Adaptive Packet Combining for IPv6 Soft Handover applied to Network Mobility

Hirokazu Naoe
Advanced Telecommunication Laboratory
SHARP corporation
Chiba, Japan

Michelle Wetterwald
Mobile Communication Department
EURECOM Institute
Sophia Antipolis, France

Christian Bonnet
Mobile Communication Department
EURECOM Institute
Sophia Antipolis, France

***Abstract*—Seamless vertical handover is one of the major challenges toward seamless inter-networking between heterogeneous access networks. To perform seamless vertical handover even under bad radio conditions in Network Mobility (NEMO) environments, NEMO-SHO has been proposed as an IPv6 soft handover extension that allows packet bicasting via two different wireless links simultaneously. However, the performance gain for TCP traffic was limited because spurious TCP retransmissions were invoked due to the difference of bandwidth and transmission delay between the two links. This paper proposes adding flow control to the packet combining process in NEMO-SHO, which is called Adaptive Packet Combining. Adaptive Packet Combining estimates packet arrival time from a slow link and delays packets coming from a fast link based on the estimation, which can hide the difference of bandwidth and transmission delay between two links from TCP without any modification on nodes inside a moving network. Adaptive Packet Combining is thus able to reduce spurious TCP retransmissions. Using 3G and IEEE802.11b link, our experiments show that Adaptive Packet Combining achieves better TCP goodput than Make-Before-Break handover under lossy radio conditions.**

I. Introduction

Recent mobile communication networks are composed of multiple radio access technologies to take advantages of different characteristics of radio access technologies in terms of transmission bandwidth, radio coverage and so on. Thus, one of the major challenges is seamless vertical handover between different access technologies. In those heterogeneous access networks where no common L2 architecture exists, vertical handovers need to be handled at IP layer or upper layers.

To perform seamless vertical handover even under bad radio conditions, NEMO-SHO [1] has been proposed as an IPv6 soft handover extension to NEMO-Basic Support [2] that is a network mobility (NEMO) protocol standardized by the Internet Engineering Task Force (IETF). NEMO-SHO introduces packet bicasting and combining operations on a Mobile Router (MR) and a serving Access Router (AR) using two different wireless links during handover so that packet losses can be reduced if either link can successfully deliver a packet. It has a positive effect on UDP traffic, but the performance gain for TCP traffic is limited because of spurious TCP retransmissions caused by the difference of bandwidth and transmission delay between the two links.

In this work, we propose adding some flow control into the packet combining operation of NEMO-SHO called Adaptive Packet Combining (APC). APC measures transmission delay per link, estimates packet arrival time from a slow link, and buffers a packet copy until the estimated time if the first copy comes from a fast link. This process hides different delay and bandwidth between two links from TCP taking into account the maximum common link capability. Using bulk TCP traffic, we evaluate APC through experiments on our testbed with 3G and IEEE802.11b link emulation. We show that using APC can reduce spurious TCP retransmissions and consequently achieves better TCP goodput than Make-Before-Break handover under high error rate conditions.

This paper is organized as follows. In Section II, we give an overview of work related to seamless vertical handover and then state the issues on TCP during IPv6 soft handover in Section III. Section IV describes our proposed scheme called APC and its protocol operations. Our testbed and handover scenarios used for measurements are explained in Section V. We evaluate and analyze the handover performance of NEMO-SHO with APC in Section VI and Section VII, respectively. Finally, Section VIII concludes the paper.

II. Related Work

The IETF has standardized Mobile IPv6 [3] as an IPv6 host mobility protocol and NEMO-Basic Support as a NEMO protocol. To support multihoming in Mobile IPv6 and NEMO-Basic Support, multiple care-of addresses (CoAs) registration extension [4] is also being standardized. It creates a Binding Identifier (BID) per CoA, and allows a Home Agent (HA) to have multiple bindings for each Mobile Node (MN) or MR. Using this extension, Flow-Bindings [5] proposes a way to forward a particular IPv6 flow to a given BID. It defines a new mobility option called Flow Identification Option that contains policies to identify an IPv6 flow and a special treatment for the flow including N-casting action, which duplicates a packet and routes them from a HA to multiple BIDs of a MR.

To perform seamless handover, Make-Before-Break handover, which establishes a new wireless link before breaking the current link, is effective because the impact of L2 disruption time is removed. Pentander et al. [6] show that using Make-Before-Break handover in NEMO provides good Quality of Service for the nodes behind the MR, called Local Fixed Nodes (LFNs). However, as pointed out in [7], Make-Before-Break handover on heterogeneous access networks causes out-of order delivery due to considerable differences of link capability between two links. This has a negative impact on TCP since spurious TCP retransmissions are invoked despite of no packet loss.

