Providing Network Services at the Base Station in a Wireless Networking Environment

Marcello Lioy and J. P. Black
University of Waterloo
{*mvlioy,jpblack*}*@uwaterloo.ca*

Abstract

Due to current demands for anytime, anywhere access to information, wireless and mobile computing will play a large role in future internetworks. TCP, the current de facto transport protocol, is tuned for stationary hosts in a wireline environment, and so does not behave ideally in mobile wireless environments. Much work has been done to improve TCP throughput in wireless environments, but the issues of disconnection and competition for limited bandwidth have not been dealt with adequately. In this paper, we propose a Base Station Service Protocol that enables base stations to provide services for mobile computers in wireless networks. Two TCP-related services are investigated which address both the disconnection and limited bandwidth issues. Support for TCP level handoff is also discussed. The disconnection service is implemented by having the base station keep the mobile's TCP connections open on its behalf when it is disconnected, and a priority service allows selected streams to use more or less of the available bandwidth. Both services have been implemented and shown to achieve the intended results.

1 Introduction

The current public demand to have information access at any time regardless of location indicates the need for strong support of wireless and mobile computing. Communication in wireless networks is characterized by low throughput, high latencies, sporadic high bit-error rates, and periodic temporary disconnections. Mobility complicates matters further by increasing the complexity of routing data to machines that are not only away from their home network but also potentially engaging in handoffs from cell to cell as they move within cellular networks. These handoffs can cause significant packet loss as well as variance in packet delays.

These conditions are very different from those in current wireline networks, with stationary hosts, very low bit error rates, high throughput, and very infrequent disconnections. It is for these environments that TCP [6, 16, 17] was tuned. TCP behaves quite well in wireline environments by using round-trip time to calculate retransmission timeouts, which allows it to adapt to variability in the network while attempting to maintain peak throughput. TCP considers packet losses to be caused by congestion, which in the low bit-error environment of most wireline networks, is accurate. When this occurs, TCP invokes the exponential backoff and slow start algorithms [13]. These are intended to restabilize the network and avoid congestive collapse, and they do so very effectively.

The problem is that packet loss in wireless networks is often a result of corruption, handoff losses or handoff delays. None of these losses requires the invocation of congestion control, but that is how TCP reacts to them. This degrades the already meager throughput on the channel, and is not the behaviour one would like to see. Ideally, if a packet is lost due to handoff or corruption, the sending TCP should retransmit immediately, rather than throttle back the data flow. Much work has been done to improve TCP throughput on wireless links, some with high degrees of success, but there are aspects of TCP behaviour over wireless links that still need to be addressed.

In this paper, we propose the Base Station Services architecture and protocol, which can be used to improve the quality of wireless computing by providing value-added services to the mobile at the base station (the last-hop router before the wireless link to the mobile host). The services discussed in this paper are all designed explicitly to improve TCP behaviour in a mobile wireless environment. These include TCP-level handoff, support for disconnections, and stream prioritization. There is the possibility of other services that are not directly related to TCP, such as a reservation service based on RSVP [5, 18], or even some kind of mobile proxy services. The TCP-related services do not require any changes to either the TCP protocol stack or applications that use it.

The rest of this paper is organized as follows. In Section 2, the goals of the architecture and protocol are discussed, as are some related works and their success in meeting these goals. Sections 3 and 4 provide a general overview of the protocol, a description of the services and how they are provided. The experiments and results are discussed in Section 5. Future work is discussed in Section 6, and conclusions are drawn in Section 7.

2 Goals and Related Work

The goals of this research are to improve TCP behaviour in wireless environments. These improvements are not simply limited to improving TCP throughput, but also include providing resilience against temporary disconnection, and prioritizing streams to improve the delivery of bursty traffic in the presence of a steady stream of data from another connection.

There are several criteria against which we evaluate proposed solutions.

1. The TCP protocol and its semantics must not be altered.
2. There should be only minimal changes to TCP implementations.
3. Any changes or special software should be confined to the base station and/or the mobile.
4. No changes to TCP-based applications should be required.

