# Modeling and Evaluation of Network Applications and Services

Ajit K Jena[1], Parag Pruthi[2] and Adrian Popescu[3]

[1]University of Karlskrona/Ronneby, 371 79 Karlskrona, Sweden. E-mail: akj@its.hk-r.se.
[2]University of Karlskrona/Ronneby, 371 79 Karlskrona, Sweden. E-mail: ppr@its.hk-r.se.
[3]University of Karlskrona/Ronneby, 371 79 Karlskrona, Sweden. E-mail: adrian@its.hk-r.se.

## Abstract

The global Internet has seen tremendous growth in terms of nodes and user base as well as of types of applications. One of the most important consequences of this growth is related to an increased complexity of the traffic experienced in these networks. Each application has a set of unique characteristics in terms of its performance characteristics, its transactions as well as the way the transaction processing profile maps onto unique network resource requirements. In order to support Internet applications effectively, it is therefore important to understand and to characterise the application level transactions. Recent advances in high resolution traffic monitoring and analysis capabilities have enabled us to build up realistic models for diverse network applications. In this paper we report investigations of classical applications such as FTP, SMTP, and HTTP to evaluate end-to-end network performance requirements. Our results show the presence of a robust correlation structure in the traffic streams that has a fundamental bearing on the user perceived quality of the applications.

## 1 Introduction

One of the most important questions today is related to the impact predominant protocols and end-to-end congestion control mechanisms in the TCP/IP protocol stack may have on application-level message flow parameters, especially the parameters related to self-similarity. Generally, there is a lack of results showing quantitative contributions of protocol stacks and their effects on source parameters. For instance, at the application level, heavy-tailed distributions of the server responses (resulting from heavy-tailed distributions of file sizes) may result in fractal or self-similar aggregate traffic at the network link. How do the long-range dependence (LRD) parameters characterizing the server message size distributions get modified by various network level protocols such as TCP and IP? The application messages are fragmented into application layer buffers. These buffers are in turn broken up into smaller portions (so-called ”cascading effect” [2]) to be carried by the specific network media being used. These packets are subject to delays and errors. They may even be discarded by a congested router/switch. The flow control mechanism of TCP regulates the packet flow by coordinating the sender and receiver windows. How does the heavy-tailed distribution of messages map onto the packet flow inside the network? Does the protocol dynamics have any effect on the distributional characteristics of the messages and if so, can the influence be quantified?

It is important to understand and to quantify the effect of controls (e.g., TCP flow control) on application-level parameters since the management of the application traffic has requirements that are different from those for the management of transport-level traffic. Application-level traffic management occurs by decisions on the admission control and resource management hierarchy whereas transport- and network-level management requires the delivery of implied (e.g., IP) or contracted (e.g., ATM) Quality of Service (QoS). Thus, in delivering end-to-end guaranteed service it is not just important to understand the traffic parameters at the application layer, but also to model and to quantify the effects of network control mechanisms on the application-level parameters. For example, if the application happens to be delivering multimedia service, then a stream of video and audio may be delivered to the client where the parameters of the video and audio stream are impacted by network congestion and recovery mechanisms. By the time the stream arrives at the client node, it may have been shaped significantly and the parameters of the stream may have changed drastically from the actual parameters negotiated during client/server connection setup. The case of interactive multimedia communications is more challenging as the source itself will shape the stream based upon feedback from the client. For the purpose of this study we only consider the case where the feedback loop is only closed between the transport layers (i.e. TCP) and not the application layers (e.g., image retrieval, streaming video, which is typically the case on the Internet today).

## 2 Performance Model

The goal is to study the contribution of TCP/IP protocol stack on self-similar traffic descriptors and end-to-end network performance. For this, a client-server framework model is developed in a simulation model (Fig. 1). The model closely mimics real-life events that occur in an Internet consisting of clients and servers. The client and the server software reside on hosts connected by an Internet cloud. The hosts are running the following applications: FTP, SMTP and HTTP.

