

# Sensitivity Analysis of Event Driven Simulation Results

M.W. Pearson  
The University of Waikato,  
Private Bag 3107, Hamilton,  
New Zealand  
mpearson@cs.waikato.ac.nz

A.J. McGregor  
National Laboratory for Applied Network  
Research  
(NLANR), San Diego Super Computer Center,  
10100 Hopkins Drive, San Diego, CA 92186-0505,  
USA  
tonym@cs.waikato.ac.nz

*Abstract*—Simulations have been used to study a wide variety of network problems. Traces collected using passive and active measurement techniques are a common source of data for simulations. When a simulation is constructed some details of the real system, that the designers believe have little impact on the results, are omitted. This is done to reduce complexity and to make the simulations more manageable. Whether the omitted detail does indeed have a significant effect on the results is not always obvious. One approach to estimating the impact of the omitted components is to undertake a sensitivity study. This paper describes a sensitivity analysis of a simulation designed to study heavily used international Internet links. The parameters studied include network structure, traffic load generation and protocol modelling. The results show that in most cases the behaviour of links with composite traffic, made from many simultaneous TCP connections, are not sensitive to variation of these parameters. The results of these simulation studies are to be used to drive our measurement program to collect data for future simulation studies that we plan to carry out.

*Keywords*— TCP/IP performance, trace-driven simulation, sensitivity analysis

## I. INTRODUCTION

One use of passive measurement is to provide traces that can be used as the input to event driven simulation. Such simulations have been used to study a wide range of network problems including strategies to achieve good performance over high bandwidth delay satellite[1], terrestrial[2] and asymmetric satellite/terrestrial[3] international links, without requiring users to upgrade or tune their TCP stacks.

Simulations, by their nature, are approximations. Some details of the real system are intentionally omitted to make the simulation more manageable. For example it is possible to drive a simulation from a trace of high level protocol events (e.g. HTTP requests) or from a packet level trace. In the case of TCP, packet control level behaviours (such as slow start) are important. They are not always important with UDP based traffic such as a game or mpeg replay. It is part of the art of simulation to omit details that are difficult to provide but make little difference to the final result while maintaining the essential elements of the situ-

ation being modelled. In trace-driven simulation we want to omit details that are difficult to collect. This might be either because they are not readily available in passive traces or because they require a lot of resources to collect, store and simulate. Unfortunately it is not always obvious which aspects of a simulation can be omitted without unduly affecting the result.

Over the last couple of years we have been studying the performance of TCP/IP traffic on high-performance links; in particular international links between New Zealand and the US as shown in Figure 2. The simulator developed for these studies is based on the ATM-TN simulator[4]. The simulator contains a TCP model that includes the actual TCP code from 4.4 BSD Lite, modified to suit the simulation environment. Each of the TCP connections are simulated on a packet by packet basis and include the modelling of slow start, congestion control, fast retransmit, and fast recovery algorithms.[5]

The simulated network can be considered as a number of connected components. These are the HTTP requesters (the Web clients in the New Zealand Internet), the NZ proxy, the international link, including the routers that feed it (which is the key component under study), the US network and the HTTP servers.

An HTTP traffic model is responsible for creating TCP connections, sending HTTP GET requests, receiving the request at the destination and returning the results, and for recording the time required to complete the HTTP requests. The HTTP model makes use of information about hosts and a URL trace collected from a real network using passive measurement.

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 applied when the page was fetched over the real network.

When developing the simulation a number of assumptions about aspects of the network structure that could be omitted without adversely affecting the simulation results were made. In this paper we examine the sensitivity of the simulation results to some of these parameters. The following list outlines the different cases that are considered:

This work is funded, in part, by NSF Cooperative Agreement No. ANI-9807479. The U.S. government has certain rights in this material.

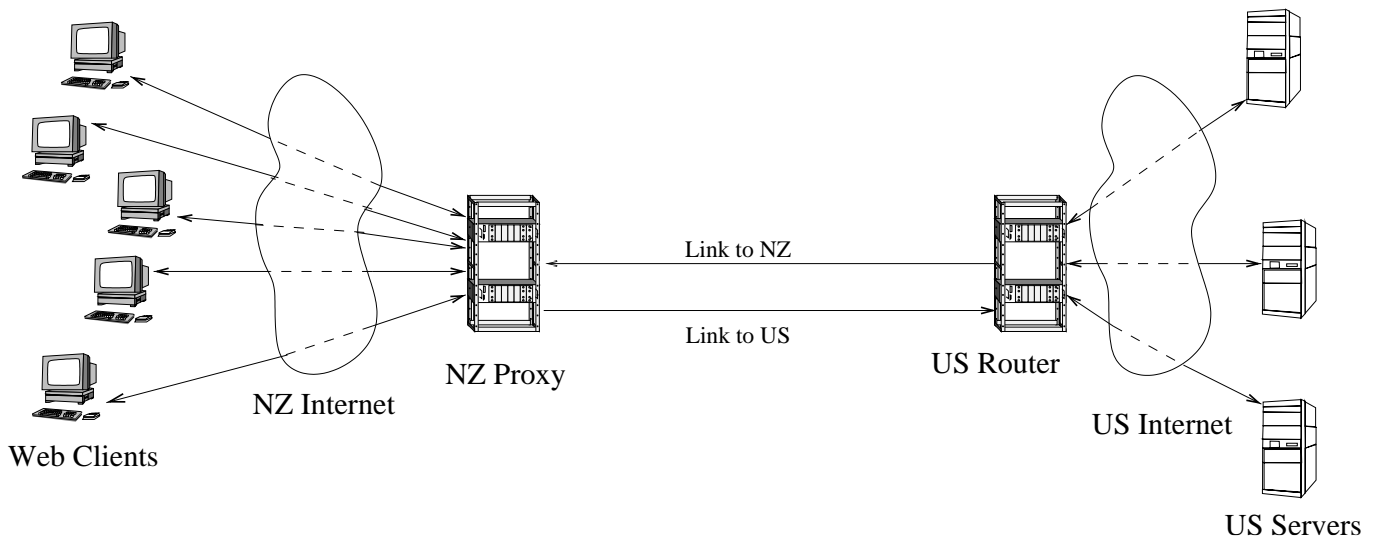


Fig. 1. Network Architecture for HTTP traffic

- The effect of overlaying HTTP trace files to generate sufficiently high workloads
- The importance of modelling TCP Connection establishment and termination
- The importance of modelling TCP mechanisms such as slow start, fast recovery and fast retransmit
- The importance of using the same maximum segment sizes (MSS) and TCP window sizes as were used by each of the TCP connections.
- The importance of modelling of Feeder Networks (i.e. The US Internet)
- The impact of modelling persistent HTTP.

In addition to these simplifications there are a number of other simplifications that we do not consider. The most important of these are:

- Most real networks 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.

The goal of this work is to guide the selection of parameters to be included in a measurement program that will feed event driven simulation.

To make the descriptions in the paper simpler the components of the simulated network are described in terms of an international connection between New Zealand and the United States. The results are, of course, more widely applicable.

The rest of this paper is organised as follows. Section II describes the workload including its main characteristics and how heavier workloads were formed to simulate high loads on the links. The simulator design is explained in section III and the sensitivity studies are described in section IV. The results of the simulation runs are shown in section V. The paper ends with the primary conclusions we draw from the results.

## II. 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<sup>1</sup> per interval.

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 minimise the effect of the artificial self correlation of the trace generated in this way.

The TCP MSS (maximum segment size) 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.

The methods for generating workloads and determining parameters such as the MSS are some of the design features that we study in this paper. They will be more fully discussed in section 5.

## III. SIMULATOR DESIGN

The simulation process is shown in figure 1. It can be considered as three interlinked processes, pre-processing, simulation and post-processing.

