Mobile-End Transport Protocol: An Alternative to TCP/IP Over Wireless Links*

Kuang-Yeh Wang
Department of Computer Science
University of Maryland
College Park, MD 20742
kwang@cs.umd.edu

Satish K. Tripathi
Bourns College of Engineering
University of California
Riverside, CA 92521-0425
tripathi@engr.ucr.edu

Abstract

Unlike wired links, wireless network bandwidth is highly limited and the channel usually suffers from frequent and bursty loss due to its vulnerability to various kinds of interference. At the same time, the traditional TCP with its congestion control is well known for its poor performance over wireless links. This paper proposes a new protocol that (1) replaces TCP/IP over the wireless link by a simpler protocol with smaller headers, if the link is the last hop along a data path, (2) shifts functions needed to communicate with an Internet host using TCP/IP from the mobile host to the base station, so that the distinct wireless link is hidden from the outside Internet, and (3) exploits link-layer acknowledgments and retransmissions to quickly recover losses over the wireless link. Our simulation results show a substantial performance improvement achieved by the new protocol.

1 Introduction

Recently, the emergence of the World Wide Web as a major media all over the world and the explosive growth of business of the Internet access providers have highlighted the success of the inter-networking technologies, specifically the Transmission Control Protocol (TCP) and the Internet Protocol (IP), which have integrated a wide range of different physical networks into a global Internet. However, with the introduction of wireless technology, the traditional inter-networking technologies, which was developed based on wired, fixed networks, have been facing a great challenge. The design of Mobile IP [19] has solved the operability problem of integrating mobile networks into the Internet. But the distinct characteristics of the wireless networks, the mobility they provide and the limitation of the mobile devices have also caused substantial performance problems with the traditional Internet protocols. First, the TCP/IP header consumes the more limited bandwidth of a wireless link and running TCP/IP on the less powerful mobile machines causes more overhead. Second, losses over a wireless link are usually bursty and frequent due to its vulnerability to interference, and host mobility also causes data loss during a transition period. However, the congestion control mechanism in TCP assumes that data loss is caused by congestion. It reacts to losses due to mobility and wirelessness by backing off and hence results in performance degradation.

We propose a new *Mobile-End Transport Protocol* (METP) to solve these problems. The protocol replaces TCP/IP over the wireless link by a simpler one with smaller headers, if the link is the last hop along a data path. By doing so, it shifts functions needed to communicate with an Internet host using TCP/IP from the mobile host to the base station, so that the distinct wireless link is hidden from the outside Internet. An important feature is that it exploits link-layer acknowledgments and retransmissions to quickly recover losses over the wireless link.

The rest of the paper is organized as follows. More detailed background and motivation are presented in Section 2. Some related work by other researchers can be found in Section 3. Then we give a description of the protocol in Section 4. Section 5 presents our simulation results, and finally Section 6 gives some concluding remarks and plans for future work.

2 Background

2.1 Limited Wireless Resources

Compared to traditional wired networks, a wireless link has much less bandwidth. As an example, the bandwidth of the AT&T WaveLAN 915 MHz is about 2 Mbps, while an Ethernet can achieve 100 Mbps. In addition, a wireless connection has higher loss rate due to its vulnerability to interference and disconnection, and hence the effective bandwidth is further reduced.

The packet headers used by various layers of network protocols cause a significant amount of overhead. The size of a minimal TCP header is 20 bytes, and a minimal IP version 4 (IPv4) header occupies another 20 bytes. With the new IP version 6, the combined TCP/IP header takes up 60 bytes. In such a connection as TELNET, a TCP segment may contain only 1 byte of data, which means the wireless link is spending most of its capacity carrying administrative traffic. Even with TCP bulk transfer, which typically uses 512 byte segments over a wide area, the overhead is 11.7%, which is still very significant given such scarce network resources.

*This work is supported in part by NSF grant CCR 9318933, IBM equipment grants, and Novell.

0-7803-4383-2/98/$10.00 © 1998 IEEE.

2.2 Congestion Control in TCP

TCP's congestion control mechanism was designed to work on wired inter-networks, where packet losses are often caused by congestion at the routers, not by errors on the links. Its exponential back-off [14] and slow-start [12] policies were particularly aimed at relieving congestion.

However, earlier researches showed that wired and wireless media have significantly different error characteristics. While data losses on a wired link are very rare and random in nature, errors on a wireless medium are bursty [4, 10]. This means the effective bandwidth over this link may drop down to almost 0 during a burst of errors, and then return to the full capacity. Such bandwidth drop has nothing to do with the number of packets being transmitted over the link. On the other hand, when there is congestion, the network has to digest the traffic gradually and the effective bandwidth recovers slowly.

Mobility also poses a similar problem on a connection involving wireless links. When a hand-off happens, data packets will be lost after the mobile host leaves the transmission range of the old base station and before the mobile host completes its registration with the new base station and its home agent and starts receiving packets through the new base station. But after the registration completes and the new forwarding channel is established, the effective bandwidth of the connection comes back up immediately.