Below is a brief summary of various protocols designed to improve TCP behaviour in mobile/wireless environments.

- **Link-level retransmissions [14]:** The link layer uses retransmissions to guarantee that data crosses the wireless link without error. This has the advantage of working independent of the higher-level protocols. This is also a disadvantage, as it may result in duplicated effort if the link layer is trying to transmit a TCP segment, and the TCP layer retransmits the same segment. The problem here is that the link layer will deliver them *both*. Each will make it across the wireless link when only one needs to. The appropriate behaviour would be to stop retransmitting the original packet and instead transmit the TCP retransmission, however, there is no shared knowledge between the link and transport layers.
- **Indirect TCP [1, 2]:** The TCP connection is split at the base station: regular TCP connections are used between the stationary host and the base station, while a transport protocol more suited to

wireless environments is used between the base station and the mobile. This approach has the advantage of isolating the flow control across the wireless link from that of the wired portion of the network. However, there are two fairly significant problems with this approach: TCP end-to-end semantics are broken, and it requires the relinking of all TCP-based applications on the mobile in order to work.

- **Fast Retransmit [7]:** This approach was designed to reduce the latency caused by handoffs. By having the receiver send a certain number of duplicate acknowledgments, the sender will realize that a packet has been lost, but the loss is not due to congestion, and so will immediately retransmit the packet without invoking congestion control mechanisms. This approach is effective in reducing latency due to handoff, but it does not solve any of the other problems that TCP experiences in wireless/mobile environments.

- **Explicit Acknowledgments [8]:** Explicit acknowledgments are generated at the base station to indicate that the TCP segment has made it to that point error-free. This allows the sending TCP to differentiate between congestive losses and corruptive losses. If the sender is a stationary wired host and it receives an ACK from the base station, it knows that if the packet is then lost, it is due to corruption. If the loss occurs before the base station ACK is received, the sender knows that it is congestion-based. If the mobile is the sender and it receives an ACK from the base station, it knows that any subsequent loss is congestive, while any losses prior to receiving the ACK are due to corruption. The most significant drawback of this approach is that it requires modifications on of the all hosts involved: the base station, the mobile and also any stationary wired hosts.

- **Extensions for Space-Based Communication [11]:** SCPS-TP is a new protocol for wireless environments, based on TCP. The protocol itself is capable of differentiating between congestive and corruptive losses. It is also capable of dealing with temporary disconnections without invoking congestion control mechanisms. The protocol solves almost all of the problems of TCP in wireless environments. However, while it is capable of interoperating with unmodified TCP stacks, it is only generally useful if implemented at all hosts involved.

- **Snoop [3]:** Snoop is a combination of protocols used to improve both handoff latency and TCP throughput. Handoff latency reduction is achieved by a routing protocol based on IP Multicast [9, 10] rather than Mobile IP [15]. This is rather limiting, as IP Multicast routing is not widely supported in the Internet. The Snoop module caches packets at the base station, and retransmits them if it perceives that they have been corrupted while crossing the wireless link. It is similar to link-level retransmissions, but because it is so closely tied to TCP, there is no duplication of effort.

While Snoop solves the throughput problem admirably, it does not solve the problems associated with link outages or of poor interactive performance in a shared low-bandwidth environment. A link outage invokes TCP congestion-control mechanisms, and if the outage lasts too long, the TCP connection will be dismantled.

Another problem is that in a low-bandwidth environment, bursty traffic and steady traffic do not interact very well. The steady traffic will cause the bursty traffic to become even more bursty, as those segments will be separated by large numbers of steady-stream segments. When the bursty traffic is a result of an interactive session of some kind, this spreading of the bursty traffic can be most irritating to the user.

