MOBILE-TCP: AN ASYMMETRIC TRANSPORT PROTOCOL DESIGN FOR MOBILE SYSTEMS

Zygmunt J. Haas

School of Electrical Engineering
Cornell University
Ithaca, NY 14853
haas@ee.cornell.edu

ABSTRACT

In this paper, we advocate the use, propose the design, and describe the implementation of an asymmetric transport-layer protocol for mobile systems. Portability of computing equipment is achieved by reduction in the physical parameters, which is facilitated by the ever increasing device integration density. To maintain constant time between recharging, reduction in the battery power of a portable system translates into reduced CPU cycles. Thus, to maintain a constant level of performance of mobile applications, there is the need to reduce the processing load of computing algorithms and communication protocols for mobile devices. One approach to offload processing of communication protocol from a mobile device without sacrificing the performance and features is through the use of asymmetric design. In an asymmetrically designed protocol, peer functions are implemented through algorithms and procedures that are of substantially different complexity, with the lower complexity procedures used on the mobile device. We have devised and implemented an example of such a communication protocol that provides the TCP functionality, yet places a significantly lower load on the mobile machine. Although Mobile-TCP is based on the indirect interaction model, the semantics of the TCP protocol are preserved. Details of our design are presented here.

INTRODUCTION AND MOTIVATION

Chief enabling technology of the ubiquitous multimedia access for Personal Communication Services is miniaturization — portability is everything! However, as the portable machines shrink in their physical parameters (weight and size), their performance capabilities are also reduced. Batteries are, and continue to be, an important weight and size factor in portable computers and devices. However, since portable devices are expected to operate at least 4-6 hours without battery replacement or recharge, one cannot just simply equip the portable devices with smaller batteries. Nevertheless, smaller power sources are possible, if such a reduction is accompanied by a corresponding reduction in the power requirements of the portable device's components. One way to accomplish this is by reducing the clock rate of the device.* From software processing point of view, such as protocol processing, lower clock rate corresponds to limited performance, such as increased delay or reduced throughput.

*Recall that dissipated power is: $P = C \cdot V^2 \cdot F$, where C is the circuit capacitance, V is the operational voltage, and F is the clock's frequency.

Consequently, a reduced complexity software (without compromise in functionality) would, indirectly, increased portability of future mobile devices.

The general challenge that we are addressing in this work is that of protocol and application design in a power-limited environment. More specifically, the primary concern of this work is the execution of transport-layer protocols on the portable device that is both computing- and communication-power limited. We have proposed and implemented a protocol that allows emulation of the TCP functionality between the mobile host and a stationary host within the network, in a way that reduces the processing requirements (i.e., complexity) on the mobile host and minimizes the use of the wireless medium. Reduction of complexity increases the protocol performance, allowing slower clock rate and, thus, reduced power expenditure. Minimization of the processing requirements eliminates the use of the wireless transmitter, reducing, again, the power usage.

Our approach taken here stems from the fact that a transport-layer connection between a mobile and a corresponding stationary host can be partition into two connections: the connection between the mobile host and a local fixed host (referred to here as the Mobile-Gateway), and the connection between the local fixed host and the corresponding host. The first connection is called here the *wireless segment*, while the second portion is the *wireline segment*. This division, by itself is not a new idea and was already employed by the Indirect TCP ([1]). What is new is in the design methodology of the wireless segment, being implemented with minimum communication overhead on the mobile machine by employing the asymmetrically-based protocol design. In the asymmetrically-based protocol design, a protocol function is implemented with different code complexity on the two ends of the protocol. In our case, the higher complexity code is executed on the local fixed machine and the lower complexity code on the portable machine. This methodology is possible due to the observation that the wireless segment of a transport connection is, in fact, a single-hop connection, very much like a Data-Link protocol. Consequently, as we will see in this paper, many functions can be either simplified or totally eliminated in the wireless segment of the transport-layer connection, leading to a lean protocol code on the mobile machine.

The entities involved in our protocol execution are: the *mobile host*, the *Mobile-Gateway*, which is a fixed machine with which the mobile host maintains a wireless link, the *corresponding host*, which is a stationary or mobile machine with which the mobile host is communicating. The transport layer protocol on the mobile host

Our approach in this study is to transparently "emulate" the TCP functionality, so that the application executed on a mobile machine interfaces with the TCP in the usual manner through the Socket Interface (see Figure 1). Similarly, the application on the corresponding host side is unaware of the TCP emulation.

Our work builds strongly on the indirect interaction model: the Indirect TCP (I-TCP) in [2, 3] and the Mobile Client Server Protocol in [4]. In those works, an intermediary, Mobility Support Router (MSR), which is analogous to our Mobile-Gateway, establishes two TCP connections, one with the mobile host and one with the corresponding host. It then performs the transport layer functionalities separately on each one of the two connections, forwarding the data between the mobile host and the corresponding host. Software residing on the Mobile-Gateway and called the *Redirector* integrates both of these segments, in such a way that the split into two connections is unnoticed by the mobile and the corresponding hosts. In this paper, we concentrate on the asymmetrical design of the Mobile-TCP and assume that the wireline segment and the Redirector are implemented according to [2, 3].

