Downlink TCP Performance Enhancement at Handoff for FMIPv6-enabled Nodes

Jorge Espi, Robert Atkinson, David Harle and Ivan Andonovic
Centre for Intelligent Dynamic Communications
University of Strathclyde
Glasgow, United Kingdom
{jorge.espi, r.atkinson, d.harle, i.andonovic}@eee.strath.ac.uk

Colin Arthur
Agilent
Edinburgh, United Kingdom
{colin_arthur}@agilent.com

***Abstract*—TCP's downlink performance during FMIPv6-enabled IEEE802.11g handoff is examined. The use of a user-initiated Handoff Notification, sent prior to handoff, is proposed. Therefore, the correspondent node is aware of the handoff-induced disruption, hence avoiding dropping the congestion window size and triggering Fast Retransmission mechanisms. During handoff, it continues to send data packets which are buffered by the new access router. After handoff, the user notifies the handoff completion to the correspondent so that normal TCP operation resumes. The Handoff Notifications are implemented as TLVs within a TCP option. Simulation results show a network throughput increase of up to 35% over standard TCP.**

I. Introduction

Fast Mobile IPv6 (FMIPv6) [1] enables a proactive approach to handoff: before handoff, the mobile node (MN) forms a new care-of address and solicits the present/previous access router (PAR) to start forwarding packets to that address at the next/new access router's (NAR) link. As a consequence, the communication disruption is limited to the link layer procedures, i.e., synchronizing to the new IEEE 802.11 access point. Theoretically, the packets addressed to the MN's Previous Care-of Address (PCoA) are not lost, as they are forwarded to the New CoA (NCoA) and buffered by the NAR until the MN has joined the NAR's link and requests them.

The correspondent node (CN), unaware of the MN's handoff, continues to send TCP packets while reducing the TCP usable send window size. However, TCP was engineered for wired, fixed topologies and therefore it assumes that any losses or round-trip time (RTT) deviations are due to congestion. Therefore, during FMIPv6 handoff, even if no packet loss occurs, TCP assumes network congestion due to the disconnection time, and reacts triggering congestion control mechanisms, thereby decreasing the network throughput.

In those cases where the timeout for the outstanding data occurs before the handover termination, TCP will perform Slow Start. Otherwise, in those cases where the handoff process is finished before the retransmission time-out (RTO) expires, the CN will perceive a larger RTT and it will re-compute TCP connection status variables. This could entail higher RTO values thereby delaying lost packet retransmissions.

Moreover, since the CN is un-aware of the MN's handoff, it continues sending TCP packets while reducing the usable TCP window size as the MN is not able to acknowledge them. When the TCP usable window size decreases to zero, the CN stops sending packets.

Although these issues have been extensively explored, previous work only addresses the RTO expiration of unacknowledged data at the CN ([2], [3], [4]). The TCP enhancement proposed in this paper aims to make an efficient use of the NAR's buffering capability while dealing with the issues targeted by previous work, i.e., RTO expiration and congestion window size dropping.

The rest of this paper is structured as follows. Section II introduces the proposed scheme. Section III explores the performance of TCP Reno and the suggested algorithm via simulation. Finally, Section IV presents the conclusions of this work.

II. Proposed Solution

This article introduces a MN-initiated handoff notification scheme to the CN. The proposed scheme works as follows. After TCP's three-way session establishment phase the CN keeps track of the average amount of bytes sent each RTT ($\bar{V}$) in 500ms intervals, to avoid data transmission rate spurious consequence of bursty nature of TCP.

Prior to handoff, the FMIPv6-enabled MN triggers the signalling tasks according to [1]. In this proposal, the Proxy Router Advertisement message is modified to include the available capacity of the NAR's buffer allocated to the MN. Subsequently, the MN sends a HO Start Notification message to the CN. The notification is embodied in a TCP option header field. This option header, shown in Fig. 1, has a 1-byte kind field and a 1-byte length field, followed by a 4-bit expected handoff delay field and a 4-bit buffer size field. The kind field is arbitrarily set to the first non-allocated TCP option kind number, i.e., 29 [5]. The length field is set to three (byte addressing).

