End-User Performance of WWW Services

Ajit K Jena and Adrian Popescu
Department of Telecommunications and Signal Processing,
Blekinge Institute of Technology,
371 79 Karlskrona, Sweden.
E-mail: akj@its.hk-r.se, adrian@itm.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 models experienced in the networks. Each application has a set of unique characteristics in terms of the way it performs its transactions as well as the way transaction processing profile maps onto unique network resource requirements. In order to support Internet applications effectively, it is therefore important to understand and to characterize the application level transactions and also to investigate their scaling properties. Recent advances in high resolution traffic monitoring and analyzing capabilities have enabled us to build up realistic models for the TCP/IP protocol stack with diverse network applications. In this paper we report investigations of classical applications such as FTP, SMTP, and HTTP to evaluate end-to-end performance requirements and accordingly to assess end-user performance like Service Level Agreement (SLA) for WWW. 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

The high variability of user/session characteristics heavily impacts on the end-to-end performance. Recent research has shown that the Long Range Dependence (LRD) behavior observed in the network traffic is mainly caused by application characteristics [1]. One of the main consequences is that of a fundamental bearing on the user perceived quality of the applications. Furthermore, it has been also observed that the self-similar scaling behavior over large time scales has little to do with network-specific controls. On the contrary, (feedback-based) network protocols (like TCP) seem to cause a much more complex scaling behavior, with richer dynamics over small time scales, and which is called multifractal scaling [3].

Such a complex traffic behavior poses tremendous difficulties when dimensioning diverse Internet resources for QoS guarantees. Also, the picture becomes even more complicated when considering the multi-dimensional nature of the Internet resources, i.e., link/network resources, transport resources (TCP/UDP) and application resources.

While the answers to these questions are not easy to find, good insight can be obtained via simulation and analytical studies. It is therefore the purpose of this paper to report results in developing models for facilitating efficient resource allocation while providing appropriate QoS guarantees to traffic streams under conditions of self-similar traffic behavior. Main questions are related to understanding the impacts the variability over large time scales has on Internet resources as well as to engineering of these resources (e.g., buffers and bandwidth in the case of link engineering, buffers and timers in the case of transport engineering and buffers and transaction models in the case of applications). Reference parameters that can be considered for the end-to-end QoS are throughput rate and delay at the application level. Based on these experiments, suitable SLA levels can be defined for WWW services.

2 Performance Model

The goal is to study Client-Server model driven with Internet applications like SMTP, FTP and HTTP. This involves understanding and simulation of application sessions and their constituent transactions. Towards this end, an object-oriented view has been developed into a client-server simulation framework. The model closely mimics the real-life events that occur in an Internet consisting of clients and servers. The client and the server nodes running the above-mentioned applications are connected by an Internet cloud.

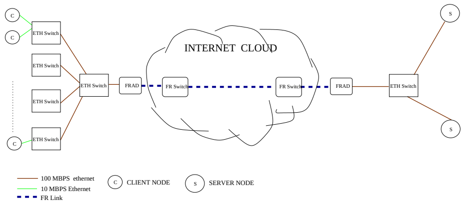

Figure 1: The simulated network architecture

2.1 Simulation Tool

The OPNET simulation tool is used as a simulation environment [6]. OPNET is a discrete event simulation package. It provides a layered framework to build models for communication networks. The layer hierarchy consists of network layer, node layer, and process layer. Various objects exist at each layer. When building objects, a given layer may use: 1) pre-defined objects at the current layer; and 2) objects defined at a layer immediately below itself. 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. For example, 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. The processes resident within a given node communicate with each other using ideal communication paths called packet streams. For example, client and server nodes consist of application process, 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. A typical example of the network layer definition would be an Ethernet or a Token ring network built by interconnecting several hosts. This layer defines therefore the interconnection strategies and the topological aspects of the communication network.

2.2 Network Architecture

The aim of the simulation work is to model real life environments as precise as possible. To that end, a network architecture is used which is common in typical business and academic organizations and in that sense it represents the state of the art (fig. 1).

The desktop access to the network is still via Ethernet. Though ATM to the desktop with access rate of 25 Mbps has been sometimes considered in this case, there are currently no applications which can justify the cost associated. The other development is related to the ubiquitous deployment of 100 Mbps Ethernet to the desktop. Even here, there are no applications that need that kind of bandwidth but the cost is a lot cheaper compared to ATM interface cards and the associated software. The bus oriented Ethernet has almost disappeared and is replaced by the twisted-pair equivalent. The aggregation points for the desktop machines are the hubs or switches. After multiple levels of aggregation, the ingress and egress points to the global Internet is via routers or switches which feeds into an ISDN, Frame Relay or ATM WAN link. In the simulated environment, the clients and the servers are organized into two separate subnets. Each of the client nodes are 10BaseT Ethernet hosts. The servers are 100BaseT hosts. The traffic from the 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 (FR) WAN via their Frame Relay Access Devices (FRAD). The WAN infrastructure consists of two Frame Relay switches interconnected via a single T1 link.

