Technical Report

PS-Service Continuity between WLAN and Cellular Networks: Performance of the TCP Dynamics on Handovers

Pedro Sobral, Universidade Fernando Pessoa, Portugal
Luis Bernardo and Paulo Pinto, Universidade Nova de Lisboa, Portugal

ABSTRACT

The integration of different wireless technologies is always a difficult task because they have evolved independently. Each technology addresses issues like user control, cell displacement and numbering, or mobility management in a different way making a uniform approach hard to achieve. One way to minimize these problems in 4G wireless systems is to provide greater importance to the cellular network and have *data vertical handovers* instead of *full vertical handovers* to integrate the other access networks. The control part remains always within the cellular network and other wireless networks are used for the data path. If the architecture integrates the various networks in the core of the cellular network, data handover latencies are very similar to 3G horizontal handovers, minimizing serious consequences. Nevertheless differences on bandwidth and link delays cause problems when the reliable transport TCP is used. This paper studies these problems and analyses the possibilities to overcome them assuming both that harsh conditions exist (there is a high probability of loosing the WLAN signal) and that the TCP or routers must not be changed. It will be seen that for a handover from 3G to WLAN the problem can be easily solved in a scalable way for the proposed architecture. A handover from WLAN to 3G poses problems to an end-to-end solution always assuming the TCP is not modified.

1. INTRODUCTION

There are two major problems in the integration of wireless networks to form 4G wireless systems: 1) the different network technologies and how they manage mobility; and 2) the existence of user control features (paging, routing areas, etc.) that the more advanced networks use (i.e., cellular). They both forest conditions on the definition of an architecture for network integration and on the performance of handovers. This paper extends a previous work on integration of 3G and WLAN systems proposing a *data vertical handover* concept and analysing its consequences on active TCP connections.

Regarding the first problem, WLANs were designed to be a low-cost solution for connecting machines leaving most of the mobility management to higher layers, whereas cellular networks were designed to handle mobility as one of their main concerns having very complete mechanisms with cross-layer interaction.

In terms of user control, the current situation on 3GPP [1] is to define access via WLAN to cellular network services (SMS, MMS, IMS-based, etc.) without much concern yet to interruptions due to the change of access network. A future step will be to make this access continuous (seamless). A complete handover from one network to another has not been seen as relevant yet. In this paper we revisit a solution for extending the 3G access using the WLAN as a bulk *data access* complementary to the UTRAN (*Universal Terrestrial Radio Access Network*). Changes to the 3G core were minimised – they are mainly the introduction of a controlling component responsible for the WLAN operation and a connection from this component to the controlling

component of the 3G operation (SGSN – *Serving GPRS Support Node)*. The idea is not to perform a *full* vertical handover between networks (that moves the data and control parts to the new network) but a *data* vertical handover that keeps the control part in the cellular network avoiding some control problems due to technological differences between the networks (paging, location, numbering of cells, definition of routing areas, etc.). *Data* vertical handovers can be performed within the same order of magnitude of horizontal cellular handovers without any loss of packets.

In this article we study the impact of *data* vertical handovers on the dynamics of TCP. Every handover, either *data* or *full* handover, creates an interruption of service and a transitory interval after it happens, temporarily reducing the TCP throughput. The study in this paper follows our work in [2] designated *core-level* approach of interconnecting the networks. In [2] we addressed the radio parts and their control entities, and we demonstrated that *data* vertical handovers from cellular to WLAN can be achieved in 16.7 ms and from WLAN to cellular in 106.5 ms. In this paper we consider the influence of higher layers and the characteristics of the network links in order to complete the study. Some mechanisms are proposed to improve the TCP loss of performance during the handovers.

The remaining of this article is organized as follows. We make an overview of the system architecture avoiding the description of some of the motivations and background on interconnection of wireless systems which can be found in [2]. We then define some types of handovers and present the characteristics of the TCP's control engine that are most relevant for our case. Section 5 presents the simulation setting and results followed by an analysis that includes a possible solution and a discussion of the related work. The final section contains some concluding remarks.

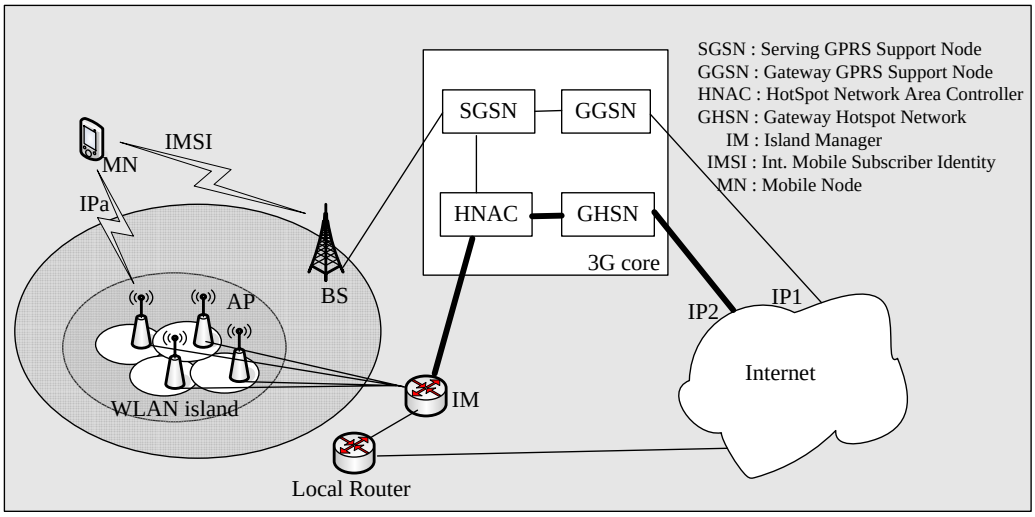
2. INTERCONNECTING NETWORKS AT CORE LEVEL