The significance of the expected handoff delay is relative, as the MN may suffer unexpected delays at handoff and, due to the different nature of heterogeneous networks, data rates and handoff delays are expected to vary within a wide dynamic range. They are, however, correlated to the radio access technologies involved in the handoff and therefore they can be estimated. For this reason, this field does not set a fixed value but an order of magnitude (in base 2). The expected

978-1-4244-8016-6/10/$26.00 ©2010 Crown

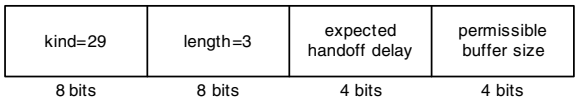


Fig. 1. TCP Handoff Notification Option

handoff delay (EHD) field is computed according to equation (1a). This approach allows for a wide span of 0-320 seconds. Similarly, the MN's permissible buffer size (PBS) field value is calculated through (1b), obtaining a span of 0-32MB. These two equations permit encoding the *real* PBS and EHD values, as advertised by the NAR by virtue of the FMIPv6 facilities, reducing the quantization error for small values.

$$EHDfield = (roundup)\, log_2 \frac{realEHD}{10ms} \tag{1a}$$

$$PBSfield = (rounddown)\, log_2 \frac{realPBS}{1kB} \tag{1b}$$

On the option fields' values computation, a conservative approach has been taken: While the EHD value is rounded up, the BS is rounded down. Therefore, the CN is prompted to send a lower volume of data while expecting a higher handoff delay as the MN may suffer eventualities at handoff; e.g.: higher latencies on the connection establishment with NAR, NAR's buffer size diminution or packet loss.

On receipt of the HO Start Notification message from MN, the CN cancels the Time Out timers for the MN's outstanding unacknowledged data (Fig. 2). The CN also freezes the RTT value and cancels its Persist timer so that no probes are sent. Next, CN sets a Handoff Delay timer using the value retrieved from the EHD field and starts sending $\bar{V}$ bytes each RTT until the Handoff Delay timer expires, or the total amount of bytes sent since the HO Notification was received exceeds the *buffer size*, which is retrieved from the BS field. In any of those cases, once the handoff delay timer times out, the CN sets Time Out timers for the unacknowledged data, making use of the RTT.

Through the facilities provided by FMIPv6, all the packets sent to the PCoA during this period are forwarded from PAR to NAR, which buffers them. The NAR may also buffer packets addressed to the NCoA from other sources. After the MN has joined the new link, and has performed the necessary Layer-3 signalling issues (e.g. Duplicate Address Detection), it starts receiving and acknowledging the buffered TCP packets. When the CN receives HO Finish Notification message, it assumes the MN has succesfully performed handoff, so it cancels the Handoff Delay Timer and resumes normal TCP operation. Also, it sets the Time Out timers for the unacknowledged data, making use of the RTT.

On expiration of a RTO for data sent at MN's handoff, the CN assumes handoff failure (at transport level). Generally, RTO expiration is a consequence of either failure of the Layer-3 handoff (MIPv6-related procedures); or impossibility of the MN to acknowledge data before the RTO expires, due to either network congestion or low throughput in the new link.

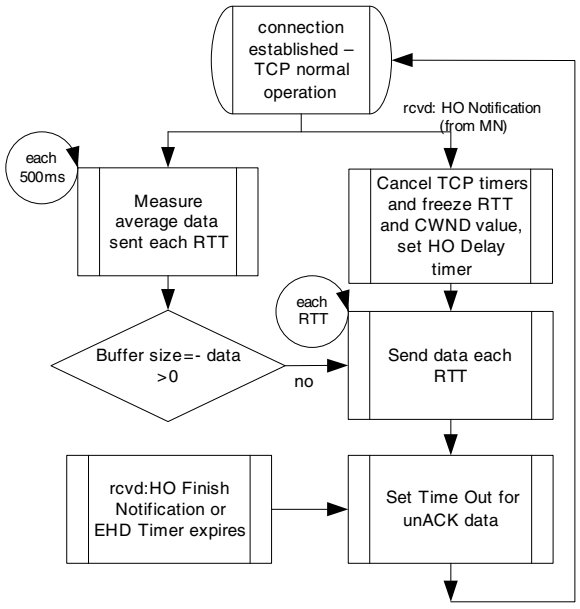


Fig. 2. Sender's algorithm

Also, RTOs may expire as a result of packet loss. This packet loss can be consequence of Layer-2 procedures, Layer-3 signalling mechanisms or buffer overflow at NAR.