The client and the server in this 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 time instants 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 (e.g., browsing a web page in HTTP) consists of a bunch of transactions (e.g., fetching the main page and consecutively the embedded documents in HTTP). The time gap between two consecutive transactions is usually very small (in the order of milliseconds) so that it 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 the 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" [9]), whereas the application traffic is usually modeled with a distribution which is a mixture of Uniform (or Lognormal) and Pareto distributions ("Noah effect") [1, 7].

The statistical inferences summarized from actual traffic flows are used to model the application layer parameters and events relevant to the specific application under consideration. The application layer process parameters (e.g., total number of bytes accessed from various servers on Internet during a busy hour, data connections for FTP file transfers, number of embedded documents in a HTTP session, etc.) have been derived from datasets collected using a combination of various data collection methods such as monitoring packet traces, processing application log files, and even directly downloading HTTP content pages themselves [4]. The traffic has been collected from many different environments such as commercial Frame Relay network [2], access networks at different universities, etc.

2.3 Implementation Details

While various standard implementations have been used as-is from the OPNET modeler, it is also important to point out that various aspects of the modules have been modified to suit the goals of the simulation experiment. Moreover, new modules have been developed such as: 1) client and server application layer modules, and 2) intermediate layer module (such as socket layer). In the following, a short description is given on the new modules developed as well as the standard OPNET modules where significant changes were incorporated to facilitate the objectives.

2.3.1 Client and Server Nodes

Fig. 2 shows the structure of the client and server nodes. The application layer on the client side is a representation of human activities such as accessing a particular type of application service (e.g., FTP, SMTP, or HTTP). At this level, the principal tasks are modeling the user session arrivals into the network, modeling the type and size characteristics of information objects, and transmitting them to the designated server node. The processes corresponding to the application (APP) process on the server side are SMTP, FTP, and HTTP server processes. The APP processes are shown using a shaded elliptical block to indicate that these processes are not explicitly present in the node structure. They only occur as child processes of the DISPATCHER process.

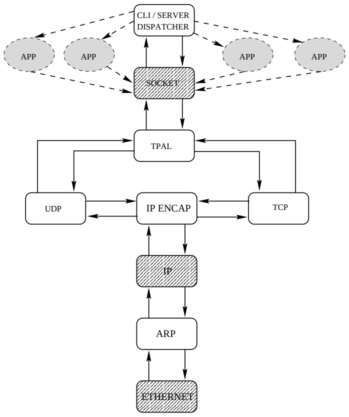

Figure 2: Layered process structure of the client and server nodes

The node structure is a typical example of layered communication protocol stack. The participating processes are interconnected by ***packet streams.*** The ***packet stream*** is a communication paradigm characterized by zero-loss and zero-delay flows. In the process hierarchy shown in fig 2 some processes are shaded. These are special processes known as ***Queues*** in the OPNET enviroment. These are processes where queue management capabilities are incorporated. In the diagram, the SOCKET, IP, and the ETHERNET layer are instances of such processes. In the IP and ETHERNET processes, the use of ***Queues*** is easily accounted for. The IP process has to store the TCP segments and process them one at a time. Similarly, the ETHERNET layer has to process one frame after another. The processing involved in the SOCKET layer is discussed below. Further, the TPAL process is a standard OPNET module that abstracts away the underlying communication framework by presenting an uniform interface. Below the TPAL module there are several possibilities like TCP, UDP, ATM, FR, or X25 modules. For example, using this process, it is possible to directly build applications on top of ATM or FR without using TCP or UDP layers. In such environments, the application layer messages are directly transported by ATM or FR connections without being fragmented into IP packets. However, as indicated in the previous section, ETHERNET hosts are used as end nodes and therefore the TCP is utilized as the transport layer.