Many TCP extensions have been proposed to avoid spurious TCP retransmissions. Ludwig et al. [8] propose TCP-Eifel that detects whether a retransmission is spurious by using TCP timestamps option and then restores the size of Congestion Window to the previous state. The essential factor in causing the issue is that TCP congestion control is executed by a TCP sender that has neither an explicit handover notification nor information about the link capability on a target link. TCP-HO [9] thus introduces new TCP options that convey the information about the bandwidth on the target link to a sender so that it can quickly adapt the target link.

III. TCP ISSUES DURING IPV6 SOFT HANDOVER

NEMO-SHO can reduce packet losses during vertical handovers using two wireless links simultaneously, which has a positive effect on UDP traffic such as VoIP over UDP in lossy radio conditions. However, the performance gain on TCP traffic is limited because of spurious TCP retransmissions caused by different transmission delay between two links even with the TCP extensions presented above.

TCP is a reliable transport protocol having its congestion control. As defined in [10], the TCP congestion control algorithm relies on a Congestion Window (*cwnd*), the amount of data that a TCP sender can transmit into a network without receiving an acknowledgment (ACK). When a TCP session is established, the *cwnd* is initialized as one segment. According to the slow start and congestion avoidance algorithm, a sender then transmits TCP segments up to the *cwnd* while adjusting it in order to fully utilize the available throughput to a receiver. A TCP retransmission is triggered when the sender detects a segment loss by either receiving three duplicate ACKs or the absence of an ACK for TCP Retransmission Timeout (RTO) that is determined from the smoothed average Round Trip Time (RTT) between TCP nodes. Once a TCP retransmission occurs, the *cwnd* is either initialized to 1 in case of RTO expiration or halved in case of three duplicate ACKs, which both crucially degrade TCP performance.

The packet combination used in NEMO-SHO is a simple procedure that forwards only the packet copy arrived earlier without waiting for the another copy. Hence, a packet lost on a fast link often causes out-of order delivery or a delay spike in the case where the two links have different transmission delay and bandwidth. RTO is also set inaccurately for a slow link. They invoke TCP retransmissions in spite of successful reception of packets. Moreover, NEMO-SHO injects the same amount of packets into two links regardless of the difference in link capability between them. This may cause accumulated queuing delay on the slow link, especially in case of bulk transfer as it tries to fully utilize the available throughput to a receiver. Finally, NEMO-SHO becomes ineffective since TCP retransmissions are always triggered prior to the reception of packets from the slow link.

Spurious retransmissions decrease TCP goodput, but also wastes network resources. Thus, it is required to deliver packets in-order and with stable delay regardless of packet losses on the fast link. At the same time, shorter delay is preferred since maximum TCP throughput is determined by transmission delay and bandwidth between TCP nodes.

IV. NEMO-SHO WITH ADAPTIVE PACKET COMBINING

NEMO-SHO with APC introduces some new packet operations as an extension to NEMO-Basic Support with Flow-Bindings. Along with packet bicasting over two links, APC has a flow control based on transmission delay measurements in its packet combining process. During handover, it buffers packets from a fast link while estimating the packet arrival time from a slow link in order to hide the differences of transmission delay and bandwidth between two links from LFNs and their correspondent nodes (CNs). Although the maximum throughput is limited by the maximum link capability on the slow link, probabilities of causing out-of order delivery and delay spikes can be minimized. This contributes to reduce spurious TCP retransmissions without any modification to LFNs and CNs.

Fig. 1 shows the signaling flow of NEMO-SHO with APC. An MR has two different wireless interfaces namely I/F1 and I/F2 for vertical handover. It is also assumed that the MR has information about the link rate per interface beforehand, thus the MR can know if a handover is from a slow link to a fast link, or not. When the MR makes a handover, the following operations are carried out.

A. Wireless link and Bi-directional Tunnels establishment

To determine a time to start soft handover, a handover threshold is defined based on wireless link quality. The MR triggers a handover when it detects that the link quality on the current link decays below the threshold even if the new link has a lower link quality than the current one. Then it establishes a new wireless link (I/F2) to the target AR while still using the bi-directional tunnel established between the MR and the HA via the current link (I/F1) simultaneously.

