# Improving the Performance of HTTP over High Bandwidth-Delay Product circuits

**A.J. McGregor**
**National Laboratory for Applied Network Research**
**San Diego Super Computer Center**
**10100 Hopkins Drive, San Diego, CA 92186-0505, USA**
**tonym@nlanar.net**

**M.W. Pearson and J.G. Cleary**
**Department of Computer Science**
**The University of Waikato,**
**Private Bag 3107, Hamilton, New Zealand**
**{mpearson, jcleary}@cs.waikato.ac.nz**

**Keywords**: TCP/IP performance, satellite, asymmetric delay

## ABSTRACT

As the WWW continues to grow, providing adequate bandwidth to countries remote from the geographic and topological center of the network, such as those in the Asia/Pacific, becomes more and more difficult. To meet the growing traffic needs of the Internet some Network Service Providers are deploying satellite connections. Through discrete event simulation of a real HTTP workload with differing international architectures this paper is able to give guidance on the architecture that should be deployed for long distance, high capacity Internet links.

We show that a significant increase in the time taken to fetch HTTP requests can be expected when traffic is moved from a long distance international terrestrial link to a satellite link. We then show several modifications to the network architecture that can be used to greatly improve the performance of a satellite link. These modifications include the use of an asymmetric satellite link, the multiplexing of multiple HTTP requests onto a single TCP connection and the use of HTTP1.1.

## INTRODUCTION

The growth of the Internet is placing strain on the worldwide telecommunications infrastructure. In particular it is no longer possible to purchase capacity on terrestrial cables to some parts of the world (for example between the Asia Pacific region and the US). The demand for network services continues to grow despite the lack of terrestrial capacity. To meet this demand Network Service Providers (NSPs) must move traffic to satellite circuits, despite the inability of standard window size TCP [1] implementations to perform well in high bandwidth-delay product environments [2].

When moving traffic to a satellite link the NSP must choose an architecture for the international component of the network. In the simplest case the terrestrial link can be replaced with a satellite link. In this case each of the HTTP requests is carried directly over the link in its own TCP connection. The additional latency of carrying traffic over a satellite link coupled with the fact that the TCP window limits the number of bytes that can be transmitted at once can lead to a significant increase in time to fetch a HTTP request. As most web pages are made up of a number of HTTP requests, the additional latency of fetching each request has a multiplicative effect in terms of how long it takes to fetch an entire web page.

Another feature of TCP that adversely affects its performance when used over a satellite is slow start. Slow start is a congestion control mechanism that ramps up the rate at which a TCP connection will send data. If the round-trip time (time taken to send a packet and receive an acknowledgement) is increased then the effective rate that TCP will ramp up the transmission of data is slowed down. If every request is sent in its own TCP connection then every request will have to overcome slow-start.

To reduce the page latencies there are a number of changes to protocols and architectures that can potentially increase the performance of the satellite links. These include:

- the use of asymmetric satellite links
- the use of persistent HTTP (HTTP 1.1)
- the use of proxy servers at each end of the satellite link

Other important techniques that are commonly used to improve the performance of TCP include big windows and fast retransmit are not considered in this study. This is because using them requires the end systems to deploy them and most NSPs do not have control over these machines.

To make the descriptions of components in this simulation study easier, they will be written in terms of an international connection between New Zealand and the United States. The results are, of course, more widely applicable.

### Asymmetric Satellite Link

Traffic volumes between the US and other countries are typically asymmetric. To minimize the additional latency imposed by a satellite circuit and also because of cost considerations, an attractive solution is to use a unidirectional satellite circuit. In this case the high volume inbound traffic is carried on a satellite circuit and the low volume outbound traffic on a terrestrial circuit. This causes asymmetric delays in the TCP connection, with a high bandwidth-delay product in one direction only. Such connections have not been widely studied. Allman et al [3] when discussing asymmetric satellite links, note that “This asymmetry may have an impact on TCP performance.”

### HTTP1.1

Another approach that can be used to improve TCP performance over the satellite link is to reduce the effects of slow start and the overhead associated with opening tcp connections. One possible way to achieve this is to use the persistent connection feature of HTTP1.1 [4] that retains TCP connections for a period of time after a request has finished being fetched. If a subsequent request is made to the same web server within this period of time the open connection is used again. While the first connection to a particular site has to overcome slow start, subsequent HTTP requests that use the same TCP connection do not. As most HTTP requests are for small files then this can led to a significant increase in performance. The disadvantage of using persistent connections is that more TCP connections each with memory requirements need to be open at any time.