Fig. 1 shows the *core-level* approach for integrating cellular and WLAN networks proposed in [2]. WLANs form islands controlled by an *Island Manager* (IM). The IM is a standard router that connects to a core node called *Hotspot Network Area Controller* (HNAC). The HNAC controls one or more islands and is, in fact, a core component responsible to include other radio/MAC technologies within the core. The main reason for its existence is the fact that it operates with the user equipment using Internet technology (in the case of WLAN) not disclosing any critical signalling interfaces to the exterior of the core but integrating the terminals really at the core level (and not at IP level). However, as it is inside the core, it has a trustworthy relation towards the other components of the core. It connects to the SGSN, and to the *Gateway Hotspot Network* (GHSN). The GHSN has similar functions to the ones of the *Gateway GPRS Support Node* (GGSN) (and could even be the same physical device). The interface between the HNAC and the IM is a public interface protected by ciphering and the AAA proxy remains inside the core. The data path used by the WLAN circulates through the thick lines, not overloading the existing core. The data path used by the 3G as well as all the control use the standard links.

If other types of alien radio systems other than WLAN are integrated, a specific HNAC should exist but there is no need to have a second GHSN. The IM must exist because it has to be very near to the WLAN to control local issues (e.g. assign local addresses to the mobile node, possibly handle micromobility, etc.). The connection between the HNAC and the SGSN allows the core to use any radio system indistinguishably, as it will be seen.

A mobile node (MN) has different possibilities to use this architecture. It can create a Packet Data Protocol (PDP) context and be given an IP_1 address at SGSN – it accesses

the Internet and services as specified by the 3GPP. It can also create another context using a different Access Point Name (APN) and it is given an IP_2 address at GHSN. This latter context can be created either via UTRAN or WLAN and it survives WLAN disconnected periods of the MN.



■ **Figure 1.** *Architecture for core-level integration (data part)*

Therefore, a mobile node can have two IP addresses: IP_1 and IP_2. IP_1 and IP_2 addresses are active all the time and can use either the 3G or the WLAN due to the connection between HNAC and SGSN. The difference is the semantic of their usage: IP_1 follows the 3GPP standard and would use the UTRAN exclusively. Any use of WLAN is an optimisation and should be decided by the network without user awareness; IP_2 is a user-controlled channel, based on a slightly different protocol stack. In more concrete terms, we propose a signalling mechanism to inform applications on the active radio link to make them decide whether they want to transfer data regardless of the radio connection or wait for a next appearance on a WLAN island.

The decision to have a second address (IP_2) is to make the system legacy compliant. Keeping IP_1 as it is means that current 3G devices and applications can still work using the standardised PDP context. New devices and applications can have a smarter use of the networks at the expense of an extra APN identifier, through IP_2. Note also that both IP_1 and IP_2 have anchor points that never move (IP_1's is GGSN, and IP_2's is GHSN). Therefore, Mobile IP is never used in this architecture.

2.1. PHASES OF A HANDOVER

In general terms, vertical handovers comprise four phases that contribute the most for the total latency:

1. Authorization, authentication and accounting (AAA) procedures;
2. Context creation (e.g. obtain the IP address);
3. Quality of Service (QoS) assurances in the new network; and
4. Fixing/re-establishing data paths (e.g. Mobile IP, micro-mobility, etc.).

In order to reduce the handover time the duration of each of these phases must be minimised or the architecture must allow some of the phases to be performed in advance (outside the time critical path). This latter option is easily done in soft handovers but

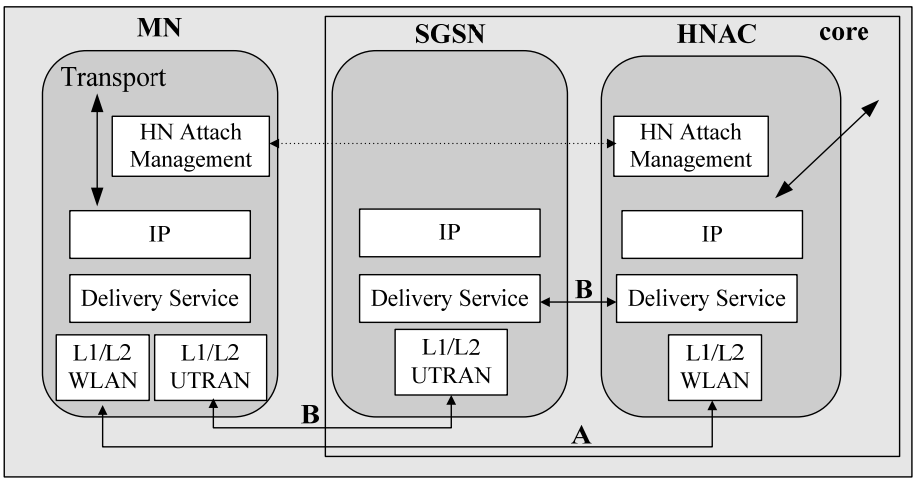
unfortunately hard[1] handovers are more common in the WLAN environment. Our cellular-centric view of the problem helps in this situation because the cellular network is ubiquitous, overlaps the hotspot regions and the decision to change the *data connection* to the WLAN can be done at the proper time. Unfortunately, the data handover from WLAN to the cellular network is not so simple. Most of the times a loss of signal from the access point happens before any procedures can start. [2] covers phases 1 and 2 and the duration is minimal (do not forget that contexts survive dark periods and the AAA procedures via WLAN can be done while still connected to the 3G). This paper covers phase 4.

