Adaptation of Transport Protocols for an IP-Micromobility Scheme

Ana Victoria Delgado Martín[1], Andrej Mihailovic[2], Nikos Georganopoulos[3], A. H. Aghvami[4]

[1] Accenture Technology Park, Networks Technology Specialty, 449, route des Cretes, BP 99. 06902, Sophia Antipolis, France
www.accenture.com, Tel: +33 4 92 94 17 56, Fax: +33 4 92 94 67 99. E-mail: ana.m.delgado@accenture.com

[2,3,4] Centre for Telecommunications Research, King's College London, Strand, London, WC2R 2LS, UK
www.ctr.kcl.ac.uk, Tel: +44 (20) 7848 2889, Fax: +44 (20) 7848 2664
e-mail: andrej.mihailovic@kcl.ac.uk, nikolaos.georganopoulos@kcl.ac.uk, hamid.aghvami@kcl.ac.uk

Abstract - **TCP is a reliable connection-oriented Transport Protocol that performs well in traditional networks. However, in networks with wireless and other lossy links the protocol suffers from losses and delays due to frequent handoffs like in wireless networks running Mobile IP. Many Micromobility Protocols have been proposed to reduce handoff latency and the load on the network when a Mobile Host moves among small wireless cells. In this paper, the performance of several TCP and UDP schemes over one of these protocols: MMP (Multicast for Mobility Protocol) is compared and improvements to TCP in a multiple handoff network are proposed. The simulation results are presented using the Network Simulator-*ns2* and different metrics for comparison. Our results show that the number of UDP packets lost is reduced when using MMP instead of Mobile IP in a micromobility environment and that TCP provides better throughput performance for a traditional scheme like Tahoe TCP when it is adapted to the conditions of the network.**

I. INTRODUCTION

Current trends in wireless communications have shown that future internetworks will include large number of portable devices moving among small wireless cells. That is why an IP-Micromobility Protocol and a reliable Transport Protocol such as TCP play an important role in the operation of Internet. Although an IP-Micromobility System such as MMP (Multicast for Mobility Protocol) [1], Cellular IP [2], or HAWAII [3] helps to decrease packet losses or delay during handoffs, other modifications in higher layers of the hierarchy are also needed to improve performance in the overall transmission. These modifications affect primarily the Transport Protocols and the way establishment, connection and transmission are brought about.

In theory, Transport Protocols should be independent of the technology of the underlying network layer. In particular, TCP should not care whether IP is running over fibre or over radio. In practice, it does matter because most TCP implementations have been carefully optimised based on assumptions that are true for wired networks but which fail for wireless ones. This results in a poor performance and it is necessary to introduce some modifications: while in a wired network when a packet is lost the sender should slow down, in a wireless network, the sender should try harder. Although UDP does not suffer from the same problems as TCP, wireless communication also introduces difficulties for it. The main problem is that applications use UDP expecting it to be highly reliable. Although no guarantees are given, it is still expected to be near perfect. In a wireless environment, it will be far from perfect. For applications that are able to recover from lost UDP messages but only at a considerable cost, suddenly going from an environment where messages are rarely lost to one in which they are constantly lost can result in a performance disaster.

In this paper, UDP and TCP are analysed over MMP and the possible effects of handoff losses in those implementations are studied. Attention is specially focused on TCP because its congestion control and transmission policy affects a connection with multiple handoffs in a particular way. In the following sections the tools used to obtain simulation results in *ns2* are described. Discussion and future work are eventually presented.

II. MULTICAST FOR MOBILITY PROTOCOL (MMP): ARCHITECTURE AND IMPLEMENTATION

Although many IP-Micromobility Protocols have been proposed to reduce delay and packet losses during handoffs, e.g. [2] and [3], MMP has been chosen as the protocol to run the simulations due to the good results obtained in [1]. In this section, the details of the network implementation in *ns2* and how MMP works are described.