After connecting to the target AR, the MR establishes the second bi-directional tunnel via I/F2 using multiple CoAs registration extension. The Binding Update (BU) sent from the MR has a Flow Identification Option that starts packet bicasting through the two tunnels.

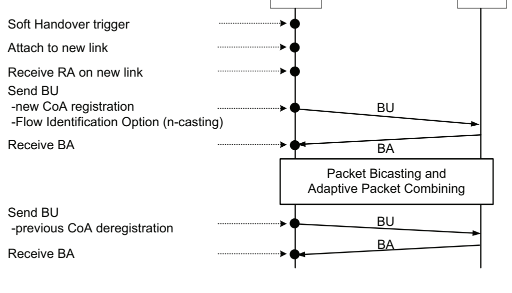


Fig. 1. NEMO-SHO signaling flow.

B. Packet Bicasting and APC operations

Packet bicasting and APC operations are applied to the down-link traffic from the HA to the MR and/or the up-link traffic. Fig. 2 shows details of the operations in case of the down-link traffic. $T_{combining}$ is initialized on the MR when the BU is sent from the MR. $T_{bicasting}$ is initialized on the HA when the BU arrives. Both base times are used to measure transmission delay per link.

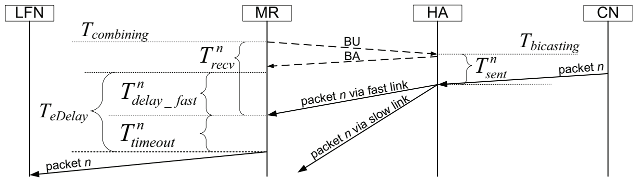


Fig. 2. Packet Bicasting and Adaptive Packet Combining.

In packet bicasting operation, the HA captures a packet destined for the MR, duplicates it and routes them to the MR through the two tunnels. When a packet is duplicated, a new IPv6 destination option is inserted after IPv6-in-IPv6 tunnel header. As depicted in Fig. 3, this option, called Packet Identifier Option (PIO), has two fields: a 16-bit sequence number field and a 16-bit timestamp field. The sequence number is used for identifying if duplicate packets are generated from the same original packet, and incremented by one each time the HA duplicates a packet. The timestamp indicates the sent time of a packet with 1 ms clock resolution. Each time the HA creates a duplicate packet having sequence number *n*, the timestamp is set to T^n_{sent}: the elapsed time from $T_{bicasting}$.

When the MR receives a packet from the fast link before arrival from the slow link, it buffers the packet until the estimated packet arrival time from the slow link. The estimation is carried out as follows.

The MR first extracts the T^n_{sent} from the PIO and then it calculates transmission delay as below:

$$T^n_{delay\_fast} = T^n_{recv} - T^n_{sent}, \tag{1}$$

where T^n_{recv} is the elapsed time from $T_{combining}$. Then the packet is buffered until the timeout given as:

$$T^n_{timeout} = \begin{cases} T_{eDelay} - T^n_{delay\_fast} & (T_{eDelay} > T^n_{delay\_fast}) \\ 0 & (T_{eDelay} \leq T^n_{delay\_fast}) \end{cases} \tag{2}$$

where T_{eDelay} is the estimated transmission delay on the slow link. T_{eDelay} is updated each time the MR receives a packet from the slow link as:

$$T_{eDelay} \leftarrow \alpha \cdot T_{eDelay} + (1 - \alpha) \cdot T^n_{delay\_slow}, \tag{3}$$

where α is a filter gain constant and $T^n_{delay\_slow}$ is the measured transmission delay of sequence *n* on the slow link, which is given as the same way with (1). The initial value of T_{eDelay} is set to zero in case of handover from the fast link to the slow link so that smooth increase of delay is performed. If a handover direction is reverse, the initial value of T_{eDelay} is set to sufficient value of avoiding the situation where a duplicate packet from the fast link overtakes outstanding packets that have been sent on the slow link before the handover. The filter gain constant α smoothes the transition of T_{eDelay}.

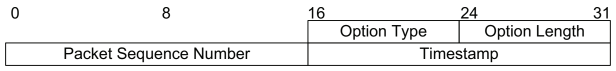


Fig. 3. Packet Identifier Option.