For phase 4, some proposals of optimisation assuming an equal role of the cellular and WLAN networks are, for instance, bicasting techniques on routers and storage of packets in the core [3], or categorization of movement patterns of mobile nodes [4]. The base technology is Mobile IP. Our approach assumes a greater importance of the 3G network and uses a handover procedure very similar to the 3G horizontal handover.

2.2. HANDOVER PROCEDURE

The procedure for handovers is the following: when the MN senses a WLAN the three first phases above are performed while it is still fully connected to the cellular network [2]. It is given a local IPa address within the IM context and a tunnel is created from the MN until the HNAC (the IPa has a similar role to the core as the 3G terminal identifier IMSI (*International Mobile Subscriber Identity*)). At this moment it can start using the WLAN and the handover is simply a switch to the new path (16.7 ms). While connected to the WLAN, as the SGSN and the HNAC are connected to each other, even the traffic using IP_1 can go through the WLAN instead of the UTRAN.

The protocol stacks of the MN, SGSN, and HNAC are displayed in fig. 2. The figure also shows the entities that are involved when the HNAC communicates with the MN via WLAN (A), or via UTRAN (B). Path B is active all the time and the control packets only use it. However, data packets can use path A or B. Below the IP layer there is a Delivery Service (DS) sublayer that offers an acknowledged service between the HNAC and the mobile node via the WLAN, and a non-acknowledged service via the UTRAN. Applications see a stable IP_2 address, which is routed through IPa or IMSI depending on the DS state. Thus, TCP sessions, for instance, are always maintained.



[1] In soft handovers the mobile node can communicate with more than one Access Point simultaneously, whereas in hard handovers it cannot.

■ **Figure 2.** *Protocol stack for mobile node, SGSN and HNAC (shows hotspot operation)*

When the WLAN path fails, packets are delivered through the SGSN and UTRAN. The DS timeout value is around 40 ms, assuming a delay of 15ms between the HNAC and the MN. After a failure, data packets that were queued in the DS at the HNAC travel to the SGSN for delivery. The timeout period and the time to re-send the packets take around 106.5 ms. Note that 40 ms needs a buffer of 80 K bits if a 2Mbps full speed stream is established (this is not a very demanding condition, and is much less than in [3]).

The anchor address IP_2 is given by the GHSN of the cellular operator responsible for the WLAN where the session began. If the user travels long distances keeping the session on (e.g. crossing countries) routing inefficiencies can arise. In such a case, tunnels between the serving HNACs and the initial HNAC must exist to convey the packets. Note, however, that this should be very rare. In any case, there is never the need to use Mobile IP.

This handover procedure is very similar to the mobility management of the 3G system. In our experiments we compare it with the Mobile IP (MIP) procedure. We assume a latency of 800ms for MIP, which can be considered good [5]. Note also that no packets are lost in our system whereas they are when MIP is used.

Returning to fig. 2, there is the HotSpot Attach Management entity above the IP layer that performs the following main tasks: AAA proxy, procedures for attachment/re-attachment (recovering the context), and reception of signals indicating failure of the WLAN path to warn the applications (considering new applications that would be able to receive these signals).

With such setting any transport layer above the IP works normally either using IP_1 or IP_2. If the mobile is inside an island the WLAN RAN can be used, otherwise packets are delivered via the UTRAN. New network applications can receive signals about the WLAN connection status and enter in a residual state of communication waiting for a future WLAN connection to transfer large quantities of data.

3. TYPES OF HANDOVERS

Two different situations lead to two types of handover between the Hotspot Islands and 3G network:

i) losses of WLAN coverage force the change to the cellular network that is ubiquitous. This situation is called ***forced*** handover. Note that the user never loses contact with the network;

ii) another type of handover, called ***user*** handover, consists on the user explicitly deciding to change to the cellular network when he is still under coverage of the WLAN (this is when a change to the cellular can have the three phases performed outside the critical path).

Additional micromobility handovers can also occur within the Hotspot Islands. A Hotspot Island can be composed by various Access Points (AP), and the MN must handover from one to another remaining inside the island. We can consider two micromobility handover situations:

iii) ***soft*** handovers exist if the terminals can contact more than one AP simultaneously (the current WLAN standard still prevents this);

iv) ***hard*** handovers can take place if a new AP is sensed just after losing the existing one (if it takes too long a *forced* handover will happen).

