WTCP: an Efficient Transmission Control Protocol for Wired/Wireless Internetworking

SHENG-TZONG CHENG^{*}, YEH-HSI CHENG^{*}, AND YUH-RONG LEU^{**}

*Dept. of Computer Science and Information Engineering National Cheng Kung University Tainan, Taiwan, R.O.C. **Information Technology Division Institute of Information Industry Taipei, Taiwan, R.O.C.

(Received April 12, 1999; Accepted November 11, 1999)

ABSTRACT

The characteristics of a wireless network environment are different from those of a wired network environment. To increase the throughput of data transmission, we propose a wireless transport control protocol (WTCP) that is suitable for the internetworking of wired and wireless networks. The features of WTCP include fast acknowledgement, on-demand retransmission, a K-success status report and a time-out mechanism at the receiver module. WTCP is capable of tackling problems caused by the vulnerable wireless environment and TCP flow control mechanism at the sender side. We are developed analytical models to evaluate the performance of WTCP. The analytical results are verified by simulation results. The analytical and simulation results show that WTCP performs better than a well-known module-based protocol (i.e. SNOOP) and TCP do.

Key Words: TCP, wireless LAN, flow control, SNOOP, internet

I. Introduction

Through rapid advances in very large scale integrated (VLSI) circuit technology, powerful computing devices, such as desktop, notebook, and palmtop computers, have become widely accepted and used in daily life. Furthermore, wireless communication greatly enhances the utility of carrying these devices and allows mobile networked communication. The natural call for mobile computing has led to a new computing paradigm in which mobile users communicate with other people, access information, and expedite timely and urgent notifications throughout the world.

Internetworking with wired and wireless links in a heterogeneous environment offers a way to support mobile computing. Wireless local area networks (WLANs) are integrated into such an environment so that users are free to move from one place to another while still maintaining their connections to the networks. The characteristics of WLANs are notably different from those of wired networks. Higher error-bit rates and temporal disconnection are encountered with the wireless links. On the other hand, congestion often occurs in a wired network. These differences significantly impact communication protocols.

In the transport layer of the network protocol stack, reliable and efficient protocols are needed to handle connection establishment and data transport in wired/wireless internetworking. The Transmission Control Protocol (TCP) (Comer and Stevens, 1994) has been tuned to handle congestion in wired networks. When the sender of a connection sends out a packet, TCP sets a timer for receiving an acknowledgement of the packet. If the timer expires, TCP assumes that the loss of the packet is due to congestion in the networks. Therefore, it reduces the size of the sliding window and retransmits the lost packet. However, in wired/wireless network environments, the loss of a packet could be due to an unreliable wireless link. Misinterpreting the reason for the packet loss and employing a slow start mechanism, such as by reducing the transmission window at the sender side of a TCP connection or using exponential back-off, will cause performance degradation.

Several approaches (Amir *et al.*, 1995; Bakre and Badrinath, 1994, 1997; Balakrishnan *et al.*, 1995a, 1995b, 1997) have been proposed to tackle the above problem. Among them, two main categories can be identified (Caceres and Iftode, 1994), namely, end-to-end approaches and split-connection approaches. These two types of approaches exploit the existence of base stations (or mobile support stations) to distinguish a loss due to congestion from a loss due to wireless links. In this paper, we take a step further and investigate the delay spread and multi-path effect on wireless links and include the mobile host into the protocol structure. We propose a protocol called the wireless transport control protocol (WTCP). According to WTCP, there is no modification at the sender side of TCP if the sender is a fixed host. If the receiver of a connection is a mobile host, WTCP uses the "selective repeat" scheme, in which the mobile host replies "Non-Acknowledgement (NACK)" to inform the base station of the lost of packets. By means of NACK, the base station is able to perform local on-demand retransmission if the packets are cached. Note that no time-out mechanism is required at the base station. In addition, in the case of several successful receptions, the mobile host reports the status back to the base station such that the cache buffer can be freed.

