I-TCP: Indirect TCP for Mobile Hosts

Ajay Bakre
bakre@paul.rutgers.edu

B.R. Badrinath
badri@rags.rutgers.edu

Department of Computer Science
Rutgers University, Piscataway, NJ 08855.

DCS-TR-314
October, 1994

Abstract — IP-based solutions to accommodate mobile hosts within existing internetworks do not address the distinctive features of wireless mobile computing. IP-based transport protocols thus suffer from poor performance when a mobile host communicates with a host on the fixed network. This is caused by frequent disruptions in network layer connectivity due to — i) mobility and ii) unreliable nature of the wireless link. We describe the design and implementation of I-TCP, which is an indirect transport layer protocol for mobile hosts. I-TCP utilizes the resources of Mobility Support Routers (MSRs) to provide transport layer communication between mobile hosts and hosts on the fixed network. With I-TCP, the problems related to mobility and the unreliability of wireless link are handled entirely within the wireless link; the TCP/IP software on the fixed hosts is not modified. Using I-TCP on our testbed, the throughput between a fixed host and a mobile host improved substantially in comparison to regular TCP.

1 Introduction

Integration of mobile hosts into the existing internetwork consisting mostly of stationary hosts gives rise to some peculiar problems because of the special requirements of the small low power mobile hosts and also because of the special characteristics of the wireless link. Several Mobile-IP proposals[20, 8, 17] have addressed the problem of delivering IP packets to mobile hosts regardless of their location. In theory one can use existing fixed network transport protocols such as UDP and TCP on the mobile

[1] This work was made possible in part by NSF equipment grant CDA 93–20300–EQT.

hosts to communicate with the fixed network. This naive approach however, gives rise to performance problems, especially when a mobile host switches cells or is temporarily disconnected. More seriously, all such proposals attempt to keep mobility, disconnection and other features of mobile hosts transparent above the network layer which does not allow any application specific handling of wireless features. On the other hand, use of a new protocol stack for mobile hosts causes interoperability problems. An Indirect model for mobile hosts[4] allows the development and use of specialized transport protocols that address the performance issues on the comparatively low bandwidth and unreliable wireless link. Protocols developed based on this model can also mitigate the effects of disconnections and moves while maintaining interoperability with existing protocols.

This paper presents the design and implementation of I-TCP which allows a mobile host to communicate over a transport layer connection with the fixed network via its current Mobile Support Router (MSR). The TCP connection with the fixed host is actually established by the MSR on behalf of the mobile host (MH). If the MH moves to another cell during the lifetime of the TCP connection, the new MSR takes over the connection from the old MSR. Experiments with I-TCP on our testbed show substantial throughput improvement over regular TCP when one of the end points is mobile.

2 Related Work

Previous research work in the areas related to network protocols for mobility and low speed links can be broadly classified as follows.

2.1 Solutions for Slow and Lossy Links

Problems related to the unreliable nature of wireless media are somewhat similar to the ones which surfaced in the early eighties when telephone and serial lines were used to connect personal computers to the Internet. Thinwire protocols[7] attempted to alleviate some of those problems. Header compression[11] for TCP connections was suggested for improving the response time of interactive applications such as telnet on low speed links. Although these solutions are applicable to some extent to wireless links, they do not deal with host mobility. In addition, such solutions cannot adapt to the changes in the wireless link characteristics such as available bandwidth, which may change from one wireless cell to another. Link layer retransmissions can be used on error-prone wireless links to bring their error rate on par with that on the wired networks but such an approach interferes with the end-to-end retransmissions of TCP and does not always result in improved performance[6].

2.2 TCP/IP in Mobile Environment

Mobility can give rise to severe performance problems in TCP throughput[5]. The main reasons for throughput degradation is the loss of TCP segments during cell crossovers especially with non-overlapped cells. Lost segments trigger exponential back off and congestion control at the transmitting host and the congestion recovery phase may last for several seconds even after network layer communication is reestablished in the new wireless cell. Fast retransmission coupled with modification of the TCP software on the mobile hosts[5] solves only part of the problem because the transmitting host still performs a *slow start* if more than one segment is lost per window, thus limiting the effective throughput[12]. Other proxy-based approaches have been suggested[2] for mobile hosts but they do not pertain to the transport layer.

3 Indirect TCP Overview

This section gives an overview of indirect TCP and describes the benefits of using indirection at the transport layer. We begin with a brief description of the Indirect Protocol model [4] on which indirect TCP is based.

3.1 Indirect Model for Mobile Hosts

The indirect protocol model for mobile hosts suggests that any interaction from a mobile host (MH) to a machine on the fixed network (FH) should be split into two separate interactions — one between the MH and its mobile support router (MSR) over the wireless medium and another between the MSR and the FH over the fixed network. This provides an elegant method for accommodating the special requirements of mobile hosts in a way that is backward compatible with the existing fixed network. All the specialized support that is needed for mobile applications and for the low speed and unreliable wireless medium can be built into the wireless side of the interaction while the fixed side is left unchanged.