For all these handovers, packets are never lost (due to the DS sublayer) but suffer different delays depending on the type of handovers.

Concerning service continuity there are three main problems that have different impacts depending on the type of handovers: latency of the handover; loss of packets and change of order in delivering the packets (both data and ACKs).

This paper focuses solely on the consequences of handover between the Hotspot Islands and 3G network on TCP. [6] presents a study of the TCP dynamics for micromobility. We describe in general terms how the TCP handles these problems in the following section and study the cases of *user* and *forced* handovers in our system and Mobile IP handovers in general.

4. HANDOVER INTERFERENCE ON TCP

To analyse the consequences on the control engine of TCP the first observation is the formula for the maximum throughput a stream can have. TCP uses the following equation:

$$B(t) = {W(t)}/{RTT} \quad (1)$$

where *W(t)* is the current minimum window of the receiving window (*rwnd*) and the congestion window (*cwnd*), and the RTT is the round trip time. For very good transmission conditions, when the *cwnd* is no longer a problem, the *rwnd* can prevent the throughput to rise to certain values. Two other problems can still exist: reordering and sudden changes in the RTT. If the new link is faster, overtakes of packets (data and ACK) can happen that trigger recovery mechanisms and reduce the *cwnd*. On the other hand, if the RTT suddenly gets longer, a timeout happens and the congestion window reacts also. This is called spurious timeout. Vertical handovers use links with very different characteristics and different protocol stacks making these types of problems relevant. These problems are not relevant for horizontal handovers (the delays are similar and the protocol stacks are the same).

The currently most used version of TCP (NewReno [7]), reacts to segment acknowledge timeouts by reducing the *cwnd* and the throughput. It contains also a powerful mechanism, called fast recovery, to make it robust to multiple segment loss in a single transmission window. Fast recovery is entered when some (three by default) repeated ACKs are received, and the *cwnd* is adjusted depending on the number of packets still inside the network or on the reception of additional repeated ACKs. While inside the fast recovery, certain conditions can trigger a fast retransmit algorithm to allow a swift transmission of segments and an enlargement of the *cwnd*. TCP SACK is an extension that uses the selective repeat techniques and we will see that it does not help in our case.

A segment acknowledge timeout introduces a *cwnd* reset and a "slow start" growth phase, until t_{SS}, and linearly after that, accordingly to following equation (for a timeout at instant zero and ignoring the speeding effect of packets buffered in the network):

$$cwnd(t) = \begin{cases} 2^{\frac{t}{RTT}} & 0 \le t \le t_{SS} \\ \frac{t - t_{SS}}{RTT} + sstresh & t > t_{SS} \end{cases} \quad (2)$$

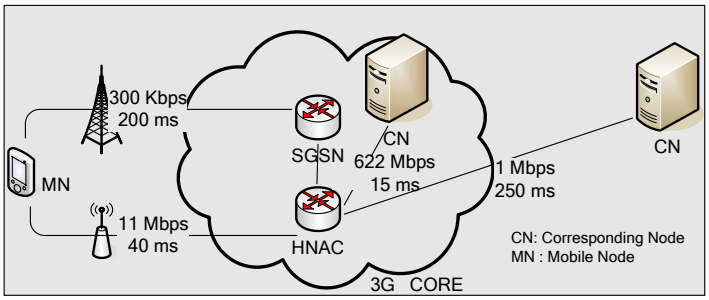
where t_{ss} is the instant when *cwnd* reaches sstresh (slow start threshold value). Eq. (2) reinforces the important role of RTT (already seen in Eq. (1)) for the determination of the throughput of the flow.

5. SIMULATION FRAMEWORK

Our simulations intend to study the performance of the TCP dynamics (the NewReno version with and without the SACK option) when delays and reordering happen in

consequence of handovers. We used the Network Simulator (ns-2) with the topology shown in figure 3. The number of intermediate routers between the corresponding nodes and the core were replaced by the delay time shown near the lines. The bandwidth of each link is also displayed.

From the core to the mobile node, taking the results from [2], and from [8], we decided to be conservative and used a 200 ms delay for the UTRAN and 40 ms for the WLAN. The Delivery Service timeout value was set to 90 ms (longer than in [2]). We also tested two scenarios for the placement of the applications. They can run on application servers on the Internet ("Internet" case) or at application farms in the core, or close by ("Core" case). For each of the two scenarios we tested three cases: a *user* handover; a *forced* handover; and the usage of Mobile IP.



■ **Figure 3.** *Simulated testbed*

In terms of traffic and protocols we simulated a unidirectional flow from the server to the MN with a 500 Kbps CBR traffic using TCP, corresponding to a call that needs more resources than the 3G can provide – i.e., a saturated TCP sender (it is also the most interesting case because if there is no traffic, there are no packets, and therefore no problems). The TCP traffic used packets with 1,000 bytes of data. The source generated a packet each 2 ms. *rwnd* in the ns-2 was changed from 20 to 40 not to restrict the throughput (see Eq. (1)).

There is always a single call. We can assume that a differentiating mechanism in the 3G to protect the resources is used. In all experiments the user started in the 3G network, moved to WLAN after 100 sec., and moved again to the 3G at second 300, staying there until the end. For our types of handovers the attachment to the WLAN, as well as the AAA procedures, was performed outside the critical path (see [2]). So, it is simply a switch of the RANs. Regarding the move to 3G, *user* handovers are also simple switches of paths, whereas *forced* handovers are performed after the timeout of the Delivery Service expires.