Related Work

In general, issues of mobility has been quite extensively addressed on the IP layer (for example, [5, 6, 7, 8, 9, 10]). Also, there has been studies indicating problems with simple porting of TCP onto mobile machines (e.g., [11, 12]). A number of approaches have been proposed to cure this ill-effect of mobility on TCP performance (e.g., [2, 13]). Finally, there have been some studies indicating that mobility needs to be addressed on layers above the transport layer as well (e.g., [14, 1, 15, 16]). This is, however, currently a much less explored area.

The topic of low-power hardware design for increased portability has also been investigated [17, 18, 19]. Low-power protocol design, has been addressed by a number of studies on the lower protocol layers, such as the data-link layer [20] and MAC layer [21]. However, to our knowledge, asymmetrical

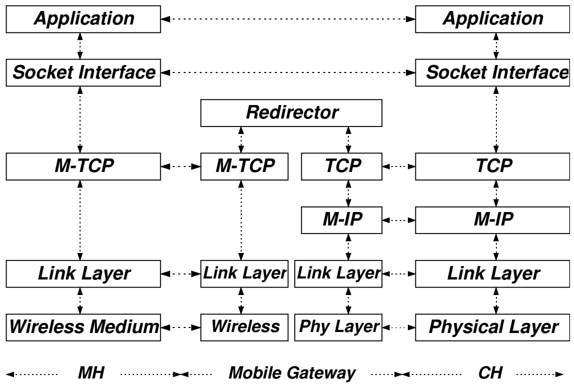


Figure 1: Mobile-TCP Protocol Stack

design of transport-layer protocols have never been implemented before.

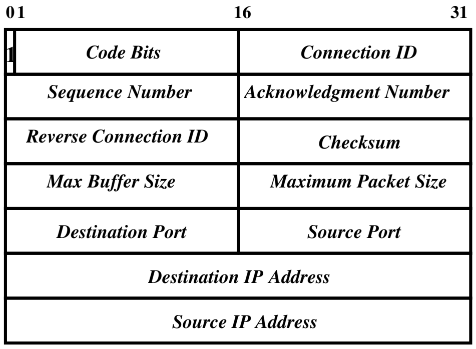
The Network Model

We base our work on the widely publicized Internet Engineering Task Force (IETF) Mobility Support protocol [5], which we briefly describe here. The description includes only the main features of the protocol and does not intend to be rigorous. But first, we introduce the nomenclature used here.

- MH – Mobile Host: A machine capable of mobility. Also called a *mobile*.
- CH – Corresponding Host: A machine communicating with a Mobile Host. Corresponding Host can be a stationary or a mobile machine.
- HN – Home Network: The administrative network of a mobile machine. The network portion of a machine's IP address equals the IP address of its Home Network.
- VN – Visiting Network: A network, other than the mobile's Home Network, to which the mobile is connected on a temporary basis.
- HA – Home Agent: A process residing on the Mobile Host's Home Network which is capable of capturing traffic addressed to the Mobile Host, while the Mobile Host is away from the Home Network.
- FA – Foreign Agent: A process resident on a visiting (wireline) network and which communicates with the Mobile Host and its Home Agent.
- COA – Care-Of-Address: An IP address on the Visiting Network through which packets are forwarded to the Mobile Host.
- POA – Point-Of-Attachement: An adaptor on the wireline network through which a Mobile Host can communicate with the wireline network.
- Handoff – A procedure of replacing a binding between a MH and a network entity with a new binding between the MH and another network entity.

Four entities participate in the Mobile IP: the Mobile Host (MH), the Corresponding Host (CH), the Home Agent (HA), and the Foreign Agent (FA). The MH maintains his HN IP address; i.e., there is no assignment of temporary IP addresses to MH-s. Upon migration from the MH's Home Network, the MH searches for a local FA on the VN. This process is referred to as *Foreign Agent Discovery*. Upon identifying a FA, the MH performs a *Registration* procedure, at the end of which the MH identity is added to the visitor's list maintained at the FA and the identity of the FA (actually the COA) is registered with the HA.

A MH sends out its IP packets using the traditional IP procedures. CH sends its packets to a MH by addressing the packets with the regular IP address of the MH. When the packets are delivered to



a.

Bit	Name	Meaning (when set)
1-2	Reserved	
3	*RET*	Retransmit request
4-6	*MFC*	MH Flow Control bits
7	*ONF*	Mobile Gateway ON/OFF Flow Control
8	*SYN*	Synchronize the connection parameters
9	*ACK*	Acknowledgement field is valid
10	*URG*	Urgent pointer is valid
11	*PSH*	This packet is to be pushed
12	*RST*	Reset the connection
13	*POL*	Poll for the status of the receiver
14	*FIN*	Terminate the connection
15	Reserved	

b.

Figure 2: Segment formats: a. Connection establishment packet   b. Significance of Code Bits

the HN, the HA is responsible for capturing the packets in place of the absent MH. The HA then tunnels the captured packets (through IP-in-IP, for example) to the FA (using the COA). The FA locally delivers the packets to the MH. As the MH roams from one network to another, the Foreign Agent Discovery and the Registration procedures are repeated. This concludes our presentation of the IETF Mobile IP protocol.

On the physical layer, mobile machines connect to the fixed network through POA-s, referred to here as the Mobile-Gateway-s. Mobile-Gateway-s maintain wireless links† to mobile clients. It is assumed that a mobile communicates with its Mobile-Gateway through a multiple-access scheme and that the bandwidth of the wireline network links is considerably larger than that of the wireless links. The wireless links are point-to-point links. Each Mobile-Gateway oversees some area, termed *coverage area;* mobiles currently located within the coverage area of a Mobile-Gateway maintain connectivity with this Mobile-Gateway. Thus the wireless network forms a cellular infrastructure. We exclude here the possibility of mobile peer-to-peer interconnection.