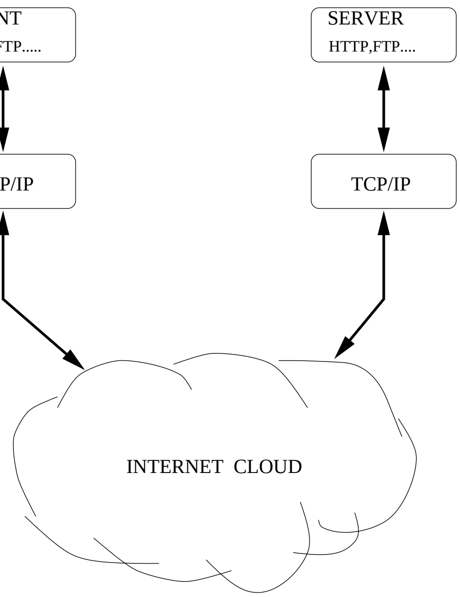


Figure 1: *Client/Server Framework Model*

A network architecture commonly used in typical business and academic organizations is used for the simulation model, as shown in Fig. 2. The clients and the servers are organized into two separate subnets interconnected by a WAN link. Each of the client nodes are 10BaseT Ethernet hosts. The traffic from these client nodes are aggregated via 16 port Ethernet switches. The Ethernet switches connect in turn to a second level switch via 100BaseT uplinks. The second level Ethernet switches connect to a router. The routers connect to a Frame Relay WAN via their Frame Relay Access Devices (FRAD). In this setup, the WAN infrastructure consists of two Frame Relay switches interconnected via a single T1 link. Our experiments are focussed onto the shared Frame Relay WAN link.

The client and the server in a client-server environment interact by exchanging a series of protocol messages. The server waits for requests from clients and processes them in a FIFO order. The traffic stream generated by the client-server interaction depends on the content transferred as well as the moments when the transfers are initiated (the timings). Furthermore, the nature of the information content transferred depends on the application type, e.g., a data file, a digitized picture or voice, a web object, a program output, or keyboard input collected from the user. In addition to these, there are messages exchanged between hosts to control the transfer operation. The timings are determined by the user actions. Typically the transfers are organized into application sessions where each session

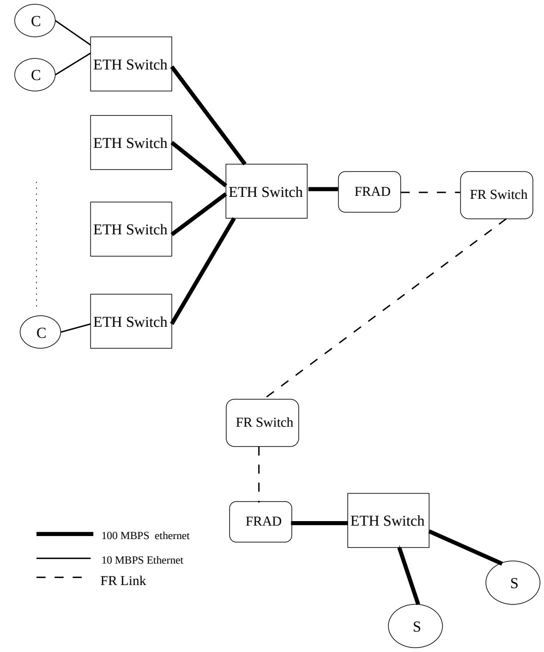


Figure 2: *Network Architecture Simulated*

(e.g., browsing a web page in HTTP) consists of a bunch of transactions (e.g., fetching the main page and consecutively the emdedded documents in HTTP). The timegap between two consecutive transaction initiations are very small (in the order of milliseconds) so that they can be ignored for modeling purposes. In other words one user session results in an ON period and an inter session gap gives rise to an OFF period. The distributions of ON and OFF durations depend upon the distribution of content sizes and the distribution of number of transactions within a session. The combination of ON and OFF periods produces in this way the correlation pattern observed in different traffic streams.

A fractional Brownian model (fBm) is used to model the network traffic (according to the so-called ”Joseph effect” [7]), whereas the application traffic is usually modeled with Pareto distributions (”Noah effect”) or even with uniform distribution for specific cases [3]. Using specific data traffic collected during a busy hour from a production Frame Relay network (Pacific Bell’s Internet Access network [1]), key statistics are calculated for diverse applications (e.g., number of embedded documents in a HTTP session, total number of bytes accessed from various servers on the Internet during a busy hour, data connections for FTP file transfers) [6]. These statistics are finally used to properly model the application traffic according to the specific application.