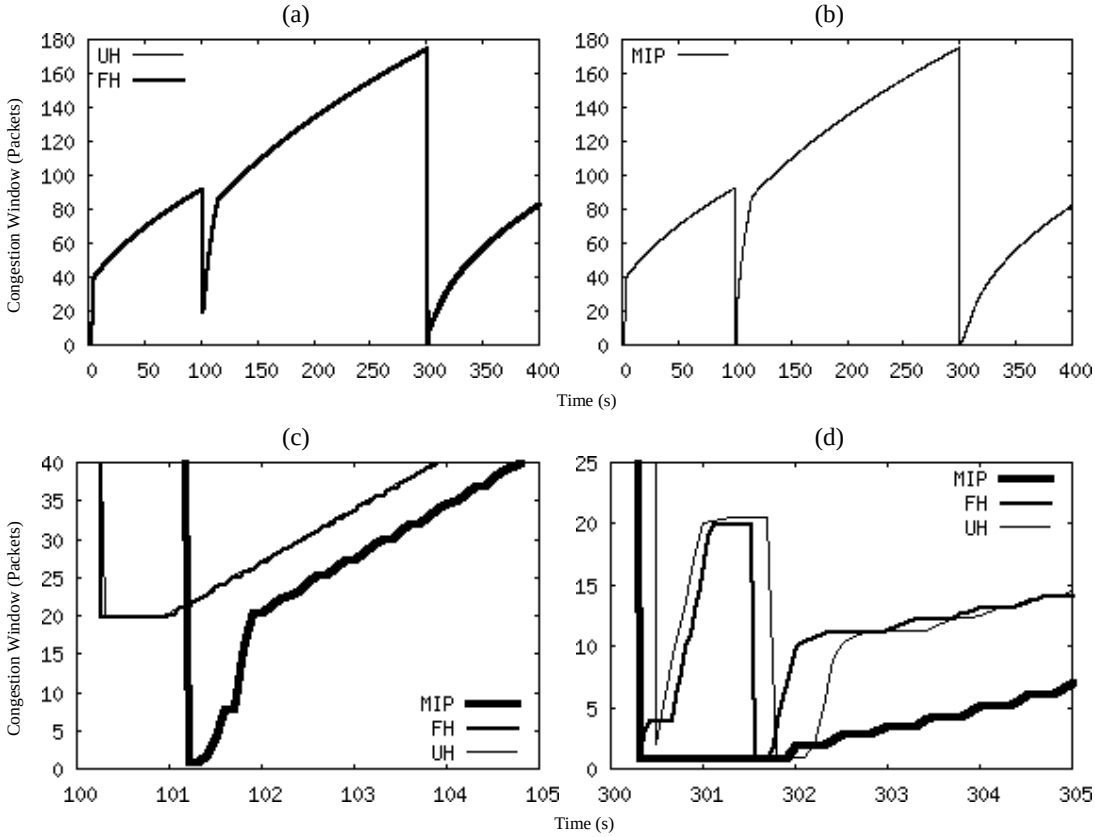
5.1. SIMULATION RESULTS

The most interesting case is the "Core" case because the differences on the RTT are more important in relation to its current value. In the "Internet" case the differences on RTT using our architecture do not cause disturbances (the greater values of RTT and its variation can absorb the difference on links and protocols). For MIP the situation is worse, due to the larger unavailability period. Due to the less relevance of the "Internet" case no results are shown.

Figure 4 shows the congestion window behaviour using TCP NewReno for user and forced handovers (a) and MIP (b). Details are shown in (c) and (d). At the entrance to the WLAN (100s) the new data packets travel faster through the WLAN, arrive before the ones just sent via UTRAN, and force the receiving TCP to send various repeated

ACK packets. On reception of these repeated ACK packets, the TCP sender reacts, enters in fast recovery mode, and decreases the *cwnd*. For this particular kind of problem there are no differences between the use of NewReno or SACK. There are no differences either between the *user* and *forced* handover because the phases are performed outside of the critical path (as can be seen in figures 4 (a) and 4 (c)). The use of MIP is slightly worse because packet losses are recovered by timeout and the *cwnd* drops to 1 (figures 4 (b) and 4 (c)).

At the re-entrance into the 3G (300s) a spurious timeout happens. The sending TCP is transmitting segments to the mobile node through the HNAC. The change happens and the sender stops receiving ACKs, since they are now travelling back through the UTRAN link. Refer to fig. 4 (d) for the behaviour of the *cwnd* and fig. 5 for the segment sequence number evolution. At second 300.28 the timer for the segment 18716 expires. As a consequence, the *cwnd* is reduced to 1 and the *SSthresh* is halved to 20 (it was set to 40 at the beginning of the simulation). Due to this timeout the sender starts to retransmit all the segments from 18716 (however segment 18736 is already in flight), doubles the timeout value but does not calculate the RTT estimation. At second 300.33 the ACKs travelling through the 3G connection start arriving at the sender and the congestion window gets inflated accordingly.