The existing solution is designed so that reacts in the same way to any of these eventualities: dropping the congestion window size.

As there are multiple potential sources of packet loss, the authors recommend implementing SACK-based acknowledging mechanisms ([6], [7]) to avoid the retransmission of the packets correctly sent.

On another note, [8] explores the reordering extent of a UDP stream at NAR for different FMIPv6-based signalling protocols. Packet reordering tends to increase due to the diversion of the data stream towards the new link from the previous link. This result is expected to hold for TCP flows.

III. Experimental Results

Simulations have been conducted to measure a range of parameters related to the TCP goodput. The congestion window evolution and the average transmission rate during handoff are the metrics of interest. Packet reordering may occur as a result of flow diversion on PAR and HA and that this may have a negative impact on upper layer protocol performance. However, the effects of packet reordering are out of the scope of this paper.

A. Experimental Set-up

The simulations have been carried out using the INET Framework for OMNeT++ [9]. Fig. 3 shows the simulated system model. The network is composed by 6 participating nodes, and the links interconnecting them are configured with a packet latency and throughput values consistent with the anticipated latencies in a real system between the CN, HA, MN, PAR, and NAR. The wired links are Gigabit Ethernet

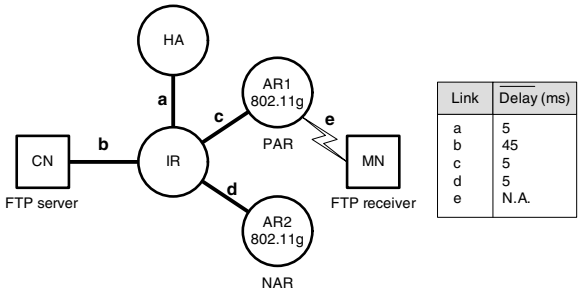


Fig. 3. System model

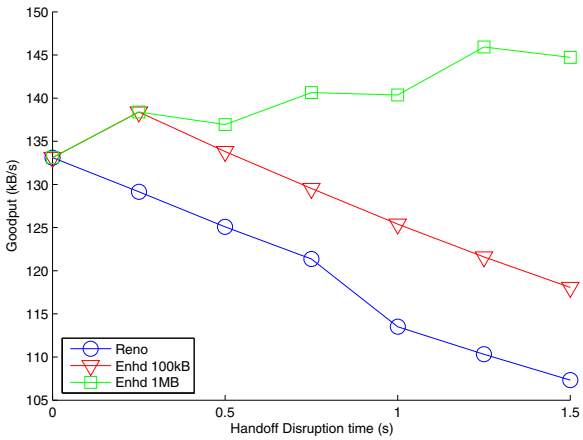


Fig. 4. Average throughput for a 1MB file download

type. Link delays follow a Gamma distribution. The wireless link is 54 Mbps IEEE802.11g.

The performance of the proposed scheme, referred as *enhanced TCP*, is compared with TCP Reno's as the handoff disruption time varies in two scenarios differentiated by the buffer size available at NAR: 100kB or 1MB (the choice of these values will be explained in next section). The handoff delay ranges between 0s and 1.5s. A handover delay of 0s, is of particular interest because, from a practical perspective, it provides a realistic assessment of the performance that may be expected where the MN were multihomed ([10], [11]) - this is particularly pertinent given the plethora of terminals with multiple interfaces on the market at present. Other handover delay settings permit comparative analysis for a range of handover delays since previous studies have demonstrated that relative performance is dependent on layer 2 handover latency. All cases assume an identical service: a downlink TCP flow from the CN to the MN. At a predefined time a layer 2 hint [12] is simulated which initiates the PRO-FMIPv6 protocol-driven handover [8].