2.3.2 Intermediate Nodes

Data transfer service inside the network cloud is handled in the form of Permanent Virtual Cir-

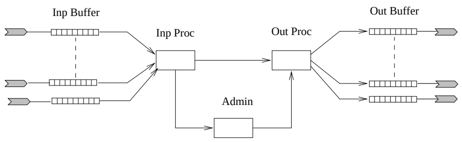

Figure 3: Model of a Frame Relay switch

cuits (PVC), with the associated parameters being Committed Information Rate (CIR), Committed Burst Size (B_c) and Excess Burst Size (B_e) [8]. By enforcing a suitable choice of these parameters, the burstiness of the resultant traffic flowing out on the PVC can be controlled. The ratio of B_c and CIR parameter values results in a quantity called T_c, the measurement interval. The PVC rate control mechanisms measures the traffic over intervals whose length are equal to T_c. If the total amount of traffic during that interval is less than or equal to B_c, then the packets are allowed into the network. If the traffic exceeds the B_c value but is less than B_e, then the packets are still allowed to enter the network but the ***discard eligibility bit*** is set in the frame header. Such traffic are liable for discard in case there is congestion inside the network. All traffic that exceed the B_e value are not allowed into the network and are dropped at the PVC entry point. The PVCs are setup between the FRADs of the communicating subnetworks. The routers of the connecting FRAD modules sense the PVC configurations and build their routing tables accordingly. The Ethernet packets originated by the client or the server traverse through the local switched LAN up to the router. The router then constructs Frames out of the Ethernet packets and sends them across the FR link on the appropriate PVC selected according to the destination IP host specified in the Ethernet packet. The OPNET implementation currently supports store-and-forward switching, with Congestion Avoidance, Notification, and Recovery functionalities provided in the switches. Figure 3 shows the Frame Relay switch model used in the simulation model.

The ***Inp Proc*** module has a certain processing rate specified by a simulation parameter. This is the raw speed at which the switching fabric processes the packets. At each of the frame processing epochs, the input module selects the oldest frame waiting at the input buffer. If the frame is an administrative frame, the frame is directed to the ***Admin*** module. The ***Admin*** module maintains the virtual circuit table and participates in the PVC establishments. Data frames are switched to the appropriate output port by consulting the virtual circuit table. The ***Out Proc*** module receives the switched frames and enqueues them at the specific output buffer stage. The output port transmits the frames at a rate determined by the rate of the physical link. As a result, the instantaneous PVC utilization may exceed 100%. The switch maintains threshold values and uses them to mark congestion and discard eligibility status of the frames.

By default, the transmitters in the OPNET switch and router nodes have infinite buffer size. As a design goal for the simulation experiments, the transmitters connected to the WAN links have an upper limit. The upper limit is specified in seconds. The transmitter buffers are dimensioned according to the specified time limit and the speed of the link they connect to. Using this mechanism, the packet delay at the transmitter can be controlled. The implementation of this scheme has been done using a statistic wire from the transmitter to the FRAD. The ***statistic wire*** generates an interrupt to the FRAD process each time its queue length changes. Both the leading and falling edges are monitored using this scheme. This enables the FRAD module to have up-to-date information about the queue length status of the transmitters. If a given packet satisfies the PVC control criteria but might cause overflow at the transmitter queue, it is dropped at the FRAD module.

In summary, the packets flowing through the switches or routers may get dropped either due to the incident traffic violating the FR flow control parameters for the established PVC, or to congestion conditions in the network, or due to limited buffer space at the WAN link transmitters.

3 End-to-End Delay Performance

One of the most important end-user performance of Internet applications is given by the 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 within 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 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.

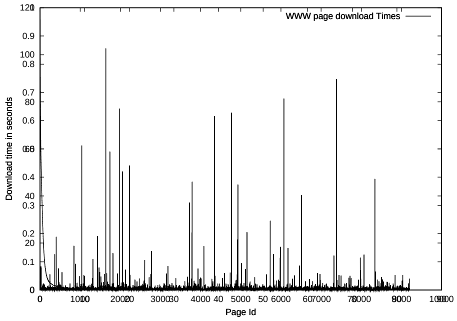

Figure 4: Download times for HTTP documents

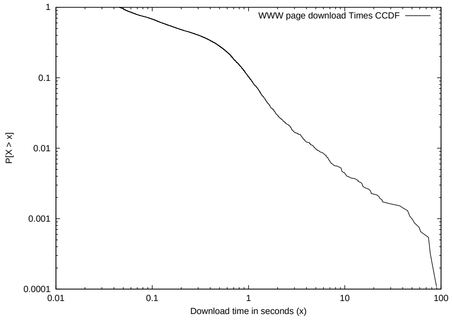

Figure 5: CCDF of the download times for HTTP documents

Fig 4 shows an example of end-to-end application level delays for web pages for a link load of about 30%, i.e., a mean rate of about 52 Kbytes/sec transmitted over a T1 link of 1.544 Mbps. Fig 5 shows the complementary cumulative distribution function (CCDF) of the web page access delays. 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 figure shows the high burstiness at the application layer object access delays seen by the end-users. There are clusters of clients and servers on both sides of the WAN. The total application load consists of a mix of HTTP and SMTP flows. The load levels can be controlled by varying the number of client hosts and the user session arrival rates. For the experiments reported, 75% of the clients were HTTP and 25% were SMTP. The object size distribution for the SMTP message sizes at the application layer is also heavy-tailed. The SMTP sources serve as LRD background traffic generators with the Pareto scaling parameter α being set to a higher value than that for HTTP.