The OPNET simulation tool is used as a simulation environment [5]. OPNET is a discrete event simulation package that provides a layered framework. The lowest layer is the process layer. This layer handles the core of the system functionalities and is organized in the form of finite-state machines. Each of the protocol layers in the simulation model (e.g., SMTP, TCP, IP) are process models. The next level of hierarchy is the node layer. This layer uses process models as building blocks and interconnects them to form nodes. For example, client and server nodes consist of an application process (HTTP, FTP, or SMTP), TCP and IP process models interconnected in a hierarchical fashion. Finally, the OPNET network layer is the topmost layer and consists of nodes interconnected via communication links.

Data transfer service under OPNET Frame Relay implementation is supported in the form of Permanent Virtual Circuits (PVC), with the associated parameters Committed Information Rate (CIR), Committed Burst Size (CBS) and Excess Burst Size (EBS). These parameters can be independently specified for both directions of data transfer. The OPNET implementation also supports store-and-forward switching, with Congestion Avoidance, Notification, and Recovery functionalities provided in the switches.

# 3 Effects of TCP/IP on Application Parameters

The aggregate link level traffic is a superimposition of several TCP flows whose durations are derived from the distributions of the objects being transferred (according to the specific application). The flow of objects is affected by diverse network and congestion controls and the traffic patterns have been shown to be of self-similar nature [7]. A fBm model is used to model the (exactly) self-similar traffic flows, with the following specific parameters: $m$ (mean rate), $a$ (peakedness), and $H$ (Hurst parameter)[4].

Fig. 3 shows the estimate of the Hurst parameter of the traffic flow on the Frame Relay link connecting the client and server subnets for various applications and for various levels of link utilization. The term ”xmt” refers to the data flow from clients to server, whereas ”rcv” refers to the data flow from server to the clients. It is noticed that the Hurst estimate remains generally unaffected as a function of reasonable link load (up to 90%). That means that the TCP/IP stack has almost no influence over self-similar traffic properties ($H$). Furthermore, the Hurst parameter estimated in the link traffic is decisively attributed to the distribution of the file size (or mail messages typed in by the user) that are transported by the application layer protocol. Given the $\alpha$ parameter chosen for the application file size (Pareto distributed ON-OFF sources) the H estimate seen in the link traffic is easily computed with $H = (3 - \alpha)/2$. The simulation results closely match the theoretically calculated values for H [6]. Furthermore, when SMTP, HTTP, and FTP traffic types are mixed, the self-similar nature of HTTP traffic (being characterized by the Pareto distributed document sizes with lowest $\alpha$) is predominating.

As the number of clients is increased, the mean traffic rate $m$ on the bottleneck link increases as well (Fig. 4). Due to the application layer protocol implementation the

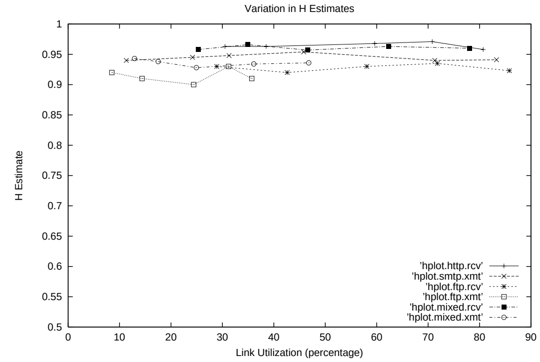


Figure 3: *Variation of Hurst Parameter with Link Load*

traffic load may rise on the clients-to-server or server-to-clients half of the Frame Relay PVC. The SMTP messages and the FTP PUT requests create data flow from the client subnet to the server subnet. On the other hand, HTTP and FTP GET operations initiate flows from the server subnet to the client subnet. Some form of load balancing effect is created on the two halves of the PVC, in spite of the fact that the link utilization in the clients-to-server half of the PVC is still low. This is because the FTP PUT operations occur only with a probability of 0.25 and the number of SMTP client nodes is only one third of the total mix. The distribution of the SMTP message body follows the same structure as the HTTP document size distribution with the minimum message size being 300 and the Pareto process having a $\alpha$ parameter of 1.1.