Now consider a connection between two hosts, of which one or both are mobile and connected with a base station through a wireless link. The sender TCP's response to a bursty loss (due to either a hand-off or the nature of the wireless link) has the following performance problems.

1. The multiplicative-decrease/additive-increase algorithm is activated, i.e. the congestion window shrinks by half and grows back up slowly when losses disappear. But since the losses are bursty, the effective bandwidth comes back up quickly and hence the transmission window is unnecessarily reduced and the throughput is degraded.
2. Fast retransmit and fast recovery may fail to detect packet loss early so that the sender will not retransmit until a timeout occurs, causing a long period of silence. The reason is that during a burst of errors, most segments in a window may be lost, and the acknowledgments to those few successfully delivered may be lost as well. In addition, the window size is usually small due to the low bit rate, and therefore probably there are not enough segments in the window to enact the fast retransmit in the first place.
3. Karn's algorithm [14] may aggravate the silence period described above by exponentially increasing the timeout in case of retransmission, when a retransmission is also lost during a bursty loss period.

Such performance deteriorations have been demonstrated in earlier work [4, 6].

2.3 Mobile Computing Environment With Infrastructure Support

In order to alleviate the performance problems of the existing TCP/IP over wireless networks, we intend to take advantage of the base station support and the fact that functions of mobile devices are limited.

With the Internet infrastructure in place, a typical connection consists of mainly wired hops, plus possibly one wireless hop at one or both of the ends. Given that mobile hosts have only limited resources such as memory, disk space, and power supply, it may be better to delegate part of the networking task to the base station from the mobile host. Shifting the bulk of network protocols to the base station has two advantages. First, a more powerful base station takes over some work from a mobile machine with limited resources, improving performance in general. Second, the communication between the base station and the mobile host is hidden from the outside, so it can be specialized and simplified.

Shifting functionalities from mobile machines to the base station naturally gives rise to the question of scalability. A base station may be overwhelmed if it has to serve a large number of mobile hosts with multiple network connections. However, a closer look at the general use of mobile computers shows that they are not used in the same way as a fixed workstation. First, a mobile computer is unlikely to have multiple users login on at the same time. Most of the mobile machines are single-user ones. Even for multi-user systems, because its resource is limited, its connection to the outside is volatile and the machine is largely controlled by the person who carries its console, other users would not like to remotely log on it. Second, while people may work on their laptop computers on an airplane or in a car, it is usually inconvenient to do so while walking. Smaller, palm-size mobile devices are developed to perform several specific functions, such as paging and simple message passing, carried by people walking around. Even with general purpose machines, people are unlikely to perform complicated work that requires a lot of network connections on the fly.

3 Related Work

3.1 TCP/IP Header Compression

Van Jacobson [13] proposed TCP/IP header compression for low-speed serial links, and Degermark et al. [8] provides both UDP/IP and TCP/IP header compression mechanisms over wireless networks. Their basic observation is that among a stream of consecutive packets, a large part of the header does not change very often. So the sender may transmit a full header for the first time, and send only the small changes for the successive packets so long as the core part of the header remains the same. The receiver saves a full header and uses it to recover the headers of the next packets until it receives another full header.

Since the link between the sender and the receiver may drop packets, the major problem that such header compression schemes have to solve is that if the original full header is lost, the receiver may not be able to recover the headers of the subsequent packets and may have to discard them. Great efforts have been made to deal with this issue in order to improve performance.

3.2 Split Connection

Bakre and Badrinath [1] proposed the *Indirect TCP for Mobile Hosts* (I-TCP). This protocol splits a traditional TCP connection

between a mobile host and a corresponding host into wired part and wireless part, with the base station at the middle point.

While [1] envisioned the use of a better adapted transport-layer protocol over the wireless link, Yavatkar and Bhagawat [20] carried out this idea by employing TCP selective acknowledgment options [16] With this mechanism, a receiver returns a selective ACK (SACK) when it receives an segment out of sequence to inform the sender the missing segments by specifying the left and right edges of consecutive data blocks that it has received. In this way more than one lost segment can be recovered in one round-trip time and hence the throughput can be improved.

Haas and Agrawal [11] proposed *Mobile-TCP*, which also splits a TCP connection into two parts at the base station. But they point out that the part between the mobile host and the base station consists of a single link, and hence a much simplified mechanism specially tuned for wireless links can replace TCP on this half of the connection. They use the *asymmetrically based* protocol design for the connection management, retransmission and timer, and flow control over the wireless link to shift much of the work from the comparatively limited and volatile mobile host to the more powerful and stable base station.

Brown and Singh [5] propose a network architecture for mobile computing, where a mobile network is not considered part of the Internet but connected to it through a *supervising host*. In such a mobile network, the supervising host is connected to several *mobile support stations*, which in turn communicate with mobile hosts over wireless links. Virtual circuits are implemented at the network layer in the mobile network. A TCP connection between a mobile host and an Internet host is split into two parts at the supervising host, which talks to the Internet host using traditional TCP and communicates with the mobile host through a simpler protocol (M-TCP) on top of a reliable virtual circuit connection.