When the same copy arrives from the slow link before the expiration of $T^n_{timeout}$, the MR forwards only one copy to LFNs. If $T^n_{timeout}$ expires, the MR forwards the buffered packet to LFNs, and records *n* on its own Sequence Number Table that stores sequence numbers forwarded to LFNs before. This avoids injecting multiple copies of the original packet into LFNs. In order to deliver packets in-order, when the MR decides to forward the sequence number *n* to LFNs, it checks whether the sequence number smaller than *n* still remains in the buffer. If packets are there, the MR forwards them in-order even prior to $T^n_{timeout}$ of them. Due to a packet loss on the fast link, the first arrival can be from the slow link. In this case, the MR forwards the packet to LFNs without buffering.

Due to the transmission delay of the BU, $T_{bicasting}$ and $T_{combining}$ are different clock. However the effect of the clock discrepancy can be ignored when $T^n_{timeout}$ is calculated by (2). Thus time synchronization between the MR and the HA is not required.

C. Handover Completion

When the MR detects that the link quality on the new link reaches the handover threshold, it decides to complete the handover and then sends a BU to the HA in order to deregister Previous CoA and stop Packet Bicasting. Afterward, all packets tunneled from the HA come only into I/F2, and the true capability on the new link becomes available immediately.

V. Experiment setup

To evaluate the gain of APC, we carried out experiments by implementing NEMO-SHO with APC and setting up our testbed and vertical handover scenarios.

A. Implementation and testbed

We implemented NEMO-SHO with APC on Linux using NEPL-MCoA [11] as a basis. Fig. 4 shows the network topology of our testbed. 3G (UMTS) cellular link and IEEE802.11b WLAN link were selected as wireless links between the MR and two ARs (3G AR and WLAN AR). We used wireless link emulation over a wired Ethernet instead of radio transmission in order to simulate the movement of the MR between access networks. Details on link emulation and handover scenarios are explained later. The HA and the ARs are connected to the same core IPv6 subnet. The CN is directly connected to the HA. The MR has three interfaces: two interfaces for vertical handover and an ingress interface to a LFN. The HA, the ARs and the MR run on Linux kernel version 2.6.16. The CN and the LFN run on Linux kernel version 2.6.8.1.

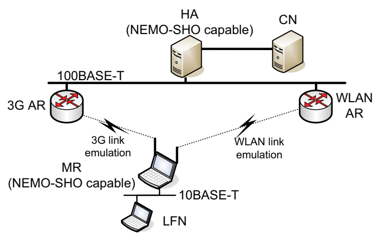


Fig. 4. Network topology of the experiment setup.

TABLE I
CONFIGURATIONS OF WIRELESS LINK EMULATION

	3G (UMTS)		IEEE802.11b
Link Rate	384 kb/s bearer (UL/DL)		1 Mb/s
Link Parameters	RLC-AM		Distributed Coordination Function
	Transmission Time Interval (TTI)	10 ms	
	Transport blocks per TTI	12	
	Block size	336 bits	
	Iub Delay	20 ms	
Retransmission	MaxDAT 5		Retry Limit 2

B. Wireless Link Emulation

Wireless link emulation was realized by using Linux NETwork EMulation (NETEM) functionality that is a waiting queue of being able to delay or drop packets. We made modifications on the NETEM to add delay or drop a packet depending on uniform random errors given by Block Error Rate (BLER) on the 3G link or Bit Error Rate (BER) on the WLAN link. The emulation models of the 3G and the WLAN link were derived from [12] and [13], respectively. All configurations are summarized in Table I.

C. Handover Scenarios

We made two handover scenarios depending on a handover direction: downward handover is defined as a vertical handover from the 3G link to the WLAN link, and upward handover is reverse direction. Each scenario consists of 20 seconds, and it defines the linear change of BLER on the 3G link and BER on the WLAN link over time as shown in Fig. 5. We focused on bad radio conditions where packet losses frequently occur due to high error rate on wireless links because the best performance of NEMO-SHO can be expected as presented in our previous result [1].

In both scenarios, the MR triggers a handover at 5 seconds, and then completes the handover at 15 seconds where the MR gets good link quality from the target link in case of NEMO-SHO. In case of Make-Before-Break handover, the MR makes a handover at 15 seconds.

VI. Performance Evaluations

We measured the handover performance on TCP bulk traffic using NEMO-SHO with APC, NEMO-SHO without APC (packet combining only based on sequence number), and Make-Before-Break Handover in the following metrics: the number of spurious TCP retransmissions and TCP goodput.