The rest of the paper is organized as follows. In Section II, we survey the related work on improving TCP performance over wireless links. In Section III, we describe our approach to lossy and burst-error prone wireless links. In Section IV, we present the analytical model used to compare our approach with others. In Section V, we present the simulation results obtained using our approach and others. Conclusions are drawn in Section VI.

II. Background and Related Work

1. TCP

TCP is a connection-oriented protocol. A connection must be established before any data is transferred. The receiver side passively waits for a connection from the sender side. When a sender wants to establish a connection with a receiver, it will send out a synchronization (SYN) segment, indicating the connection request. The destination replies with a SYN segment, once the former SYN segment arrives. Then the sender acknowledges the second SYN segment by means of an acknowledgement (ACK) segment. Afterwards, the connection is established. This method is called three-way handshaking (Tanenbaum, 1995), as illustrated in Fig. 1.

Since the buffer size at the receiver side is limited, if the sender sends too many packets at a time, overflow occurs. Consequently, some packets are dropped, and the sender needs to re-transmit those packets. It is easy to see that this situation will waste network bandwidth. To reduce this chance of overflow, the receiver advertises a window size by means of which the sender can check if the receiver has a large enough buffer (Comer, 1994).

In the course of transporting data, the sender will send as many packets as possible, up to the sliding window size advertised by the receiver. However, if the intermediate routers have a stringent buffer limitation, then packets may still be dropped in the middle of transmission. Therefore, TCP employs a slow start mechanism to deal with this problem. In the mechanism, another window, called a congestion window, is used. The congestion window is initialized to one segment. Each time the



Fig. 1. Three-way handshaking.

sender receives an acknowledgment, it adds one segment to the congestion window. The minimum value of the congestion window and that of the sliding window determines how many packets the sender is allowed to send at any time.

When the sender sends a packet, it sets a timer for receiving an acknowledgement from the receiver. If no corresponding acknowledgment arrives before the timer expires, the sender will retransmit a pending packet. The timeout at the sender side signals a possible congestion situation in the network. Therefore, the sender reduces the congestion window and resets the retransmission timer in an exponential form. The slow start mechanism used by TCP alleviates congestion in the network.

2. SNOOP

The main features of SNOOP are its ability to cache packets at the base-station side and to perform the local retransmission (from the base station to the mobile host) if necessary. Consider a scenario in which a fixed host is sending packets to a mobile host and the packets are passing through a base station. At the base-station side, SNOOP keeps track of the last sequence number seen on the connection. When a new in-sequence packet arrives, SNOOP caches the packet for further use. On the other hand, if a new out-of-sequence packet arrives, it signals a possible congestion situation in the wired part of the connection. Note that it is not necessary for SNOOP to respond to the congestion situation. Congestion in the wired network will eventually be detected by the sender due to the time-out at the sender side of the TCP connection. Whenever the base station receives a packet, SNOOP forwards the packet to the mobile receiver on the wireless side.

Local retransmission is performed when SNOOP

sees a duplicate ACK in a sequence number. A duplicate "ACK p" indicates that the packet p is wanted by the mobile host, and that packets with higher sequence numbers (than p) may have arrived at the mobile host. Therefore, the base station sends packet p, which has been cached previously, to the mobile host with higher priority.

The advantages of SNOOP are its ability to preserve the protocol semantics and the fact that it is compatible with existing applications using TCP. However, SNOOP still suffers from burst losses in wireless links.

3. Other Approaches

The split-connection approach has been developed to handle data transmission in wired/wireless internetworking environments. Bakre and Badrinath (1994) proposed Indirect TCP (short for ITCP) for mobile hosts. ITCP aims to distinguish packet loss over a wireless link from that over a wired link. According to ITCP, each TCP connection is divided into two separate connections: one is between the sender and the base station, and the other one is between the base station and the mobile receiver. Consequently, every packet has to undergo TCP protocol processing twice at the base station. Another disadvantage of ITCP is that the end-to-end semantics are violated, such that the sender is informed of successful transmission (by acknowledgements) before the packets really reach the receiver. Consequently, the sender might close the connection or begin to work on sequent jobs while the packets are still in transit over the wireless link.