At the transport layer, use of indirection results in the following benefits:

1. It separates the flow control and congestion control functionality on the wireless link from that on the fixed network. This separation is desirable because of the vastly different characteristics of the two

kinds of links — the fixed links (ethernet or long-haul links and ATM in the future) are becoming faster and more reliable every day whereas the wireless links (especially the outdoor links) are still very slow and are extremely vulnerable to noise and loss of signal due to fading which result in higher bit error rates.

2. A separate transport protocol for the wireless link can support notification of events such as disconnections, moves and other features of the wireless link such as the available bandwidth etc. to the higher layers which can be used by *link aware* and *location aware* mobile applications.
3. Indirection allows the base station (mobile support router or MSR) to manage much of the communication overhead for a mobile host. Thus, a mobile host (e.g. a small palmtop) which only runs a very simple wireless protocol to communicate with the MSR can still access fixed network services such as *WWW* which may otherwise require a full TCP/IP stack running on the mobile.

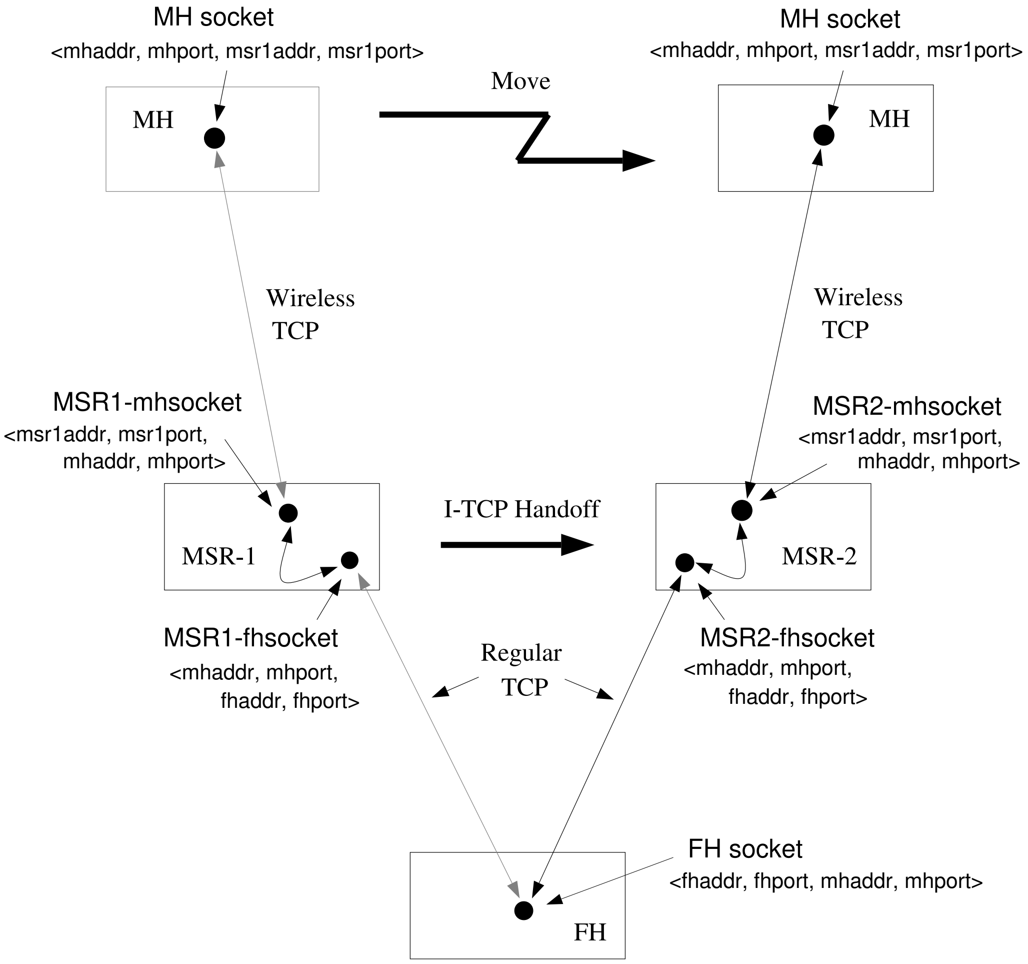
3.2 I-TCP Basics

I-TCP is a transport layer protocol for mobile hosts which is based on the Indirect Protocol model. I-TCP is fully compatible with TCP/IP on the fixed network and is built around the following simple concepts:

1. A transport layer connection between an MH and an FH is established as two separate connections — one over the wireless medium and another over the fixed network with the current MSR being the center point.
2. If the MH switches cells during the lifetime of an I-TCP connection, the center point of the connection moves to the new MSR.
3. The FH is completely unaware of the indirection and is not affected even when the MH switches cells i.e. when the center point of the I-TCP connection moves from one MSR to another.

When a mobile host (MH) wishes to communicate with some fixed host (FH) using I-TCP, a request is sent to the current MSR (which is also attached to the fixed network) to open a TCP connection with the FH *on behalf of* the MH. The MH communicates with its MSR on a separate connection using a variation of TCP that is tuned for wireless links and is also aware of mobility. The FH only sees an image of its peer MH that in fact resides on the MSR. It is this image which is handed over to the new MSR in case the MH moves to another cell.