### Proxy Servers

A third method for improving the performance of TCP over satellite links is to use a proxy server at each end of the satellite link. Rather than carrying each HTTP request in it own TCP connection between the proxies multiple requests are carried over the same connection. This improves slow start behavior as slow start only occurs once for each inter-proxy TCP connection. When an HTTP request is made slow start does not normally occur for the international part of the connection. It will still occur locally within NZ and the US. Further performance may be gained in the proxy-to-proxy case because the TCP stacks operating over the international link are under the control of the NSP and may be tuned. In particular a large buffer size may be selected. Finally the aggregation of several HTTP connections over a single TCP connection may allow TCP to better package the data and to carry more piggy-backed acknowledgments. Opposing these performance gains, performance may be limited by the number of TCP connections available between the proxies.

In this paper we describe a series of simulation studies that compares the performance of a terrestrial link and a satellite link between New Zealand and the US. The studies also show that with modifications to protocols and network architectures that it is possible to achieve a better performance than a terrestrial link without the same modifications. Of course most of the suggested modifications apply equally well to the terrestrial case as well. The simulations include a real TCP/IP protocol stack and are driven by a trace of HTTP activity collected from the NZ international exchange (NZIX).

The rest of this paper is organized as follows. First a description of the network under study is given. This is followed by descriptions of the workload and the design of the simulator. The results of the simulations are then given. The paper ends with the primary conclusions we draw from the results.

## SIMULATED NETWORK

Figure 1 shows the basic structure of the simulated network. In the simulations three sets of international links are modeled; terrestrial links in both directions, satellite links in both directions and a terrestrial link to the US with the return link back to NZ being a satellite link. For each of these configurations three different architectures are evaluated

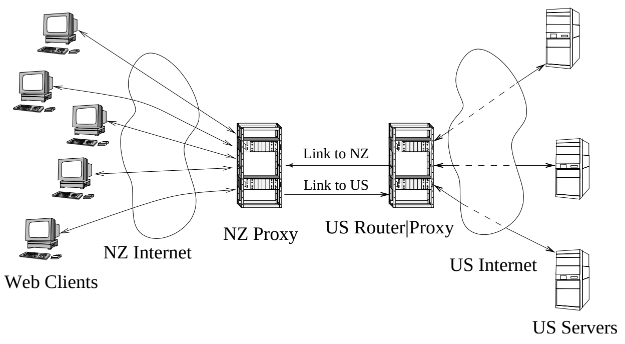

**Figure 1** Network Architecture for HTTP traffic

### Base Case

In the base case a router is placed at the US end of the link. For each web page that is fetched two TCP connections are involved. The first, from the web client to the NZ proxy. The second connection is from the NZ proxy to the US server providing the web page. In this case there is a TCP connection across the international link for each concurrent HTTP request.

### US and NZ proxies

In this case the router in the US is replaced with a proxy server. As in the base case the web clients located in NZ connect to a NZ proxy. However instead of directly connecting to the web server in the US the NZ proxy connects to a US based proxy that, then connects to US based web server. There are three TCP connections involved in fetching a web page. The first connects the web client to the NZ proxy, the second is between the two proxies and the third is from the US proxy to the US server.

Multiplexing is implemented between the proxies. That is the data for different replies may be interleaved on a single TCP connection between the proxies. The overhead of multiplexing is assumed to be, on average, 20 bytes per HTTP reply segment received from an HTTP server. It is expected multiplexing will improve the efficiency of the international link. Because TCP does not maintain the boundaries between application requests the data from (possibly different) HTTP reply packets may be repackaged for more efficient TCP transmission. In most cases the TCP segments will be the maximum segment size (MSS).

Because the connections between the proxies persist indefinitely the effect of TCP slow start is greatly reduced over the satellite component of the network.

### Persistent HTTP

The third case to be considered is the use of persistent HTTP that can be used in either of the two previous cases. In our study we have assumed a simple strategy that when a request is made a check is made to see if there is an idle connection to the same web server open. If so the idle connection is used to satisfy the new request otherwise a new connection is opened. If after a specified timeout period a subsequent request has not been made over a persistent TCP connection it will be closed.

When persistent HTTP is used in the base case then the persistent connections are opened from the NZ proxy to the US web server. However in the US and NZ proxy case persistent HTTP is only used between the US proxy and the web server as the TCP connections between the two proxy servers are kept open anyway.