As presented in [1], MMP uses a sparse mode multicast routing protocol Core Base Trees (CBT) [4] to handle the movement of mobile nodes within a Foreign Network. This scheme uses a shared-tree, to and from a centre point called the Core of the network. MMP relies on Mobile IP Agent Discovery procedure in order for mobile hosts to discover relevant Mobility Agents and obtain a multicast *care-of-address*. The base stations will transmit periodic beacons with *Agent Advertisement* messages including a multicast *care-of-address*. After acquiring the *care-of-address*, the Mobile Host transmits a *Registration Request* to the base station. The base station will send a *Join Request* to the Core and create a permanent group. When the Mobile Host

0-7803-7097-1/01/$10.00 ©2001 IEEE

moves to another cell, it will initiate handoff when it receives a stronger beacon from another base station and it will include in its *Registration Request* the *care-of-address* of the group it belongs to, so the base station can extract the multicast address and send a *Join Request* to the core to join the group already established.

The test network used in *ns2* is shown in figure 1. The simulation starts when the mobile host is in cell 1 and moves from one base station to the other until it reaches cell 12. The cell size is 30 metres and the speed of the mobile host is 0.83m/s. Internet and foreign network data rates are set to 7Mbps and 2.5Mbps respectively. Point to point links have a delay of 10ms for all the links except for the links between the base stations and their correspondent router where the delay is 1ms.

III. UDP PERFORMANCE

UDP [5] traffic was used as the traffic load generated by the Corresponding Host with throughputs between 1024Kbits/s and 2512Kbits/s for a packet size of 64 bytes and a constant and exponential traffic generation.

The results show that packet losses are insignificant during handoffs even for the worst case when the handoff distance, that is, the number of hops a *CBT Join Request* traverses before an old on-tree router is reached, is three.

The exponential character of the graphic demonstrates the good performance of MMP until the maximum throughput of the links is reached. The average packet losses in the case of exponential traffic mode, see figure 3 in [6], shows the same results as for the constant model hence the extreme cases of maximum number of lost packets are considered.

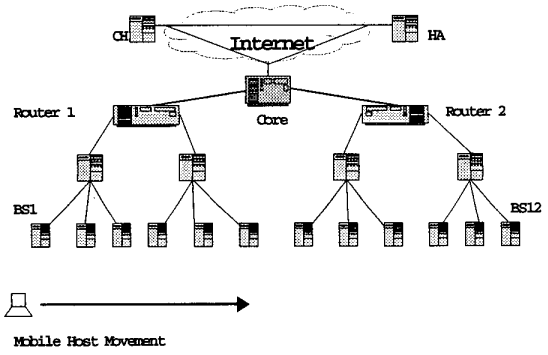

Fig. 1. Testbed Configuration. HA stands for Home Agent, CH for Correspondent Host, BS for Base Station

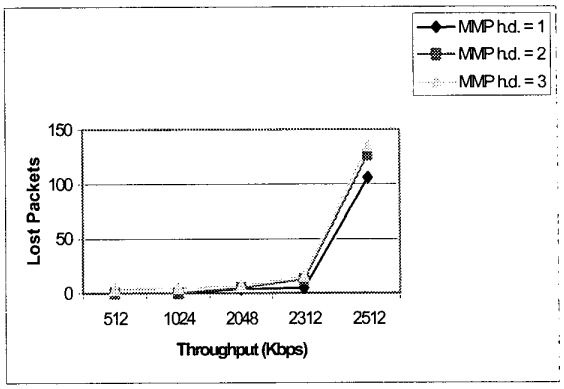

Fig. 2. Lost packets for CBR Traffic Model, constant packet size of 64 bytes.

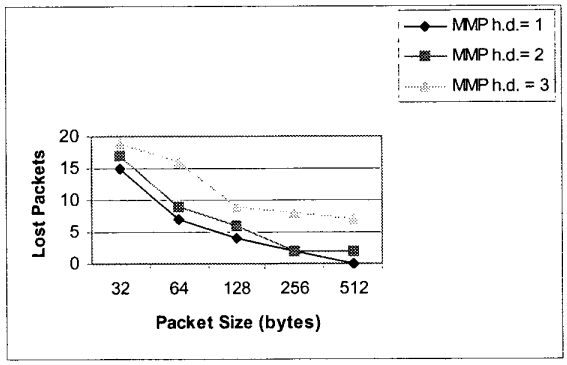

Fig. 3. Lost packets for CBR Traffic Model, constant throughput of 2312Kbits/s.

With a fixed throughput of 2312Kbits/s for both constant and exponential traffic generation, when a smaller packet size is used, the packet stream is more dense so the number of packets that are lost in a handoff for different handoff distances rises.