Figure 1 I-TCP Connection Setup



As an example, figure 1 shows the setup for an I-TCP connection. The mobile host (MH) first establishes a connection with a fixed host (FH) through MSR-1 and then moves to another cell under MSR-2. When the MH requests an I-TCP connection with the FH inside the cell of MSR-1, MSR-1 establishes a socket with the MH address and MH port number to handle the connection with the fixed host. It also opens another socket with its own address and some suitable port number for the wireless side of the I-TCP connection to communicate with the MH.

When the MH switches cells, the state associated with the two sockets for the I-TCP connection at MSR-1 is handed over to the new MSR (MSR-2). MSR-2 then creates the two sockets corresponding to the I-TCP connection with the *same* endpoint parameters that the sockets at MSR-1 had associated with them. Since the connection endpoints for both wireless and the fixed parts of the I-TCP connection do not change after a move, there is no need to *reestablish* the connection at the new MSR. This also ensures that the indirection in the TCP connection is completely hidden from the FH.

I-TCP Semantics One consequence of using I-TCP is that the TCP acknowledgments are not end-to-end but instead we have separate acknowledgments for the wireless and the wired parts of the connection. Most applications that use TCP for bulk data transfer such as ftp however, also have some kind of support built-in for application layer acknowledgment and error recovery. Such acknowledgments are often required because TCP does not provide any notification to the sending application when the data is actually received by the peer application. One can therefore argue that using I-TCP does not yield *weaker* end-to-end semantics in comparison to regular TCP, *provided that there are no MSR failures and that the MH does not stay disconnected from the fixed network for too long*. It is important to note however that the wireless link between the MSR and the MH is highly fragile and so it is desirable that applications using I-TCP provide some mechanism for error recovery to deal with failures on the wireless link.

3.3 I-TCP Interface at the MH

To establish an I-TCP (indirect) connection with a remote host, MH applications must use special I-TCP calls instead of the regular socket system calls. I-TCP calls are provided to replace connect, listen, accept and close socket system calls and have the same interface as their corresponding socket system calls. The I-TCP calls only provide a wrapper around the regular socket system calls to perform the necessary handshake with the MSR to open or close an I-TCP connection. Once an I-TCP connection is established, normal socket system calls can be used to send or receive data on the connection.

4 Performance Results

We present performance figures for experiments conducted using the ttcp benchmark which measures TCP throughput between two hosts. The throughput experiments were conducted on our wireless testbed which is described below.

4.1 Experimental Wireless Testbed

The wireless testbed had three MSRs, all of them 33 MHz 386 PC-ATs with 16 MB memory and 400 MB disk drives. The mobile host was a 66 MHz 486 PC-AT. All the machines use 2Mbps NCR

Wavelan cards for wireless communication. The MSRs are also connected to 10 Mbps ethernet segments which are part of a single administrative domain. The MSRs run Mach microkernel with Unix server (MK84/UX40)[1] and use Columbia Mobile-IP protocol to support wireless cells. The MH has similar configuration but without Columbia Mobile-IP. I-TCP daemon processes running on the MSRs provide support for I-TCP connections. Modified versions of *msrmicp* and *mhmicp* programs run at the MSRs and the MH respectively which constitute the user level part of Mobile-IP. The fixed hosts used were Sparc machines running SunOS.

Two MSRs in the setup described above were used for supporting the wireless cells for the MH while the third one merely acted as a gateway to our mobile subnet which routed packets destined for the MH to one of the other two MSRs. Thus, all the packets arriving at the MH had to go through 1 hop of IPIP encapsulation (from the gateway to the current MSR) in a steady state and possibly two hops for a brief interval immediately following a move. We experimented with two distinct cases to study the performance of I-TCP for connections spanning over local area and wide area networks i.e. — *i)* when the FH to MH communication involves only a few hops within our campus and *ii)* when the FH to MH communication involves a long-haul link over the Internet.

Our experiments were inspired by similar experiments reported by Caceres and Iftode[5] to study the effect of mobility on reliable transport protocols. Tables 1 and 2 compare the end-to-end throughput of an I-TCP connection between an MH and a fixed host (FH) with that of a direct TCP connection for local area and wide area connections respectively. In all our experiments, the FH sent a few megabytes of data (4 MB in case of local area and 2 MB in case of wide area) to the MH using a window size of 16 KB. We chose to make the MH to be the receiving host which we expect to be a typical situation with most mobile applications that will download more data from the fixed network rather than sending data over the uplink. Cell switching was implemented in software to allow precise control over the instant when the MH crosses cells. The end-to-end throughput was measured at the MH under four different cell configurations —

i) **No Moves —** The MH stays in one wireless cell during the lifetime of a connection.

ii) **Moves between overlapped cells —** MH switching between overlapped cells every 8 seconds such that the MH stays in contact with the previous MSR during handoff. For a brief period after switching cells, the MH continues to receive packets from the previous MSR before the Mobile-IP routing adjustments take effect.

iii) **Moves between non-overlapped cells with 0 second between cells —** In case of non-overlapped cells, the cell boundaries are sharply defined and therefore no communication is possible with the previous MSR after a move to another MSR. The MH starts looking for a beacon from the new MSR immediately after a move and thus in the worst case the link layer connectivity may be lost for one full interval between successive beacons which was 1 second in our testbed. The cell switching again occurs every 8 seconds.

iv) **Moves between non-overlapped cells with 1 second between cells —** Same as in *iii)* above but now the MH starts looking for a beacon 1 second after moving out of the previous cell. As in the previous case, an additional 1 second may elapse before a beacon is received by the MH and the link layer connectivity is reestablished.