The main parameters of the network are shown in Table 1. The values have been chosen to match real network parameters where possible. Real architectures would be not be as simple as the one described in this paper. Most will need more than a single proxy at each end of the international link to support the required load. The NZ proxy would almost certainly include a cache that satisfies some of the HTTP requests locally. There are many routers not shown, some of which are central to the international feed. Most real networks include equal terrestrial capacity in both directions, the satellite link serving to boost the incoming bandwidth. The simpler architecture used in this paper makes the simulation easier and allows the difference between the ways of using a satellite link of this type to be examined without interference from the full range of factors that would impact the performance of a real system.

## SIMULATED WORKLOAD

Most of the information required to generate the simulation input files (described in the next section) was gathered from HTTP log-files collected from the New Zealand Internet exchange (NZIX). The trace files used were collected from 3:00pm to 3:10pm in July 1997. There were, on average, 421 requests[1] per interval.

**Table 1** Main Network Parameters

| | |
|---|---|
| Satellite Bandwidth | 34.368Mbps (E3) |
| Satellite Delay | 320ms[5] |
| Terrestrial Bandwidth | 34.368Mbps (E3) |
| Terrestrial Delay | 60ms |
| Persistence Timeout | 15sec |
| US-NZ link buffer size | 256,000B |
| TCP buffer size: - Proxies<br>- Servers | 32767<br>as measured |
| MSS: - Between proxies<br>- Elsewhere | 1460<br>as measured |
| Delayed in US cloud | as measured |
| Delays in NZ cloud | not simulated |

To generate higher loads than that experienced when the trace files were collected, traces for the same time on successive days in July were integrated into a single trace. When higher still loads were required more than one copy of each trace was integrated into the log-file. Each copy was offset in time to minimize the effect of the artificial self correlation of the trace generated in this way.

The TCP MSS and server buffer sizes were not recorded in the HTTP traces we used. To discover these parameters a connection was established to each host while the network traffic was monitored using tcpdump. From the tcpdump output the MSS and window size was discovered for most hosts. Some hosts did not advertise their MSS. In this case the most common MSS (1480) was used.

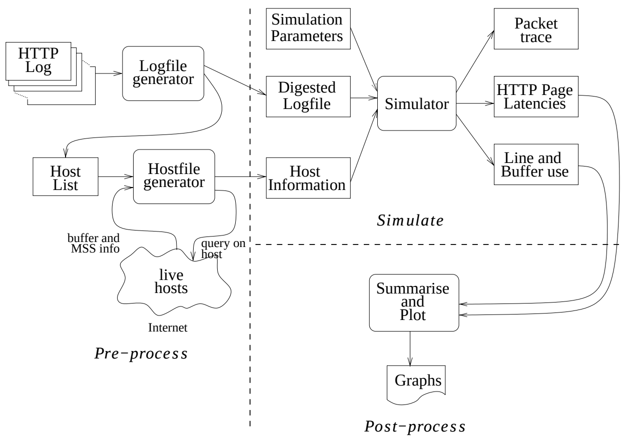

**Figure 2** Simulator Design

[1] The actual trace includes more requests. This number is the number of successful international requests that were not satisfied by the cache hierarchy

## SIMULATOR DESIGN

The simulation process as shown in Figure 2 can be considered as three interlinked processes, pre-processing, simulation and post-processing.

### Pre-Processing

In the preprocessing stage the input files for the simulator are prepared. These are:

- A “hostfile” containing an entry for each server accessed during the simulation. Each entry contains: a unique ID, the DNS name, the maximum window size and the TCP maximum segment size (MSS) for connections to the host.
- A “log-file” which contains an entry for each HTTP request. The entry contains the host ID for the server the request is fetched from, the size of the HTTP GET request, the size of the HTTP response and the time taken in the US component of the network.
- The simulation parameters including: the buffer sizes used by the proxies, the presence or absence of a US proxy, the number of connections between the proxies and the international link speed and delay in each direction.

### Post-Processing

Post processing is mostly a matter of collecting the results of interest from many simulation runs into a single set of plots. This was done with an array of perl scripts. GNU plot was used to draw the plots.

### Simulation

The simulator used in this study was based on the ATM-TN simulator[6] with modifications for this problem. The changes include replacing the ATM infrastructure with a simpler and more general bit serial interface.