Other approaches, such as Explicit Feedback and Fast Retransmission (Caceres and Iftode, 1994), have also been proposed in the literature. Interested readers are recommended to obtain further details.

III. The Proposed WTCP

1. Overview of Our Approach

In a wired/wireless internetworking environment, a TCP connection consists of one wired part and one wireless part, as shown in Fig. 2. In our approach, the sender of a connection transmits data without making any modifications to the original TCP protocol structure. A module is added to the base station to process incoming packets from senders via wired link. The base-station module also responds to communication of the acknowledgement status received from the receiver in the wireless network. In addition, at the mobile receiver side, an acknowledgement mechanism is adopted to handle status reporting back to the base station. Note that there is no time-out mechanism at the base station side. The base station caches a packet and performs local retransmission only when the receiver asks it to do so. We call this *on-demand retransmission*.



Fig. 2. Illustration of wired/wireless internetworking.

The receiver asks the base station to perform on-demand retransmission when the wireless link resumes normal communication after a temporary disconnection or a time out occurs at the receiver side. On-demand retransmission is able to reduce the accumulation of traffic during a temporary disconnection in the wireless network. Consider the case in which the base station times out several times (if the base station is equipped with a time-out mechanism) while the wireless link is experiencing error transmission. In such a case, the time and resources used to perform transmission are spent in vain. Therefore, in our approach, the time-out mechanism is moved to the receiver side, where the receiver makes use of the delay spread and multi-path effect to correctly estimate the time-out interval.

The proposed mechanism is suitable for error-prone and low-bandwidth wireless environments. It reduces the amount of up-link traffic by summarizing the receiving information of several consecutive packets into one ACK. In addition, when the wireless environment is experiencing temporary disconnection, the receiver is able to request retransmission of lost packets after it receives a successful packet.

2. Base-station Module

One module is added to the base station. The basestation module maintains a "*last_ACK*" counter to preserve the TCP semantics. The *last_ACK* counter indicates the sequence number of a packet that the receiver expects to receive. Each time a new packet arrives, the module checks whether the sequence number of the packet equals *last_ACK*. If it does, then the value of *last_ACK* is incremented by one. Otherwise, the value of *last_ACK* is left unchanged. Possible events that are handled by the mod-



Fig. 3. Basic scenario of WTCP.

ule are described in Subsections III.2.A and III.2.B.

A. Arrival of a Packet from the Sender

- (1) Arrival of a new in-sequence packet When a new in-sequence packet arrives at the base station, the module forwards the packet to the receiver. In addition, it checks if there is any available buffer in the system for caching incoming packet. If there is, then the module puts the packet in the cache and sends an acknowledgement (indicated by last_ ACK) back to the sender immediately. This fast acknowledgement signals a congestion-free status in the wired-line part of the connection and can increase the sending rate at the sender side. The packet is cached for possible retransmission in the future if the packet is lost over the wireless link. On the other hand, if the cache space for this connection is used up, the module simply forwards the packet without sending an acknowledgement.
- (2) Arrival of a new out-of-sequence packet A new out-of-sequence packet signals a possible congestion situation in the wired part of the network. Packet(s) with sequence number(s) less than (or greater than) that of the current packet might be lost. In such circumstances, the module caches the new packet for possible local retransmission if storage space is available. The module also performs fast acknowledgement, if the packet is cached, by sending an acknowledgement, indicated by the *last_ACK* counter. In addition, it forwards the current packet to the receiver.
- (3) Arrival of an old packet There are two possible cases in which the base station receives an old packet. One case is where the sequence number of the packet is less than *last_ACK* and the acknowledgement of the packet is lost on its way back to the sender. The other case is where the sequence number of the packet is greater than *last_ACK* and the packet is in cache. In the former case, the mod-

ule simply drops the packet since forwarding of the packet has been taken care of. In the latter case, the module needs to forward the packet to the receiver again. By distinguishing the former case from the latter case, our approach is able to reduce retransmission traffic in the wireless part.