The wireless cells were completely overlapped in our setup and non-overlapped cells were simulated with the two MSRs transmitting using different Wavelan (MAC layer) network IDs. Cell switching was implemented in software to allow better control on the timing of cell crossovers.

Table 1 I-TCP Throughput Performance over Local Area

Connection type	**No moves**	**Overlapped cells**	**Non-overlapped cells with 0 sec between cells**	**Non-overlapped cells with 1 sec between cells**
Regular TCP	65.49 KB/s	62.59 KB/s	38.66 KB/s	23.73 KB/s
I-TCP	70.06 KB/s	65.37 KB/s	44.83 KB/s	36.31 KB/s

4.2 Performance over Local Area

In case of local-area experiments, we observed that I-TCP performed slightly better compared to regular TCP when the MH stayed within one cell. This is remarkable considering the copying overhead incurred by I-TCP at the MSR. In the second case when the MH switches between two completely overlapped cells, the link-layer connectivity is maintained at all times since the MH is in contact with both the new MSR and its previous MSR during handoff. There is still some degradation in throughput since the TCP segments that are in transit during handoff are delayed because of IP layer routing adjustments by the MSRs. I-TCP performance suffers only marginally in this case despite the additional overhead of I-TCP state handoff between the two MSRs on every move. We believe that the main reason for improved performance with I-TCP in the first two test cases is that the sending host (FH) sees more uniform round-trip delays for TCP segments as compared to the regular TCP. Loss of TCP segments over the wireless link, although infrequent, was also responsible for the difference in performance since I-TCP seemed to recover faster from a lost packet than regular TCP.

The two cases of non-overlapped cells where the MH lost contact with the fixed network (for 0 and 1 second respectively) before such contact was reestablished at the new MSR, affected the end-to-end throughput more severely. With regular TCP, congestion control kicked in at the FH on every handoff because of packet loss and it took some time after a cell crossover before the FH was able to send data again at full speed. In addition, the exponential back off policy of TCP resulted in the FH going into long pauses that continued even after the MH was ready to communicate in its new cell. In case of I-TCP however, a cell crossover by the MH manifested itself in the form of shrinking receive window size at the MSR which forced the FH to stop sending data when the MSR buffers were full. After a handoff, the new MSR could accept more data from the FH and the data rate on the connection quickly came back to normal. Congestion control did kick-in on the wireless link between the MSR and the MH however and so did exponential back off. We found that a simple reset of the TCP retransmission timer at the new MSR immediately after an I-TCP handoff forced the MSR to initiate a slow-start on the wireless link, and was enough to quickly get the wireless part of I-TCP out of congestion. In the worst case when the MH lost connectivity with the fixed network for 1 second, I-TCP showed an improvement by a factor of about 1.5 over regular TCP.

4.3 Performance over Wide Area

Our wide area experiments highlight the benefits of I-TCP even more clearly. Because of relatively long round-trip delays over wide area connections, any packet losses over the wireless link severely limit the end-to-end throughput of regular TCP. This is because *the time needed to recover from falsely triggered congestion control increases with the round-trip delay*. Similarly any perturbations (such as cell crossovers or transient changes in the observed round-trip delay) have a more drastic effect over wide

Table 2 I-TCP Throughput Performance over Wide Area

Connection type	**No moves**	**Overlapped cells**	**Non-overlapped cells with 0 sec between cells**	**Non-overlapped cells with 1 sec between cells**
Regular TCP	13.35 KB/s	13.26 KB/s	8.89 KB/s	5.19 KB/s
I-TCP	26.78 KB/s	27.97 KB/s	19.12 KB/s	16.01 KB/s