Mobiles can “roam” throughout the coverage of the wireless network. While changing its physical location, the mobile’s POA also changes to support adequate wireless (e.g., radio) connectivity (e.g., sufficient RF signal strength). Such changes of the POA are referred to as physical-level handoffs or handovers.

Since our work is targeted at issues above the network layer, we are not concerned here with, what are generally referred to as “lower-layers” issues. For instance, since the physical-level handoff operation is performed at the MAC layer, the manner in which the physical handoff is achieved is of no concern to us. However, the effect of handoffs, resulting in changing of the mobile’s Mobile-Gateway, is of primary importance to us.

DESCRIPTION OF THE MOBILE-TCP FEATURES

We now proceed with detailed description of the Mobile-TCP functions, emphasizing the asymmetrical features and the simplified complexity of the proposed design. The major factor in our simplified design is the fact that Mobile-TCP operates over a single point-to-point connection. Thus, in many aspects, Mobile-TCP resembles the design of a data-link layer protocol. As a consequence, one can assume that as long as the mobile station remains connected to the same POA to the wireline network, some connection characteristics will not change substantially. An example of such a connection characteristic is the connection Round Trip Time (RTT) delay. Of course, handoff from one POA to another does not guarantee consistent connection characteristics. Thus, to differentiate between the operation at handoff instances from the operation at other times, our Mobile-TCP protocol requires handoff indication from the “lower layers.” This is, usually not a serious constraint, as most of the lower-layers protocols has this information readily available.

† Our network model also covers mobile machines that frequently change their location by plugging into different networks. Although, in this work, we mainly address the wireless, rather than the ‘plug-in’ networks, nevertheless, our work is applicable to the ‘plug-in’ environment as well.

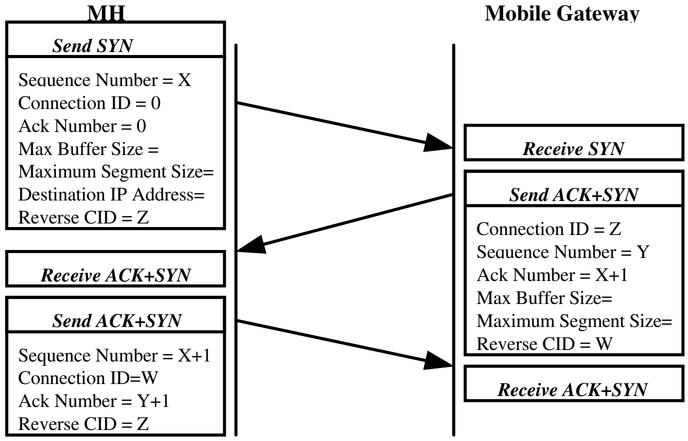


Figure 3: An active open

Connection Control

Connection control is concerned with setting up and tearing down connections. Under TCP, a host can initiate a connection with another host (called an active open) or can listen for and accept connections from other hosts (called a passive open). In our indirect model of interaction, the Mobile-Gateway acts as an intermediary, which establishes independent connections with the MH and with the FH. When the MH executes an active open, it communicates to the Mobile-Gateway the identity of the destination of the intended connection, as well as a set of parameters that will govern the operation of the connection between the MH and the Mobile-Gateway. Furthermore, some information related to the wireline segment of the connection is also relayed to the Mobile-Gateway at the connection setup time. The Mobile-Gateway then proceeds with setting up both segments of the connection. Similarly, when a MH executes a passive open, it informs the Mobile-Gateway that it is prepared to accept connections. This allows the Mobile-Gateway to accept or reject connection requests from FH-s without communicating a priori with the MH.

We propose to employ a technique similar to the Van Jacobson's header compression [22] to reduce the amount of wirelessly transmitted data, As all the packets from the Mobile Host pass through the Mobile-Gateway, in Mobile-TCP, there is no need to communicate the full TCP-layer source and destination addresses. At the connection establishment stage, a connection ID (CID) is assigned to each direction and it is used in any future exchange of data over the wireless segment. (CID_{MH} represents the CID from the MH to the Mobile-Gateway and CID_{MG} is the CID in the reverse direction.) The connection ID information includes the source and the destination IP addresses and the corresponding port numbers. These are cached at both ends of the connection. When sending a packet from the mobile to the network, the TCP address is translated into the corresponding CID at the MH. At the Mobile-Gateway, the CID is expanded back into the TCP-layer address, which is used on the wireline segment of the connection. Similar operation is performed in the reverse direction.

The full format of the Mobile-TCP control packet is shown in Figure 2 (a). Value of "1" in the first bit of the first octet indicates that this is a control packet, as opposed to a user data packet. The presence or absence of a particular field in the control packet is govern by the value of the *Code Bits* field, significance of which is shown in Figure 2 (b). In particular, the SYN control packet (a packet with the SYN bit set) utilizes the full packet format shown in Figure 2 (a). An ACK packet contains only the first 3 words.