3.3 Lower-Layer Solution

Balakrishnan et al. [3] proposed the SNOOP module between TCP and IP layers. At the base station, it monitors the packets being sent to the mobile host and the acknowledgments sent by the mobile host. Packets that have not yet been acknowledged are buffered. If a packet loss is detected through an acknowledgment, the packet will be retransmitted to the mobile host if it is in the buffer without relaying the acknowledgment to hide the loss from the corresponding host.

The SNOOP module by itself does not address the problem of packet losses and delay due to hand-offs. To deal with this issue, they use multicast [7] and intelligent buffering in nearby base stations [15]. In this scheme, the mobile host uses a multicast address as its care-of address. When visiting a foreign domain, it invites the base stations that it has contact with or may hand-off to in the near future to join this multicast group. All packets destined for the mobile host are encapsulated by the home agent and tunneled to the multicast address, and therefore to all these base stations. Only the base station that the mobile host is currently in touch with actually forwards the packets to the mobile host, and other participating base stations simply buffer them. When the mobile host hands-off to a buffering base station, it can start receiving packets from the new base station immediately.

4 Mobile-End Transport Protocol

We propose to eliminate the TCP and IP layers from mobile hosts and hence also the TCP/IP headers from the packets transmitted over a wireless link. A mobile host will have only a simple multiplex and demultiplex mechanism, and the TCP/IP header in packets over a wireless link will be replaced by a header containing essentially only demultiplex keys (port numbers) and IP addresses of the source and the destination (Section 4.3).

Under the new design, all Internet traffic to and from a mobile host is handled by the base station. From the viewpoint of an outside machine that is talking to the mobile host, the communication between the base station and the mobile host resembles that between an application process and a transport layer protocol within a usual machine.

4.1 Shifting IP Layer to the Base Station

We can take advantage of the fact that the hop between a mobile host and its base station is the first or last one along a data path. Therefore, the mobile host does not perform datagram forwarding, and only part of the IP functionalities have to be shifted to the base station.

METP at the base station accepts IP datagrams destined for the mobile host as if they were destined for itself. It strips the datagram of its IP header and delivers it to the higher layer, since the transport layer of the mobile host is also shifted to the base station. Re-assembly of IP fragments is also done at the base station. The header checksum (and similarly any higher layer checksum) is replaced by the link-layer CRC since there is only one hop between the mobile host and the base station.

4.2 Transport Layer

An IP datagram destined for the mobile host is accepted by the base station instead. If it is a TCP segment, METP acts as a proxy TCP in the way described below. If it is a UDP datagram, METP strips its UDP/IP header and delivers it to the mobile host.

In the other direction, when the mobile host wants to send out a UDP datagram, it simply sends the data to the base station, which in turn adds the UDP/IP header, and then set off the resulting datagram. The source and destination information is provided by the mobile host in a new METP header (Section 4.3).

Other unreliable datagrams such as the Internet Control Message Protocol (ICMP), except the Mobile IP registration messages, are handled similarly.

All TCP connections are handled at the base station by METP on behalf of the mobile host. METP negotiates with another host in the Internet to open or close a TCP connection, possibly with a request from the mobile host, and keeps the connection state and sending and receiving buffers. When the mobile host has data to send through a TCP connection, it actually sends the data to the base station, which puts them in the sending buffer of the connection for METP to send out in TCP segments to the destination. When a TCP segment destined for the mobile host arrives at the base station, METP puts it in the receiving buffer and sends an acknowledgment back to the source. A separate process tries to

send data in the receiving buffer to the mobile host, and a receiving buffer is not freed until the data it contains have been received by the mobile host. Figure 1 shows an METP-TCP connection between two mobile hosts through the Internet.

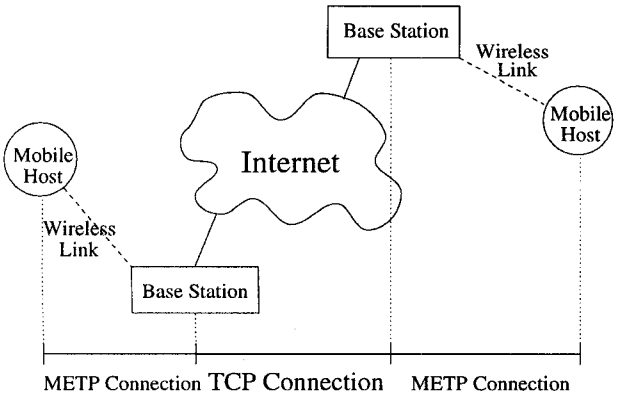

Figure 1: A Typical TCP Connection Under the METP Framework

Since METP at the base station sends out acknowledgment as soon as a segment arrives at the base station, any congestion control or avoidance mechanism of TCP reflects the state of only the wired portion of the connection. Therefore the problems with the TCP congestion control mechanisms over wireless links no longer exist.