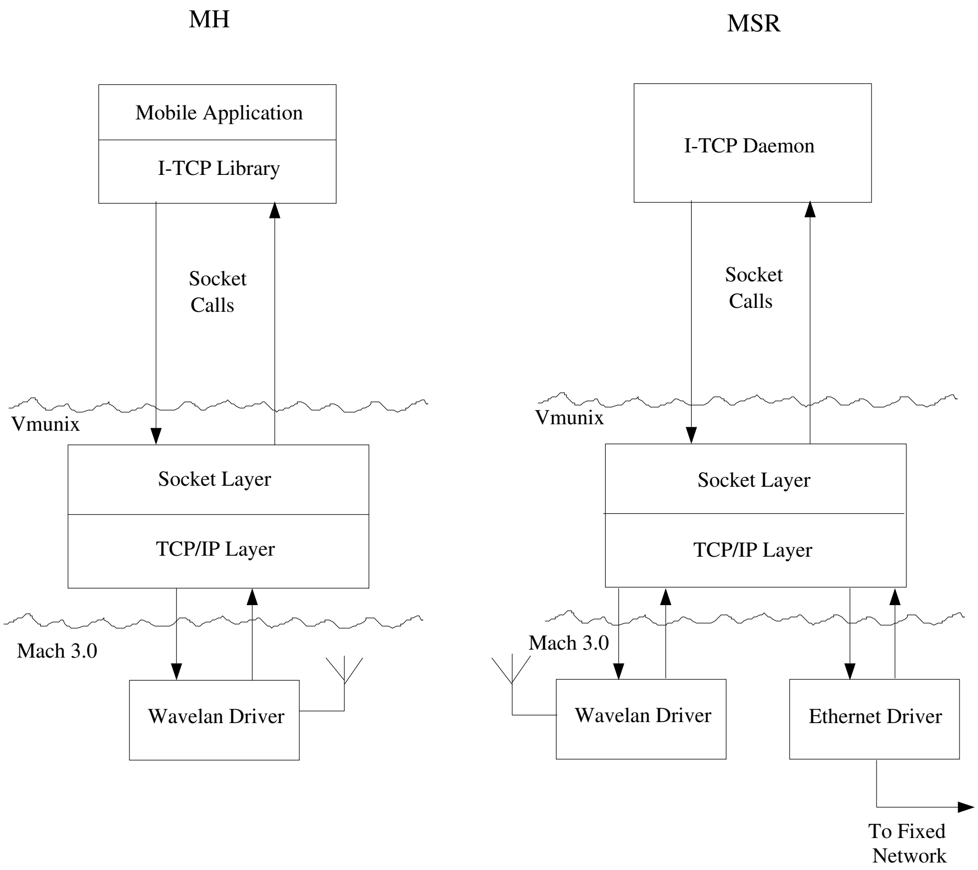
area connections than over local area connections. For the first two test cases i.e. when the MH stayed within once cell and when it switched between overlapped cells, the observed performance of I-TCP was about 2 times better than that of regular TCP. We did not observe any significant degradation in performance with the MH switching between overlapped cells either with I-TCP or with regular TCP which suggests that the effect of variation in round trip delay because of IP level routing changes was negligible for wide area connections.

In case of moves between non-overlapped cells, the throughput with regular TCP dropped to almost a third (61% degradation) of the no-moves throughput in the worst case when the MH lost contact with the fixed network for 1 second. With I-TCP, the corresponding degradation in throughput was only 40%. The net effect was that I-TCP throughput in the worst case was 3 times better than that of regular TCP. The main reason for this improved performance with I-TCP is that the retransmissions due to packets lost on the wireless link were confined only to the wireless part of I-TCP which can recover much faster from the congestion control phase because of the following two factors — *i)* much shorter round-trip delay between the MH and the MSR as compared to the delay between the MH and the FH and *ii)* we reset the retransmission timer at the MSR immediately after a handoff.

5 I-TCP Implementation

This section describes the implementation of various software components that constitute the I-TCP system. These include modifications to the TCP and Mobile-IP code on the MSRs. No modifications are needed in the Unix kernel at the MH for I-TCP to work.

Figure 2 I-TCP Implementation on Mach 3/Unix



5.1 MSR Kernel Support

First we describe the changes required in the network management module of the Unix kernel at the MSRs followed by a description of other user level programs at the MSR which manage the I-TCP connections.

TCP/IP layer support At any MSR, we allow binding sockets to the addresses and port numbers of MHs that are currently local to the MSR. This is essential to grab the TCP packets originating from fixed hosts which are addressed to the MH on a per connection basis. This also allows us to move an I-TCP socket to another MSR without changing any connection parameters and therefore no cooperation from the fixed peer is needed to move the TCP endpoint from one MSR to the other. The port numbers used for the I-TCP sockets on behalf of one MH cannot conflict with the same port numbers used by the MSR or by other MHs having indirect sockets through the same MSR because the corresponding addresses are different.

A small change is needed to the IP input routine which sends the IP packets that are addressed to I-TCP port numbers at the MH, upwards to the TCP layer instead of forwarding them to the MH. A list of such I-TCP port numbers is maintained at the MSR on a per MH basis.

Mobility Support As an MH moves from one cell (say, under MSR-1) to another (under MSR-2), all the I-TCP sockets active at MSR-1 on behalf of the MH must be moved to MSR-2. Moving a connected socket from one machine to another is a fairly complex task. In addition to transferring the state maintained by the socket and TCP layers from MSR-1 to MSR-2, it involves restarting the connection at MSR-2. Also, this migration of sockets needs to be completely transparent to the fixed host at the other end of the connection.