The main two components used from ATM-TN are the conservative (as opposed to parallel) simulation engine and the TCP model. ATM-TN's TCP model includes the actual TCP code from 4.4 BSD Lite, modified to suit the simulation environment. Connections are simulated on a packet-by-packet basis and include slow start, congestion control, fast retransmit, and fast recovery algorithms [7].

The simulator design for the US-proxy case is shown in Figure 3. The simulator simulates the connections between the NZ proxy and the servers in the US. It does not include the NZ proxy to web client component of the network because this not significant to the study. Additional delays that are dependent on the type of connection (e.g. modem or direct connect) will be incurred in the NZ component of the real network.

The non-US-proxy case is similar to the US-proxy one with the omission of the US proxy and the replacement of the two TCP connection modules with a single TCP connection module and a single set of end-to-end TCP connections.

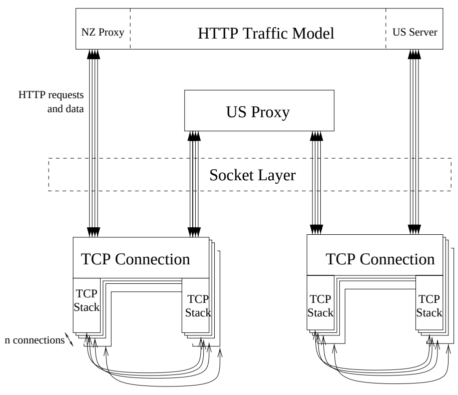

**Figure 3** Simulator Design

#### HTTP traffic model

The HTTP traffic model is responsible for creating TCP connections, sending HTTP GET requests, receiving the request at the destination and returning and results and for recording the time required to complete the HTTP requests. The HTTP model makes use of the hostfile and the log-file to control the simulation. The delays in the US part of the network are simulated by the HTTP traffic model which releases the packets that make up the HTTP response at a regulated rate so that the complete response arrives at the US proxy at the same mean rate as it did when the page was fetched on the real network.

#### TCP Connection

The TCP connection model simulates an end-to-end TCP connection and is based on a pair of TCP stacks, one for each end of the connection. It is assumed that the effect of errors is negligible.

#### US Proxy Model

The US proxy accepts HTTP GET requests across the TCP connections from the NZ proxy and forwards the request to the US server over a new or existing connection to that server. When the first packet of the reply arrives from the server a TCP connection is chosen to carry the HTTP reply to the NZ proxy. The connection with the smallest number of bytes awaiting transmission or transmitted but not acknowledged is chosen. As the packets of the reply arrive they are queued for transmission over the chosen TCP connection. Because multiplexing is used the data from different HTTP replies can be intermixed on a single TCP connection between the caches.

The simulation assumes that the proxy has sufficient CPU and memory to manage the workload and that the delay imposed by processing on the proxy not due to TCP queuing and transmission is negligible.

## RESULTS

Using the simulator 12 different simulation experiments have been carried out. The results of the simulations are presented in the following graphs. Each point on a graph represents a simulation run. The points have been joined with (straight) lines.

To interpret the labels that define each of the lines on the graphs (e.g. `noprxyST1.0_cxns`) a definition of there format is now given. Each of the labels is made up of three significant parts. The first part of the label (proxy or nopxy) defines whether the US proxy was present or not in the set of simulations. The is followed by a pair of letters (TT, ST, SS) that defines configuration of the international link (TT for terrestrial both ways, ST for terrestrial to US and Satellite to NZ and SS for satellite both ways). Following the pair of letters is a number (1.0 or 1.1.) that defines whether version 1.0 or 1.1 of HTTP was used in the set of simulations.

### Latency

Perhaps the most interesting result of the study is shown in Figure 4. The graph shows the average time required to fetch a set of sample HTTP requests that were present in all simulations.