However, any problems on the wireless link should still be made known to the corresponding host, and reacted to *differently* without modifying the traditional TCP running there. Fortunately, TCP allows a very natural solution. If the wireless link fails temporarily while no problem exists over the wired networks, METP at the base station continues to receive data from the corresponding host but cannot send any to the mobile host. The result is a decrease in the free receiving buffer space, which causes METP to send out acknowledgments with a smaller *advertised window*. The *allowed window* at the sender shrinks, suppressing data flow. Later when the wireless link comes back up, the advertised window will grow to the original level, and the allowed window at the sender can jump right back without going through any additive-increase or slow-start phase. Since the sender detects no data loss and makes no retransmission, the exponential back-off mechanism is never activated.

From the viewpoint of the TCP semantics, the mobile host now resides in the base station, and migrates from base station to base station as hand-offs occur. In a traditional system, an acknowledgment from the TCP receiver does not guarantee that the data has been received by the *application program*. Under METP, such a gap is now stretched between the application on the (physical) mobile host and the TCP/IP stack on the base station.

One implication of this design is that the mobile host becomes more dependent on the base station. Any failure in the base station will be seen by the outside world as a failure in the mobile host. For example, when a data packet has arrived at and been acknowledged by the base station, it is considered by the sender that the mobile host has already received it, but the base station may fail thereafter and thus the packet may never actually arrive at the mobile host. The mobile host should always take appropriate recovery measures after such a break-down as if the failure had happened in itself. We argue that this is tolerable since the base stations are part of the infrastructure and much less likely to fail than the mobile hosts themselves.

4.3 Communication Between a Mobile Host and the Base Station

Because TCP is eliminated between the mobile host and the base station (over the wireless link), METP has to guarantee reliable and in-order delivery over this link whenever needed. We note that now this link is hidden by the base station from the outside world running TCP/IP, so we are free to any efficient mechanism to achieve reliable delivery.

1. Retransmission at the link layer.

Since a wireless medium may encounter a high rate of packet losses, a need for explicit MAC layer acknowledgment for each data packet has been widely recognized within the IEEE 802.11 subcommittee. One of the recommendations [9] is to use CSMA/CA with priority ACK. In addition to CSMA/CA, which reduces the collision probability among multiple stations accessing the same medium, the proposal adds an immediate ACK to allow MAC level recovery from lost frames. For traffic with a single destination, an ACK is returned by the destination immediately after a successfully received frame. The host sending an ACK frame has higher priority for access of the medium than all others waiting to send a normal data frame. An ACK is not transmitted unless the CRC of the received frame is found correct. If no ACK is received immediately after a data frame transmission, the sender quickly times out and attempts a retransmission after a random back-off time delay to recover from the failure. This recover process continues until a pre-determined number of retransmissions fail, in which case the data frame is dropped. The upper layer can be notified whether an ACK is received or the link layer has given up retransmitting and dropped the frame.

2. Reliable delivery over the wireless link.

In addition to a quick attempt of retransmissions in case of data losses, the link layer retransmission mechanism described above makes it possible for METP to provide orderly transmission. When METP transmits a packet, it waits for the feedback from the link layer. If an ACK is received, it transmits the next packet. If the packet has been dropped by the link layer, the receiver *may or may not* have received the packet since an ACK may have been lost. In this case, METP assumes the packet has been lost and retransmits it. The receiver therefore gets the packets in order except perhaps consecutive duplicates. The sequence number in the METP header described in the next section enables the receiver to detect any duplicates.

The link may suffer a prolonged period of errors or the mobile host may have failed or moved away. METP retransmits the data packet after a back-off delay, and become idle after a few such attempts. It will be waken up if one of the following three happens.

(a) New traffic with the mobile host occurs. Such traffic includes beacons, agent solicitation or registration request [19], or other data packets. In this case, METP retransmits data packets immediately.

(b) Notification arrives, indicating that the mobile host has moved to the new base station. A hand-off process will follow. See Section 4.4.

(c) A long timeout occurs. Such a timeout is close to the standard TCP timeout that would close the connection. In this case, METP assumes either the other end over the wireless link is dead or the mobile host is no longer able to connect to any base station on the Internet. All connections involving the mobile host are terminated.

METP still has to provide flow control. Even if the reliable data delivery is guaranteed at the link level, packets may still be dropped if the buffer at the receiver is full. To avoid such packet losses, METP at the receiver periodically sends out a feedback packet, which can be thought of as a coarse-grain acknowledgment, to inform the sender of how much buffer space it currently has. METP at the sender will not transmit data unless it is sure through such feedbacks that the buffer at the receiver will not overflow.

3. Reduced header over the wireless link.

Since the wireless link is now hidden from the outside Internet, we can use a smaller header for packets exchanged between the mobile host and the base station. The only information we retain for the header is the IP addresses and port numbers of the source and the destination. The size of the new header would be 12 bytes for IPv4, in comparison with 28 bytes of the UDP/IP header and 40 bytes of the TCP/IP header.