Sequencing in Mobile-TCP is done on a packet-basis, rather than on a byte-basis. This allows much smaller sequence number space and simplifies the bookkeeping. The *Connection ID* field is the CID of the connection that this packet belongs to. *Sequence Number* and *Acknowledgement Number* are fields that contain the numbering of the current packet and the number of the next expected packet in the reverse direction. *Max Buffer Size* is the maximum space allocated for the connection at the sender and *Maximum Packet Size* is the size of maximal packet that the sender allows. Port number and IP addresses are as in regular TCP. The sender assigns the CID on the reverse direction and communicates this number in the *Reverse CID* field. The *Checksum* is taken over all the present fields

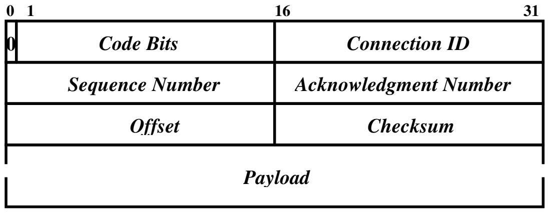


Figure 4: The format of a user data packet

of a control packet.

As in TCP, in Mobile-TCP, an active open uses a three-way handshake to reliably establish a connection. To open an active connection from a MH to a FH, a packet is sent to the Mobile-Gateway, which includes: the initial sequence number (determined by the MH), the size of the buffer size of this connection at the MH, the maximum size of a packet, the identity of the destination process (i.e., destination port and destination machine IP address), the IP address of the MH (source IP address), and the CID_{MH} assigned by the MH. The Connection ID and the Acknowledgement fields of the initial packet are set to zeros. The Mobile-Gateway responds by setting the following parameters in the reverse direction: the sequence number, the CID ($= CID_{MG}$), the maximum buffer size, and the maximum packet size. The SYN packet is then sent to the MH, which responds with an acknowledgment that concludes the active connection open process. The flow chart for active connection setup is shown in Figure 3.

Mobile-TCP's passive open uses a modified TCP three-way handshake. To open a passive connection, the MH sends an initial SYN packet, similar to the active open case, but with the destination port and destination IP address missing. In the return direction, the Mobile-Gateway acknowledges the SYN packet, but without setting most of parameters for this direction except for the Connection ID and Sequence Number fields. Upon receipt of a connection request frm a MH, the Mobile-Gateway sends another SYN packet that completes all the missing information on the reverse direction, including the destination port and IP address. After the connection has been established through the passive open procedure and the MH is still ready to accept new connections, it resends the SYN packet again. Because of space limitations, we omit here additional details of the passive open procedure.

Once a connection has been established, the actual exchange of information is through the regular user data packet format, depicted in Figure 4. The *Offset* field indicates the length of the payload in bytes – maximum 64K. Connection termination is handled similarly to TCP.

Note that the some connection control packets contain often information that was already communicated in preceding packets. However, as the connection establishment is performed relatively infrequently, once per connection lifetime, further simplification of the format of the connection control packet is not expected to have a significant effect on the overall performance of the protocol. Thus, we direct our efforts in other directions.

Retransmission and Timer Management

In TCP, each time a packet is sent, a timer is started to detect and retransmit unacknowledged packets. Retransmission schemes might result in significant load on the mobile machine, mainly because of their usage of timers. In the Mobile-TCP, we have addressed the issue of retransmission in two ways:

- use different retransmission schemes in the two directions
- where possible refrain from using timers by placing them on the Mobile-Gateway

We use the *Go-Back-N* scheme for retransmission of packets by the Mobile-Gateway, since the Go-Back-N scheme requires little processing at the receiving end — the MH in our case. On the other hand, *Selective Reject* is used in the opposite direction, as it minimizes the use of the wireless transmission.

A word of explanation is in order here. Usually, Go-back-N protocol is known to result in more retransmitted data than, say, Selective Repeat. This would suggest that Go-Back-N might not be appropriate for a wireless environment, in which one would like to preserve the use of the wireless

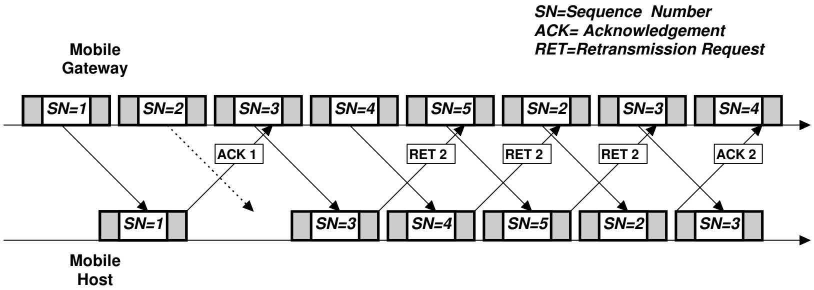


Figure 5: Retransmission by the Mobile-Gateway

resources. Nevertheless, we claim that for our application, Go-Back-N will not, substantially increase the overall amount of retransmitted data. This has to do with two factors: the short and relatively stable propagation delay between the transmitter and the receiver and the cause of packet loss being from transmission fading, as opposed to random errors. Short propagation time corresponds to less retransmitted data in case of a missing packet, especially when the round trip delay is relatively stable, and, thus, the time-out can be only a little bit longer than the round trip propagation time. Moreover, in fading transmission, it is very improbable to loose a single packet. More typically, a fade wipes out a long stream of packets, thus, requiring retransmission from the first missing packet anyway. In fact, one could argue that Selective Repeat performs much worse in such a communication environment, requiring packet-by-packet retransmission and, thus, incurring larger overhead.