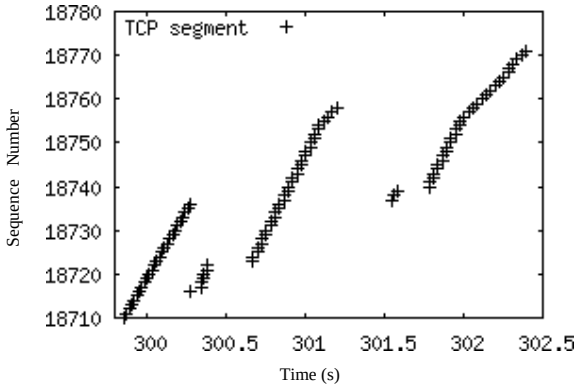


■ **Figure 4.** Behaviour of the congestion window, cwnd, for: (a) *user (UH)* and *forced (FH)* handovers; (b) Mobile IP (b). Details at 100seg (c) and 300s (d).

When the retransmitted packets arrive at the receiver, they are considered out of order provoking the sending of repeated ACKs. At second 301.1, when the sender is transmitting segment 18758, the repeated ACKs (with the value of 18736) start arriving forcing the sender to enter in fast recovery. As all the ACKs carry the same value, they are not considered partial acknowledgements. Therefore the retransmission timeout is

not reset (Slow-but-Steady variant of NewReno) and the window is maintained. The sender stops sending data packets due to lack of window and the system halts waiting for partial ACKs. As the value of the timeout is not large enough yet, a second timeout happens around second 301.54 on segment 18737. At this moment the sender transmitted already segments up to 18758 but due to the timeout it exits the fast recovery mechanism and resends TCP segments from 18737. The *cwnd* goes back to 1 and *SSthresh* goes to 10. The timeout value is doubled again, and again no adjustment of the RTT estimation is performed.

The following events are the arrival of the ACKs of the data packets till 18758 having the effect of raising the *cwnd*. When the sender is at 18768, at the time 302.32, the repeated ACKs (with a value of 18758) sent back in consequence of the retransmitted packets after the second timeout start arriving driving the sender to fast recovery again. Transmission continues and, before the timeout expires again, partial ACKs (from 18759 to 18761) make it reset and open slightly the window. Eventually the ACK from 18768 arrives forcing the sender out of the fast recovery and stabilizing the system. By this time the RTT has grown from 0,14s to 0,98s and will stabilize in 1.11 s for the UTRAN link. Note that although there were no losses the sending TCP entered twice in the fast recovery algorithm.



■ **Figure 5.** Sequence number of TCP segments for *forced handover (FH)*.

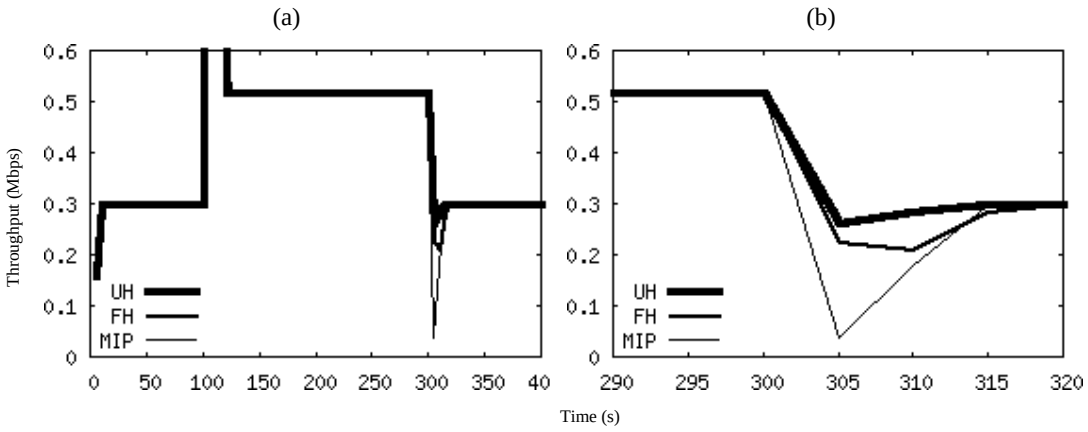
There are no differences between NewReno and SACK. As there are no missing packets the values carried in the ACK packets are the same. The difference between user and forced handover can be seen in fig 4 (d). The 90 ms timeout delays a little bit more the first ACK packets travelling via UTRAN and the *cwnd* does not rise so quickly as for the user handover case. What is very striking is the almost lack of differences to MIP which has losses and 800ms of delay (MIP induces a shrink on *cwnd* but as there is not the second effect it can recover quite well – see fig 4 (a) and (b)).

Looking at the TCP throughput during the simulation, fig 6 (a), a spike is seen after 100s due to the handover to the WLAN. Since the CBR data rate (500kbps) is greater than the available bandwidth on the 3G connection (300kbps) the buffers on the nodes along the path are filled with TCP segments before the handover. When the WLAN connection (11Mbps) becomes available, a burst of buffered data arrives at the mobile node. As can be seen in figure 6 (b), at second 300, in the case of the user and forced handover the throughput degradation is lower than for the MIP case. For all cases, 15s after the handover to 3G connection, the throughput reaches 300kbps again.