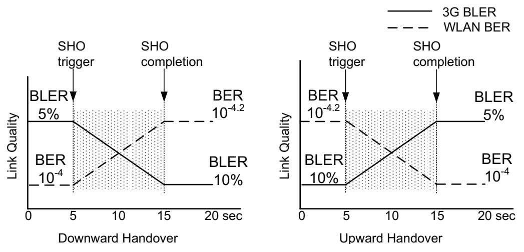


Fig. 5. Handover scenarios.

A TCP session was established between the CN and the LFN at the beginning of a handover scenario, and then the CN sent TCP Bulk traffic to the LFN for 20 seconds using a traffic generator. TCP Reno and TCP Selective Acknowledgment Option were used. We set 0.875 to the filter gain constant α in (3), and set 1 second to the initial value of T_{eDelay} in case of downward handover. 30 trials were conducted per scenario.

A. Downward Handover Performance

Table II shows the number of out-of order delivery and spurious TCP retransmissions in case of downward handover. It also shows the average TCP goodput during the time between 5 to 15 seconds. Compared to NEMO-SHO without APC, NEMO-SHO with APC significantly avoided out-of order delivery and succeeded to reduce spurious TCP retransmissions, which consequently contributed to achieve better TCP goodput than Make-Before-Break handover.

B. Upward Handover Performance

Table III shows the results of upward handover. NEMO-SHO with APC also succeeded to reduce spurious TCP retransmissions, and achieved better TCP goodput than Make-Before-Break handover. However, the number of out-of order delivery slightly increased. In case of upward handover, the initial value of T_{eDelay} is set to zero. Thus incoming packets from the WLAN link are immediately forwarded without

TABLE II
TCP PERFORMANCE OF DOWNWARD HANDOVER.

	Out-of order delivery	Spurious TCP Retransmissions	Avg. TCP goodput
Make-Before-Break HO	0 (σ 0.7)	1 (σ 0.8)	122 kb/s (σ 28.0)
NEMO-SHO without APC	152 (σ 29.0)	34 (σ 5.8)	248 kb/s (σ 28.4)
NEMO-SHO with APC	39 (σ 18.8)	5 (σ 2.9)	283 kb/s (σ 35.8)

TABLE III
TCP PERFORMANCE OF UPWARD HANDOVER.

	Out-of order delivery	Spurious TCP Retransmissions	Avg. TCP goodput
Make-Before-Break HO	0 (σ 0.0)	1 (σ 1.1)	153 kb/s (σ 32.5)
NEMO-SHO without APC	153 (σ 33.3)	34 (σ 7.0)	245 kb/s (σ 25.9)
NEMO-SHO with APC	46 (σ 8.5)	6 (σ 2.8)	292 kb/s (σ 21.5)

buffering at the beginning of handover. This is to hide abrupt increase of delay from TCP, but the probability of causing spurious TCP retransmissions slightly increases.

VII. Analysis

APC has two configurable parameters: the initial value of T_{eDelay} and the filter gain constant α used in (3). These parameters affect the estimation of packet arrival time from the slow link, thus need to be configured adequately.

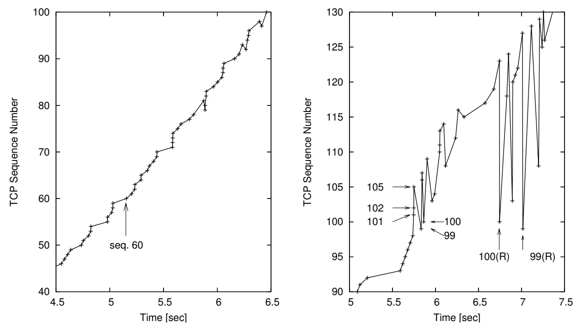
Concerning the initial value of T_{eDelay}, we set 1 second to it in case of downward handover and zero in case of upward handover. Both of them worked in our experiments as spurious TCP retransmission were almost removed. However, arrival time difference between two links can vary depending on link rate and also network congestions. Thus a fixed value of the initial value will not fit in all possible cases.

Fig. 6 shows the order of TCP sequence received at the LFN during downward handover in case of single TCP session and two TCP sessions (two TCP bulk traffic, and one session is shown in Fig. 6b), respectively. In case of single session, the first duplicate packet (TCP seq. 60) from the 3G link arrived at the MR only 102 ms later than the WLAN link. Hence, setting 1 second to the initial value was effective to avoid the situation where duplicate packets from the fast link overtake outstanding packets that have been sent on the slow link before handover. On the other hand, in case of two TCP sessions, the arrival time difference became quite large due to many outstanding packets on the 3G link. As a result, the first duplicate packet from the 3G link arrived approximately 1.1 seconds later than the WLAN link. Then the duplicate packets (seq. 101, 102 and 105) from the WLAN link overtook previous 2 packets (seq. 99 and 100) that had been sent before handover. The LFN thus received seq. 101, 102 and 105, which immediately returned three duplicate ACKs to the CN, and then invoked spurious TCP retransmissions of seq. 99 and 100. A possible solution will be to wait for the first packet from the slow link, and then use the first measured delay as the initial value.