As it will be demonstrated in the next section, in contrast to TCP, UDP does not react to packet losses and thus often achieves a higher throughput than TCP. It never performs backoff and always sends data when transmission is possible, that is why, UDP presents the optimum throughput a TCP connection could achieve, if additional measures were taken to prevent TCP from backing off in case of non-congestion related packet loss.

IV. TCP PERFORMANCE

TCP (Transmission Control Protocol) was formally defined in RFC 793 [7], after that, some bugs were detected and new versions of the protocol were introduced [8, 9, 10, 11, 12, 13, 14]. TCP relies on Flow Control, Error Control and Congestion Control to maintain its connections: Flow Control in TCP is based on a window that limits the number of bytes the sender can transmit before receiving any acknowledgement from the receiver. At time *t*, the window

0-7803-7097-1/01/$10.00 ©2001 IEEE

size is denoted by *W(t)*, and is equal to the maximum allowed number of acknowledged packets (not counting retransmissions). The window varies dynamically in response to acknowledgements and to detection of packet loss. The destination sends back cumulative acknowledgements: if all data up to *(n-1)* have been received, then the receiver sends back the acknowledgement "*next expected* = *n*." When varying the size of the window the throughput of a TCP connection can be set to an appropriate value for the actual transmission. At any time, this throughput can be calculated as the window size divided by the Round Trip Time of the connection. Error Control, on the other hand, is implemented using sequence numbers, retransmission timers and cumulative acknowledgements (ACK). Whenever the destination receives a packet, it is acknowledged, this ACK contains the sequence number of the last byte received. Thus, a single packet loss can be detected by consecutive acknowledgements having the same "*next expected*" number, so that TCP retransmits the packet after the number of such duplicate acknowledgements exceeds a threshold (typically three). At the same time by receiving an ACK we can estimate the Round Trip Time of the connection and set a timer when the packet is sent, if the timer expires (Timeout) the first packet sent and not yet acknowledged is retransmitted. In the end, Congestion Control is based on the change of the window size and the throughput of the connection as a function of network conditions. Every sender will then maintain two windows, the window the receiver has granted and the congestion window (each of them reflects the number of bytes the sender may transmit). Therefore, the effective window is the minimum of what both, the sender and the receiver, think is all right.

Different schemes of TCP implement Flow, Error and Congestion Control in a certain way, that is why they present a particular performance depending on the scenario they are running. For instance, Tahoe [13] calls slow-start after it discovers a packet loss, Reno [8] considers that every duplicate ACK is a signal that a packet has left the network (fast retransmit) and injects a new packet if the congestion window allows (fast recovery), New Reno [9] stays in fast recovery until all the losses in the same window have been recovered, Selective Acknowledgement [10] reads the information contained in the header of a TCP packet, the three blocks of contiguous data most recently received at the destination, so more than one loss per Round Trip Time can be detected, Forward Acknowledgement uses the additional information provided by the SACK option to keep an explicit measure of the total number of bytes of data outstanding in the network and Configurable Delay per ACK delays only in-order packets, meanwhile out-of-order packets cause immediate ACK generation, while Vegas [14] establishes a "more efficient" retransmission and congestion avoidance mechanism.

The first round of simulations run in *ns2* showed that Tahoe TCP presented a very good performance, so this protocol was adapted fixing in simulation time slow-start option to false, configurable delay per ACK and number of duplicate ACKs received to start a retransmission to 2.

The performance of TCP over MMP is compared through the distribution of the sequence numbers of the packets sent versus simulation time as well as the overall throughput received and sent in the different connections.