<sup>1</sup>The actual trace includes more requests. This number is the number of successful international requests that were not satisfied by the cache hierarchy

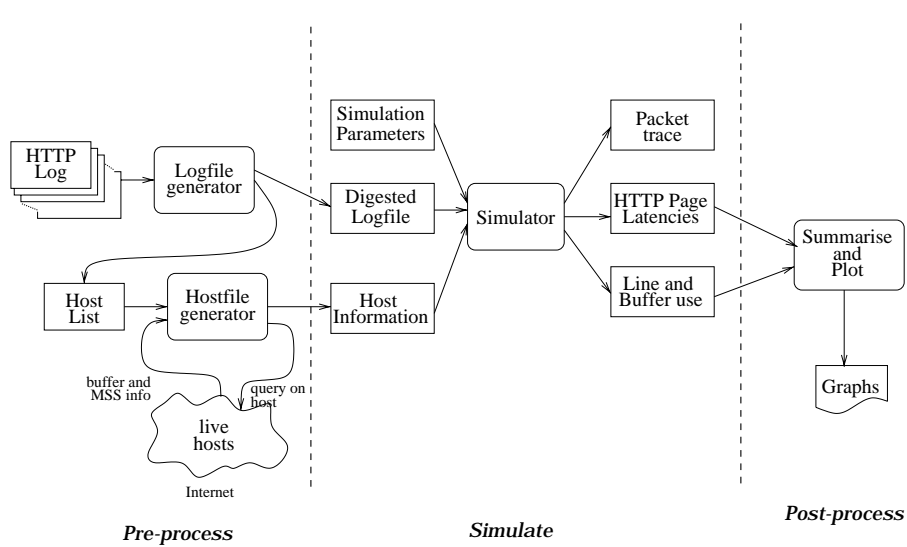


Fig. 2. Simulator Design

### Pre-Processing

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

- A “hostfile” which contains an entry for each server accessed during the simulation. The entries contain: a unique ID for each host, the DNS name of the host, the maximum window size for the host 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 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[4] 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 algorithm[5]

The simulator design is shown in figure 3. The simulator simulates the connections between the NZ proxy and

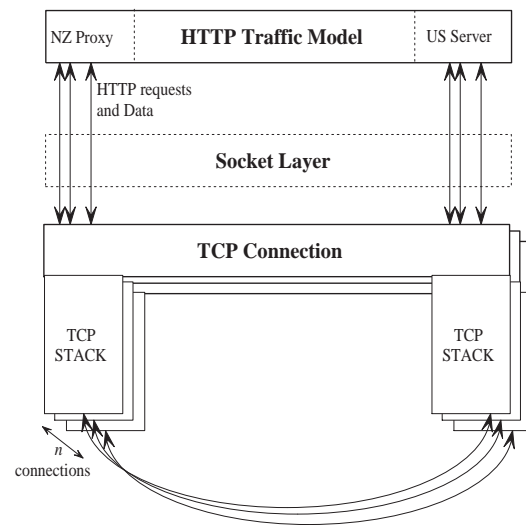


Fig. 3. Simulator Design

the servers in the US. It does not include the web client to NZ proxy to component of the network because this has not been significant to the studies carried out in the past. 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.

### 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 are based on a pair of TCP stacks, one for each end of the connection. It is assumed that the effect of errors is negligible.

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.

### IV. SENSITIVITIES

This section describes our experiments to determine the sensitivity of the results of the simulation to variations in key aspects of the simulator design. We have run a set of simulations for each aspect of the design that we are interested in. The following sub-sections describe each of the simulation studies that have been performed.

#### *Connection Establishment*

The original TCP model contained within the ATM-TN simulator did not model the opening and closing of TCP connections. Because the effect of the open and close sections of a TCP connection seemed important to our results we extended the simulator to include them. To undertake the sensitivity study switches have been incorporated into the TCP stack code to allow the modelling of connection establishment and termination to be turned on or off. A series of simulations were run for each of the combinations (no open or close, close and no open, open and no close, and both open and close).

#### *Overlaying Workloads*

In many communications contexts, burstiness exists at all layers of a traffic hierarchy. This means it is not always valid to add together multiple copies of the same workload to generate a higher load. However, in most of the simulation studies performed using the simulator multiple copies of the same trace file have been overlaid (as described in section II) so that sufficiently high traffic loads could be generated to drive the simulations. To assess the impact of overlaying trace files in this way a set of simulations that vary the total number of trace files used (before they have to be overlaid) to construct the traffic loads have been performed. We compare the results of simulations where the same overall workload is constructed from many overlaid copies of just a few trace files with simulations where fewer overlaid copies of more files are used.

#### *Modelling of Feeder Networks*

A great deal of complexity is contained in the structure and performance of the network that collects data to be carried over the link under study. In the simulation studies that we have performed the feeder networks that provide traffic for the link under study is the US Internet.

It is infeasible to model the whole of the US Internet so the simulator uses a simplification of the network. Packets from the servers, that make up the HTTP reply, are delivered to the US link at a rate that makes the page complete in the same time as it took when the trace was collected if the simulation parameters match those of the real system from which the trace was collected. This is achieved by calculating an effective bit rate for delivery of the packets. This in turn was discovered by undertaking tuning simulations, with the simulation parameters set to those of the real network, to discover this bit rate for each request. Successive approximation was used because varying the value for one page affects others that share the link with it.

Note that this bit rate is nominally and is applied to each packet from a server individually. In essence it is intended to provide a spacing of packets so that they arrive at the US link at appropriate times.

We study the sensitivity to the design of the feeder network by adding two switches. The first allows us to remove the feeder network from the simulation completely, the second allows us to replace it with a constant bit rate.

#### *MSS*

While it is well known that MSS has a significant affect on the performance of an individual TCP connection it is less clear how it affects the overall aggregate performance of a link that is carrying many TCP connections. In previous simulation studies we attempted to set the MSS for each connections to the same value used to satisfy the original measured HTTP request. This required a set of active measurements to be made to determine the MSS used by each of the hosts referenced in the trace files as these values were not contained in the original HTTP traces. Unfortunately, this process is error prone. It assumes that hosts returned the same MSS when we connected to them that the used when the page was originally fetched. Some machines were no longer available, or did not advertise MSS in their TCP connection packet so it was not possible to determine all of the MSS values in this way. As a consequence, it was necessary to estimate some of the MSS values. The process of simulation could be greatly simplified if it was not necessary to accurately model the MSS used by each connection. This set of simulations compare the results of using the actual measured MSS values with using a fixed MSS size for all TCP connections.

#### *TCP mechanisms*

As TCP has developed additional mechanisms have been added to improve its performance, especially when loss and congestion are experienced. The simulator models slow start, fast recovery and fast retransmit but it is not clear how these affect the performance of a link with many TCP connections.

- All mechanisms turned
- Slow start only
- Slow start and fast retransmit only
- All mechanisms turned on

One of the results of previous simulation studies is that the use of persistent HTTP, as found in HTTP 1.1, has a big impact on the results of the simulation. In the context of the simulator, persistent HTTP is implemented as follows. 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 is 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.

Although all the aspects of the simulation design described above are independent of one another persistence has the potential to significantly affect the others. Consequently each of the simulation studies has been run with, and without, persistent HTTP between the NZ proxy and US Web servers.

## V. RESULTS

The results for each of the simulation studies described above are presented in this section. Graphs that show the performance of the US-NZ link under a range of loads are presented for each of the simulation studies. Different lines on the graph represent different configurations of the simulator design. Each of the points in a graph represents a simulation run.

In addition to the parameters under study there are a number of other network parameters that are used by the simulator. The main parameters and their values are shown in table I. The values have been chosen to match real network parameters where possible.