3 The Base Station Service Protocol

The Base Station Service Protocol (BSSP) provides value-added services to mobile computers in wireless/mobile networks. It was designed to be general-purpose enough that any number of services could be provided, such as the transport-level services that are discussed later in this paper. Other services could be provided at the network level, such as a providing advanced bandwidth reservations at other base stations using a reservation protocol such as RSVP, or even a high-level service such as a mobile proxy. The discussion in this paper will be limited to the transport-level services mentioned above.

The protocol is implemented both at the mobile and the base station. It provides the services by maintaining state information at the base station that represents both the services being provided and the state the service is currently in. It operates in four distinct phases: protocol discovery, authenticated key exchange, state maintenance and service requests.

Protocol discovery is used when a mobile computer first initializes its network connection, and, while it is roaming, to determine whether it has roamed into a network that supports the BSSP. If protocol discovery fails, that is, the BSSP is not supported in this network, the mobile sends a message to the previous BSSP base station requesting that it discard any state being maintained.

For concreteness, we assume the use of an asymmetric public-key system for identification and authentication of the mobile and the base stations. After protocol discovery, the mobile and base station exchange key certificates issued by a trusted certification authority. Detailed research into more appropriate security measures for this environment is outside the scope of this paper.

After successful completion of the previous two phases, the mobile enters into the state-maintenance phase if it has been roaming and has BSSP state stored at a previous base station. In this phase, the mobile requests that the new base station retrieve state from the old base station. In this request, the mobile includes a “chit,” which is used by the old base station to authenticate the message as having originated from the mobile. The contents of the chit are the request for state transfer, the IP address of the base station to which the state should be transfered, and the certificate of the new base station.

Upon receipt of the request, the base station can refuse it if it has insufficient resources, or forward it to the old base station. In the latter case, the new base station sends the chit, as well as an authentication chit of its own, that consists of its signed IP address. The base station chit allows the old base station to be sure that the new base station is not only who it claims to be but also identifies it as the machine specified by the mobile.

After authenticating the mobile and the new base station, the old base station transfers the state information to the new base station. Once the transfer is finished, the new base station resumes the services that the old base station had been providing.

After these initialization phases, the base station is providing whatever services the mobile had negotiated previously, and the mobile can request and cancel services. This phase is intended to support an array of services. The service-specific request information is included at the end of a standard protocol header. There are provisions for requests, and cancelations of service, as well as acceptances and refusals from the base station. A service can be refused if the base station has insufficient resources, or if the service is not supported at this particular base station.

Service requests can be made for any combination of port and machine numbers, possibly including “wildcard” values. The only field that is always specified explicitly is the address of the mobile. It is therefore possible, for example, to request that all FTP connections have a particular service applied to them.

As mentioned above, one of the criteria that we sought to satisfy with the BSSP was that existing TCP applications need not be relinked, or rewritten, to take advantage of these value-added services. As it is not necessary for the application using the TCP connection to be the one that makes a service request, it is possible for another process to make requests on behalf of the particular processes. This means that processes that are not aware of BSSP can still take advantage of the services through the auspices of another process. Such a process could take the form of a control panel of some sort that would allow the user to select streams associated with particular applications and assign a service to them, or potentially to assign a service to a particular application type, such as FTP.

4 Transport-Level Services

As mentioned above, there are three transport-level services defined: TCP-level handoff, connection maintenance, and stream prioritization. Each of these is discussed in more detail below.

All of these services use the same technique, which is the alteration of the window size advertised in the TCP segment. In the case of the first two, the window size is actually set to zero; these are called "zero window-size messages" (ZWSM). A ZWSM is an acknowledgment sent by the base station which appears to come from the mobile.

The use of ZWSM inhibits the sending TCP from invoking-congestion control mechanisms and stops the flow of data except for occasional window probes. Upon receipt of an acknowledgment with the send window set to zero, the sending TCP enters the "persist" state. In this state, all TCP timers are suspended except for the persist timer. The persist timer is responsible for sending window probes. TCP does not reliably transmit acknowledgments that contain no data, that is, it does not acknowledge acknowledgments, only segments containing data, so the ACK from the receiver may get lost. These window probes are sent to the receiver to determine if a receive window has opened since the last received acknowledgment. The persist timer generally behaves in the same way as the retransmit timer: it exhibits exponential back off, that is, the time between probes doubles until some maximum interval is reached. These window probes will continue indefinitely, as long as the base station continues to send acknowledgments with advertised windows of size zero.

