January 2002

Characterization of Internet Traffic and User Classification: Foundations for the Next Generation of Network Emulation

Rigoberto Chinchilla
Eastern Illinois University, rchinchilla@eiu.edu

John Hoag
Ohio University

David Koonce
Ohio University

Hans Kruse
Ohio University

Shawn Osterman
Ohio University

See next page for additional authors

Authors

Rigoberto Chinchilla, John Hoag, David Koonce, Hans Kruse, Shawn Osterman, and Yufei Wang

Characterization of Internet Traffic and User Classification: Foundations for the Next Generation of Network Emulation

Rigoberto Chinchilla, John Hoag, David Koonce, Hans Kruse,
Shawn Osterman, Yufei Wang
Ohio University, Athens, OH 45701

Abstract

Rigorous experimentation is emerging as the preferred method for validating the benefits of protocols, especially those affecting quality of service. The emulation of traffic generators provides a compact and efficient method for testing protocols and techniques in a realistic context. Models of user behavior serve as the basis for the *trafgen* traffic emulator and are subject to refinement as our knowledge expands. The project in this study uses a layered approach to study different Internet applications and protocols including the hypertext transfer protocol, http.

Introduction

Many external forces are working to change the study of network performance. Analytical models, simulations, and experiments are in turn increasingly elusive, not credible, or not feasible. The distance increases daily between the Internet's heritage as a research entity and its future as a virtual marketplace, where economic efficiency reigns.

The National Research Council has called for a return to experimentation as the primary means to expand our knowledge of network performance. "Researchers, research funders, and network operators should work together to find opportunities that would allow more network research to be done in realistic operational settings."[1] Testbed facilities once supported by the National Science Foundation now have no contemporary equivalent, public or private. Testbed facilities feature extensive instrumentation and may not interconnect with the commodity Internet. The NRC further states that networks with comprehensive measurement ability provide better information that, "... would provide the basis for understanding the implications of introducing new ideas on the network."[2] The credibility of stochastic simulation studies, an alternative to network experimentation, has been undermined recently due to improper assumptions regarding random number generation.."[3]

This project has embraced the testbed approach to experimentation for network performance analysis, using emulators to generate traffic and to model the subtleties of wide-area networks. The core technology for traffic generation is a Java-based module known as *trafgen*, whose presence may substitute for one or more point sources of traffic and whose characteristics may be tuned for one or more applications or protocols.

This report provides insight into the methodology to be employed in network performance testing, the role of user modeling within the project as a whole, the architecture of trafgen and the testbed, the structure of experimental results, and the future research questions to be considered.

Proper modeling of user behavior is an important input to emulation experiments, as will be discussed in depth in a later section. A multilayered approach is necessary, for instance, to describe the traffic involved in web browsing: distinct but related models exist at the session, page, connection, and packet layer.[4]

In general, user modeling may involve any of the following approaches:
- Markov: requiring some definition of system state, with memoryless transitions.
- Petri Nets: involving transitions based on meeting preconditions such as "discoragement," "exhaustion," and "satisfaction," gleaned through inspection of logs.
- Hierarchical: selecting behaviors based on user selection and invocation of rules.
- Other: considering psychological or sociological profiles of users, etc.

Not only will the *trafgen* module and experiments embody known user models, the testbed also as the ability to refine user models. The testbed provides the means for testing hypotheses regarding user behavior in the presence or absence of controlled traffic factors.

Methodology

This project incorporates a data collection infrastructure that supports statistical modeling and hypothesis testing.   Infrastructure for experiments consists of (a) an equipment configuration that includes local and wide are network components, (b) *trafgen* sources and sinks, and (c) instrumentation nodes. The notion of an experiment involves (a) configuration of *trafgens* for particular user models and their traffic characteristics, and (b) execution of the *trafgens* while under instrumentation, and (c) data collection and storage for later analysis. Description of specific testbed components appears in a later section.

Instrumentation nodes run a modified version of tcptrace that provide summary data for each TCP connection. On a per-connection basis, we collect the protocol type, the number of bytes transferred, and the total connection time. This program is being modified to collect retransmission data and to identify related connections within, for instance, a single http page view. All per-connection data is intended to be identified by experiment and session within a single table whose rows can be selected via Structured Query Language.

In addition to providing descriptive statistics, experimental data are amenable to various forms of hypothesis testing with regard to user effects. Regression models, analyses of variance, and goodness-of-fit techniques are available to assess the validity of research hypotheses to be developed over the course of the project.   The next section of this report outlines some of the research questions related to user models.