TABLE I  
MAIN NETWORK PARAMETERS

International Bandwidth	34.368Mbps (E3)
International Delay	60ms
Persistence Timeout	15sec
US-NZ link buffer size	256,000B
Default TCP buffer size	
Proxies	32767
Servers	as measured
Default Maximum Segment Size	as measured
Delayed in US cloud	as measured
Delays in NZ cloud	not simulated

### A. Connection Establishment and Termination

Figure 4 shows the average time (page latency) it took to fetch a fixed set of web pages for each of the simulation runs. The top set of lines are for the HTTP1.0 simulation runs while the bottom set show the results for the HTTP1.1 runs. In both sets a line is shown for each of the

combinations (no open or close, open and no close, close and no open and close and open) that were tested. From this graph it can be seen that in the case that open is not modelled (close-1.0\_have and close-1.1\_have) there is a slight improvement (0.2 seconds for HTTP1.0 and 0.1 seconds for HTTP1.1 under light loads) in the average page latency.

From these graphs it can also be seen that the modelling of the closing of connections has no impact on the average time taken to fetch a page whether it is turned on or off.

Figures 5 and 6 show the load carried and loss experienced on the US to NZ link. On close inspection it can be seen that the load carried is slightly higher when open and/or close are modelled due to the extra overhead associated with connection establishment and/or termination. The loss graph shows that loss is experienced on the line slightly sooner when open and/or close are modelled, again because of the extra overhead.

### B. The Effect of Overlaying Workloads

Figure 7 8 9 show the average page latency, load carried and loss experienced on the US-NZ link for both HTTP1.0 and HTTP1.1. From these graphs it can be seen that altering the number of unique trace files used to generate this load has very little impact on the simulation results proved that more than a few files are used. The one obvious anomaly occurs when only four files are used to generate the traffic load. There are two aspects of this study that we wish to study further, but time did not permit during the preparation of this paper. Firstly, we do not understand the shape of the  $n = 4$  line, particularly why it increases so much more slowly under high load than the other cases. Secondly, we are unsure why the simulation with only two files does not show any unusual behaviour.

### C. Distribution of Arrivals from Feeder Networks

Figures 10 and 11 show the page latency graphs for various configurations of the feeder network for HTTP1.0 and HTTP1.1 respectively. The graphs show that the lines are all essentially the same shape, only the offset changes. Likewise the lines on the load and loss graphs are almost identical. This indicates that simulation of the feeder network is less important than we imagined when we originally designed the simulator. However, notice that the difference between the no US cloud case (with no fixed bandwidth) and the with US cloud case is close but not the same for HTTP 1.1 and HTTP 1.0

### D. Maximum Segment Size (MSS)

Figures 14 and 15 show the results of simulations with various fixed MSS sizes and the original, per host, MSSs.

The results show that the average page latency increases as the size of the MSS increases. This is the opposite of what was expected as the page latency should decrease as the MSS increases and each packet is able to carry a larger payload. Closer inspection of these results has identified a weakness in the way the US cloud is modelled in simulator.

As described in section IV a delay is calculated for each packet that passes through the US cloud. This delay is intended to incorporate both the bandwidth and latency components of the real network. In essence the effect of this approach is a fixed latency and with infinite bandwidth. If two packets are sent from a host in the US cloud at the same time they will arrive at the US Router at the same time. This seems reasonable, but it has the side effect that contained in multiple small packets will travel through the US cloud much faster than if the same data were contained in a single larger packet. This means that HTTP requests get satisfied more quickly if small packets are used.

Once this problem was identified a second set of simulation runs were performed with the modelling of the US cloud turned off as shown in figures 16 through figure 19. These graphs exhibit the expected behaviour; that is as the size of the MSS increases, the average page latency decreases.

As a consequence of this result we plan to review the way the US cloud is modelled, with a view to introducing links between the servers and the international link that model both bandwidth and latency.

### E. Window Sizes

Figures 20 through 23 show the results of simulations with various fixed TCP window sizes and the the original, per host, Window sizes. These graph show the exhibit the expected behaviour; that is as the size of the window sizes is increased, the average page latency decreases.

### F. TCP Mechanisms

The results from the simulations that vary the TCP mechanisms (slow start, fast retransmit and fast recovery) used are shown in figures 24 through 27. These graphs show that the results of the simulations are very sensitive to these mechanisms. In particular they show that turning off slow start does have a major impact on both the average page latencies and the amount of loss experienced on the international link. Fast retransmit and fast recovery impact on the results in situations where load presented to the network is close to the capacity of the international link.

## VI. CONCLUSIONS

In most cases our results indicate that simulations of the behaviour of links with composite traffic, made from many simultaneous TCP connections, are not sensitive to variation of the parameters described above. The main exception is the TCP mechanism class which show that including slow start is very important. We are not surprised by that result, it is consistent with all the literature.

This study is specific to our simulator and modelled network. There are many possible event driven simulations and the needs many of these simulations vary. However, many simulations have elements in common with ours and, to that extent, we believe our results will be generally useful.

- [1] Pearson M. and McGregor A., "A simulation study of network architectures to support http traffic on symmetric high-bandwidth\*delay circuits," in *Proceedings of the Asia Pacific Web Conference*, 2000, pp. 19–25.
- [2] Pearson M. and McGregor A., "Reducing us/nz web page latencies," in *Networks '99*, Waikato University, Hamilton, New Zealand, Jan. 1999, pp. 54–63.
- [3] McGregor A.J., Pearson M.W., and Cleary J., "The effect of multiplexing http connections over asymmetric high bandwidth-delay product circuits," in *SPIE Conference on Routing in the Internet*, Boston, Massachusetts, USA, Nov. 1988, pp. 398–409.
- [4] M. Arlitt, Y. Chen, R. Gurski, and C.L. 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.
- [5] W. Stevens, "TCP slow start, congestion avoidance, fast retransmit, and fast recovery algorithms," Tech. Rep. RFC2001, IETF, Jan. 1997.