The I-TCP state handoff was carefully implemented with kernel support to be efficient even with relatively frequent moves. We also buffer the TCP segments that are in transit during a handoff at the new MSR even though they cannot be immediately processed. This is necessary to avoid congestion on either side (fixed or wireless) of the I-TCP connection. After an I-TCP handoff, we reset the TCP retransmission timer on the wireless side so that the MSR immediately begins a *slow start*. Complete details of handoff are described in a later section. Here we restrict ourselves to describing the primitives needed to achieve an I-TCP socket handoff. In particular, we implemented basic primitives for I-TCP sockets to do the following:

1. Freeze a connected socket and capture its state.
2. Create a *connected* socket *without* any cooperation from the peer.
3. Establish the state of a socket and restart the connection.
4. Delete (*not* close) a socket which has been moved to another MSR.

5.2 MSR I-TCP Daemon

An I-TCP daemon process running on every MSR is responsible for managing all the I-TCP connections through that MSR for all the MHs that are currently local to the MSR. Managing the I-TCP connections at the MSR from a process in user space involves additional copying overhead on each half of the duplex connection. The data sent by the MH on an I-TCP connection has to go up through the TCP and socket layers in the Unix kernel and into the user space and down again on the fixed side of the connection through the socket and TCP layers of the kernel to the IP output routine. On the other hand, a regular TCP packet from the MH for a direct connection would be forwarded by the IP layer in the kernel to the fixed network with nominal processing overhead.

Figure 3 I-TCP MH Modules

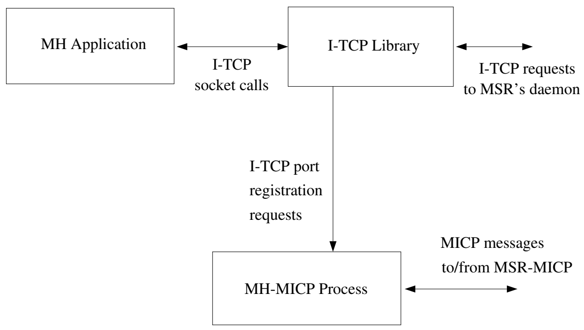
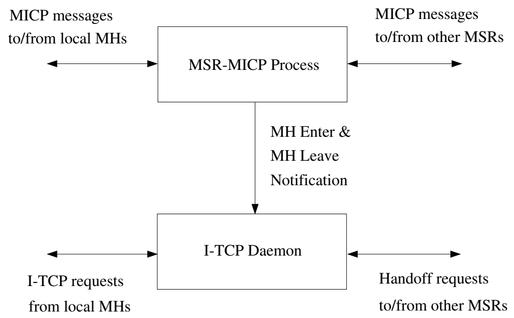


Figure 4 I-TCP MSR Modules



The I-TCP daemon is a threaded process with different modules to communicate with the local MHs, the *msrmicp* process on the MSR and with the I-TCP daemons on other MSRs. In its current form the daemon performs the following functions:

1. Handle requests from locally registered MHs for opening I-TCP connections. Such requests can be either passive (listening for connections) or active (initiating a connection with a remote host).
2. Copy data from wireless side of I-TCP connections to the fixed network side and vice versa.