TCP retransmission protocol relies on the Sliding Window [23] concept to prevent acceptance of old instances of packets that arrive with excessive delay. Since our TCP connection operates over a single hop, packets will not live in the wireless segment much beyond their transmission time plus propagation time. Consequently, there is no need for the Sliding Window mechanism in the Mobile-TCP. Furthermore, receipt of a packet with a higher-than-expected sequence number, indicates loss of the preceding packet and immediately triggers the retransmission mechanism.

As mentioned before, in general, retransmission of transport layer protocols rely on timers; each time a packet is transmitted, a timer is initiated so that if the packet or its acknowledgement is lost, it is retransmitted by the source. Thus, if a number of packets is outstanding, the same number of timers is created. Timers usually contribute substantial processing load and, as such, their use should be avoided on the MH. Therefore, we use a simple mechanism at the Mobile-Gateway to replace the timers in the MH, while regular timer mechanism is used in the Mobile-Gateway. This mechanism will be explained now.

First, lets concentrate on the Mobile-Gateway-to-MH direction, which, in principle, operates based on the Go-Back-N scheme. The Mobile-Gateway sends a packet to the MH, which, upon reception of the packet returns an acknowledgement (ACK) in the reverse direction in the *Acknowledgement* field (see Figure 4). Upon receipt of the ACK, the Mobile-Gateway can remove the packet from its buffer. If a packet is missing, the MH requests its retransmission by setting the *Retransmission* bit in the Code Bits field and the missing packet's sequence number in the Acknowledgement field. This is done in <u>every packet</u> in the reverse direction, until the missing packet is received at the MH. When the Mobile-Gateway receives the retransmission request, it retransmits all the packets again, starting from the indicated missing sequence number. In addition, the Mobile-Gateway starts a timer for the duration exceeding the Round Trip Delay. If, upon the timer expiry, the Mobile-Gateway continues to receive the retransmission request for the same sequence number,‡ the Mobile-Gateway repeats the same process again. There is no timer implemented on the MH. The main aspects of the scheme are shown in Figure 5.

Now consider the opposite transmission direction, from the MH to the Mobile-Gateway. Upon receipt of a packet from the MH at the Mobile-Gateway, the Mobile-Gateway generates an acknowledgement which, upon receipt at the MH is used to remove the corresponding packet from the MH's transmission buffer. If a packet is missing, the Mobile-Gateway requests retransmission of this packet only, by placing the missing packet's sequence number in the Acknowledgement field and by setting the *Retransmission* bit in the Code Bits field. This is done in a <u>single</u> packet only (see Figure 6). A timer is started at the Mobile-Gateway to protect against the loss of the retransmission request; such

‡This is the case in which the initially lost packet is lost again for the second time.

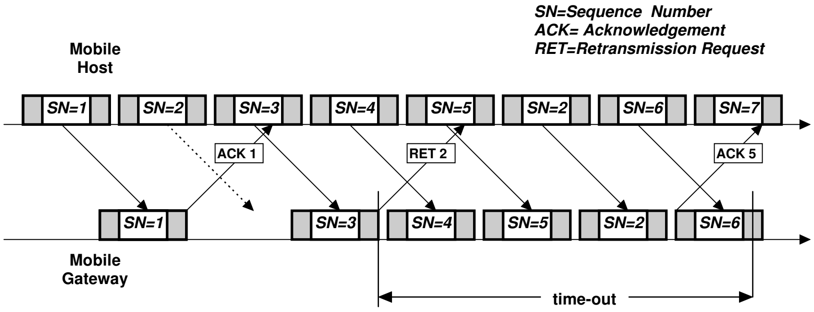


Figure 6: Retransmission by the MH

cases are handled according the the regular Selective Reject scheme. Also here there is no need for a timer on the MH.

The operation of the schemes presented here relies on continuous availability of traffic in both of the directions. To ensure this, a *persistence timer* is implemented on the Mobile-Gateway for each connection. When there is a break of longer than *persistence timeout* [24] in the traffic from the Mobile-Gateway to the MH, the Mobile-Gateway inserts Dummy Packets; i.e., packets with no payload field. If there is no traffic received at the Mobile-Gateway from the MH for the period exceeding the *persistence timeout*, the Mobile-Gateway sends a *Polling Packet* to the MH. A Polling Packet has the bit 13 (the *POL* bit) of the Code Bits field set. It prompts the MH to respond by sending a Dummy Packet back to the Mobile-Gateway. This way, the synchronization of both the ends of the connection is guaranteed. Again, this is accomplished without creation of a timer on the MH.

Flow Control

Controlling the flow to the Mobile-Gateway in Mobile-TCP is achieved through a simple ON/OFF scheme. Mobile flow control, on the other hand, uses a new scheme that relies on reporting the status of the receiver buffer occupancy. Thus, flow control function is also asymmetric. Because of the space limitations, we will explain here the main concepts of the schemes involved and leave out specific details.

Mobile-Gateway Flow Control The receive buffer at the Mobile-Gateway is partitioned by two thresholds: the *high.threshold* and the *low.threshold*. The idea of the ON/OFF scheme is to send a single-bit signal to the MH: ON – continue to send, OFF – stop sending. The ON signal is sent to the MH if the buffer occupancy exceeds the high.threshold watermark, while OFF is send when the buffer occupancy is below the low.threshold level. This creates a hysteresis, avoiding frequent state changes at the MH. Thus, processing of the flow control information at the MH is quite simple. The scheme is depicted in Figure 7.