B. Performance Evaluation

Fig. 4 illustrates the average throughputs of TCP Reno and Enhanced TCP (100kB and 1MB buffer sizes). Handoff delay has not been taking into account for the throughput averaging, providing a fairer view of the transmission rates out of the L2 handoff disconnection period. As expected, the TCP Reno throughput decreases while increasing the handoff disruption time. This decrease is more noticeable when the handoff takes longer than 1s, which is exactly the RTO value that computed by the CN. When this happens, CN triggers Slow Start as it assumes network congestion. Alternatively, enhanced TCP offers higher throughputs. In case NAR's buffer is limited to 100kB, the CN injects up to 100kB of unacknowledged data. If the handoff delay, defined as

$$HOdelay = HOfinish - HOstart \tag{2}$$

satisfies the inequality

$$PBS < \int_{HOstart}^{HOfinish} txrate_{noHO}(t)dt \tag{3}$$

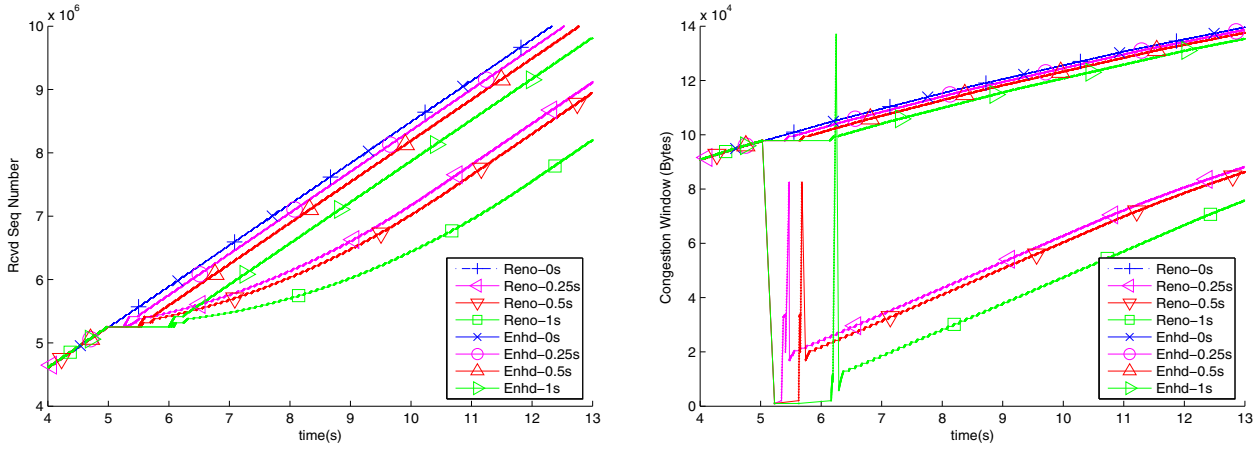
where $txrate_{noHO}$ stands for transmission rate on the no-handoff scenario and BS for buffer size, then the throughput is lower than the throughput in the no-handoff scenario. The larger the inequality, equation (3), the lower is the performance. If the NAR's buffer size is augmented from 100kB to 1MB, Fig. 4 reports an augment of the throughput as the handoff delay increases. The rationale behind this is that the send window advertised by a MN to optimize its download throughput, while keeping in with a conservative congestion avoidance philosophy, should be computed as:

$$capacity(bits) = bandwidth(bits/s) \cdot RTT(s) \tag{4}$$

However, in the simulation environment, the send window size is arbitrarily set to 64*MSS, i.e., 65536 bytes, and therefore the 1MB-buffer at NAR acts as a 1MB extension of the send window size, which entails a higher performance. Also users advertising higher window sizes will perceive an improvement on network goodput. This result highlights the fact that, for an optimum performance, the MN should advertise a PBS value as close as possible to that in Equation (3), but limited by two factors: firstly, it has to be lower or equal than the NAR's advertised PBS. Secondly, the MN, and based on the available bandwidth at the NAR, must be able to receive and acknowledge every buffered segment within RTO seconds.