3. Perform I-TCP handoffs in coordination with similar daemons at other MSRs and the local *msrmicp* process.

5.3 MH I-TCP Library

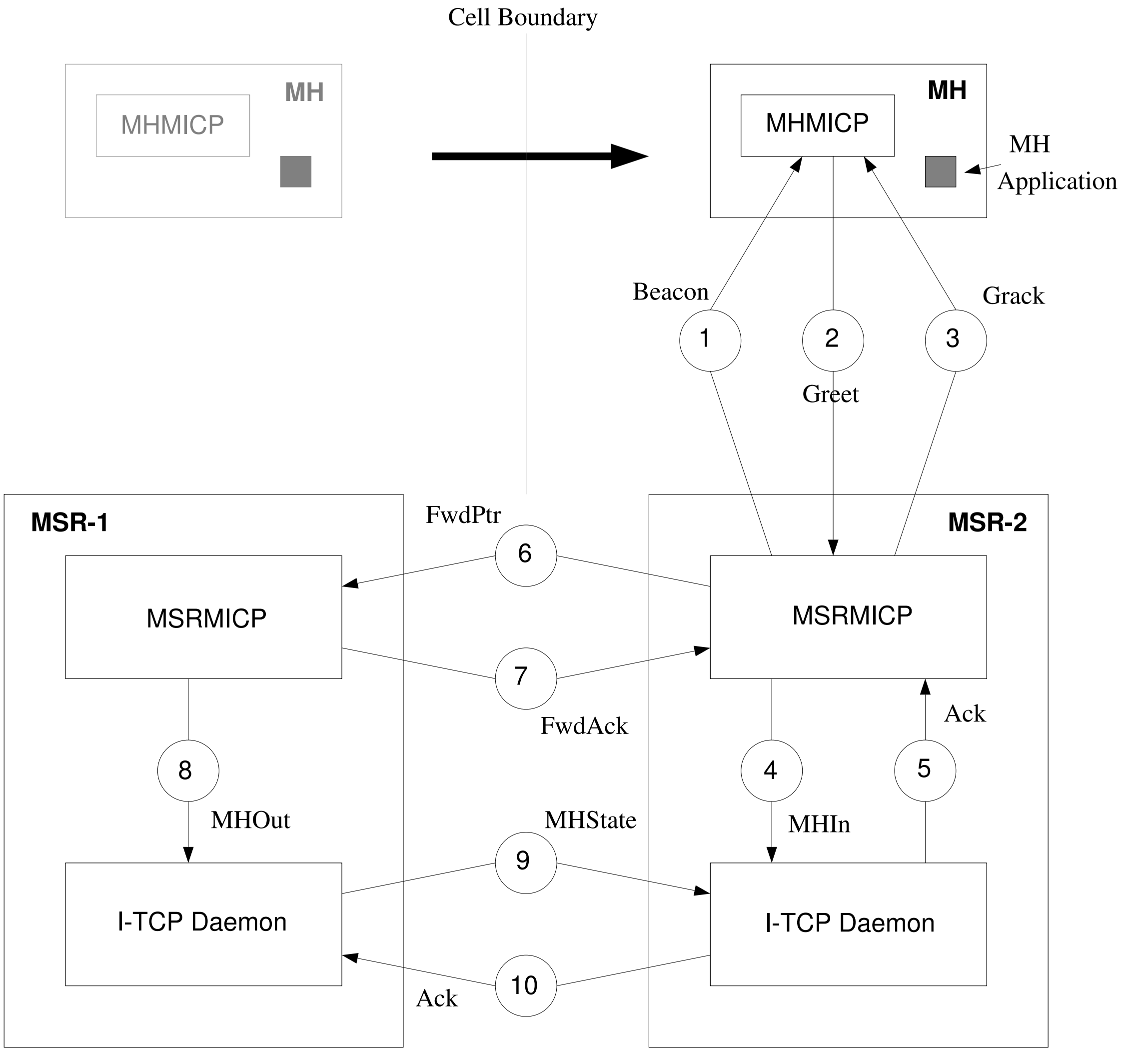
The I-TCP library on the MH provides the Application Programmer's Interface (API) for the I-TCP functions. This library provides a familiar interface similar to the socket related system calls in Unix. This library manages the indirect connections on a per process basis in coordination with the I-TCP daemon of the current MSR and the local *mhmicp* process. MH applications that wish to avail of I-TCP instead of using regular TCP simply need to replace the regular socket system calls for connection initiation and termination by their equivalent I-TCP calls.

5.4 Handoff Management

Figure 5 shows the handoff sequence for I-TCP connections when an MH that has open I-TCP connections moves from one cell (under MSR-1) to another (under MSR-2). The handoff procedure is closely integrated with the MH registration procedure of Columbia Mobile-IP for the sake of efficiency and thus the user level modules, namely the *msrmicp* and the *mhmicp* processes running at the MSR and the MH respectively, also participate in an I-TCP handoff. In the following description the item numbers correspond to the step numbers shown in figure 5.

1. A beacon is received by the MH from the new MSR (MSR-2).
2. The *mhmicp* process sets the new MSR to be the default router and sends a greeting message to the MSR containing the connection endpoints of all the active I-TCP connections at the MH and also the address of its previous MSR (MSR-1).
3. The *msrmicp* process at MSR-2 sends an acknowledgment for the greeting to the MH.
4. The *msrmicp* process at MSR-2 sends an MHIn message to the local I-TCP daemon containing the list of I-TCP connection endpoints received from the MH.
5. The I-TCP daemon establishes sockets for both the wireless and the fixed network parts of the I-TCP connections for the newly registered MH and prepares itself for an I-TCP handoff request from MSR-1. The daemon then sends an ACK to the local *msrmicp* process.
6. The *msrmicp* process sends a forwarding pointer to MSR-1.
7. The *msrmicp* process at MSR-1 sends a forwarding ACK to MSR-2.

Figure 5 I-TCP Handoff Sequence



8. The *msrmicp* process at MSR-1 sends an MHOut message to its local I-TCP daemon with the address of the MH that moved out.
9. On receiving and MHOut message, the I-TCP daemon freezes all the I-TCP connections for the indicated MH. It then makes a handoff request to the I-TCP daemon of MSR-2 to make sure it is ready and then sends the state of each I-TCP connection to MSR-2.
10. The I-TCP daemon at MSR-2 receives the state of each I-TCP connection for the newly registered MH and restarts each connection. It then sends an ACK to MSR-1 signalling the completion of I-TCP handoff.

The handoff procedure described above assumes that the wireless cells are non-overlapping i.e. no direct communication is possible between the MH and its previous MSR after switching cells. In case of overlapped cells, the MH can continue to receive IP packets during steps 1 through 6 of the above handoff procedure while it sends the outgoing IP packets through the new MSR. The I-TCP handoff thus does not interfere with other IP traffic to and from the MH. For the I-TCP connections, there is a brief interruption in the traffic between steps 6 through 10 of the handoff procedure. The TCP segments in transit during this short period are buffered (without processing) at the new MSR and are acknowledged as soon as complete state information is available for I-TCP connections at the new MSR.