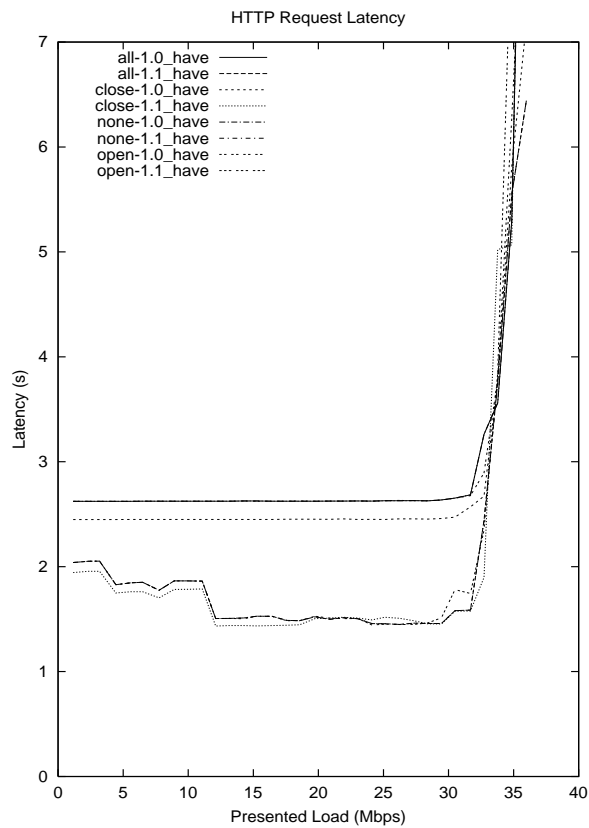


Fig. 4. Page Latencies for connection runs using both http1.0 and http1.1

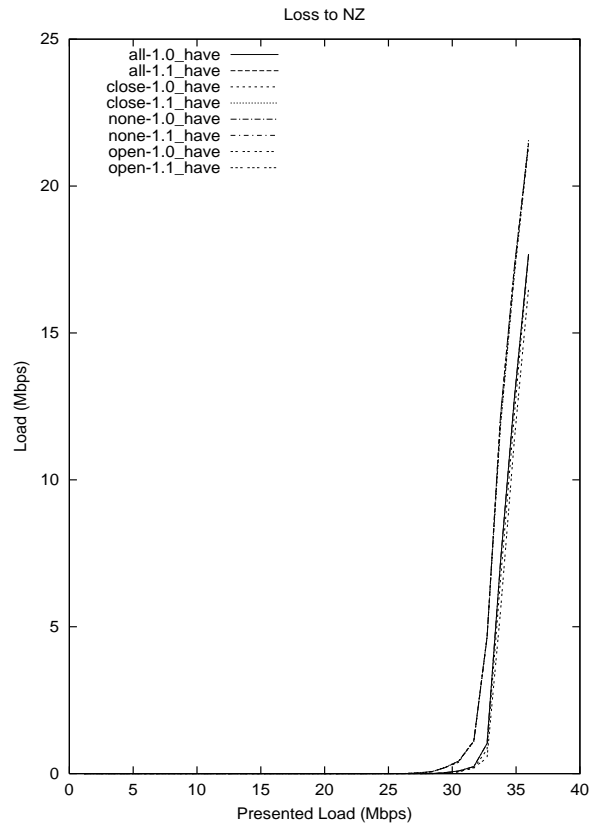


Fig. 6. Loss on US-NZ link for connection runs

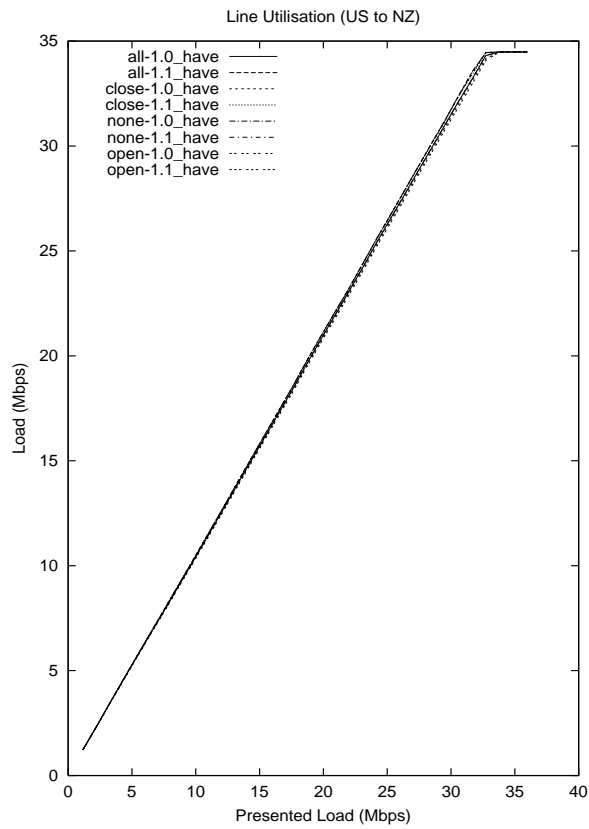


Fig. 5. Load on US-NZ link for connection runs

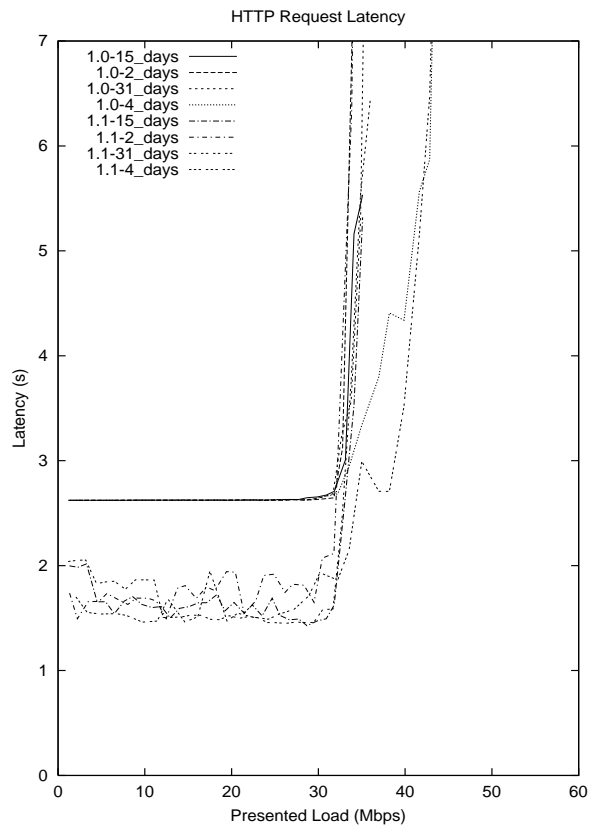


Fig. 7. Page Latencies for page overlay runs using HTTP1.0 and HTTP1.1

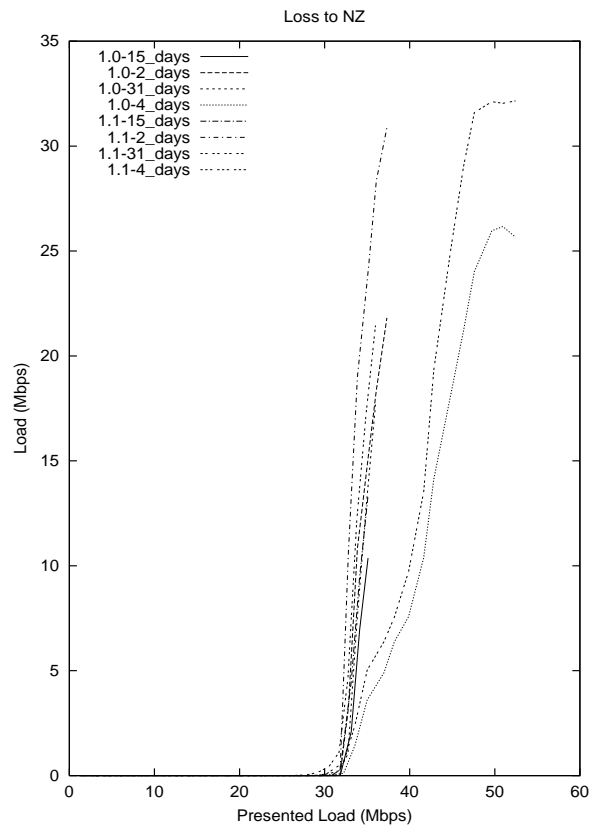