As a matter of interest, using (4) is not always the case because the maximum allowable window size advertisement is 65536 bytes. The Window Scale Option circumvents this issue, extending the size up to 230 bytes (1GB), but it can only appear in a SYN segment. Therefore, it must be set at the connection establishment.

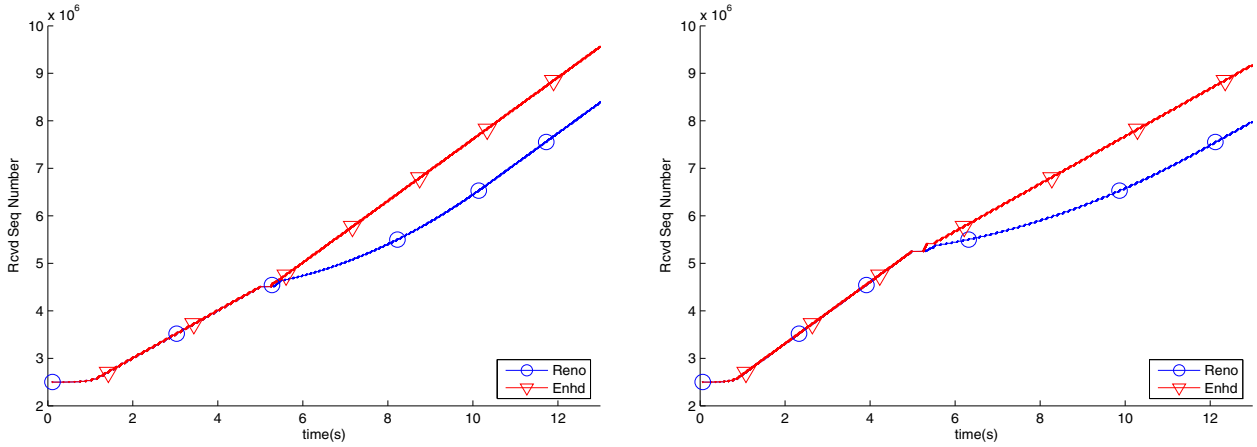
Fig. 5a shows the TCP Reno's and Enhanced TCP's (1MB buffer size) congestion window evolution during the 1MB file download for different layer 2 handoff delays (0, 0.25, 0.5 and 1 s). At t=0s the MN sets a TCP connection with the CN. At t=5s the MN, already in congestion avoidance stage, starts handoff. For a 0.5s handoff delay, a TCP Reno-enabled sender waits until an acknowledgment is received. TCP would treat this as episodic network congestion, but it triggers no course of action except updating the RTT Variance (RTTVAR) and Smooth RTT (SRTT) connection variables. Conversely, if the



(a) Mobile node's received sequence number evolution

(b) Correspondent node's congestion window evolution

Fig. 5. Effect of layer 2 handoff disruption



(a) c=20ms, d=5ms.

(b) c=5ms, d=20ms.

Fig. 6. Effect of different link delays

handoff delay is larger than the RTO (1s in the simulation environment), TCP Reno's congestion window is reduced to one MSS and Fast Retransmissions are triggered.

Alternatively, the Enhanced TCP-enabled CN maintains the same data transmission rate. On receipt of HO Finish Notification from MN, the CN resumes normal TCP operation: Packet acknowledgement receipt results on an augment of the congestion window and CN keeps track of all the connection-related parameters, such as RTT, SRTT, RTTVAR and RTO.

Fig. 5b depicts the sequence number of the received TCP segments versus time for different layer 2 handoff delays (0, 0.25, 0.5 and 1 s), from which a comparison between TCP Reno and Enhanced TCP can be observed. TCP connection starts at t=0, sequence number set to 25000. The Enhanced TCP curves show how, after handoff, the MN receives the buffered data packets (at NAR) so that it resumes normal TCP operation without decreasing the average data rate. Reno TCP curves, however, are severely affected by handoff, especially for handoff delays higher that the RTO value (they trigger Slow Start).