With non-overlapped cells, the MH can start sending out IP packets immediately after step 1, but it cannot receive any IP packets until step 6 because the rest of the network does not know about its new location. This disruption in the network layer is inevitable with non-overlapped cells. For I-TCP connections, the I-TCP handoff has to be completed in step 10 before data can flow in both directions normally which causes a brief delay as in the case with overlapped cells.

6 Conclusion and Future Work

We have described I-TCP as a robust approach to improve transport layer performance in a mobile wireless environment. Our approach first confines the mobility related performance problems to the wireless link and then attempts to alleviate such problems by adapting the TCP/IP software on the wireless link in a way that requires no modifications to the hosts on the fixed network. I-TCP is particularly suited for applications which are throughput intensive. Experiments with I-TCP on our testbed showed greatly improved throughput in comparison to regular TCP under simulated mobility conditions. The performance improvement for wide-area connections was higher than for local-area connections.

We would like to study I-TCP performance in different wireless environments especially those with high error rates. We are also planning to build a flexible and lightweight transport protocol for the wireless side of I-TCP which can adapt to changes in the wireless environment and can support planned disconnections. Presentation layer services can also be built on top of I-TCP which will allow mobile applications to dynamically choose a format for data transmitted over the wireless medium. Other work includes testing throughput intensive applications such as ftp and mosaic with I-TCP.

7 Acknowledgments

We are thankful to Darrell Long (UC, Santa Cruz) for allowing us to use one of their machines for conducting experiments for wide area connections. We would also like to thank John Ioannidis for providing us with the source code of Columbia's Mobile-IP implementation. The Wavelan driver for Mach used in our experiments was originally written by Anders Klemets and adapted for our network configuration by Girish Welling. Arup Acharya provided valuable comments in the preparation of this paper.

References

[1] M. Accetta, R. Baron, W. Bolosky, D. Golub, R. Rashid, A. Tevanian, and M. Young. Mach: a new kernel foundation for UNIX development. In *Proc. of the USENIX 1986 Summer Conference*, July 1986.

[2] A. Athan and D. Duchamp. Agent-mediated message passing for constrained environments. In *USENIX Symposium on Mobile and Location-dependent Computing*, August 1993.

[3] B.R. Badrinath, A. Acharya, and T. Imielinski. Structuring distributed algorithms for mobile hosts. In *Proc. of the 14th Intl. Conf. on Distributed Computing Systems*, pages 21–28, June 1994.

[4] B.R. Badrinath, A. Bakre, T. Imielinski, and R. Marantz. Handling mobile clients: A case for indirect interaction. In *4th Workshop on Workstation Operating Systems*, October 1993.

[5] R. Caceres and L. Iftode. The effects of mobility on reliable transport protocols. In *Proc. of the 14th Intl. Conf. on Distributed Computing Systems*, pages 12–20, June 1994.

[6] A. DeSimone, M.C. Chuah, and O.C. Yue. Throughput performance of transport-layer protocols over wireless LANs. In *Proc. of Globecom '93*, December 1993.

[7] D.J. Farber, G.S. Delp, and T.M. Conte. A thinwire protocol for connecting personal computers to the Internet. *Request for comments 914*, September 1984.

[8] J. Ioannidis, D. Duchamp, and G.Q. Maguire. IP-based protocols for mobile internetworking. In *Proc. of ACM SIGCOMM*, pages 235–245, September 1991.

[9] J. Ioannidis and G.Q. Maguire. The design and implementation of a mobile internetworking architecture. In *Proc. of USENIX Winter Technical Conference*, January 1993.

[10] V. Jacobson. Congestion avoidance and control. In *Proc. of ACM SIGCOMM*, pages 314–329, August 1988.

[11] V. Jacobson. Compressing TCP/IP headers for low-speed serial links. *Request for Comments 1144*, February 1990.

[12] V. Jacobson, R. Braden, and D. Borman. TCP extensions for high performance. *Request for Comments 1323*, May 1992.

[13] S.J. Leffler, M.K. McKusick, M.J. Karels, and J.S. Quarterman. *The design and implementation of the 4.3BSD UNIX Operating System*. Addison Wesley, 1989.

[14] A. Myles and D. Skellern. Comparison of mobile host protocols for IP. *Journal of Internetworking Research and Experience*, 4(4), December 1993.

[15] J. Nagle. Congestion control in IP/TCP Internetworks. *Request for Comments 896*, January 1984.

[16] J. Postel. Transmission control protocol. *Request for Comments 793*, September 1981.

[17] Y. Rekhter and C. Perkins. Optimal routing for mobile hosts using IP's loose source route option. *Internet Draft*, October 1992.

[18] J.H. Saltzer, D.P. Reed, and D.D. Clark. End-to-end arguments in system design. *ACM Transactions on Computer Systems*, 2(4), November 1984.

[19] C.K. Siew and D.J. Goodman. Packet data transmission over radio mobile radio channels. *IEEE Transactions on Vehicular Technology*, 32(8):95–101, May 1989.

[20] H. Wada, T. Yozawa, T. Ohnishi, and Y. Tanaka. Mobile computing environment based on internet packet forwarding. In *Proc. of the USENIX Winter Technical Conference*, January 1993.