We propose to further reduce the METP header size by employing the idea of header compression in [13] and [8]. We add 4 bytes to the full METP header to include a *connection ID*, a *sequence number* and a few flags. When a TCP connection is established, a full METP header is used and both the mobile host and the base station record a binding of the connection ID and the address information of this connection. The ensuing METP packets of this connection contain only the added 4 bytes in the header. Since the link layer reliability mechanism mentioned above enables the sender to detect any potential loss of packets, all possible performance degradation caused by losses that [8] tries to alleviate does not exist in our case. The connection ID is assigned by the base station and is invalidated when the connection is closed. In the case of UDP datagrams, the connection ID is assigned and a full METP header is used the first time the source-destination pair appears. The connection ID binding expires after a time-out and a full METP header is sent.

A full METP header format is shown in Figure 2. A reduced METP header consists of only the first 4 bytes of a full header.

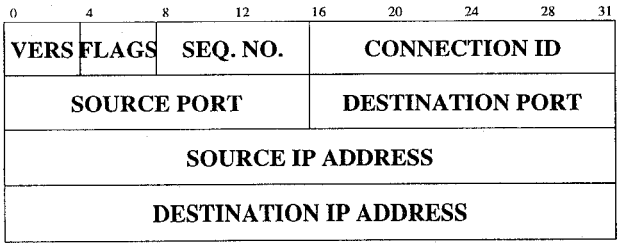

Figure 2: Format of a Full METP Header (with IPv4 Addresses)

4.4 Hand-Offs

When a hand-off occurs, all information of the TCP connections opened on behalf of the mobile host, including their states, current sending and receiving windows, and all data in their sending and receiving buffers, has to be handed over to the new base station. We require that the new base station, while receiving a registration reply from the home agent, notify the old base station at the same time, and the old base station in turn open a TCP connection with the new base station and send the above information.

If the new base station receives any data packet of a TCP connection *before* it completes setting up this connection, it should buffer the packet and move it to the receiving buffer as soon as the set-up completes and there is enough buffer space for it. If the new base station runs out of memory for such temporary buffer space, which we expect to happen rarely, it will drop the packet.

Since the connection ID in the METP header is assigned by the base station, it may have to be changed when the connection is moved to a new base station to avoid conflicts. Therefore, all connection ID bindings at the mobile host should be invalidated, and a full header should be used the first time a packet of a connection is sent. We note that the flow of the sequence numbers described in the previous paragraph remains unchanged during the lifetime of a connection, unaffected by a hand-off.

The new base station restarts activities of an inherited TCP connection as soon as it receives the information about the connection except the data in sending and receiving buffers. It sends out data to one end as soon as they are available, from either the other end or the old base station. On the other hand, if the new base station does not have enough buffer space for such an inherited TCP connection, it slows down the old base station handing over connections through the usual TCP advertised window mechanism, and temporarily buffers any overflow segments from the mobile host or the corresponding host.

Because in most cases, a hand-off does not cause packet losses in a TCP connection but merely shrinks the advertised window, no congestion control mechanisms will be activated at the sender running traditional TCP, and hence the traffic can quickly recover to its previous level after the hand-off completes.

4.5 Impact on Application Programs

There is no change required for application programs currently utilizing traditional TCP/IP to run on top of METP. It is achieved by keeping intact the socket (or other similar) interface between the applications and the system. If an application wants to open or close a connection, METP handles the socket calls traditionally handled by TCP/IP stack, talks to its counterpart at the base station, where the connection is created or closed on behalf of the mobile host. When an application wants to listen to a port (passive open), METP notifies the base station and if there is any connection opening request arrived for that port, a connection is opened at the base station and a notification is sent to the listening application through METP. METP handles sending and receiving socket calls in a similar manner.

In short, the METP design maintains an illusion for an application program as if it were talking to a traditional TCP/IP stack on

the local host while it is really communicating with the base station through METP.

5 Simulation Results

5.1 Simulated Environment

We have measured the TCP performance with METP through simulations. Our simulation is based on the Network Simulator (*ns*) developed by the Lawrence Berkeley National Laboratory [17]. We incorporated into the simulator the basic Mobile IP mechanism and METP. For comparison, we added four other schemes:

1. A straightforward split connection TCP. Such a split connection TCP is very similar to I-TCP, where a usual TCP connection between a mobile host and a fixed host splits at the base station, and the TCP control blocks are transferred to the new base station when a hand-off occurs. We call this "split connection" in the following discussions.
2. A split connection TCP with selective ACK over the wireless link, similar to the mechanism proposed by [20]. We call this "split SACK" for short.
3. A TCP-aware link layer together with multicast forwarding and intelligent buffering. This is very similar to the SNOOP module with a smooth hand-off optimization proposed in [3]. We call this "LL-MC" (Link Layer with MultiCast) for short.