B. Arrival of a Status Report from the Receiver

The status report received from the receiver takes the form of "NACK p to q and/or FREE r," which indicates that the packets from p to q have been lost, and that the packets up to r have been successfully received at the destination. When packets with sequence numbers greater than q reach the receiver side, the wireless link resumes the connection following temporary disconnection, and the packets from p to q were lost before. An illustrative example is shown in Fig. 3. We consider the example as the basic scenario in a wireless network because a wireless link is error-prone and may be temporarily disconnected. In addition, the "Free r" message tells the base station to remove the packets up to r from the cache since the packets have been successfully received. If removal of the packets is not acknowledged, the base station sends an acknowledgement back to the sender.

For each packet n in the range NACK p to q, the base-station module processes the packet based on the following rules:

- Packet n was cached previously In the case of a cached packet, the module simply retransmits it again.
- (2) Packet n was not cached The module discards the message. Eventually, the timer at the sender side will expire. Packet n will be re-transmitted by the sender. When the packet arrives at the base station again, the module will handle it according to case 3 described in Subsection III.2.A.

The handling of packets and status reports, as described above, is summarized in Fig. 4.

3. Receiver Module

A module is added at the receiver side to trace the sequence numbers of the packets received so far. A " R_ACK " counter is maintained, which is similar to *last_ACK* used in TCP. The R_ACK value indicates the next sequence number of the packet that the mobile host expects to receive via the wireless link. In addition, one variable, *head_not_ACK*, is used by the receiver module. Each time the mobile host receives a packet, the module sets a timer for receiving the next packet (indicated by the new R_ACK). Besides setting the timer, the module checks whether it needs to send a status report back to the base station. Recall that our approach reduces the uplink



Fig. 4. Handling a packet at the base station.

traffic by summing the information about receiving several consecutive packets (controlled by the system-parameter K) into one ACK message. The ACK message takes the form of "FREE r." Therefore, *head_not_ACK* points to the first of several consecutively received and not-yetacknowledged packets. If the difference between R_ACK and *head_not_ACK* exceeds the value of K, it is time for the receiver module to reply with a status report by sending a "FREE R_ACK " message. In Fig. 5, we summarize the actions that may be taken by the receiver module. As shown in Fig. 5, on receiving a packet p:

(1) If p equals R_ACK and no packet greater than p was received previously, then this is a normal situation. In this case, the value of R_ACK is increased by one. Also, if this is the K-th reception since the last FREE message, a "FREE head_not_ACK+K" message is sent back to the base station.



Fig. 5. Handling a packet at the receiver module.

- (2) If p equals R_ACK and if some packet(s) greater than p was/were received previously, then there might be a gap between p and the higher packets. Therefore, we need to check whether the concatenation of p and higher packets forms a contiguous block. If it does, then the value of R_ACK is updated to a higher value (i.e. the tail of the contiguous block), and the steps after the updating step are similar to those described for Case 1. If it does not, then it is still necessary to check if the packet p is the K-th reception.
- (3) If p is different from R_ACK and is the highest number received so far, then this indicates possible recovery from temporary disconnection as in the case of receiving packet 6 shown in Fig. 4. Therefore, it is essential to report the status back to the base station, as indicated by the message "NACK x to y and/or FREE R_ACK-1."

A. Time-Out Mechanism at the Receiver Module

In our approach, the base-station module caches the packets received from the sender and performs on-demand retransmission. On-demand retransmission occurs when a NACK message is received. There are two cases in which the receiver will send a NACK message. One case, as described in Case 3 of Section III.1, is when the receiver module detects a recovery from temporary disconnection. The other case is when the timer expires on the receiver side. Each time a packet arrives at the receiver side, a timer is set for receiving the next packet in sequence. The time-out mechanism is essential for making data transmission continuous.

Consider the following extreme situation in which the base station receives *x* packets before the wired link is disconnected. No more packets will get through the wired link for a long time due to the disconnection. The base station forwards these packets to the receiver. Let us assume that these *x* packets get lost in the wireless link. Consequently, the base station has no way to know whether the packets reach the receiver or not, even after the wireless link resumes connection. To solve this problem, the time-out mechanism at the receiver side is required to trigger the receiver to send a NACK message. The NACK message, once it arrives at the base station, will activate on-demand retransmission and make the data transmission continuous even when the wired link is disconnected.