A performance comparison of the various end-to-end

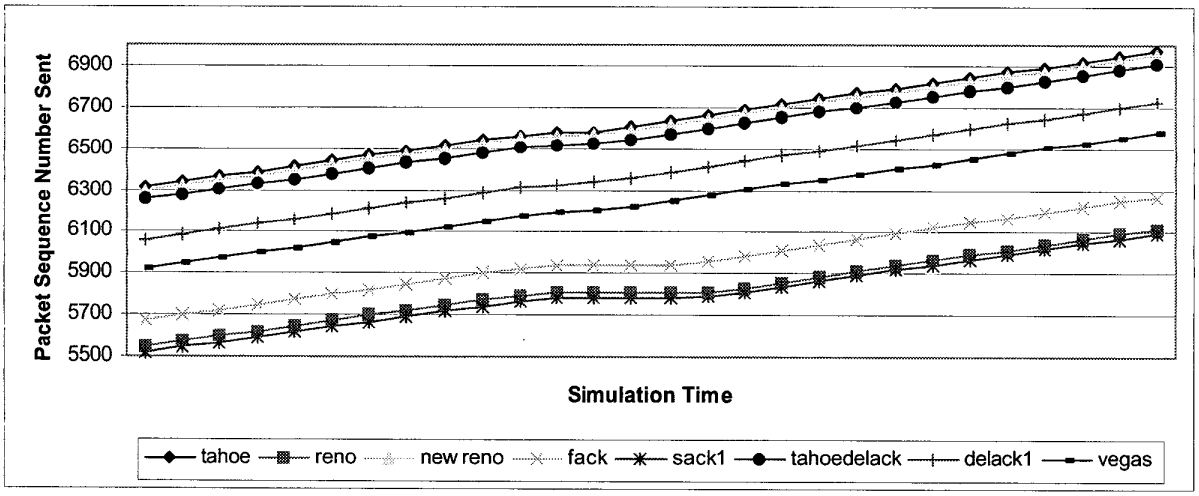

Fig. 4. Sequence Number Sent versus Simulation Time for the last seconds of the simulation: adapted Tahoe as defined in [13], Reno as in [8], New Reno as in [9], Forward Acknowledgement as in [11], Selective Acknowledgement as in [10], Tahoe TCP with Configurable Delay per ACK (tahoedelack) as in [15], Sack TCP with Configurable Delay per ACK (delack1) as in [15] and Vegas as in [14].

0-7803-7097-1/01/$10.00 ©2001 IEEE

protocols is summarised in figure 4 for the last seconds of the simulation when the last handoff occurs. Those TCP schemes that reach a higher sequence number at the end of the simulation will also present a better throughput performance as it can be shown in figure 5. Every sample represents a packet sent, so two samples with the same sequence number in different simulation times indicate a retransmit. The desirable behaviour on this graph would be a smooth line of dots extending diagonally from the lower left to the upper right, that is, the sharper the slope is the better performance is indicated. The "flat" lines in the middle of the graph show the retransmission of the same packet in different simulation times; this happens when a handoff occurs, packets are lost and it is necessary to retransmit them, therefore those schemes that present longer "flat" periods do not react as well as other schemes to packet losses in this kind of environments.

As indicated in figures 4 and 5, Tahoe TCP [13] with the modifications introduced in simulation time presents a better performance reaching the best throughput transmission. The rest of the schemes that were posterior modifications of TCP [8,9,10, 11, 12, 14] seem to be not very appropriate for a micromobility environment. This is due to the fact that TCP Tahoe attacks the lost of a packet calling a congestion control algorithm that adapts "slowly" to the new situation recovering "quickly" from the lost. The posterior modifications of other algorithms can improve performance in a wired network but they are not appropriate for a wireless micromobility environment.

Figure 6 shows the typical throughput of the adapted Tahoe TCP connection in the micromobility environment shown in figure 1, the "dents" in the throughput are caused by packet losses when a handoff occurs: TCP increases its rate until it reaches the maximum bandwidth of the link, experiences a loss, and then backs off again. It can also be appreciated that when the mobile host is moving from Base Station 6 to 7 (the sixth handoff), the "dent" is more abrupt because more packets are lost (handoff distance = 3) and the transmission is more affected. The differences between the sent throughput and the received throughput are due to the distance between both nodes, incurred delay, and the fact that the sent throughput includes the packets that need retransmission because they got lost before arriving to the sink.

The next round of simulations was focused on the study of the smoothed averaged Round Trip Time and the variation of the size of the Congestion Window along the simulation time. As it can be seen in figure 7, the estimation of the Round Trip Time is important in the way that a miscalculation of such a time will lead to a non-necessary retransmission and a consequent waste of the available throughput. It can be assumed that that an unstable network will change constantly its estimations of this time and it will result in an unstable performance. The variations on the size of the Congestion Window are related to the number of packets a sender may transmit depending on network conditions. Figure 8 shows the variation for Tahoe TCP. The window increases smoothly until a packet is dropped due to handoff and it starts again. As it can be observed, the periodicity of the variation corresponds to the periodic handoffs the Mobile Host suffers when moving in the environment of figure 1. Other examples of variation as well as other results can be examined in [6].