Fig. 9. Loss on US-NZ link for page overlay runs

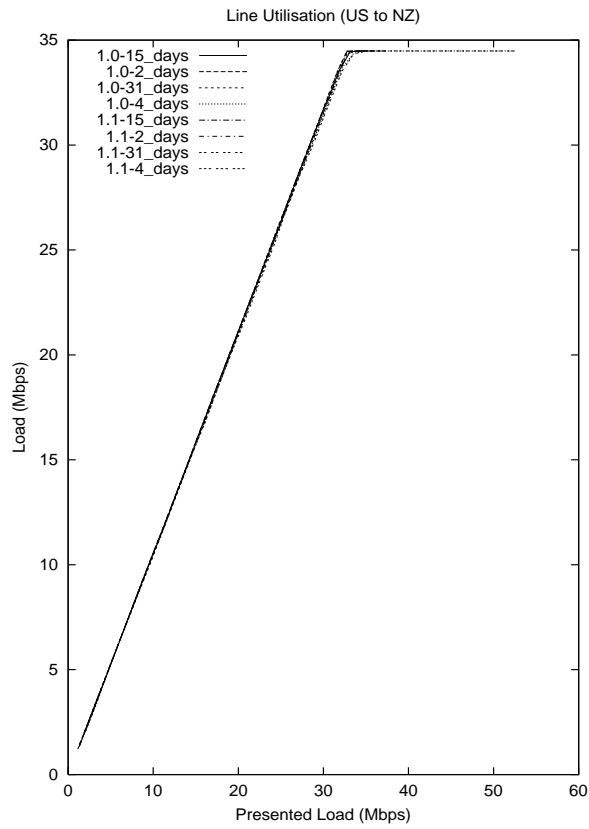


Fig. 8. Load on US-NZ link for page overlay runs

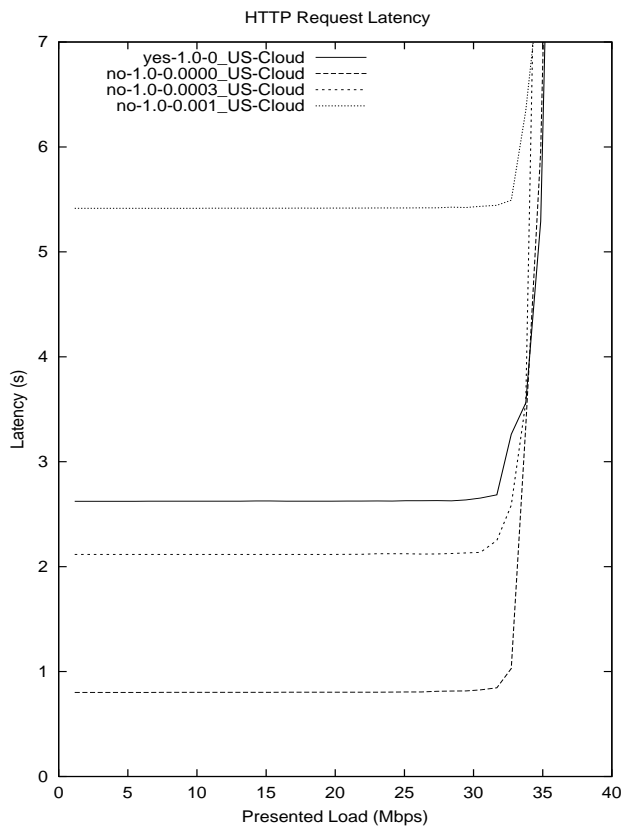


Fig. 10. Page Latencies for US-Cloud runs using http1.0

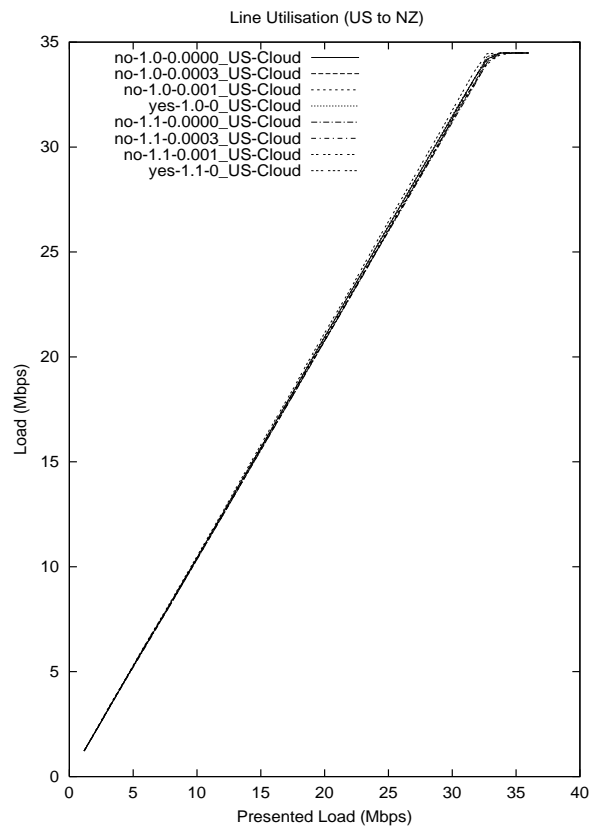


Fig. 12. Load on US-NZ link for US-Cloud runs

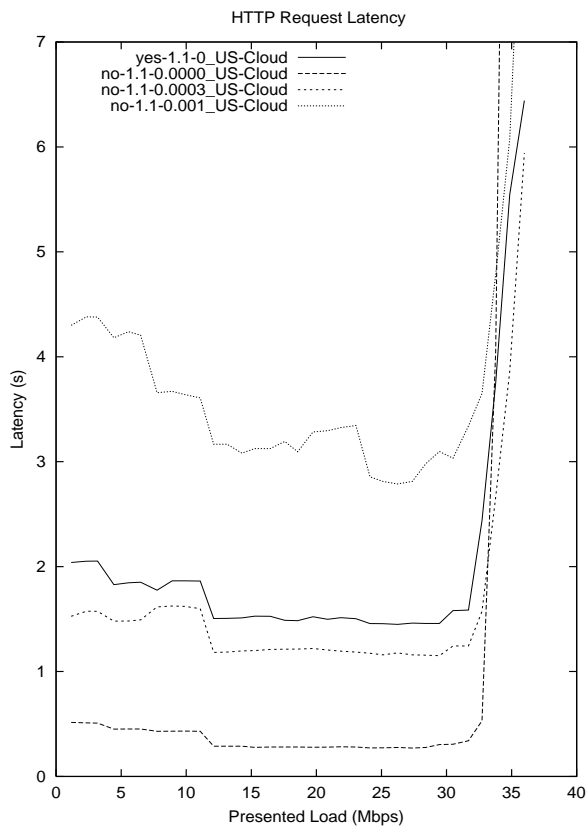


Fig. 11. Page Latencies for US-Cloud runs using http1.1

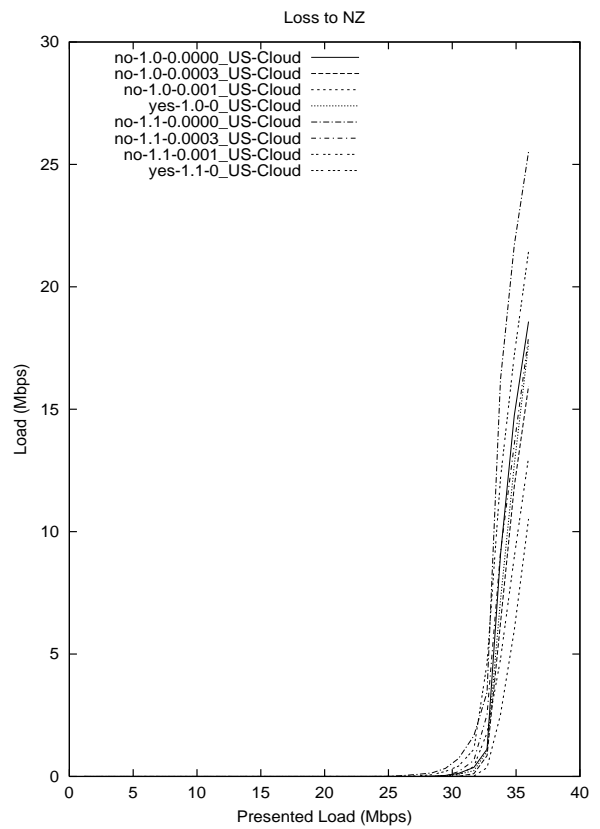