Fig.6 shows the correlation that exists between the downloaded bytes and the application layer delays experienced by the end-user. When the total bytes downloaded is large, it is natural to expect longer turnaround times. The correlation between the byte count process and the page access times clearly shows the expected linear behaviour. However, given a class of web pages having a roughly equal size, the access times vary over almost an order of magnitude. This kind of variability is due to statistical multiplexing effects occuring at the link layer when all TCP connections are aggregated onto a single PVC.

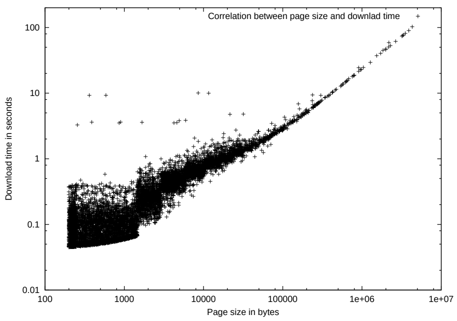

Figure 6: Relationship between the HTTP page sizes and download times

The statistical properties of application layer turnaround times experienced by the end-users occurs due to a combination of several factors: 1) heavy-tailed behaviour of application layer objects, 2) TCP windowing and retransmissions, and 3) controls at the link layer. The results reported here are for the experiments where link layer controls were purposely avoided. The experiments were performed with the PVC configuration set to ***Zero CIR***. This means that there were no resource reservations of any kind. Under this mode, all the packets are marked as discard eligible and thus the link layer is equivalent to a simple serial link. The port buffers at the routers and the switches were dimensioned such that packet losses were almost eliminated. Because of the absence of any cross traffic and the above mentioned port buffer setup, a packet, once accepted for transmission on the WAN, is never dropped. In other words, the autocorrelation structure at the packet layer is unaltered as the traffic passes through different switching stages. The link layer parameters m, a, $and H$ have been computed using the Periodogram method. The link utilization is extremely bursty and may stay close to the maximum capacity for time intervals spanning over tens of seconds.

Fig. 7 shows the CCDF of the queueing delays on a linear-log plot for traffic with Long Range Dependence (LRD) characteristics (H = 0.9) and for traffic with Short Range Dependence (SRD) characteristics (H = 0.5). The transmission and

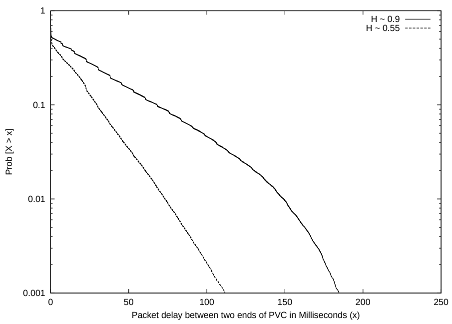

Figure 7: CCDF of packet delays between the two ends of the PVC

propagation delays have been taken out from the computation. In other words, the figure shows the interaction between the traffic arrival process from IP layer and the limited bandwidth constraints at the link layer. Notice the heavy-tailed nature of the LRD figure. The delay distribution indicates a Weibullian process for the end-to-end delay. On the contrary, the SRD figures shows exponential characteristics, as expected. It was observed that the major component of the delay on the end-to-end path occurs at the entry to the PVC and it is also the point where packet drops may occur due to port buffer overflow at the router. The routers and the switches have a processing capacity of 50000 packets/sec corresponding to a switching delay of 20 μseconds per switching stage. There is no segmentation or reassembly activity at any of the devices. 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.

4 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 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 applications like FTP, SMTP and HTTP. The OPNET simulation tool was used as a simulation environment.

Using specific data traffic collected from different environments such as universities and a commercial 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 effect of LRD properties on packet delays at the link layer and the object access time at the application layer has been studied. Using these observations accurate engineering of trunks and network resources can be done to obtain prescribed end-to-end QoS. Further work will be conducted to develop stable algorithms which can be used to dimension and engineer Internet trunks and access links.

References

[1] Crovella M.E. and Bestavros A., *Explaining Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes*, Proceedings of ACM SIGMETRICS '96, 1996.

[2] 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.

[3] 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.

[4] Jena A.K. *Modeling and Evaluation of Internet Applications*, Licenciate thesis, Lund Institute of Technology, Lund, Sweden.

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

[6] OPNET Inc. *OPNET Modeler version 6.0*, http://www.opnet.com, OPNET Inc. Washington DC, USA.

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

[8] Smith P., *Frame Relay Principles and Applications*, Addison-Wesley, 1994.

[9] 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.