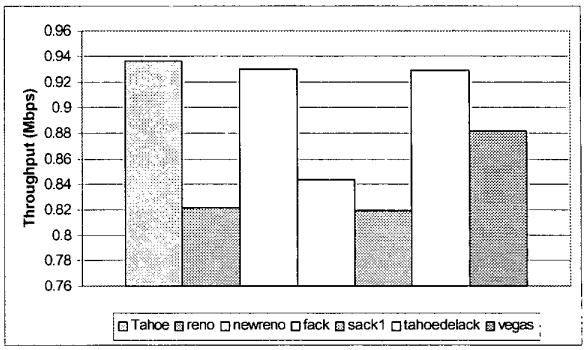

Fig. 5. Averaged Throughput received for different TCP connections.

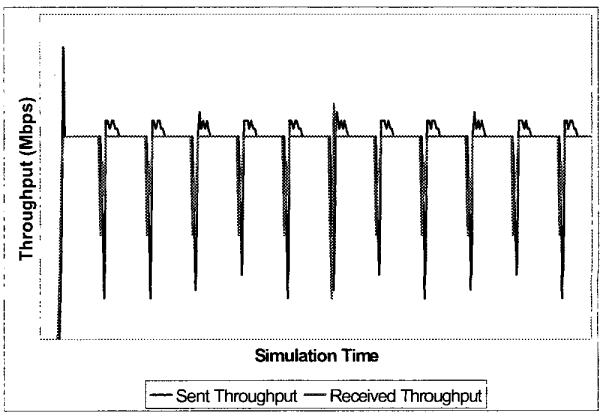

Fig. 6. Sent and received throughput comparison, adapted Tahoe TCP.

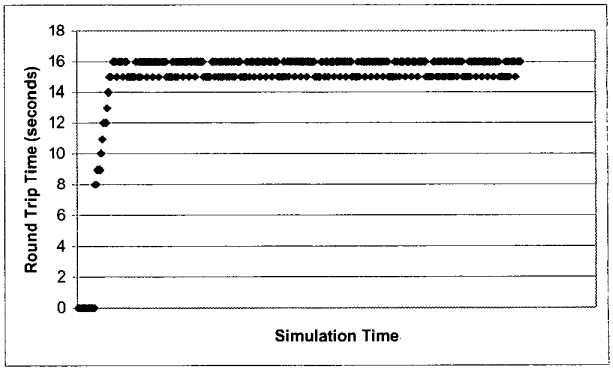

Fig. 7. Smoothed Averaged Round Trip Time versus simulation time, adapted Tahoe TCP.

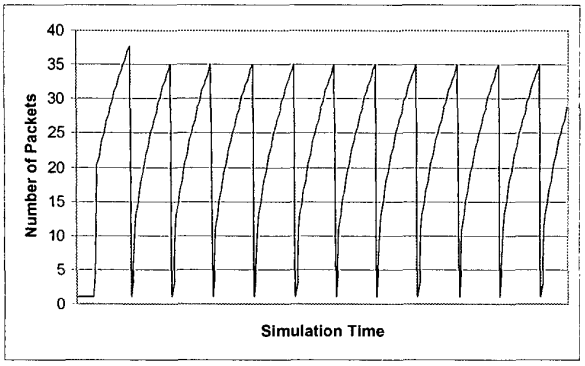

Fig. 8. Variation of the size of the Congestion Window versus simulation time, adapted Tahoe TCP.

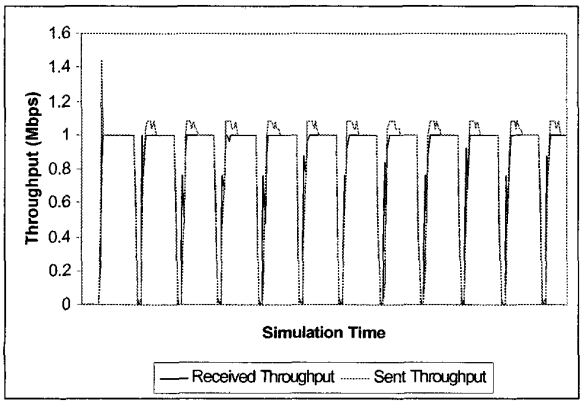

Fig. 9. Sent and received throughput comparison, SACK TCP.