Fig. 13. Loss on US-NZ link for US-Cloud runs

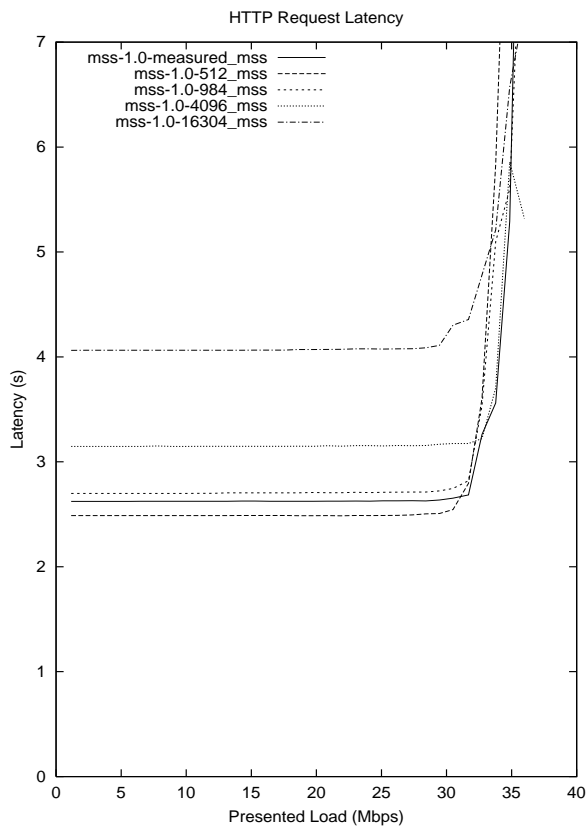


Fig. 14. Page Latencies for MSS runs using http1.0

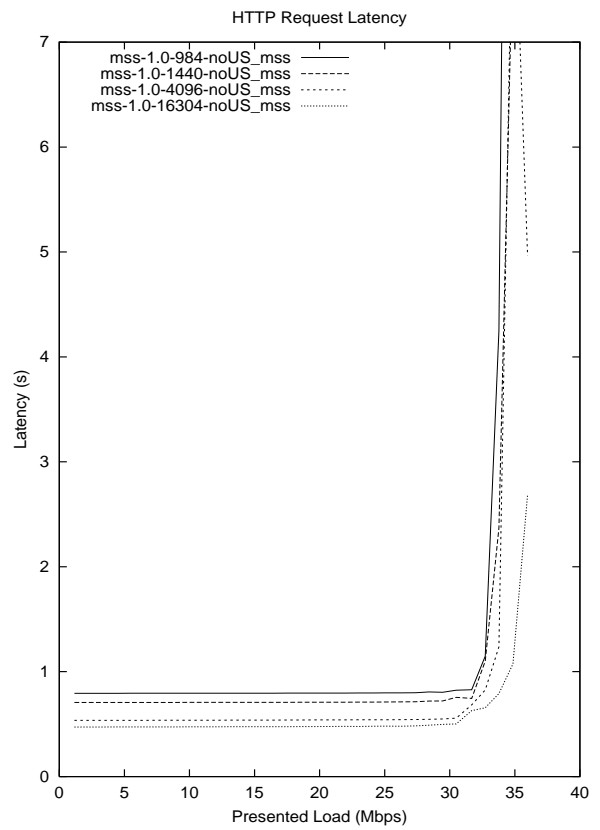


Fig. 16. Page Latencies for MSS runs using http1.0 and no US cloud

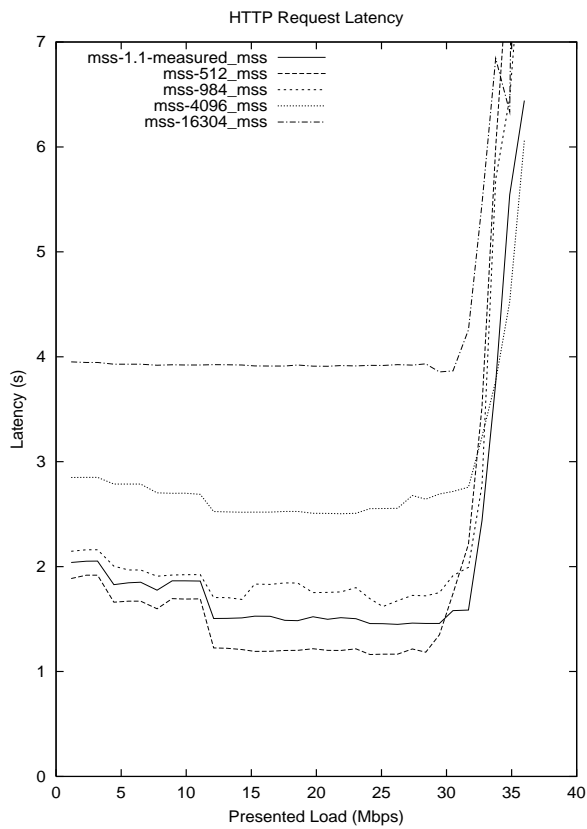


Fig. 15. Page Latencies for MSS runs using http1.1

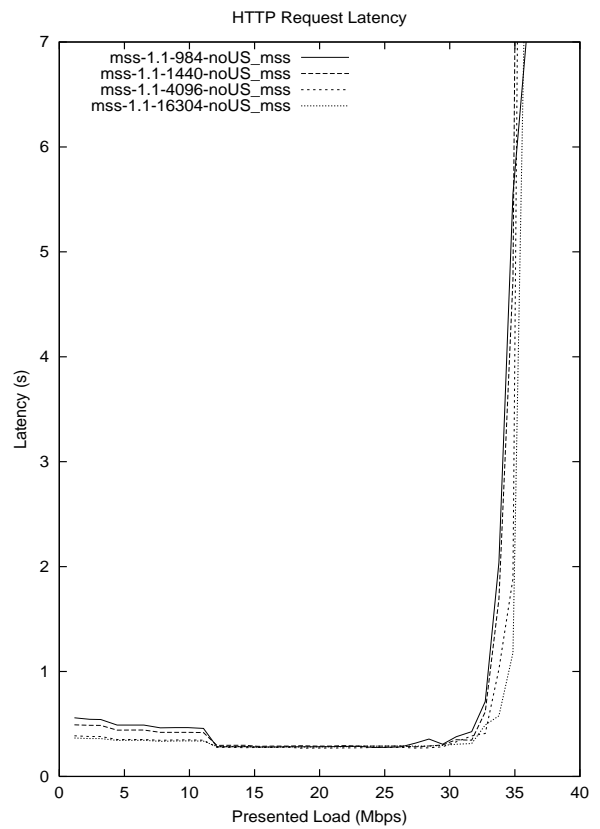


Fig. 17. Page Latencies for MSS runs using http1.1 and no US cloud

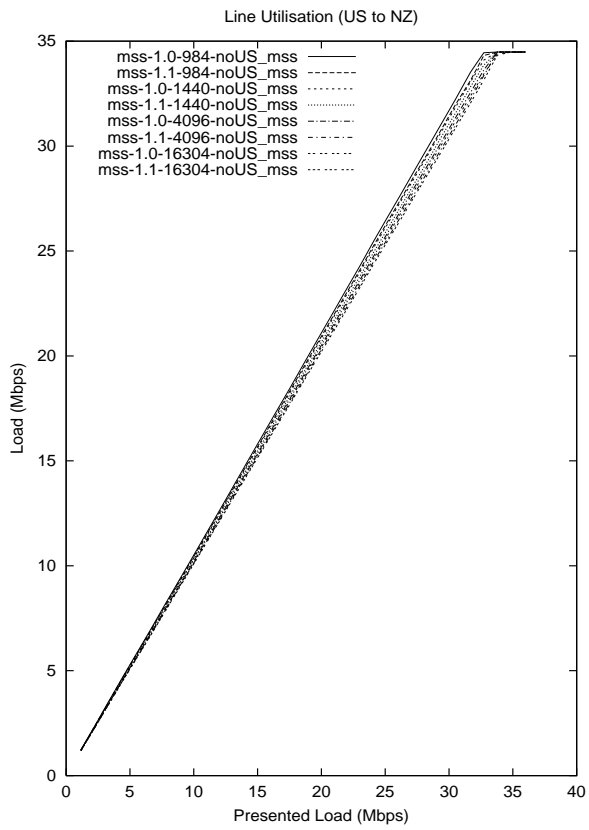