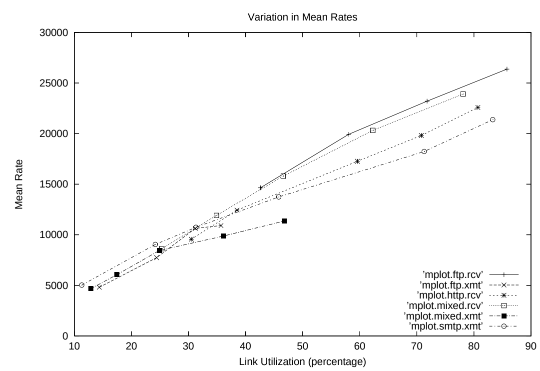


Figure 4: *Variation of Mean Rate with Link Load*

The effects of TCP/IP dynamics on the $a$ parameter (peakedness) for different types of application layer protocols are shown in the Fig. 5. The parameter $a$ measures the variation in the traffic process on smaller time scales (500 millisecond intervals). As the link utilization level goes up, the variation in the traffic flow gets affected by the finite link capacity and the $a$ parameter finally collapses. This can be viewed as some kind of choking effect where these traffic variations are eliminated. Another important observation is related to the almost flat curve showing the variation of $a$ parameter for the PUT component of FTP.

This is due to the insignificant load levels generated by the assymetric load structure as described above.

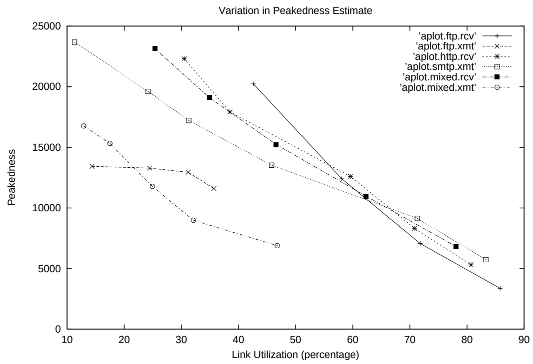


Figure 5: *Variation of Peakedness with Link Load*

The main conclusions of these experiments is that TCP has large difficulties in providing transport services over ”long” lines, e.g., sattelite, gigabit. This is based on the observation that TCP does not change the LRD characteristics of traffic. Furthermore, it is observed that diverse network performance as captured by throughput, packet loss rate, packet retransmission rate, etc., degrade gradually with increased link loads due to degradations in *a* and increase of *m*.

# 4 End-to-End Delay Performance

One of the most important end-user performance of Internet applications is characterized by transaction or session level delays. For instance, in the case of HTTP, each web page requested by the user represents a HTTP session and the embedded documents are the transactions with that session. FTP and SMTP applications follow a similar session oriented structure. By controlling the amount of port buffers at the switches, it is possible to control switch level queueing delays. However, because of the correlation inherent in the application streams, packet losses and hence retransmissions between the end-stations may result. This increases in turn link level traffic and also indirectly increases the application level delays seen by the application. It is generally difficult to arrive at a optimum bandwidth and buffer allocation strategy to control application level delays while minimizing packet loss.

Fig. 6 shows examples of end-to-end application level delays for web pages for diverse link loads (45% and 93%). These are delays representing the session times for HTTP, i.e., the time elapsed between the initiation by a user of a web page request and the moment when the result arrives including the main page and all the embedded references within the web page. The traces show the delays where the CIR value for the PVC is set to 512kbps and the port buffers are dimensioned at 256k bits. The Committed Burst Size (CBS) and the Excess Burst Size (EBS) for the PVC are set at 512 and 256 kbits respectively. The Frame Relay Access Devices (FRADs) and the switches have a processing capacity of 50000 packets/sec. There is no segmentation or reassembly activity at any of the devices. Thus the switching component introduces a delay of 20 $\mu$seconds [3]. Propagation delays are set to zero and the transmission delays are determined by the link speeds. Given the above-mentioned parameters of the simulation model, it is noticed that the queueing activity is the dominant contributor to the overall end-to-end packet delay. Notice that the variation in delay increases as the load level is increased.