4.1 TCP-Level Handoff

If a mobile informs a base station running BSSP that it is doing a handoff, the BSSP daemon will a send a ZWSM to any host sending a TCP packet to that mobile. This avoids the invocation of congestion control and allows for immediate recovery once the handoff has completed. These ZWS messages will continue to be generated as a result of TCP segments destined for the mobile for a maximum of one minute, which should be far greater than the time a handoff requires. The only exceptions will be for those connections that specifically request the connection maintenance service. After one minute, or when the base station has been informed of successful completion of the handoff, any packets destined for the mobile will be discarded.

4.2 Connection Maintenance

This service will maintain a TCP connection for a specified length of time. When it observes that that the mobile has been disconnected, the base station begins providing connection maintenance service for those connections that have had the service requested for them. The service will be provided until either the mobile is no longer disconnected, a state transfer request is made for the mobile, or the timer expires. When the service timer expires, the base station sends a reset message to the remote host, and ignores all future packets.

4.3 Stream Prioritization

This service allows the mobile to associate priorities with TCP streams. This would allow the mobile, for example, to associate higher priorities with interactive streams, in situations where limited bandwidth is being shared with other applications. The default priority for a stream would be "normal," which would not imply any alteration of the flow of the stream. Lower priorities can be applied selectively to streams. Streams whose priority has been lowered will only have their flow restricted in the presence of other streams.

The mechanism for this service is similar to that of the other two, although ZWS messages are not used. Instead, the window size of packets passing between the mobile and the stationary host is altered. The base station, being the last-hop router, is in an ideal position to make these alterations as all packets to and from the mobile must pass through it. The alteration of the window size allows the base station to reduce the rate at which data is leaving the two machines involved in the connection. While reducing the window size is safe, increasing it past that advertised by the machine would be dangerous, and could be destructive: buffers could overflow, resulting in many retransmissions. Rather than assuming that all streams should be slowed and allowing users to increase the flow rate, we decided that reduction of flow would be a fairly infrequent happening, and that from a user interface point of view it would make more sense to have the common case be the default.

5 Experiments and Results

5.1 The Experimental Testbed

The experimental testbed is composed of two networks connected by a gateway that runs the BSSP software. One network contains the wireless leg, and the other is a wireline network which is connected to the Internet. A laptop PC running Linux is connected to the wireless leg of the network via a Lucent WaveLAN card. Mobile IP is used to support mobility.

With the exception of the interactive performance tests, data streams bound for the mobile originate from a host connected via Ethernet to the gateway. In the case of the interactive performance tests, the wireline server is connected to the network via a modem to emulate conditions found in a wide area wireless network. The WaveLAN is still used as the link to the mobile, however, a data rate being approximately 1.5 Megabits per second makes it considerably faster than those of current wide-area data networks.

Two applications were written to simulate distinct traffic patterns. The first application is a bulk data transfer client and server. This application mimics the behaviour of an FTP data stream [4, 12], that is, one which has steady traffic and is heavily loaded. The second application is a time client and server. The server sends the its local time regularly at a specified interval. On receipt, the client displays server time local time and the difference between the two. The times sent are accurate to within one second. This is meant to simulate an interactive stream, that is, one which has bursty traffic and is fairly lightly loaded.

5.2 Connection Maintenance

5.2.1 The Experiments

Several experiments were performed to test the connection maintenance service, contrasting TCP behaviour with and without the service. Experiments used disconnection lengths of 1 minute, 5 minutes and 15 minutes. The aim was to see if there was a noticeable change in the speed at which data flow is resumed once the machine was reconnected.