Setting the threshold levels is not a trivial matter. The idea is that, in continuous flow of packets, the buffer occupancy oscillates approximately between the two thresholds. Placing the low.threshold too high will unnecessarily delay the packets in the receive buffer; placing it too low, may render the buffer empty too often and, thus, reduce the connection throughput. If the high.threshold is too high, packets might be lost due to too long delay in propagating the OFF signal. A high.threshold too close to the low.threshold will result in frequent changes of status at the MH.

We propose here to employ the following settings:

$$\text{high.threshold} = \mathit{Buffer} - RTT \cdot DR \qquad \text{and} \qquad \text{low.threshold} = RTT \cdot DR, \tag{1}$$

where RTT is the average *Round Trip Time*[§] between the MH and the Mobile-Gateway and DR is the average Data Rate of the MH-to-Mobile-Gateway connection. Finally, *Buffer* is the size of the receiver buffer at the Mobile-Gateway. The average Data Rate is calculated at the Mobile-Gateway by observing the arriving traffic while the Flow Control is in the ON state. Note, that we assume that $\mathit{Buffer} \geq 2 \cdot RTT$ and that all the settings are done at the Mobile-Gateway, with no processing load at the MH. Other threshold setting strategies are possible as well.

[§]RTT is periodically measured by the Mobile-Gateway and the value is conveyed to the MH. The frequency of measurements is considerable lower than what is used in TCP. This is due to the fact that on a single-hop connections the RTT does not change much. We omit here the details of the RTT calculation mechanism.

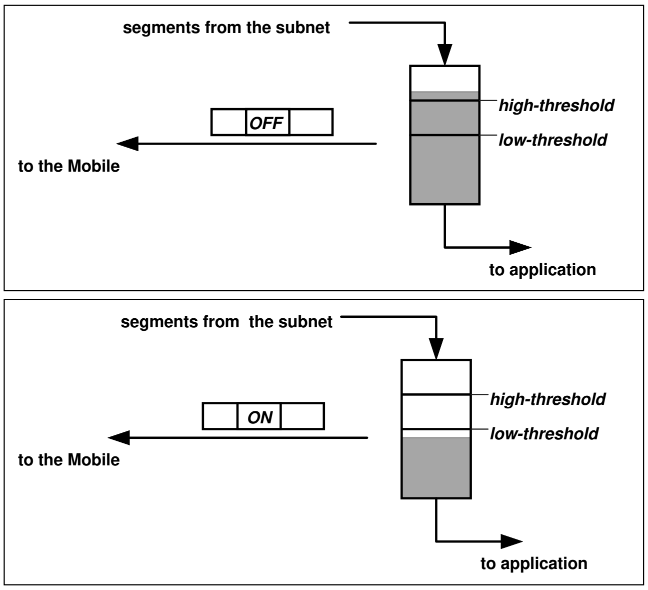


Figure 7: Mobile-Gateway Flow Control

The ON/OFF signal is transmitted in the bit number 7 (*ONF*) of the *Code Bits* portion of the packet's header (see Figure 2(b)). To ensure that the signal is properly received at the MH, in spite of possible packet losses, the MH repeats the ON/OFF signal in all its packets in the reverse direction. Also, the ON/OFF signal is continuously set by the Mobile-Gateway in all its packets to the MH. Thus, loss of few packets will, normally, have little effect on the operation of the flow control scheme.

MH Flow Control The MH flow control is accomplished by the Mobile-Gateway estimating the available buffer at the MH and stopping sending data when the estimated buffer occupancy reaches some predetermined threshold. Such estimation of the buffer occupancy at the MH is facilitated by the fact that the delay between the MH and the Mobile-Gateway is equal in both directions and is assumed to be relatively constant during the time the mobile is present in the current environment (e.g., in the current cell).

More specifically, controlling the flow in the MH-to-Mobile-Gateway direction is performed by partitioning the MH's receive buffer into a number of partitions. For the purpose of explanation, assume that the buffer is divided into 4 equal parts, as shown in Figure 8. The mobile communicates to the Mobile-Gateway the status of its receive buffer occupancy by sending two bits in each packet. These two bits correspond to the partition number at which the tail of the receive queue is currently positioned. These digits are referred to as the *Buffer Occupancy (BO)* bits.

The Mobile-Gateway, knowing the BO, the size of the MH's receive buffer,¶ the propagation delay, and the history of its transmission in the period of the last RTT, controls the buffer occupancy at the MH. Note, this operation is completely transparent to the MH; all that the MH needs to do is only to continuously communicate the BO status.

The operation of the scheme relies on continuous availability of the BO bits. This is assured by the Dummy and the Polling Packets, as explained in the "Retransmission and Timer Management" section. Also, as indicated in Figure 2 (b), our proposed implementation uses, actually, 3 BO bits and, thus divides the MH's receive buffer into 8 regions. In principle, the partitions of the MH's buffer do not need to be equal. However, the examination of the unequal partitions in outside the scope of this paper.

In summary, the MH flow control scheme requires little processing at the MH; the MH simply continuously inserts the BO bits in every packet. It also answers the POL requests.