Fig. 18. Load on US-NZ link for MSS runs with no US cloud

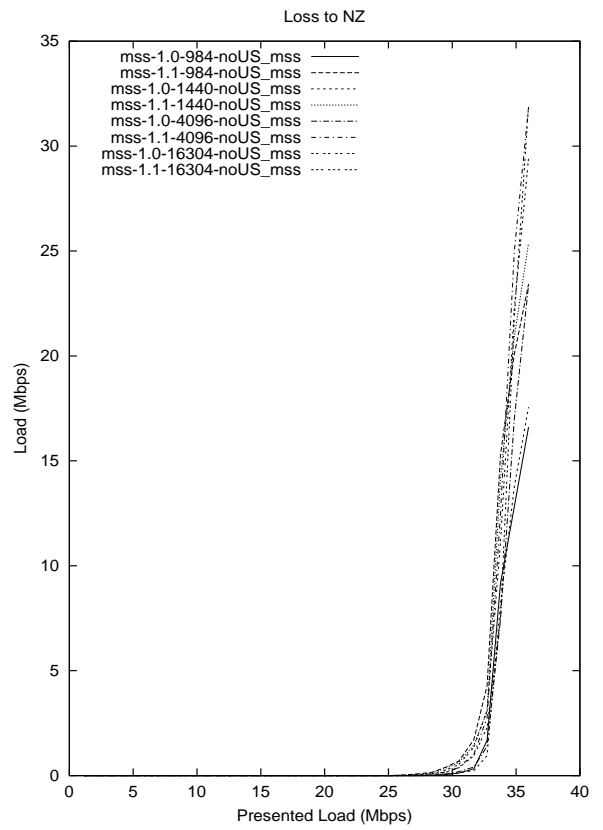


Fig. 19. Loss on US-NZ link for MSS runs with no US cloud

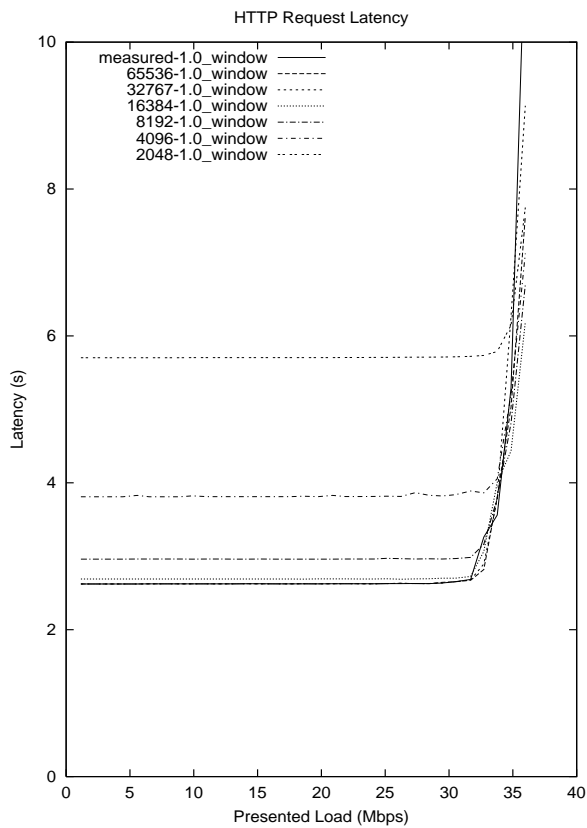


Fig. 20. Page Latencies for window size runs using http1.0

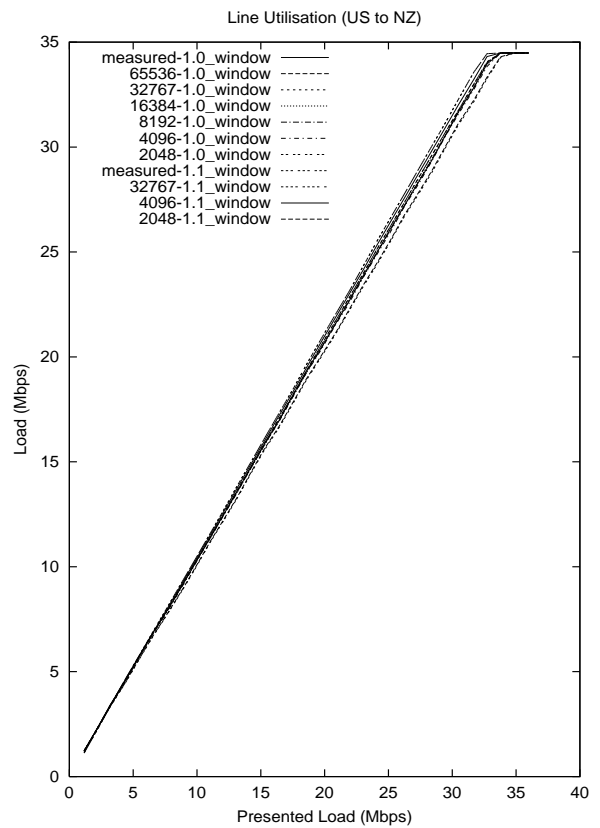


Fig. 22. Loss on US-NZ link for window size runs

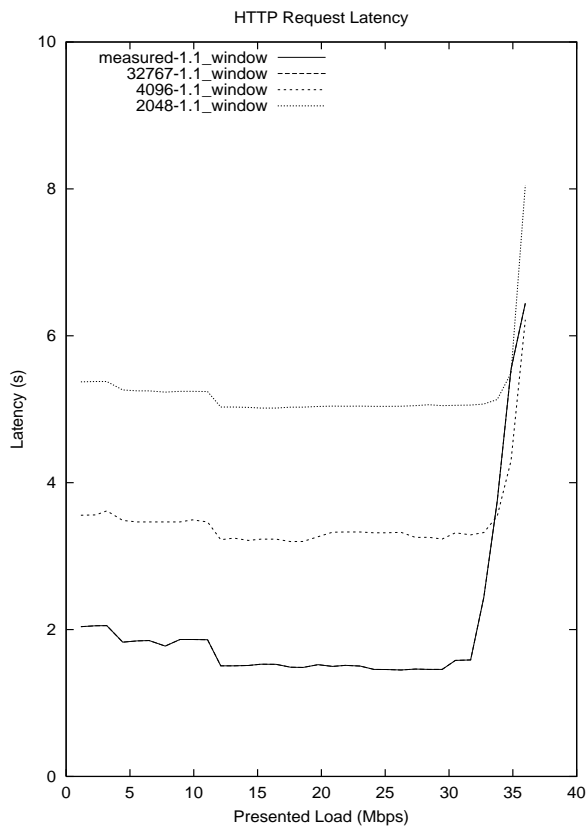


Fig. 21. Page Latencies for window size runs using http1.1