5.2.2 The Results

As can bee seen in Figure 1, all of the tests yield positive results. It is also plain that the connection maintenance service not only maintains the connection for the specified time, but also restarts the data flow up to 40 seconds before flow is resumed without the service. The results of tests for a fifteen-minute disconnection are not displayed in Figure 1 because nothing new is revealed, except that in the absence of the service, the sender resets the stream after approximately ten minutes, and the data flow cannot be resumed on reconnection.

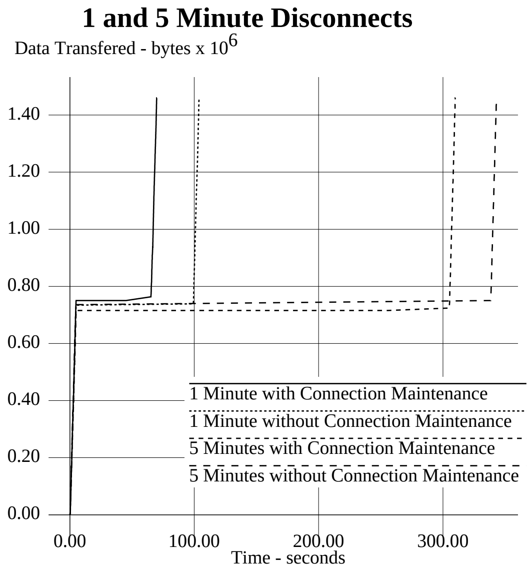

Figure 1: One and five minute disconnections, both with and without connection maintenance

5.3 Stream Prioritization

5.3.1 The Experiments

There are two distinct classes of experiments on the effectiveness of the priority service. The first involves two bulk transfers taking place simultaneously, with one set to low priority and the other to high priority. Experiments are performed with window scalings of both 4KB and 8KB. In the second experiment, a bulk transfer stream and an interactive stream compete for a low-bandwidth link, with the bulk stream being given a low priority and the interactive session a high priority. The interval on the time server was set to one second. As explained above, experiments were performed with link speeds of 14.4 Kbps and 28.8 Kbps. At each of these link speeds, window scalings of both 4KB and 8KB were tried.

5.3.2 The Results

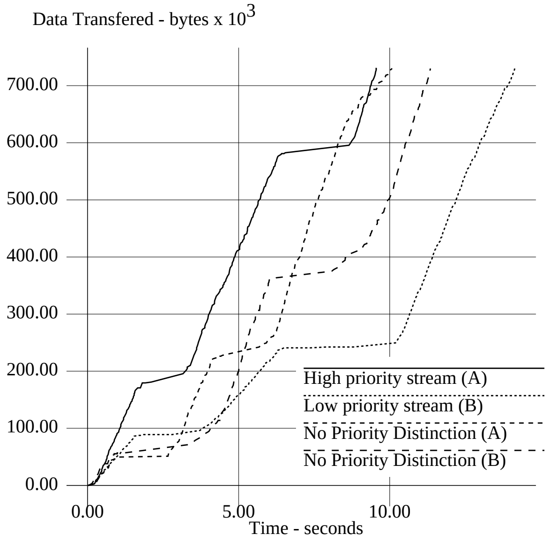

Figure 2: Two bulk transfer streams one prioritized and one not. The window scaling is set to 8KB. The two streams without prioritization are included as a baseline for comparison.

Figures 2 and 3 clearly demonstrate that the prioritization of streams yields quantifiable