The setting for the timer value is based on past history and the single-trip delay. The average time interval for receiving a packet can be derived from past history. Furthermore, the single-trip delay can be estimated by adding the push-down time at one side, the pull-up time at the other side, and the propagation delay over the wireless link. As shown in Fig. 6, the push-down (pull-up) time indicates the processing time of a packet from the transport layer (physical layer) to the physical layer (transport layer). The push-down and pull-time values can be obtained by writing a program that sends packets to the sender itself.

It is notable that the time-out mechanism in a wireless environment is dramatically different from that in a wired environment. This is because that there is no router or gateway in between the base station and the mobile receiver. This implies that the propagation delay over a wireless link is more predictable. In our approach, we adopt the multi-path effect and delay spread concept to compute the propagation delay. As shown in Fig. 7, the signal from the radio tower follows multiple paths to reach the mobile host. Consequently, the received signal at the mobile side spreads over a time interval, T_d . This value T_d is added to the propagation delay to get a more accurate timer value.

Let α_p be the average time interval for receiving a packet up to packet p at the receiver side. Let D be the



Fig. 6. Total transmission delay.



Fig. 7. The multi-path effect and delay spread.

total propagation delay, which includes the push-down, pull-up, and radio transmission delay. Let β_{p+1} be the timer value for receiring the packet p + 1. We calculate β_{p+1} according to the following equations:

$$\alpha_0 = \alpha_1 = \text{initial round-trip time over the wired link;}$$
(1)

$$\alpha_p = ((\sum_{i=1 \text{ to } p-1} \alpha_i) + \text{ time interval between}$$

receiving packets $p-1$ and $p)/p$; (2)

$$\beta_{p+1} = \alpha_p + D + T_d, \quad \forall p > 0.$$
(3)

B. Setting of the System-Parameter K

Recall that the receiver module sends a FREE message to the base station upon K successful receptions. The setting for the system parameter depends on the buffer capacity of the base station. Let f and τ be the total buffer capacity and the uplink transmission delay. Let t be the time when the receiver sent the last FREE message. If a total of e packets are received from time t to the current time t2, then the following inequality must hold:

$$\frac{e}{t^2 - t^1} \times \tau \le f - K. \tag{4}$$

Based on the above inequality, the value of K can be adjusted properly.

4. End-to-End Semantics Preservation

Split-connection approaches are criticized for violating TCP semantics. Each original TCP connection is divided into two separate connections: one is between the sender and the base station, and the other one is between the base station and the mobile receiver. It is possible for the former connection to be released while the latter connection is still active. Consequently, the sender proceeds to work on the job next to the transmission though the receiver has not yet received all the packets. The semantics are then violated.

To preserve the TCP one-connection semantics, the receiver of WTCP issues a FIN-confirm signal only after a FIN-request signal is received and all packets have reached the final destination. In Fig. 8, the left part illustrates the original TCP signaling procedure for releasing a connection. As shown on the right side, when a base station is added in between the sender and receiver, the base station forwards the FIN-request to the receiver upon receiving the FIN-request from the sender. The base station will not send back the FIN-confirm signal until it receives such a signal from the receiver. In this way, the semantics can be preserved.

IV. Analytic Results

We have developed an analytic model (Cheng and Cheng, 1999) for analyzing the performance of the proposed protocol as well as SNOOP and TCP. Interested readers are referred to Cheng and Cheng (1999) for details of the analysis. Here, we will summarize the analytical results:

$$P_{tcp} = \frac{c \times s}{(e_1 + e_2) \times E + E},\tag{5}$$

$$P_{wtcp} = \frac{c \times s}{e_{wtcp,1} \times E + t_{wtcp,2} + f},$$
(6)