Error Control

There exist a number of error-control schemes that place a substantially unequal load on both ends of the connection.‖ However, as the study of new error-control algorithms is orthogonal to

¶This information was communicated by the MH at the connection set up time.

‖Nevertheless, it is usually the case that a more powerful error-control scheme involves more processing load on both

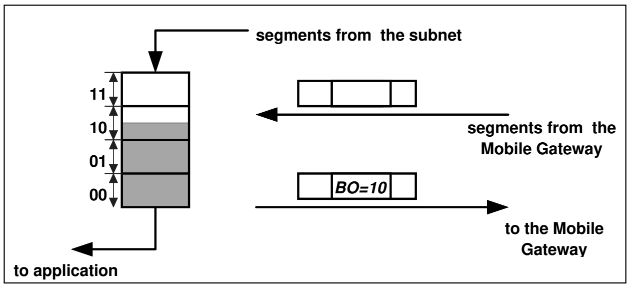


Figure 8: MH Flow Control

the proposed-here work, in our prototype, we have demonstrated the asymmetry in the error-control function through the following simple approach: error-detection feature at the MH and error-correction capability at the Mobile-Gateway. We have accomplished this by using the standard CRC-16 [23] polynomial calculation at the MH, while a similar, more complicated procedure is used on the Mobile-Gateway to identify the position of the errors and subsequently to correct those errors. This type of error-control scheme has the advantage of reducing the number of retransmission requests from the MH, thus further reducing the processing load on the MH. The use of the wireless bandwidth is reduced as well.

Other TCP Functionalities

Several functions are missing in Mobile-TCP, due to the "single-hop" nature of the wireless segment. For example, there is no need for Congestion-Control as Flow-Control function adequately handles a single-hop connection. Resequencing is also unnecessary as all the packets, except for retransmissions, will arrive in the order in which they were sent.

Handoff Support

Our Mobile-TCP protocol supports handoffs on the transport layer by migrating the parameters of the connections between the Mobile-Gateway-s as the MH roams and reconnects itself to a new POA. The function of migrating the Redirector has been already addressed in I-TCP [3]. We follow a somewhat similar philosophy here.

Recall from the "Introduction an Motivation" section that, based on the Mobile-IP protocol, IP datagrams to the MH are sent to the MH's HN and then redirected to the FA (i.e., the Mobile-Gateway , in our case). Thus, reregistration of the MH with its HA guarantees that the packets to the MH will, indeed, end up with the Mobile-Gateway. Moreover, there can be no confusions in the port numbers of the migrated connection with the already used port numbers on the new Mobile-Gateway, since the IP address of the MH is unique and different from the IP of all the other connections through the Mobile-Gateway. Thus, the handoff procedure does not involve set up changes in the MH and the CH machines.

To facilitate the recreation of the Mobile-TCP state on the handed off Mobile-Gateway, a Mobile-TCP daemon is set up in every Mobile-Gateway. As part of the lower layer handoff process, an indication is sent to the Mobile-TCP layer on the old Mobile-Gateway about the completion of the handoff process and the identity of the new Mobile-Gateway. The old Mobile-Gateway establishes a (regular) TCP connection with the new Mobile-Gateway and sends the state of all the parameters of all the MH's Mobile-TCP connections. These connections are then removed from the old Mobile-Gateway .

A new set of sockets with the TCP-layer addresses is created on the new Mobile-Gateway. As the *Connection ID* used on the old Mobile-Gateway might be in use on the new Mobile-Gateway, a new control packet is sent to the MH with the new value of the Mobile-Gateway's *Connection ID* on each connection. (See Figure 2 (a)). This concludes the Mobile-TCP handoff procedure.

---

sides of the connection.

SOME IMPLEMENTATION DETAILS

In this section, we provide, in general terms, some details of our Mobile-TCP implementation. We intend to publish the results of an extensive performance evaluation of our Mobile-TCP implementation in our next publication on this subject.

Our Mobile-TCP prototype is implemented on a PC platform in Windows 3.11. We used a commercially purchased TCP/IP protocol stack; in particular, the Winsock interface with raw socket. This allowed us to minimize processing overhead, yet leverage from the available software.

Our wireless hardware platform is a wireless local-area network, RANGELAN,** from PROXIM. We have added simple mobility management features (i.e., a subset of the Mobile-IP protocol) to support limited local-area mobility.

CONCLUDING REMARKS

In this paper, we have presented the rationale for asymmetrical design of transport-layer protocol, the Mobile-TCP, for use in mobile and wirelessly-interconnected systems by portable devices. The idea behind Mobile-TCP is in the recognition that the wireless segment of the indirect protocol is, in fact, a single-hop connection. This allows elimination of some transport-layer functions and simplification of other functions, leading to reduced communication load on the wireless link. Finally, reliance on asymmetrical design allows reduction in the protocol processing load on the mobile machine, achieving the second goal of the project.

We have surveyed the design of our prototype Mobile-TCP, discussing the implementation of its functions. Although detailed performance evaluation of the prototype is outside the scope of this paper, our measurement results indicate substantial improvement in the mobile code complexity, reduction in delay, and increase in throughput. Additional gain is in reduced memory requirements due to a much simpler code.

Although the Mobile-TCP protocol is based on the indirect interaction model, we made sure that the semantics of TCP are preserved. This is achieved through the Mobile-Gateway delaying the acknowledgements of packets to the CH until the packets are, indeed, acknowledged by the MH. This is possible, because of decoupling of the Acknowledgement and the Retransmission processes, and because of the Mobile-Gateway managing the Flow Control processes in both directions.