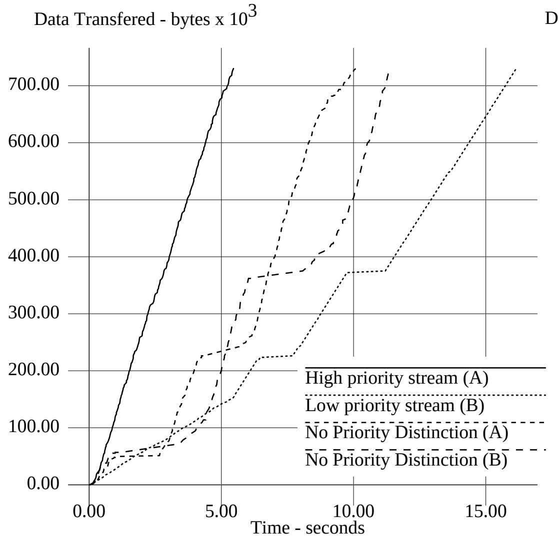

Figure 3: Two bulk transfer streams one prioritized and one not. The window scaling is set to 4KB. The two streams without prioritization are included as a baseline for comparison.

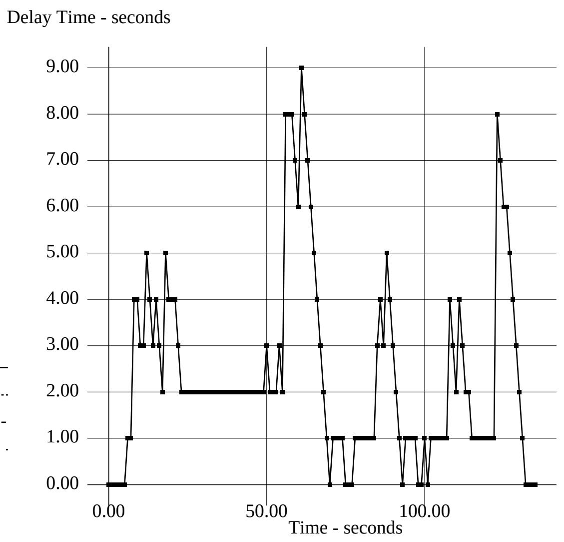

Figure 4: Packet delays for an interactive stream caused by competition with a high load steady stream. The time a packet was sent is plotted against the delay it experienced in delivery.

results. It is clearly visible that the data throughput improves for the high priority stream, and the behaviour of the stream is also more consistent. When the two streams are competing for the bandwidth and there is no priority scheme in effect, it can be seen that the throughput consistency is not very good and the data is transfered in fits and starts. When the priority service is introduced with an 8KB window size, we see a marked improvement in the consistency of the data flow, and when the window scaling is reduced to 4KB, we see even more dramatic improvements. The data flow for the high-priority stream is almost perfectly constant, and we also see that it has improved for the low priority stream. By way of comparison, we see that the unprioritized streams fall almost exactly in the middle of the high and low-priority streams. This last observation is also noticeable with the 8KB window scaling, but is not quite as dramatic.

The second set of experiments are also quite telling and show very positive results. Figure 4 shows the typical behaviour of an interactive stream on a low-bandwidth link, in the presence of a bulk stream. What often happens is that a group packets will be delayed, and then all arrive together. This is evidenced by the sharp rise in delay for a single packet and then the subsequent drop by one second increments in the following packets until either the delay drops to zero or the next spike occurs.

From Figures 5 and 6 we see the numbers of packets that were delayed by a certain amount of time. It is clear in both cases that the priority mechanism is effective in reducing delay on the interactive stream, and in particular, that using a 4KB window scaling is very effective.

In the 28.8 Kbps case with the 4 KB scaling, we can clearly see that there is no delay, as all of the packets are in the zero-delay column. The 8KB window scaling still yields fairly good results, with all delays appearing in the 1 to 2 second range. In the presence of the unprioritized heavy-load stream, we see that most delays are in the 2 to 6 second range, with some delays as high as 10