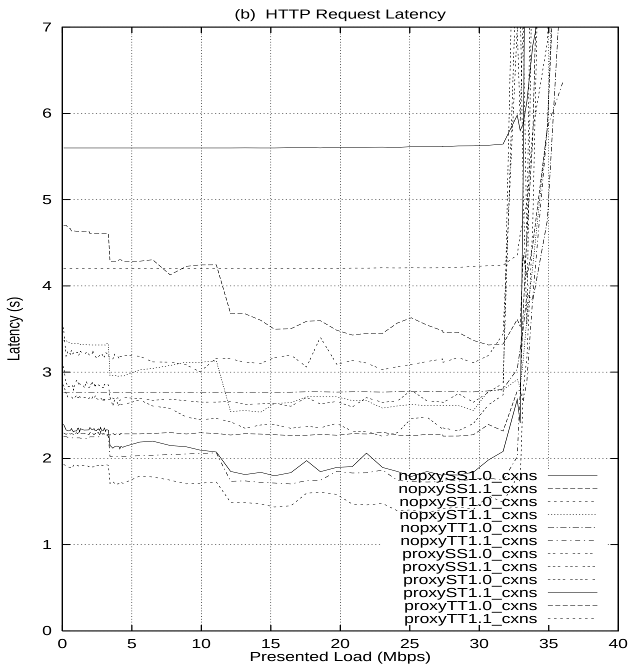

**Figure 4 HTTP Request Latency**

Starting with the bi-directional terrestrial case (noppxyTT1.0_cxns) it can be seen that it took on average 2.8 seconds to fetch one of the sample HTTP requests. If this link is replaced by a bi-directional satellite link (noppxySS1.0_cxns) it can be seen that the average time it took to fetch a HTTP request doubled to 5.8 seconds. This is directly the result of increasing the delay in the international link.

This graph also shows the improvements in performance that can be achieved when the suggested performance enhancements are used. The single enhancement that gives the biggest increase in performance is the addition of a proxy at the US end of the link (proxySS1.0_cxns) where the average time to fetch one of the sample HTTP requests is 55% of the base satellite case. This results from the reuse of the international TCP connections saving most of the cost of slow start and opening a TCP connection.

The use of persistent HTTP over the bidirectional satellite link reduces the average time to fetch one of the sample HTTP requests by between 18% and 40%. The result is not as good as the proxy case because TCP connections are only reused when there are multiple requests to the same web server within a specified time period as opposed to all the time which is the case of the proxy server. It is also worth noting the the average time to fetch one of the sample HTTP requests reduces as the presented load increases. This can be attributed to more reuse of the persistent TCP connections. We are currently trying to determine whether this is a real effect or an artifact from the method that we are using to generate the large trace files used as input to the simulator.

When an asymmetric satellite link is used the average time taken to fetch a sample HTTP request is 75% of the time that it took in the satellite base case. This can be attributed to a reduction in the bandwidth delay product.

The graph also shows that incorporating multiple enhancements leads to even larger reductions in the average time taken to fetch a sample HTTP request. For example when both a US-proxy and HTTP1.1 are used in the bi-directional satellite case a 55% improvement is observed as opposed to 50% US-proxy only case and a maximum of 41% for the HTTP1.1 case. Note also from the graph that the enhancements have similar effects in terms of increasing its performance.

Figure 5 shows the peak buffer space used in the routers that feed each the US end of the international link. From the graph it can be seen that the buffer usage never gets above 256K as this is the size of the buffer that has been used in the simulations. If the buffer is full any further packets that arrive at the link will be discarded.

Buffer (or link) usage is never heavy in the NZ to US direction in the simulation so the graphs have not been shown. (A real US/NZ link would be more heavily used in the US direction because of requests on NZ servers from US clients. These are not simulated here. We assume that sufficient NZ/US capacity exists to carry the client requests to the US.)