Fig. 6 illustrates the effect of handoff on the congestion window size for a fixed 0.25 s layer 2 handoff delay. In Fig. 6a the delay of link c (MN-AR1) is set to 20 ms, and the delay of link d is set to 5ms. In Fig. 6b, the delays of links c and d are set to 5 and 20 ms respectively. The Enhanced TCP curves show that the CN's congestion window dropping was avoided, whereas the Reno TCP curves show congestion mechanisms were triggered, thefore reducing the network goodput.

IV. Conclusions

In this article, a MN-initiated Handoff Notification scheme for FMIPv6-enabled networks has been proposed. This scheme provides more efficient downlink TCP behaviour at FMIPv6 handoffs. There are two reasons for this. In first term, while MN's handoff, the CN freezes the congestion window value while continues to send data packets, which are buffered by the NAR. In second place, after handoff, the MN sends a handoff termination message and continues to acknowledge

data packets so that CN increases the congestion window value according to the TCP standard behaviour.

Though simulation, the proposed scheme has proved to outperform TCP Reno. Results confirm an augment of the network throughput of up to 35%. These results are expected to be replicated for other TCP flavours, due to their similarities on tackling network congestion at RTO Timer expiration.

REFERENCES

[1] R. Koodli, "Mobile IPv6 Fast Handovers," RFC 5568 (Proposed Standard), Internet Engineering Task Force, Jul. 2009. [Online]. Available: http://www.ietf.org/rfc/rfc5568.txt

[2] T.Goff *et al.*, "Freeze-TCP: A true end-to-end TCP enhancement mechanism for mobile environments," in *Proc. IEEE IEEE Computer and Communications Societies (INFOCOMMS)*, Nagoya, Japan, 2000, pp. 1537–1545.

[3] M. Yoshimoto, K. Kawano, K. Kinoshita, T. Matsuda, and K. Murakami, "Handoff performance enhancement for tcp-based streaming services in heterogeneous networks," in *Local Computer Networks, 2007. LCN 2007. 32nd IEEE Conference on*, 15-18 2007, pp. 703 –710.

[4] D. Le, D. Guo, and B. Wu, "Wlc47-6: Tcp performance improvement through inter-layer enhancement with mobile ipv6," in *Global Telecommunications Conference, 2006. GLOBECOM '06. IEEE*, nov. 2006, pp. 1 –5.

[5] S. Bradner and V. Paxson, *IANA Allocation Guidelines For Values In the Internet Protocol and Related Headers*, IETF Std. RFC 2780, 2000.

[6] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow, "TCP Selective Acknowledgment Options," RFC 2018 (Proposed Standard), Internet Engineering Task Force, Oct. 1996. [Online]. Available: http://www.ietf.org/rfc/rfc2018.txt

[7] E. Blanton and M. Allman, "Using TCP Duplicate Selective Acknowledgement (DSACKs) and Stream Control Transmission Protocol (SCTP) Duplicate Transmission Sequence Numbers (TSNs) to Detect Spurious Retransmissions," RFC 3708 (Experimental), Internet Engineering Task Force, Feb. 2004. [Online]. Available: http://www.ietf.org/rfc/rfc3708.txt

[8] J. Espi, R. Atkinson, I. Andonovic, and J. Dunlop, "Proactive route optimization for fast mobile ipv6," in *Vehicular Technology Conference Fall (VTC 2009-Fall), 2009 IEEE 70th*, 20-23 2009, pp. 1 –5.

[9] INET Framework for OMNeT++. (2009, Oct.) OMNeT++ Community Site. [Online]. Available: http://www.omnetpp.org

[10] J. Abley *et al.*, *Goals for IPv6 Site-Multihoming Architectures*, IETF Std. RFC 3582, 2003.

[11] G. Huston, *Architectural Approaches to Multihoming for IPv6*, IETF Std. RFC 4177, 2005.

[12] The 802.21 Working Group. (2009, Oct.) IEEE. [Online]. Available: http://www.ieee802.org/21