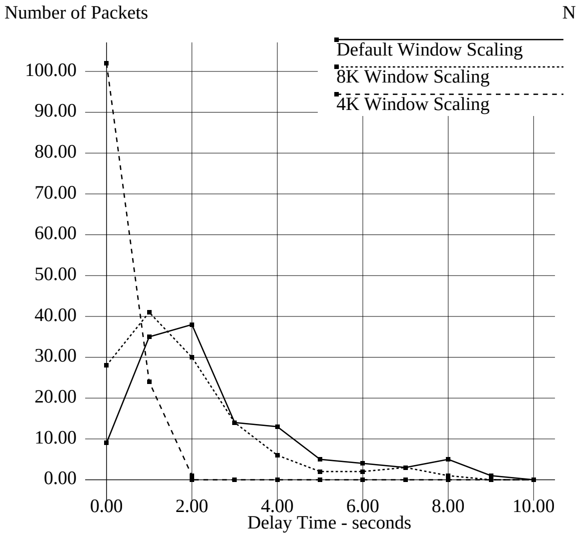

Figure 5: Delays of an interactive stream in the presence of a high load steady stream with various window scalings over a 14.4Kbps connection.

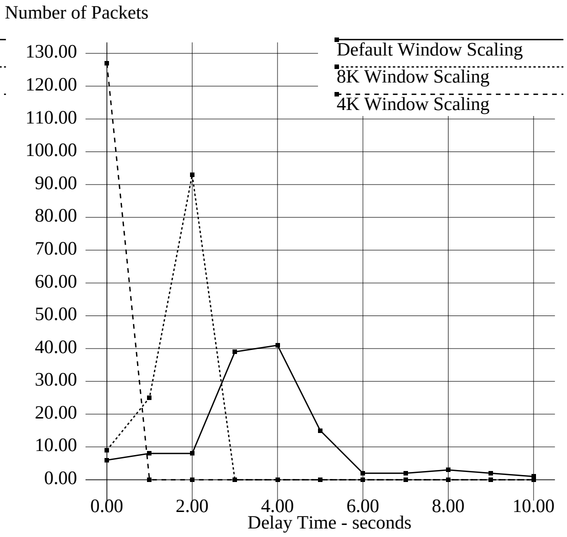

Figure 6: Delays of an interactive stream in the presence of a high load steady stream with various window scalings over a 28.8Kbps connection.

seconds. The results for the 14.4 Kbps experiments are similar. It is interesting to note that the peak time for the delays seems to be lower in both the 8KB window scaling and the unprioritized cases. This is probably an aberration due to the relatively small sampling of only 127 packets sent. The average delays and standard deviations are in Table 1.

6 Future Work

There are several areas where more work could be done. One of the first would be to investigate mechanisms for providing the connection maintenance services for streams *outbound* from mobiles. This would involve some server running on the mobile that would monitor the state of connectedness, and stop outbound TCP segments without invoking congestion control and eventually dismantling the connection.

Channel Speed	Window Scaling	Average Delay	σ
14.4 Kbps	default	2.55	2.01
	8KB	1.72	1.65
	4KB	0.20	0.42
28.8 Kbps	default	3.61	1.77
	8KB	1.66	0.61
	4KB	0.00	0.00

Table 1: Average delays and standard deviations for various bandwidths and window sizes.

Another area that needs investigation is handoff performance. It is clear that the performance will not be as good as that exhibited by the Snoop protocol, as a transfer of state information is required. However, this state transfer is unavoidable if services are to follow the mobile from point of attachment to point of attachment.

A third area might be to see how the BSSP would interact with the Snoop protocols. Some of the services they provide overlap, such as handoff support, and some BSSP services could be of use to Snoop, such as knowing if the mobile has been disconnected.

Finally, the investigation of a global priority scheme at the base station would also be of interest. Here, the base station might impose some form of global prioritization scheme on streams associated with all of the mobiles it is currently managing. This may provide a mechanism for giving privileged network access to particular mobiles.

7 Conclusions

It is clear from the data presented that providing services at the base station can greatly improve TCP performance. In the case of prioritization, it seems that scaling the window back to 4KB provides the best performance for both interactive and steady traffic streams. Both connection maintenance and prioritization are worthwhile services for mobiles in wireless environments as each provides an improvement in TCP behaviour in the wireless environment.

References

[1] A. Bakre and B. R. Badrinath. Handoff and system support of indirect TCP/IP. In *Proc. of the Second Usenix Symp. on Mobile and Location Independent Computing*, April 1995.