User Models

Up to this time we have been refining the modeling techniques used in *trafgen.* Currently, we model traffic based on applications, such as a web browser or a file transfer application. In the present upgrade, we are modeling the way users utilize multiple applications. For example, two users engaged in voice over IP conversation may be likely also to initiate a file transfer as they exchange documents relevant to their conversation. We will also begin a more detailed study of user behavior with the goals of understanding how use of network applications changes as the quality of service delivered by the network increases or decreases, with an eye towards improving future versions of *trafgen*.

Our expected outcome will be a software tool and a series of detailed applications and user models that will be available to network researchers and designers who need a cost-effective way to prototype new applications and protocols in a testbed with realistic traffic and Quality of Service characteristics. These stochastic models will be developed from analysis of network traffic. This project will encompass the inclusion of these user models into *trafgen* as the basis for its future enhancements

User Behavior

Most research on Internet user behavior focuses on demographics, level of sophistication and website path prediction. The Pew research center studies the Internet behavior of major demographic groups in order to provide insight to policy makers. Booz-Allen and Hamilton, with Nielsen Media Research, has studied Internet users to reclassify them by usage pattern and not by demographics. Their study identifies seven primary usage patterns and uses these levels to study advertisement placement and frequency. Cadez cluster users based on Website navigation patterns identified from server Log files. These clusters are then used in a Markov model to simulate users and analyze system behavior. Cisco and HP use a combination of Cisco Netflow's packet statistics capturing capability; HP's Smart Internet Usage analysis tool to create standard Internet data records (IDR). These records can then be used in a data mining tool to identify specific user patterns. The focus of the Cisco and HP systems is on marketing, customer retention and capacity allocation.

To use simulated users in quality of service research, users must be modeled with respect not only their typical behavior but also their responsive behavior with respect to service quality. Depending on users' sophistication, their reaction to low service quality performance may vary. Users may increase or decrease their traffic based on the service type, perceived reason for poor quality and their level of sophistication.

User Classification

Initial work ha been concentrated in collecting many samples of different connections in which we can "track" the activity of each user over several periods of time, by observing the user as a specific IP address as the following diagram shows
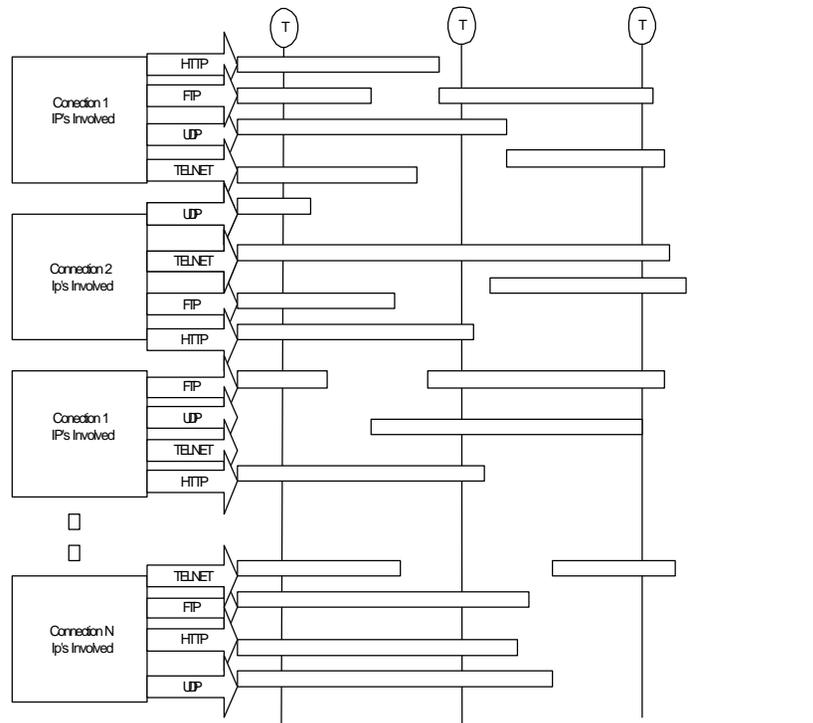
With this information and the full set of statistics per connection, we are creating a set of rules in order to extract the right information from the diagram. This set of rules determines what information we need and how we will import these information into the database from the raw data. We have structured the data in a table format showing the information of above figure .

Once we have the information in a suitable way (with thousands of samples) a data mining process with a set of queries will extract the information related with every connection and the activity per connection. Uultimately we will be able to classify users as advanced, intermediate or novices. QoS is incorporated in order to make correlation between throughput and activities like balking from FTP, Telnet or simple browsing activities.