For simplicity in our simulation, the mobile host invites a base station to join the multicast group that is the care-of address of the mobile host, and the base station never leaves this multicast group. Because we do not measure the network utilization but only TCP throughput (described below), this may only benefit performance in simulation. In addition, the administrative overhead of the multicast group management is also ignored.

The simulated network topology is shown in Figure 3. The numbers beside a link indicate the bandwidth and delay of the link. We tried to emulate a mobile host talking to a fixed one across the United States.

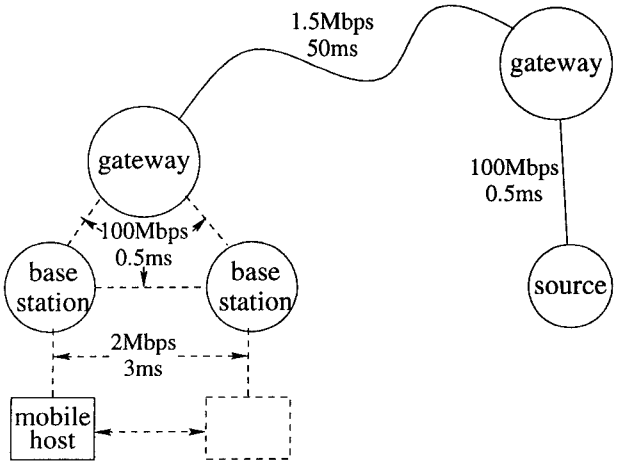

Figure 3: The Network Topology in the Simulation

The source (fixed host) transfers a file of size 1 MB to the mobile host using a TCP connection. The mobile host constantly switches between the two base stations. The time period during which the mobile host stays with one base station is a normally distributed random number, with average varying from 2 to 16 seconds and standard deviation 0.1 seconds. We used such short hand-off intervals to examine the performance in such an environment as a micro-cellular network, where split connection schemes potentially perform poorly due to the frequent hand-over of connection states and buffers. For comparison, we also did simulations where no hand-offs occurred.

The wireless link provides link-layer acknowledgments and retransmissions, as described in Section 4.3. The number of retransmissions the link will try before giving up and informing the upper layer ranges from 0 to 2.

As the loss model in our simulation, we use the improved two-state Markov model proposed by Nguyen et al., based on observations of the loss behavior of the WaveLAN interface through a trace-based approach [18].

5.2 Results

For each set of parameters described in the previous section, we ran 1000 independent simulations and measured the throughput. In addition to the average throughput, we also calculated 95% confidence intervals. The average throughput and confidence interval are shown in Figures 4 through 7.

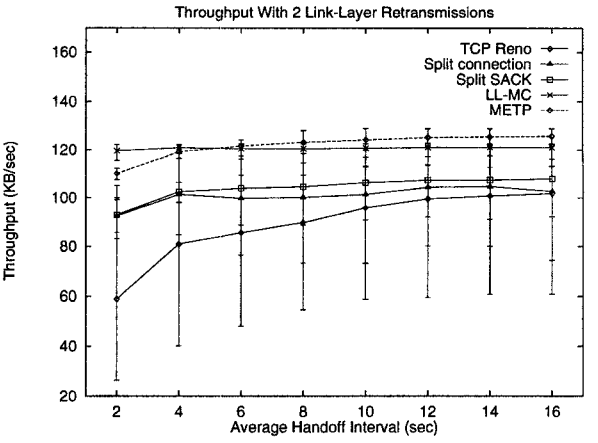

Figure 4: Average throughput, with varying hand-off interval and buffer size at the base station equal to 16 KB.

We have the following observations:

1. While the split connection TCP performs better than the traditional TCP and using selective ACK improves the throughput slightly more, METP gives a more substantial improvement. For example, when the base station allows a buffer of 16 KB, the average hand-off interval is 8 seconds per hand-off, and the link provides 2 retransmission attempts, METP achieves a 37% improvement over TCP Reno, 23% over simple split connection, 18% over split with SACK and 2% over SNOOP with smooth hand-off.
2. Without enough buffer space at the base station, METP and other split connection schemes do lose a large amount of their advantage (Figure 5). But METP keeps its lead over others. When there is a noticeable disconnection period during a hand-off, the

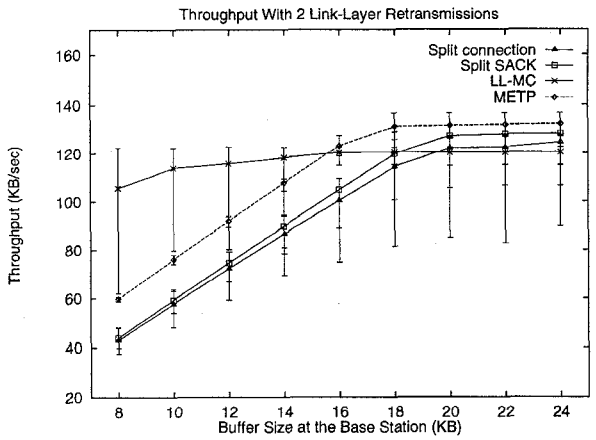

Figure 5: Average throughput, with varying buffer size at the base station and average hand-off interval equal to 8 seconds per hand-off.