The following section analyses these results and the state-of-the-art in the area to propose solutions to these problems.

5.2. ANALYSIS OF THE SIMULATION RESULTS AND RELATED WORK

The first main observation is that if no remedies are applied a very fast handover without losses behaves almost exactly as an 800 ms handover with packet losses, such is the reaction of the TCP control algorithm to reordering and differences on the RTT. The second main observation is the elapsed time of a handover. Ideally it should be very fast (around half a second or less), and any solution that considers greater amounts of time to solve any problems is not appropriate and it might be better to have the TCP control engine handle it, anyhow.



■ **Figure 6.** (a) Throughput of the TCP flow for *user (UH), forced (FH)* and *Mobile IP (MIP)* handovers. (b) Detail around 300s.

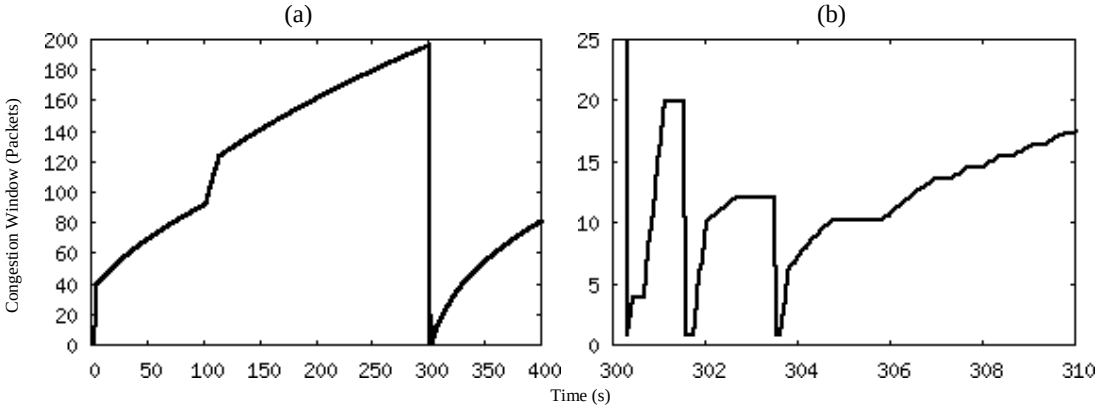
The case of the handover to WLAN is a typical case of moving to a link with less delay (and larger bandwidth) which introduces the problem of reordering. This problem has been deeply studied. [9] makes use of a reordering router to change the order of the segments in the queue to study the problem. They work at the level of modifying the TCP control algorithm (mainly trying to restore its congestion control state prior to the reordering). This is more appropriate for spurious retransmissions that happen for handovers to the 3G than for this case (see below). Our approach is to try to find solutions that impose no modifications to the existing TCP implementations. [10] studies a somehow peculiar scenario with three successive handovers to faster links without considering the corresponding return handovers. They used more aggressive (smaller) values for the delays of the links than we did. They studied the behaviour of various TCP versions and showed that if the *cwnd* has not reached a high enough value (which can mean a short stay in a particular RAN) the reordering problem can force two entrances in the fast recovery algorithm delaying even more the recovery. In fact, the condition can be more generalised: *cwnd* must be over *rwnd* (= 40 in our case) not to introduce extra problems. They provide some hints on possible solutions. A typical solution is to control the number of repeated ACKs sent to avoid triggering the entrance into the fast recovery algorithm. The TCP itself could do this. However, it implies that the TCP would have to know the network interface used by the packets, or be signalled of it. But it is a change to TCP and it is out of our options. The solution resides then in having an entity at a lower layer controlling the passage of duplicate ACKs and deleting repeated ones. Having a router, or gateway, performing this task is not feasible due to the overhead of maintaining information about TCP flows and to scalability problems. [11] and [12] propose such a solution for a slightly different problem than handovers – rate control adjustment to variable rate/delay conditions on links. With our architecture, the Delivery Service at the Mobile Node is a natural entity that can perform the job because there are no problems of scalability and it knows exactly when it should start monitoring TCP flows. Once it notices that there has been a handover to WLAN (either directly or via signalling from its counterpart at the HNAC) it starts building information about TCP flows and record ACK packets. Every repeated ACK packet is destroyed. This acting mode lasts for a certain time after which the Delivery Service returns to neutral mode. Figure 7 (a) shows the result of this intervention. As there are no repeated ACKs arriving at the sender, the TCP does not enter in fast recovery and the evolution of the *cwnd* adapts quicker to the better network conditions (100s).

The case of the handover to the 3G is a typical case of moving to a link with more delay (and smaller bandwidth), which introduces the problem of spurious timeouts. The problem can be avoided if the TCP protocol is modified. The Eifel algorithm [13] uses the TCP timestamp extension to detect spurious timeouts, allowing TCP to restore the initial congestion window state and update RTT as soon as the first ACK arrives on the sender side. Unfortunately, TCP Eifel is rarely used. Another solution is to work on the ACKs, delaying them [14], [11] and [12]. Since all TCP traffic passes through the Delivery Service (in the HNAC or SGSN), this sub layer could be used to avoid spurious timeouts as well. It could keep track of TCP flows and delay ACK forwarding to reduce the timeout probability. The DS would intercept ACK packets and forward them according to the desired source rate. The sender would keep receiving ACKs from the DS during the handover, postponing the timeout. However, the sudden increase on RTT is difficult to match with this approach because of the high number of ACK packets needed to increase significantly the sender's estimation of RTT ([14] proposes a

number as high as 62). It is a typical case of the remedy lasting longer than the disease. This solution also does not scale due to the connection tracking requirement at the intermediary router/gateway. Notice also that solutions proposed for soft handover cannot be used [14], because the lost of WLAN connections is almost unpredictable and very sudden, not giving time to pass the tenths of ACKs required to adapt the RTT estimation, and hence the timeout value.