Architecture

The first prototype version of *trafgen* was written as part of a satellite experimentation project for NASA in 1996-1997.[5] The primary purpose of both the original and the current version is to inject realistic network traffic into a network. Because *trafgen*'s emphasis in on creating realistic traffic, it does not model the traffic at the packet level, rather the traffic is modeled at the transport protocol level. When driving TCP traffic, for example, *trafgen* controls the pacing and blocking of traffic written into a TCP socket from the application level. Achieving a desired blocking of the data is accomplished by using TCP's PUSH mechanism to control the granularity of the TCP segment sizes on the network. Likewise, delay between segments is accomplished by delaying the application before providing more data to TCP. The manner that TCP chooses to inject that traffic into the network is under the control of a production TCP protocol stack. It's important to keep in mind that allowing TCP to drive the data is one of *trafgen*'s primary design choices; although it would be possible to generate actual network packets from an application program similar to *trafgen*, the behavior of those packets would not then correspond to the behavior of TCP packets on the same network due to changes in packet loss, round trip time, congestion, etc.

For our purposes, "realistic" traffic is defined as traffic that is statistically similar to traffic generated on a production network. The original prototype was built around the framework from tcplib[6] with additions by our research team to incorporate HTTP traffic. In the tcplib framework, traffic

generation is guided by random events driven by a cumulative distribution function. For example, when driving Telnet traffic, tcplib determines write-event sizes, elapsed time between write events, and total elapsed time of the entire connection. The current version of tcplib can also be driven by tcplib-style events, but can also drive traffic based on several other statistical models. Our experience with the original *trafgen* prototype lead to two important design differences for the version presented in this paper:

1. The tcplib framework was too limiting. Although we added a realistic http model to tcplib, the first version didn't include more recent advances in http modeling and didn't include any provisions for peer-to-peer networking.
2. The software was too fragile. The original prototype was written in C using the primitive thread library constructs that existed at the time. It was extremely difficult to add new models and was extremely difficult to debug. As a result, it was of limited use to other researchers.

To overcome those limitations, the current version of *trafgen* is written in Java and TCL. The low-level statistical routines and packet transmission and receptions routines are written in Java. Java gives us a great deal of portability and a mature thread implementation. Borrowing an idea from the NS network simulator, the traffic generation itself is driven by scripts written in TCL. TCL has the advantage of being a very high-level language with allows fairly complicated models such as http to be written very concisely. Several colleagues have indicated that Python might be a better choice for the scripting engine and we are currently evaluating that as well.

As an example of the scripted portion of *trafgen*, Table 1 shows the control portion for a single Telnet thread.

```
while { $timeNow < $timeEnd } {
    set sendSize [java::call $sendSizeModel getNext 50]
    set replySize [java::call $replySizeModel getNext 1000]
    set replyDelay [java::call $replyDelayModel getNext 100]

    set v [$message sendMsg $version $sendSize $replySize $replyDelay]
    puts "telnet Send $sendSize bytes"

    set v [$message receiveMsg $replySize]
    puts "telnet Receive $replySize bytes"

    set nextFire [java::call $interarrivalModel getNext 5000]
    set v [java::call sleep sleepms $nextFire]

    set timeNow [java::call Time getTime]
}
```

Table 1.

There are three approaches for characterizing Internet applications.
1. Server log
2. Client log
3. Traffic trace

Here we focused on the third method. From the packet trace we got by tcpdump, and from the knowledge about the higher-layer protocol used (ssh, ftp, http), traffic analysis can yield a model of the application and user behavior.

The following analysis is based on HTTP protocol:
1. Each HTTP transfer consists of a single request-reply pair of messages. So we need to know the distribution of **request size** and **reply size**;
2. Each document consists of multiple files, so the client and server will set up multiple connections to transfer those files. So we should record the distribution of the **number of files per document**.
3. After the page is displayed by the browser, the user will look for a while and then click some other hyper-link for other document. We use **think time** to describe the time user spend on reading the document.

Since the tcpdump files we got usually don't consist of enough information for us to decide the relationship between those files transferred, (the GET field is usually not complete and refer field is seldom captured) We can use heuristic method to find the relationship between those connections.

1. The two connection should be originated from the same client IP address, the reason is apparent, two users requests rarely belongs to a same document.
2. The two connections are usually not separated by a long time. The client will initiate more connections to retrieve for files in a short of time period, saying, less than 1 second. And if the request is originated by the user's click, it is usually longer than 2 seconds.

Outputs and Organization

In the proposed model, we describe user activity using a three-layer hierarchy:
Top Layer: *Session*

Second Layer: *Collection*
Sessions consist of one or more Collections
Third Layer: *Data Item*
Collections consist of one or more Data Items

As an illustration, Table 2 shows the definition of this hierarchy for three protocol types.