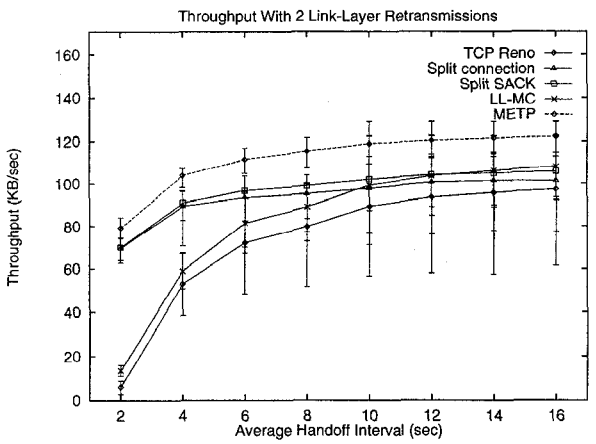

Figure 6: Average throughput, with buffer size at the base station equal to 16 KB, and a 0.5 second disconnection with either base station during each hand-off.

SNOOP-like scheme suffers a steep performance drop as the buffer size decreases (Figure 6). This is probably because with too small buffer space, the new base station may have to flush out packets that are not yet delivered to the mobile host due to a hand-off and therefore cannot hide the loss from the TCP source. On the other hand, because SNOOP-like scheme immediately passes along packets to the mobile host while buffering them, it is hardly affected by the buffer size when hand-offs are relatively smooth and outperforms all other schemes as the buffer size becomes very small. We note that since we do not take into account the cost of the multicast hand-off scheme, in reality LL-MC may be more expensive.

3. The throughput performance of METP is not greatly affected by the hand-off interval or the distance between the mobile host and the base station. Without link-layer retransmission to hide the real loss, METP loses only 7.3% of the throughput as the distance grows from 10 to 130 feet, in which case the connection is less stable and suffers higher bursty losses. In contrast, with the same change in distance, LL-MC loses 32.3%, the split TCP with

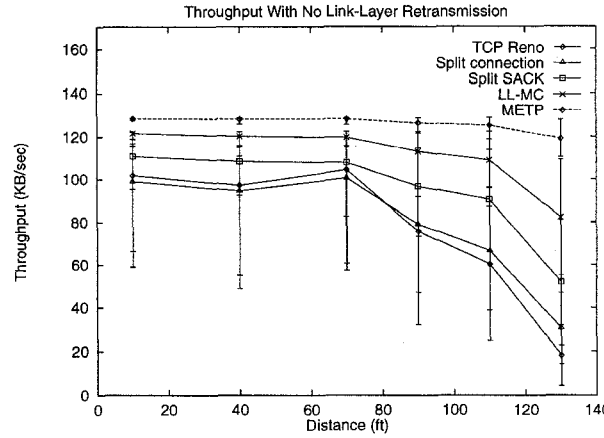

Figure 7: Average throughput, with buffer size at the base station equal to 16 KB and the average hand-off interval equal to 8 seconds per hand-off.

SACK 53.2%, the simple split TCP 69.2%, and TCP Reno 82.2% in throughput. We also note that as the bursty loss rate grows, the advantage of split SACK over simple split connection scheme becomes larger. This shows that TCP with selective ACK is better for dealing with bulk losses, a result echoed by Yavatkar and Bhagawat [20] and Balakrishnan et al. [2].

4. The performance of METP is also consistent in that it has smaller confidence intervals than other split connection schemes and TCP Reno in nearly all cases. The confidence intervals of the split TCP and TCP Reno are significantly larger where the loss rate is higher.

5. Splitting a TCP connection to hide the distinctly different wireless link from the source can avoid the inappropriate reactions of traditional TCP (mainly congestion control mechanisms) and hence improve the throughput. However, the simple split connection approach probably has gone only half-way, that is, we still need a better reliable-delivery protocol specifically adapted to the wireless link. Over a single, lossy link, TCP's coarse-grain time-out often causes an unnecessarily long silent period and hence degrades performance. In contrast, METP itself contains *no* time-out mechanisms during the normal operation. It relies on the link layer, which sets a fine-grain time-out since it deals with only one link. METP retransmits a packet *immediately* after the link layer reports a drop due to such a fine-grain time-out and therefore avoids long silent periods.

6 Conclusion and Future Work

There are two basic ideas behind the design of METP. First, we believe that the network protocols running over the wireless link should be kept simple. Since the wireless link is usually the last hop in a data path, we are able to have the base station deal with the outside Internet and employ a simple protocol for the communication between the mobile host and the base station. Second, with the link-layer acknowledgment mechanism in place for wireless links, the transport layer providing in-order reliable data delivery over a single wireless link can cooperate with the link layer

to achieve significant performance improvement. METP's coarse-grain acknowledgment policy contributes a large part of its better performance. Exploiting the link-layer recovery mechanism distinguishes METP from other split connection schemes [1, 20, 11]. The simulation results presented in the previous section support this argument.