$$P_{snoop} = \frac{c \times s}{(e_1 + e_2) \times E + f},\tag{7}$$

where

- e_1 is the number of cycles in the first phase in which the window size is less than the average value;
- e_2 is the number of cycles in the second phase in which the window size is equal to or greater than the average value;
- c is the number of packets sent in one epoch;
- *s* is the size of a packet ;
- *E* is the time period of an epoch;
- *f* is the time period for re-transmitting a packet in the wireless network;
- $E_{wtcp,1}$ is the number of cycles in the first phase in which the window size is less than the average value;
- $T_{wtcp,2}$ is the time period of the second phase in which the window size is equal to or greater than the average value;
- P_{tcp} is the throughput of TCP;
- *P*_{snoop} is the throughput of SNOOP;
- P_{wtcp} is the throughput of WTCP.

Figure 9 shows the performance evaluation based on the analytic model presented in this section. The values of E and f were set to be 20 and 20, respectively. The sliding window size was 64 Kbytes, and each packet was as-



Fig. 8. Signaling a connection release.



Fig. 9. Analytic results.

sumed to be 1024 bytes. It is seen that WTCP performed better than the other two for various error rates. The performance of the three became similar when the error rate was extremely high since the cycle time became smaller, and the benefit of fast acknowledgement became less significant.

V. Performance Evaluation

To evaluate the performance of the WTCP protocol, we wrote programs to use to conduct simulations. We compared WTCP with TCP and SNOOP. We used the throughput as the performance measurement: the total number of transmitted packets over the total elapsed time. We developed two platforms: one for TCP, and the other for SNOOP and WTCP, as shown in Figs. 10 and 11, respectively. In these two platforms, there are three main entities: a sender, base station, and receiver. In between the sender and the base station, there is an environment,



Fig. 10. Simulation platform for TCP.



Fig. 11. Simulation platform for SNOOP and WTCP.

which is controlled by the transmission delay (i.e., the total delay from the sender to the base station) and an error profile. The error profile is characterized by the error bit rate, which is also a simulation parameter. The error profile specifies how many and which successful packets and fault packets there are. The wireless environment has a similar structure, but that the error bit rate is assumed to be higher that that in a wired environment.

The major difference between these two platforms is in the caching at the base station. Since SNOOP and WTCP both perform packet caching and local retransmission, a storage module is added as shown in Fig. 11. In addition, WTCP performs fast acknowledgement, produces a K-success status report, and conduct on-demand retransmission, which are implemented in the base-station and mobile-receiver modules. The simulation programs were written in Maisie, a parallel-programmable simulation language. The simulation was based on the following parameters:

- (1) Error bit rate on the wireless link: The rate varied from 1 bit/64 Kbits to 1 bit/16 Mbits. The value of 1/r indicates that, on average, there is one bit error out of r bits. Note that the transmission of a packet is considered to fail if it contains at least one error bit. The typical values used in the simulations were 1/64 K, 1/128 K, 1/256 K, ..., 1/8 M, and 1/16 M. Furthermore, we set the error bit rate for the wired link at 1/16 M, which is more reliable than that of a wireless link in most cases.
- (2) Packet size: 1024 bytes.
- (3) Total amount of transport data: The value used in the simulations was 5 M bytes, namely 5,000 packets.
- (4) Size of sliding window: 64 Kbytes
- (5) Propagation delay in the wired network: It randomly varied from 6 to 9 simulation time units.
- (6) Propagation delay in the wireless network: It was fixed to 5 time units.
- (7) Round trip time (RTT) value at the sender TCP: It was initialized to 20.
- (8) Number of iterations for each case: One simulation case was defined as a simulation in which the error bit rate of the wireless link is set to a value (from 1/64 K to 1/16 M). For each simulation case, 50 runs were simulated.

The simulation results are summarized in Fig. 12. Note that we only show the averaged number of received packets versus time for the cases in which the error bit rate was 1/64 K, 1/512 K, and 1/4 M in Fig. 12(a), (b), and (c), respectively. In Fig. 11(d), we summarize the throughput versus the error bit rate of the wireless link for all cases. The throughput is defined as the average number of packets received per time unit.