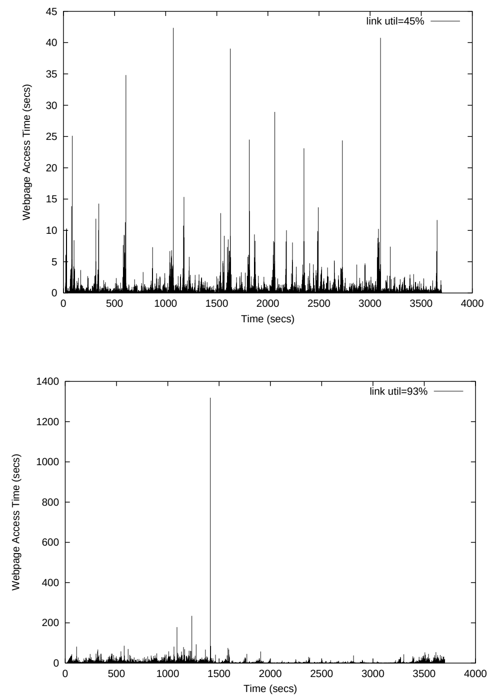


Figure 6: *Web Page Access Delays at Different Link Loads*

Fig. 7 shows the complementary cumulative distribution function (CCDF) of the above delays on a log-log plot. Notice the heavy-tailed nature of these delays. At the link utilization of 45% the probability of HTTP session delay exceeding 1 second is 0.1. When the link utilization grows to 93%, the session delays exceed 10 seconds with a 0.1 probability. This is an order of magnitude difference. The task of analytical computation and validation of the end-to-end delay distribution for document retrieval appears to be extremely difficult. At a first glance, this involves mixing of negative binomial distribution (number of embedded documents in a web page) with convolutions of Weibull distributions (end-to-end packet delays). This is an issue for future work.

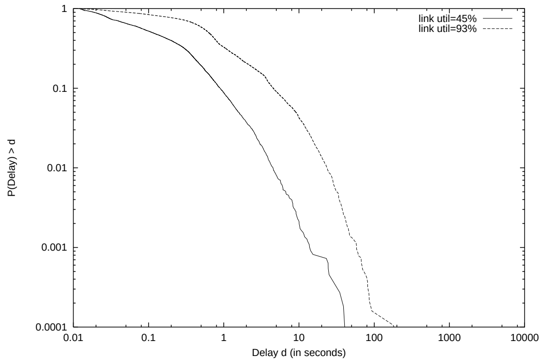


Figure 7: *CCDF of Web Page Access Delays*

# 5 Summary

A framework model has been reported for the study of end-to-end performance in a TCP/IP stack environment. A client-server model has been developed in a simulation model that closely mimics real-life interactions between the clients and servers in an Internet. The client and the server software reside on hosts connected by an Internet cloud. The hosts are running applications like FTP, SMTP and HTTP. The OPNET simulation tool was used as a simulation environment.

Using specific data traffic collected during a busy hour from a production Frame Relay network, key statistics have been calculated for diverse applications (e.g., number of embedded documents in a HTTP session, total number of bytes accessed from various servers on the Internet during a busy hour, data connections for FTP file transfers). These statistics have been used to model the application traffic according to the specific application.

The impact of TCP/IP flow and congestion control on application source parameters has been studied. It has been observed that the Hurst parameter is a robust metric that is not modified by network impairements or control algorithms. On the other hand, the peakedness parameter was shown to be dependent upon network load. Using these observations accurate engineering of trunks and network resources can be done to obtain end-to-end Quality of Service. Further work will be conducted to develop stable algorithms which can be used to dimension and engineer Internet trunks and access links.

# References

[1] Devadhar D., Pruthi P. and Gerla M. *Delay in Public Frame Relay Networks: Measurements, Modeling and Simulation*, 11th ITC Specialist Seminar: Multimedia and Nomadic Communications, Yokohama, Japan, October 1998.

[2] Feldmann A., Gilbert A.C. and Willinger W. *Data networks as cascades: Investigating the multifractal nature of Internet WAN traffic*, Proceedings of SIGCOMM98, Vancouver, BC, August 1998.

[3] Jena A.K., Pruthi P. and Popescu A. *Resource Engineering for Internet Applications*, in preparation, to be submitted to the IFIP ATM '99 Conference, Antwerp, Belgium.

[4] Norros I. *A storage model with self-similar input*, Queueing Systems Theory and Applications, Vol. 16, Iss. 3-4, 1994.

[5] Mil3 Inc. *OPNET Modeler version 5.1*, http://www.mil3.com, Mil3 Inc. Washington DC, USA.

[6] Pruthi P., Jena A.K. and Popescu A. *HTTP Interactions with TCP*, 11th ITC Specialist Seminar: Multimedia and Nomadic Communications, Yokohama, Japan, October 1998.

[7] Willinger W., Taqqu M.S., Sherman R. and Wilson D.V. *Self-similarity through High Variability: Statistical Analysis of Ethernet LAN Traffic at the Source Level*, IEEE/ACM Transactions on Networking, Vol. 5, No. 1, February 1997.