Similarly, an appropriate value on the filter gain constant α also needs to be studied. APC should ignore temporal delay due to L2 retransmission, thus larger value would be preferred, but it would be also required to adapt network congestion by setting smaller value.

VIII. Conclusion

To improve TCP performance during vertical handover under bad radio conditions, this paper proposed Adaptive Packet Combining (APC), which is flow control for IPv6 soft handover extension to NEMO-Basic Support. In addition to packet bicasting via two different wireless links between the MR and the HA, APC estimates packet arrival time from the slow link and delays incoming packets when they come from the fast link during handover. This flow control can hide the difference of bandwidth and transmission delay between the two links from TCP running on LFNs and their CNs. Our experiments using 3G and WLAN link emulation showed that APC is able to avoid out-of order delivery regardless of packet losses on the two links. This significantly reduced spurious TCP retransmissions, and resulted in better TCP goodput than Make-Before-Break handover under bad radio conditions.



(a) single session (b) two sessions

Fig. 6. The order of TCP sequence number received at the LFN.

As the future work, we will investigate appropriate values on the filter gain constant and the initial value of estimated transmission delay so that APC can achieve good performance under various situations.

Acknowledgment

We are thankful to Xiaolu Deng and Yiyu Huang for their help on measurements. We also appreciate the members of Advanced Telecommunication Lab. of SHARP giving an opportunity to achieve this work at EURECOM Institute.

References

[1] H. Naoe, M. Wetterwald and C. Bonnet, “IPv6 Soft Handover applied to Network Mobility on heterogeneous access networks,” *in proceedings of IEEE PIMRC 2007*, September 2007.

[2] V. Devarapalli, R. Wakikawa, A. Petrescu and P. Thubert, “Network Mobility (NEMO) Basic Support Protocol,” IETF RFC 3963, January 2005.

[3] D. B.Johnson , C. E.Perkins and J. Arkko, “Mobility Support in IPv6,” IETF RFC3775, Jun 2004.

[4] R.Wakikawa, T.Ernst and K.Nagami, “Multiple Care-of Address Registration,” Internet draft, IETF, February 2008; work in progress.

[5] H. Soliman, N. Montavont, N.Fikouras and K. Kuladinithi, “Flow Bindings in Mobile IPv6 and Nemo Basic Support,” Internet draft, IETF, February 2007; work in progress.

[6] H. Pentander, E. Perera, K. Lan and A. Seneviratne, “Measuring and Improving the Performance of Network Mobility Management in IPv6 Networks,” *in IEEE Journal on selected areas in communications*, vol. 24, September 2006, pp. 1671-1681.

[7] R. Chakravorty, P. Vidales, K. Subramanian, I. Pratt and J. Crowcroft, “Performance issues with vertical handovers experiences from GPRS cellular and WLAN hot-spots integration,” *in Proceedings of IEEE Pervasive Communications and Computing Conference*, March 2004.

[8] R. Ludwig and M. Meyer, “The Eifel Detection on Algorithm for TCP,” IETF RFC3522, April 2003.

[9] X. Wu, M. C. Chan and A. L. Ananda, “TCP HandOff: A Practical TCP Enhancement for Heterogeneous Mobile Environments,” *in Proceedings of IEEE ICC 2007*, June 2007.

[10] M. Allman, V. Paxson and W. Stevens, “TCP Congestion Control,” IETF RFC2581, April 1999.

[11] Multiple Care-of Addresses Registration for NEPL (NEMO Platform for Linux), http://www.nautilus6.org

[12] J. D. P. Pavon and S. Choi, “Link Adaptation Strategy for IEEE 802.11 WLAN via Received Signal Strength Measurement,” *in Proceedings of IEEE ICC 2003*, vol. 2, May 2003, pp. 1108-1113.

[13] N. Enderle and X. Lagrange, “Radio Link Control-Acknowledged Mode protocol performance modeling in UMTS,” *in IEEE workshop of Mobile and Wireless Communication Network*, September 2002.