ACKNOWLEDGEMENT

The author would like to thank Matthew Androwski for his initial implementation of the Mobile-TCP prototype.

References

[1] B. R. Badrinath, A. Bakre, T. Imielinski, and R. Marantz, “Handling Mobile Clients: A case for indirect interaction,” in Proc. 4^{th} Workshop on Workstation Operating Systems, October 1993.

[2] A. Bakre and B. R. Badrinath, “I-TCP: Indirect TCP for mobile hosts,” in Proc. 15^{th} International Conference on Distributed Computing Systems, Vancouver, Canada, June 1995, pp. 136–143

[3] A. Bakre and B. R. Badrinath, “Handoff and system support for Indirect TCP/IP,” in Proc. Second Usenix Symposium on Mobile and Location-dependent Computing, Ann Arbor, Michigan, April 1995, pp. 11–24.

[4] Z. J. Haas, “Location Independent Access in Mobile Systems,” accepted for publication in the *Kluwer Wireless Personal Communications Journal,* special issue on *Mobile and Wireless Computing* 1996.

**RANGELAN is a trademark of PROXIM.

[5] C. Perkins, editor, "IP Mobility Support," Internet Engineering Task Force, Internet Draft, draft-ietf-mobileip-protocol-15.txt, 9 Feb. 1996.

[6] J. Ioannidis, D. Duchamp, and G. Q. Maguire, Jr., "IP-based Protocols for Mobile Internetworking," Technical Report, Department of Computer Science, Columbia University, 1992.

[7] H. Wada et al, "Mobile Computing Environment Based on Internet Packet Forwarding," *1993 Winter USENIX,* San Diego, CA, Jan 25-29, 1993.

[8] C. Perkins, "Providing continuous network access to mobile hosts using TCP/IP," *Computer Networks and ISDN Systems,* Vol. 26, 1993.

[9] A. Myles and D. Skellern, "Comparing four IP based mobile hosts protocols," *Computer Networks and ISDN Systems,* Vol. 26, 1993.

[10] F. Teraoka et al, "A Network Architecture Providing Host Migration Transparency," *ACM SIGCOMM'91,* Sept. 1991.

[11] P. Manzoni, D. Ghosal, and G. Serazzi, "Impact of Mobility on TCP/IP: An Integrated Performance Study," IEEE Journal on Selected Areas in Communications, issue on *Mobile Computing Networks,* 1995.

[12] R. Cáceres and L. Iftode, "The Effects of Mobility on Reliable Transport Protocols," IEEE Journal on Selected Areas in Communications, issue on *Mobile Computing Networks,* 1995.

[13] H. Balakrishnan, S. Seshan, and R. H. Katz, "Improving reliable transport and handoff performance in cellular wireless networks," *Wireless Networks Journal,* vol. 1, no. 4, 1995.

[14] Z. J. Haas, "Location Independent Access in Mobile Systems," accepted for publication in the *Kluwer Wireless Personal Communications Journal,* special issue on *Mobile and Wireless Computing* 1996.

[15] Z. J. Haas, "The Progressive Execution Technique for Mobile Systems," in Proc. Milcom'95, San Diego, CA, November 5-8, 1995.

[16] Z. J. Haas, "On the Design of a Mobile System," in Proc. Mobidata Workshop, Rutgers University, Piscataway, NJ, Nov. 1, 1994.

[17] E. P. Harris and K. W. Warren, "Low Power Technologies: A System Perspective," in Proc. *3^{rd} International Workshop on Mobile Multimedia Communications,* Princeton, NJ, Sept. 25-27, 1996.

[18] M. Stemm, P. Gauthier, D. Harada, R. H. Katz, "Reducing Power Consumption of Network Interfaces in Hand-Held Devices," in Proc. *3^{rd} International Workshop on Mobile Multimedia Communications,* Princeton, NJ, Sept. 25-27, 1996.

[19] A. Chandrakasan, T. Simon, J. Goodman, and W. Rabiner, "Signal Processing for an Ultra Low Power Wireless Video Camera," in Proc. *3^{rd} International Workshop on Mobile Multimedia Communications,* Princeton, NJ, Sept. 25-27, 1996.

[20] E. Ayanoglu, S. Paul, T. F. La Porta, K. K. Sabnani, and R. D. Gitlin, "AIRMAIL: A link-layer protocol for wireless networks," *ACM Wireless Networks,* 1 (1995) 47.

[21] W. Mangione-Smith and P. S. Ghang, "A Low Power Medium Access Control Protocol for Portable Multi-media Systems," in Proc. *3^{rd} International Workshop on Mobile Multimedia Communications,* Princeton, NJ, Sept. 25-27, 1996.

[22] V. Jacobson, "Compressing TCP/IP Headers for Low-Speed Serial Links," *RFC 1144.*

[23] D. Comer, "Internetworking with TCP/IP, Principles, Protocols, and Architecture," *Prentice Hall,* 1988.

[24] M. Santifaller, "TCP/IP and NSF, Internetworking in the UNIX Environment," *Addison-Wesley,* 1991.

[25] D. B. Johnson and C. E. Perkins, "Route Optimization in Mobile IP," Internet Engineering Task Force, Internet Draft, work in progress, draft-ietf-mobileip-optim-01.txt, Jan. 1995.