The ORACLE database used to analyze user behavior contains records for each Data Item observed in the network packet trace. Each of these atomic records contains fields that allows the Data Item to be aggregated into the corresponding Collection and Session item. This aggregation can take place during the packet trace processing, or during various post-processing steps. The network trace processing also records indications of the network state at the time the Data Item was observed. This state information is used to characterize differences in user behavior which may be caused by changes in service quality delivered by the network.

The following data are collected:

1. Unique ID assigned by the packet trace processor (tcptrace)
2. Data Item description (e.g., URL in HTTP GET, file name in FTP)
3. Data Item start date
4. Data Item start time
5. Data Item duration
6. Data Item protocol type
7. Data Item user-side IP address
8. Data Item far-side IP address
9. Data Item user-side port number
10. Data Item far-side port number
11. Data Item total bytes sent by user
12. Data Item total bytes received by user
13. Collection ID assigned to this Data Item
14. Session ID assigned to this Data Item
15. Estimated RTT for packets exchanged within this Data Item
16. Duplicate ACKs sent by the user during this Data Item (if packet data was collected near the user)
17. Packet re-transmissions from the user (if packet data was collected near the user)
18. Duplicate ACKs sent by the far-side during this Data Item (if packet data was collected near the far-side)
19. Packet re-transmissions from the far-side (if packet data was collected near the far-side)

	Interactive Sessions (Telnet, SSH, Rlogin)	File Transfer (FTP, S-FTP)	WWW
Session	A session starts when the TCP connection is opened and end when the connection is closed or aborted.	A session starts when the control connection is opened and ends when the control connection is closed.	A session starts with the first HTTP request issued by a user after a dormant period during which no HTTP interactions were seen. The session ends at the start of the next dormant period.
Collection	Identical to the session (there is always exactly one collection in a session)	Identical to the session (there is always exactly one collection in a session)	A collection encompasses all data items exchanged as a result of an initial request (data item) which triggered further, embedded, requests which were executed without manual user input.
Data Item	One packet, or a series of packets seen with temporal separation less than a set threshold, and the data returned to the user in response to the packet(s). Transfer Data Item: Starts with the establishment of a TCP data connection, and ends with the termination of this connection. Includes all data sent and received for the duration of this connection.	Control Data Item: One packet, or a series of packets seen with temporal separation less than a set threshold, and the data returned to the user in response to the packet(s), on the control connection.	Consists of an HTTP GET command and the data returned by the server in response to the command.

Table 2

Contributors

Rigoberto Chinchilla (rigoberto.chincilla@ohiou.edu) is a Ph.D. student in Integrated Engineering at Ohio University.

Dr. John C. Hoag (j.hoag@ieee.org) is Assistant Professor in the McClure School of Communication Systems Management at Ohio University.

Dr. David A. Koonce is Associate Professor in the Department of Industrial, Mechanical, and Systems Engineering at Ohio University.

Dr. Hans Kruse (kruse@ohio.edu) is Associate Professor in the McClure School of Communication Systems Management and Adjust Associate Professor in the Department of Electrical Engineering and Computer Science at Ohio University. Dr. Kruse is also manager of the Ohio Consortium for Advanced Communications Technology, operator of the NASA-owned ACTS satellite.

Dr. Shawn Ostermann (ostermann@cs.ohiou.edu) is Associate Professor in the Department of Electrical Engineering and Computer Science at Ohio University.

Yufei Wang (wyf @bobcat.ent.ohiou.edu) is a graduate student in the Department of Electrical Engineering and Computer Science at Ohio University.

1.
National Research Council, <u>The Internet's Coming of Age</u>, Washington, D.C.: National Academy Press, 2001.

2.
National Research Council, "Looking Over the Fence at Networks: A Neighbor's View of Networking Research," Washington, DC: National Academy Press, 2001.

3.
Krzysztof Pawlikowski, Hae-Duck Joshua Jeong and Jong-Suk Ruth Lee, "On credibility of simulation studies of
      telecommunication networks", <u>IEEE Communications Magazine</u>, vol. 40, no. 1, January, 2002.

4.
Casilari, E., et al, "Modeling of HTTP Traffic," <u>IEEE Communication Letters 5:6,</u> June, 2001.

5.
Allman, Mark, Hans Kruse and Shawn Ostermann, "Satellite Network Performance Measurements Using
Simulated Multi-User Internet Traffic," Proceedings of the 7th International Conference on Telecommunication Systems, March, 1999.

6.
 Danzig, Peter B., and Sugih Jamin, "tcp-lib: A Library of TCP/IP Traffic Characteristics," USC Networking and Distributed Systems Laboratory TR CS-SYS-91-01, October, 1991, ftp://catarina.usc.edu/pub/jamin/tcplib.