On the other hand, the worst case of performance corresponds to SACK TCP, as it is shown in figure 9, the "dents" in the received and sent throughput are much deeper and wider than for the best case of performance: adapted Tahoe TCP. This is due to the fact that SACK TCP was thought to solve the problems of burst of losses in the same Congestion Window and running TCP simulations in a micromobility environment over MMP reduces the number of packets losses and therefore the probability of bursts. Nevertheless, the information provided by SACK TCP can be used by other schemes such as FACK TCP to eliminate the incurred delay when a loss in encountered.

V. Conclusions and Future Work

In this paper, the simulation results of UDP and different kinds of TCP over MMP (Multicast for Micromobility Protocol) using *ns2* have been presented.

The study has shown that the performance of UDP is much better in a micromobility environment when we get rid of the frequent location updates needed in Mobile IP.Tahoe TCP with the adapted retransmission policy has been shown to be the protocol that better works in an environment of multiple handoffs whereas other schemes that perform well for a wired network present a poor performance in a micromobility environment.

Our next study will be focused on the study of the performance of TCP and UDP over other micromobility protocols such as Cellular IP [2] or HAWAII [3] and the possible definition of a new TCP protocol specially designed for a wireless environment as proposed in [16].

Acknowledgement

The authors would like to thank Accenture Integrated Marketing and Solutions Engineering as well as Jamie J. Gylden, Partner and Specialty Leader of the Networks Technology Group in the Accenture Technology Park of Sophia Antipolis for sponsorship in the presentation and publication of the paper.

References

[1] A. Mihailovic, M. Shabeer, A. H. Aghvami, "Multicast for Mobility Protocol (MMP) for emerging Internet networks." Proceedings of the eleventh IEEE International Symposium (PIMRC 2000) September 2000.

[2] A. G. Valko, "Cellular IP, " Internet Draft, November 1998.

[3] R. Ramjee, T. La Porta, "IP micro-mobility support using HAWAII," Internet Draft, June 1999.

[4] A. Ballardie, "Core Based Trees (CBT version 2) Multicast Routing," RFC 2189, September 1997.

[5] J. Postel, "User Datagram Protocol", J. Postel. RFC 768. August 1980.

[6] A. Delgado, A. Mihailovic, N. Georganopoulos, A. H. Aghvami, "Evaluating the performance of Transport Protocols over MMP", Proceedings of the fifth London Communications Symposium (LCS2000), September 2000, London, UK.

[7] Defense Advanced Research Projects Agency. RFC, RFC 793, Sept. 1981.

[8] W. Stevens. " TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms", RFC 2001, Jan.1997.

[9] S. Floyd and T. Henderson. "The NewReno Modification to TCP's Fast Recovery Algorithm". RFC, RFC2582, April 1999.

[10] M. Mathis, J. Mahdavi, S. Floyd and A. Romanov. "TCP Selective Acknowledgement Options". RFC, RFC 2018, Oct. 96.

[11] M. Mathis and J. Mahdavi. "Forward Acknowledgment: Refining TCP Congestion Control", in ACM SIGCOM, August 1996.

[12] H. Balakrishnan and V. Padmanabhan. "A comparison of Mechanisms for improving TCP performance over Wireless Links", in ACM SIGCOM, August 1996.

[13] V. Jacobson. "Congestion Avoidance and Control", in ACM SIGCOM, CA, USA, August 1988.

[14] L.S. Brakmo, S.W. O'Malley, L.L. Peterson. "TCP Vegas: New Techniques for Congestion Detection and Avoidance", University of Arizona.

[15] K.Fall and K. Varadhan. "ns Notes and Documentation". October 1999.

[16] C. Barakat, E. Altman, W. Dabbous, "On TCP Performance in an Heterogeneous Network: Asurvey," Unit of Research INRIA, Sophia Antipolis, July 1999.