcore, the number was 0.02%. We ignored packets that have their *more fragment* flag set.

that traces collected at other sites might show a different application breakdown than the ones reported here. Future studies will be conducted to further validate our results.

The breakdown of traffic varies greatly from site to site (see Table 1) and should be accounted for when simulating sequences of conversations. However, the *characteristics* of conversations, shown later, are essentially identical between the three sites, even though the USC trace was collected one year and three months after the others. Furthermore, these characteristics are shared by two different days of UCB traces, and by a one-day trace and a three-day trace of Bellcore traffic. That is, the distributions of number of bytes transferred, conversation durations, total packets per conversation, and packet sizes are indistinguishable. For legibility, we present only UCB data in the body of the paper. Appendix 1 contains representative figures comparing data from the three sites.

Traffic Type	% Packets			% Bytes			% Conversations		
	UCB	USC	Bell	UCB	USC	Bell	UCB	USC	Bell
ftp (ctrl+data)	**12.0**	**5.0**	**18.7**	**36.2**	**10.6**	**54.9**	**2.8**	**3.2**	**4.8**
shell (rcp)	0.2	3.6	1.4	0.4	12.5	4.3	0.2	0.2	0.7
smtp	**11.6**	**3.1**	**12.6**	**11.0**	**1.9**	**10.6**	**69.6**	**52.3**	**68.0**
dc_10	—	3.5	—	—	0.8	—	—	1.5	—
vmnet (bitnet)	10.0	9.1	—	25.4	20.7	—	0.1	3.3	—
uucp	0.2	0.1	0.8	0.4	0.1	1.3	0.3	1.1	2.2
nntp	**11.6**	**36.3**	**9.2**	**15.8**	**44.5**	**15.6**	**0.4**	**1.6**	**0.8**
telnet	**28.0**	**16.6**	**36.3**	**5.5**	**2.3**	**6.5**	**4.1**	**8.7**	**8.7**
rlogin	**15.5**	**5.8**	**18.5**	**2.8**	**0.7**	**3.1**	**2.1**	**2.7**	**4.3**
x11	0.2	5.0	0.4	0.2	2.5	0.1	—	0.5	0.4
ircd	4.6	—	—	1.3	—	—	0.6	0.2	—
finger	1.1	0.4	0.5	0.6	0.2	0.2	18.3	17.8	7.6
domain	0.1	0.1	—	—	0.2	—	0.2	3.3	0.1
other	4.9	11.3	1.6	0.4	3.1	3.1	1.3	3.6	2.4

Table 1: Breakdown of unidirectional TCP traffic

NSFnet statistics for the total packet activity generated by stub-network also suggest that our traces are representative [4, 22]. The six applications identified in Table 1 as transmitting the largest number of packets (TELNET, RLOGIN, FTP, SMTP [20], NNTP [21], and VMNET) are also the TCP applications most often encountered on the NSFnet backbone. (Table 2 provides a short glossary of Internet protocols and applications.) Of the 1,174 stub-networks for which backbone activity was detected during October 1989, UCB was the 3rd busiest, and Bellcore was the 130th

busiest. Of the 2,345 active stub-networks measured during January 1991, USC was the 31st busiest. These numbers indicate that our traces capture activity generated by a range of stub-networks, from a very active one like UCB, to a moderately active one like USC, to a less active but non-trivial one like Bellcore.

DC_10	Cadre Teamwork Mailbox 10
DNS	Domain Name Service, host name resolution protocol
DOMAIN	Domain Name Service
FINGER	User information query application
FTP	File Transfer Protocol
ICMP	Internet Control Message Protocol
IP	Internet Protocol, a network layer datagram protocol
IRCD	Internet Relay Chat Program Server, a tele-conferencing application
NTP	Network Time Protocol
NNTP	Network News Transfer Protocol
RLOGIN	Remote login application
ROUTE	Routing Information Exchange Protocol
SHELL	Remote shell application, often used for remote copy (rcp) operations
SMTP	Simple Mail Transfer Protocol
TCP	Transmission Control Protocol, a reliable transport layer protocol
TELNET	Remote terminal application
UDP	User Datagram Protocol, an unreliable transport layer protocol
UUCP	Unix to Unix Copy Program, used for mail, news, and file transfer
VMNET	A method of running the RSCS protocol (usually from IBM mainframes running VM) on top of TCP; it is used to handle a majority of the BITNET backbone traffic
X11	X window system

Table 2: Internet Protocols and Applications

2.5. Traffic Pattern Analyzer

We wrote a traffic pattern analyzer to reduce the raw packet trace data and produce the probability distributions employed by the application-specific workload models. One of the first decisions we had to make was how to break up the trace into meaningful units. Should we adopt the *packet-train* model or should we maintain a state machine per TCP connection? We look at these alternatives below.

The *packet-train* model has largely replaced earlier Markov models of network traffic [9, 10, 13]. In the *packet-train* model, a stream of packets is broken up into *trains*. Two consecutive trains are delimited by a *maximum allowable inter-car gap* (MAIG). The MAIG is usually chosen