As in any effort to shift more responsibility from the mobile host to the base station, hand-off latency is always a major concern. But with well-connected base stations and generally fewer connections on a mobile machine at any time, performance improvements mentioned above can still outweigh this disadvantage, as suggested in our simulation results.

There have been several papers on providing reliability at the link layer. For example, Bhagwat et al. [4] deals with a base station serving multiple mobile hosts with different channel states. Such an approach can be easily incorporated into METP to further improve the performance.

There are several issues that can hardly be addressed through simulations. Two of them are the overhead for a base station system to actually create and hand over a TCP connection structure, and any potential savings in software overhead with a simpler protocol. To answer these questions, we plan to implement METP to measure the actual performance. Furthermore, we would like to see the performance of the unreliable datagram delivery (such as UDP) over METP and whether it can provide a better support for such applications as the network file system (NFS).

Acknowledgment

We would like to thank Charles Perkins of Sun Microsystems, Inc. and our colleagues at the University of Maryland, George Apostolopoulos, Ibrahim Korpeoglu, Frank Miller, Cynthia D. Rais and Wei Zhao for their comments and suggestions.

References

[1] A. Bakre and B. Badrinath. I-TCP: Indirect TCP for Mobile Hosts. In *Proceedings of the International Conference on Distributed Computing Systems*, Vancouver, Canada, May 1995.

[2] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz. A Comparison of Mechanisms for Improving TCP Performance over Wireless Links. *IEEE/ACM Transactions on Networking*, June 1997.

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

[4] P. Bhagwat, P. Bhattacharya, A. Krishna, and S. K. Tripathi. Using Channel State Dependent Packet Scheduling to Improve TCP Throughput over Wireless LANs. *ACM Wireless Networks*, 3(1):91–102, March 1995.

[5] K. Brown and S. Singh. A Network Architecture for Mobile Computing. In *IEEE INFOCOM '96*, pages 1388–1396, 1996.

[6] R. Cáceres and L. Iftode. Improving the Performance of Reliable Transport Protocols in Mobile Computing Environment. *IEEE Journal on Selected Areas in Communications*, 1994.

[7] S. E. Deering. *Multicast Routing in a Datagram Internetwork*. PhD thesis, Standford University, December 1991.

[8] M. Degermark, M. Engan, B. Nordgen, and S. Pink. Low-loss TCP/IP Header Compression for Wireless Networks. In *MOBICOM '96*, pages 1–14, Rye, New York, November 1996. ACM and IEEE.

[9] W. Diepstraten. *IEEE 802.11: Wireless Access Method and Physical Specification*. Doc: IEEE P802.11-93/70, May 1993.

[10] D. Duchamp and N. F. Reynolds. Measured Performance of a Wireless LAN. In *17th Conference on Local Computer Networks*, pages 494–499, Minneapolis, Minnesota, September 1992. IEEE.

[11] Z. J. Haas and P. Agrawal. Mobile-TCP: An Asymmetric Transport Protocol Design for Mobile Systems. In *ICC '97*, Montreal, Canada, June 1997.

[12] V. Jacobson. Congestion Avoidance and Control. In *ACM SIGCOMM '88*. ACM, 1988.

[13] V. Jacobson. *Compressing TCP/IP Headers for Low-Speed Serial Links*. Network Working Group, Request for Comments 1144, February 1990.

[14] P. Karn and C. Partridge. Improving Round-Trip Time Estimates in Reliable Transport Protocols. In *ACM SIGCOMM '87*. ACM, 1987.

[15] K. Keeton, B. A. Mah, S. Seshan, R. H. Katz, and D. Ferrari. Providing Connection-Oriented Service to Mobile Hosts. In *Proceedings of the First USENIX Symposium on Mobile and Location-Independent Computing*, August 1993.

[16] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow. *TCP Selective Acknowledgment Options*. Network Working Group, Request for Comments 2018, October 1996.

[17] Network Research Group, Lawrence Berkeley National Laboratory. *ns—LBNL Network Simulator*. URL: `http://www-nrg.ee.lbl.gov/ns/`.

[18] G. T. Nguyen, R. H. Katz, B. Noble, and M. Satyanarayanan. A Trace-based Approach for Modeling Wireless Channel Behavior. In *Proceedings of the Winter Simulation Conference*, December 1996.

[19] C. E. Perkins, editor. *IP Mobility Support*. Request for Comments 2002, October 1996.

[20] R. Yavatkar and N. Bhagawat. Improving End-to-End Performance of TCP over Mobile Internetworks. In *Mobile 94 Workshop on Mobile Computing Systems and Applications*, Santa Cruz, California, December 1994.