[2] A. Bakre and B. R. Badrinath. I-TCP: Indirect TCP for mobile hosts. In *Proc. of the 15th International Conference on Distributed Computing Systems*, pages 136–143, May 1995.

[3] H. Balakrishnan, S. Seshan, and R.H. Katz. Improving Reliable Transport and Handoff Perfomance in Cellular Wireless Networks. *ACM Wireless Networks*, 1(4), December 1995.

[4] A. Bhushan. File transfer protocol, August 1972. *RFC 354*.

[5] R. Braden, L. Zhang, S. Berson, S. Herzog, and S Jamin. Resource ReSerVation Protocol (RSVP) – version 1 functional specification. ftp://ietf.org/internet-drafts/draft-ietf-rsvp-spec-14.txt, November 1996.

[6] R. Branden. Requirements for internet hosts — communication layers, October 1989. *RFC 1122*.

[7] R. Cáceres and L. Iftode. Improving the Performance of Reliable Transport Protocols in Mobile Computing Environments. *IEEE Journal on Selected Areas in Communications*, 15(5), June 1995.

[8] J. A. Cobb and P. Agrawal. Congestion or corruption? A strategy for efficient wireless TCP sessions. In *Proc. of the IEEE Symposium on Computers and Communications*, pages 262–268. IEEE, June 1995.

[9] S. Deering. Host extensions for IP multicasting, August 1989. *RFC 1112*.

[10] S. Deering. *Multicast Routing in a Datagram Internetwork*. PhD thesis, Stanford University, December 1991.

[11] Robert C. Durst, Gregory J. Miller, and Eric J. Travis. TCP Extensions for Space Communications. In *Proc. of MOBICOM '96*, pages 15–26. ACM/IEEE, November 1996.

[12] G. Hicks. User FTP documentation, November 1972. *RFC 412*.

[13] Van Jacobson. Congestion avoidance and control. In *Proc. of SIGCOMM '88*. ACM/IEEE, August 1988.

[14] S. Paul, E. Ayanoglu, T.F. Laprota, K.H. Chen, K.K. Sabnani, and R.D. Gitlin. An asymetric link-layer protocol for digital cellular communications. In *Proc. of InfoComm '95*, 1995.

[15] C. Perkins. IP Mobility support, October 1996. *RFC 2002*.

[16] J. Postel. Transmission control protocol, September 1981. *RFC 793*.

[17] W. Richard Stevens. *TCP/IP Illustrated*, volume 1. Addison Westley, 1994.

[18] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala. RSVP: A new resource ReSerVation Protocol. *IEEE Network*, September 1993.

Marcello Lioy is a Masters candidate at the University of Waterloo having received his B.Sc. in Physics from the University of British Columbia in 1991. He is now writing his thesis and his current research area is distributed systems, more specifically mobile and wireless computing. Other interests include voice and data telecommunications, real-time multimedia over the Internet, and quality of service issues in mobile/wireless environments. He is a member of the mobile research group within the Shoshin Distributed Systems group at Waterloo.

James P. Black received his Ph.D. in Computer Science from the University of Waterloo in 1982, having previously studied at the University of Calgary and the Institut National Polytechnique de Grenoble, France. He has been on the faculty of the University since 1984. He is currently Associate Provost, Information Systems and Technology, and is an Associate Professor in the Department of Computer Science. As Associate Provost, he is responsible for the administration and long-term planning of all information systems and information technology at UW. His current research interests include the management of distributed applications and systems, distributed debugging and the visualization of complex executions, and mobile and wireless computing. He was involved in the design and implementation of The Newcastle Connection, an early distributed extension to Unix at the Computing Laboratory, University of Newcastle upon Tyne. More recently, he has been involved in the design and development of Poet, a suite of software for visualizing and debugging distributed and concurrent applications. His research forms part of the activities of the Shoshin research group at Waterloo.