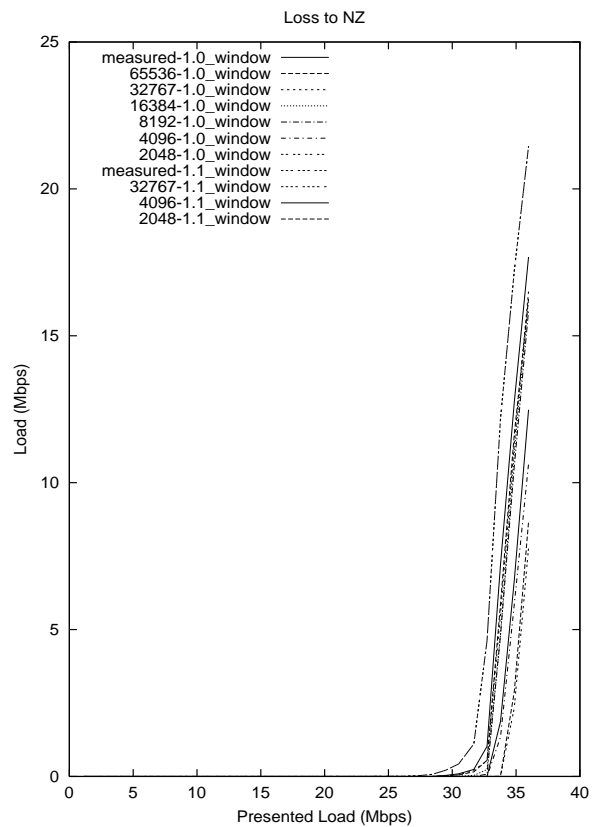


Fig. 23. Loss on US-NZ link for window size runs

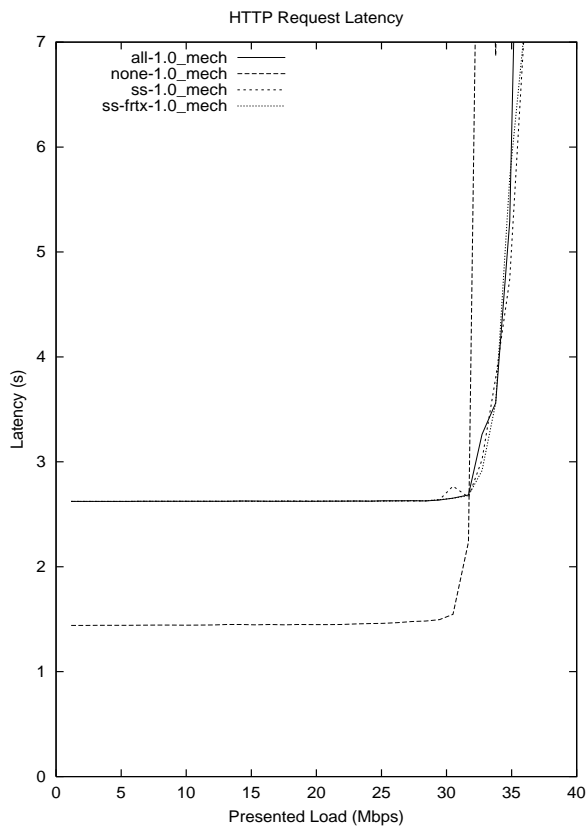


Fig. 24. Page Latencies for TCP mechanisms runs using http1.0

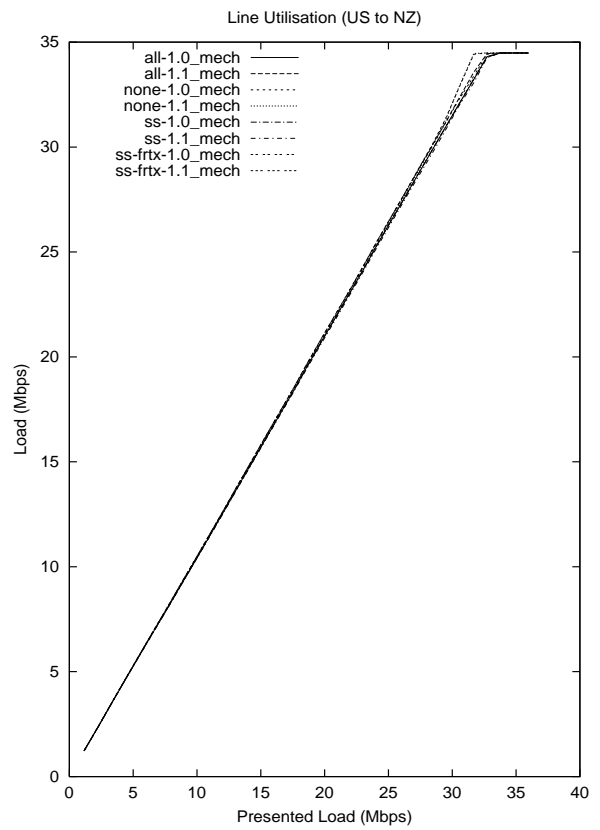


Fig. 26. Load on US-NZ link for TCP mechanisms runs

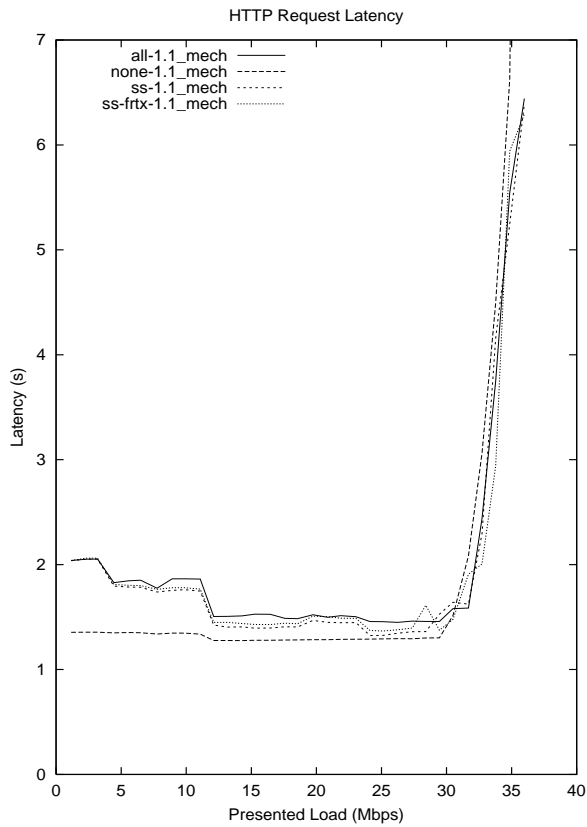


Fig. 25. Page Latencies for TCP mechanisms runs using http1.1

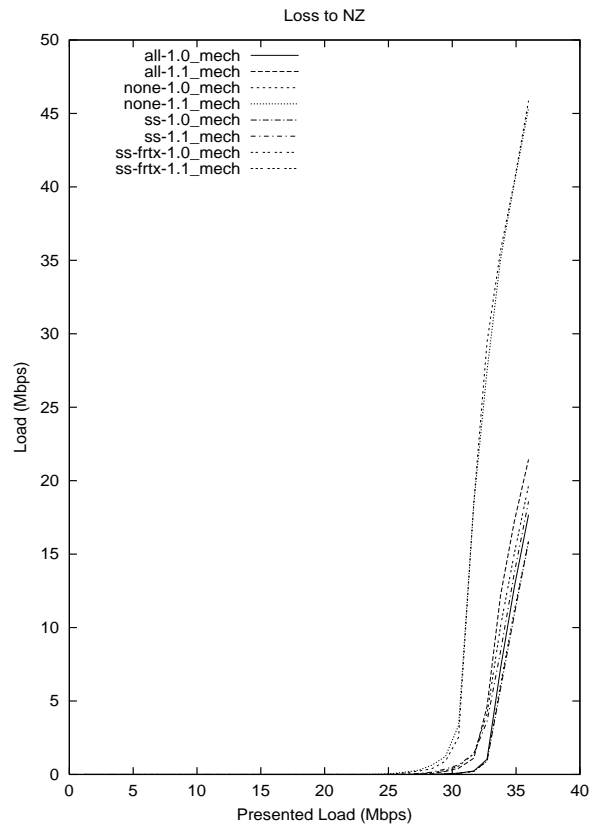


Fig. 27. Loss on US-NZ link for TCP mechanisms runs