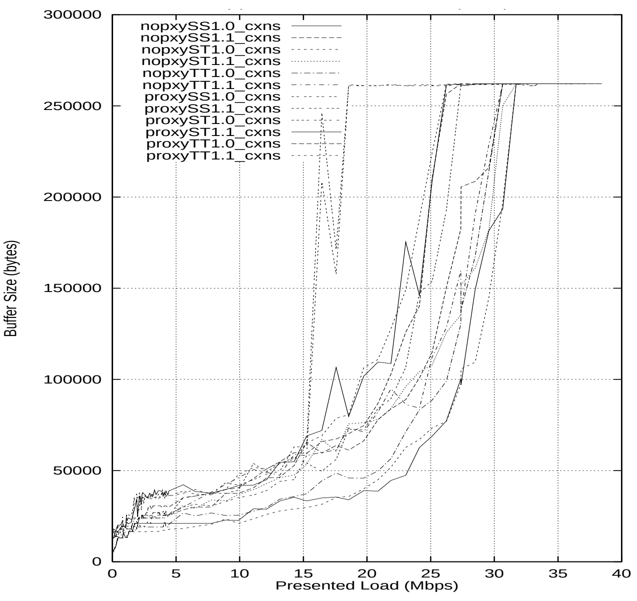

**Figure 5** Peak Router Queue Size (to NZ)

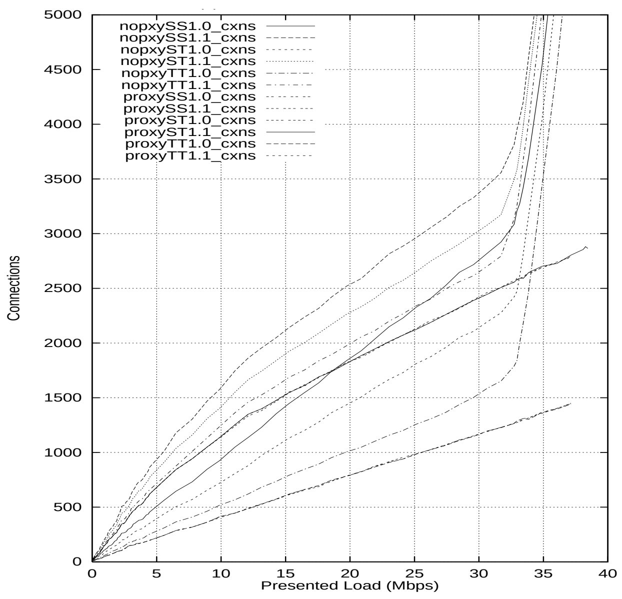

**Figure 6** Max Concurrent Sessions to Servers

Figure 6 shows the number of connections between the US proxy and servers for the US-proxy proxy configurations. In the no-US-proxy cases it shows the number of connections from the NZ proxy to US servers. In the latter case this increases rapidly when the international link is saturated because the HTTP requests take a long time to complete (see Figure 4). In general the no-US-proxy cases use more connections than their equivalent US-proxy cases because the connections take longer to complete. Also the HTTP1.1 cases use more connections than their equivalent HTTP1.0 cases because connections are not closed as soon as the request has been fetched meaning there can be many idle connections being maintained.

## CONCLUSIONS

The studies described in this paper show that NSPs need to give careful consideration to their network architectures if they plan to move traffic from terrestrial links to satellite links and they do not wish to significantly impact their users perceived performance. For example the results show the average length of time it take to fetch a HTTP request doubles if traffic is moved from a bi-directional terrestrial link to a bi-directional satellite link.

Three enhancements that can significantly improve the observed performance of a network containing a satellite were then described. These included the use of an asymmetric satellite link, the use of proxy servers at both ends of the satellite link and the use of HTTP1.1.

Simulations of these enhancements have shown that they all led to significant gains in the observed performance of a high bandwidth, long latency link such as a satellite link. Furthermore combining the enhancements leads to even greater gains in the performance of these links.

## REFERENCES

[1] J. Postel, ``Transmission control protocol," Tech. Rep. RFC793, DARPA, Sept. 1981.

[2] M. Allman, C. Hayes, S. Ostermann, and H. Kruse, ``TCP performance over satellite links," in Proceedings of the Fifth International Conference on Telecommunications Systems, Mar. 1997.

[3] M. Allman, D. R. Glover, and L. A. Sanchez, ``Enhancing TCP over satellite channels using standard mechanisms," Tech. Rep. draft-ietf-tcpsat-stand-mech-06, IETF Internet Draft, Sept. 1998.

[4] R.Fielding, J.Gettys, J.Mogul, H.Frystyk, and T.Berners-Lee, ``Hypertext transfer protocol - http1.1," Tech. Rep. RFC2068, IETF, Jan. 1997.

[5] W. L. Morgan and G. D. Gordo, Communications satellite handbook, Wiley, New York, 1989.

[6] M. Arlitt, Y. Chen, R. Gurski, and C. Williamson, ``Traffic modeling in the ATM-TN TeleSim project: Design, implementation, and performance evaluation," in Proceedings of the 1995 Summer Computer Simulation Conference, (Ottawa, Ontario), July 1995.

[7] W. Stevens, ``TCP slow start, congestion avoidance, fast retransmit, and fast recovery algorithms," Tech. Rep. RFC2001, IETF, Jan. 1997.

[8] V. Jacobson, R. Braden, and D. Borman, ``TCP extensions for high performance," Tech. Rep. RFC1323, IETF, May 1992.