From Fig. 12, we conclude that:

- (1) WTCP performed better than TCP and SNOOP in all the simulated cases. The main reason is that WTCP uses fast acknowledgement, which hides the wireless environment from the sender. Consequently, the sender is not affected by error transmission in the wireless link.
- (2) As the reliability of the wireless link increased, the throughput of the three protocols increased.
- (3) In the cases with high error bit rates, the throughput difference between WTCP and the other two protocols was higher compared to it was in the cases with low error bit rates. This implies that the WTCP protocol is suitable for wireless/wired internetworking.

VI. Conclusions

We have proposed a wireless transport control proto-

S.T. Cheng et al.



Fig. 12. Simulation results.

col (WTCP) that is suitable for internetworking of wired and wireless networks. The features of WTCP include fast acknowledgement, on-demand retransmission, a Ksuccess status report and a time-out mechanism in the receiver module. WTCP is capable of dealing with a vulnerable wireless environment and TCP flow control mechanism at the sender side. From the simulation results, we find that WTCP performs better than SNOOP and TCP. Our future research direction will be to develop a mathematical model for analyzing the performance of WTCP.

Acknowledgment

This work was sponsored by the Institute of Information Industry, Taipei, Taiwan, R.O.C., under contract 86-R-58.

References

Amir, E., H. Balakrishnan, S. Seshan, and R. H. Katz (1995) Efficient TCP over Networks with Wireless Links. Computer Science Division, University of California at Berkeley, Berkeley, CA, U.S.A.

- Bakre, A. and B. R. Badrinath (1994) *I-TCP: Indirect TCP for Mobile Host.* Department of Computer Science, Rutgers University, Piscataway, NJ, U.S.A.
- Bakre, A. V. and B. R. Badrinath (1997) Implementation and performance evaluation of indirect TCP *IEEE Transactions on Computers*, 46(3), 260-278.
- Balakrishnan, H., S. Seshan, and R. H. Katz (1995a) Improving reliable transport and handoff performance in cellular wireless networks. ACM Mobile Computing and Networking Conference (Mobicom '95), Berkeley, CA, U.S.A.
- Balakrishnan, H., S. Seshan, E. Amir, and R. H. Katz (1995b) Imporving TCP/IP performance over wireless networks. 1st ACM Int'1 Conf. on Mobile Computing and Networking (Mobicom'95), Berkeley, CA, U.S.A.
- Balakrishana, H., V. H. Padmanabhan, S. Seshan, and R. H. Katz (1997) A comparison of mechanisms for improving TCP performance over wireless links. *IEEE Transactions on Networking*, 5(6), 756-769.
- Caceres, R. and L. Iftode (1994) Improving the performance of reliable transport protocols in mobile computing environment. *IEEE Journal* of Selected Areas in communications, 13(5), 850-857.
- Cheng, S. T. and Y. Cheng (1999) Design and analysis of an efficient wireless transport protocol. International Symposium on Communications, Kaohsuing, Taiwan, R.O.C.

Design of a Wireless TCP

Comer, D. E. and D. L. Stevens (1994) Internetworking with TCP/IP volume 2, Design, Implementation, and Internals, 2nd Ed., pp. 283-308. Prentice-Hall, Englewood cliffs, NJ, U.S.A. Tanenbaum, A. S. (1995) Computer Networks, pp. 370-436. Prentice-Hall, Englewood Cliffs, NJ, U.S.A.

在有線與無線整合環境下TCP網路通訊協定之改良

鄭憲宗^{*} 鄭曄禧^{*} 呂毓榮^{**}

* 國立成功大學資訊工程學系 ** 資訊工業策進會資訊技術處

摘要

無線傳輸媒介的特性與有線網路環境有所差異。在一個有線與無線整合環境下,傳統的TCP通訊協定無法有效地解決無線介質的特性與行動主機所衍生的行動性等問題,因此本研究論文針對無線通訊與行動計算的特性,提出一個新的wireless TCP (WTCP)通訊協定。我們同時針對WTCP的效能進行評估,經過定性之數理分析與定量之模擬證實,WTCP 比TCP 與SNOOP 的效能都高。