Perhaps the best thing one could do is to work, once again, on the problem of repeated ACKs trying to minimize the consequences. Figure 7 (b) shows a detail at second 300 of figure 7 (a). The lack of these repeated ACKs prevented the sender to enter in fast recovery, not resetting the timeout on each partial ACK. The simple duplication of the timeout value is not enough and it expired for the third time producing an overall slower reaction. No simple solutions exist for this case. However, it is worth to note that the system is changing to a network with poorer performance, so a throughput reduction has to happen anyway. The problem is not so serious.

One solution that might reduce the disturbances for elastic traffic could be the reduction of the flow during handoffs. As the HNAC can behave like a router it could issue an Explicit Congestion Notification (ECN) [15] to warn the source to slow down and reduce the impact of the recovery. Unfortunately this will not work either. By the



■ **Figure 7.** (a) Behaviour of the congestion window when the Delivery Service has an active mode on controlling ACK packets. (b) Detail at 300seg.

time the ECN reaches the receiver and it is sent to the source it is already too late to react. The unique alternative would be to use it only in *user* handoff because it is predictable, but the experiments showed that it is not necessary.

6. CONCLUSIONS AND FURTHER WORK

In this paper we analysed the consequences for the TCP control engine of handovers using an interworking architecture that does not loose packets and it is the fastest due to the *core-level* approach. We showed that the reordering problems can be easily solved but the spurious timeouts are harder to fix. These problems can make such a damage that even in our system without modifications to the TCP or DS the overall performance is not much better than using the Mobile IP (with losses).

Although our solutions are easily feasible, and the interworking architecture could be implemented as a follow-up of the current state-of-the-art on 3GPP, the reordering problem is a much general problem if traffic engineering starts to be used extensively for the Internet.

As a topic for further research we are planning a new application architecture that can adjust their operation to the mobile node connection status. These applications must decide if it is worth to move to another network, based probably on past experience of

the user. This paper also showed that the placement of these applications (either in the core or outside) has different consequences to the overall performance.

REFERENCES

[1] 3GPP TR 23.234 v6.1.0, "*3GPP system to WLAN interworking; System description (Release 6)*", June 2004

[2] P. Pinto, L. Bernardo, and P. Sobral, "*Seamless continuity of PS-services in WLAN/3G interworking,"* Computer Communications, Elsevier, Vol 29/8 pp 1055-1064, May 2006.

[3] N. Shenoy, R. Montalvo, "*A Framework for Seamless Roaming Across Cellular and Wireless Local Area Networks*," IEEE Wireless Commun. June. 2005, pp. 50-57

[4] R. Hsieh and A. Seneviratne, "*A Comparision of Mechanisms for Improving Mobile IP Handoff Latency for End-to-End TCP*," MobiCom'03, Sept. 2003, San Diego.

[5] N. Montavont and T. Noël, "*Analysis and Evaluation of Mobile IPv6 Handovers over Wireless LAN*," Mobile Networks and Applications 8, 2003, pp. 643-653

[6] L. Cerdà, F. Vena, and O. Casals, "*Study of the TCP Dynamics over Wireless Networks with Micromobility Support Using the ns Simulator*," Wireless Networks, Kluwer (10) 2004, pp. 17-27

[7] S. Floyd, T. Henderson and A. Gurtov, "*The NewReno Modification to TCP's Fast Recovery Algorithm*," IETF RFC 3782, April 2004

[8] M. Kohlwes, J. Riihijärvi, and P. Mähönen, "*Measurements of TCP Performance over UMTS Networks in Near-Ideal Conditions*," Proc. of IEEE Vehicular Technology Conference (VTC 2005-Spring), Stockholm, Vol. 4, pp.2235-2239, June 2005.

[9] E. Blanton, and M. Allman, *"On Making TCP More Robust to Packet Reordering"*, ACM Comput. Commun. Rev., vol. 32, no. 1, pp. 20-30, Jan. 2002.

[10] W. Hansmann, M. Frank, and M. Wolf, *"Performance Analysis of TCP Handover in a Wireless/Mobile Multi-Radio Environment"*, 27th Annual IEEE Conf. on Local Computer Networks (LCN'02), Washington, USA. pp. 585-594, Nov. 2002.

[11] P. Narváez, and K.-Y. Siu, *"New Techniques for Regulating TCP Flow over Heterogeneous Networks"*, 23rd Annual IEEE Conf. on Local Computer Networks (LCN'98), Washington, USA. pp. 42-51, Oct. 1998.

[12] M.C. Chan, and R. Ramjee, *"TCP/IP Performance over 3G Wireless Links with Rate and Delay Variation"*, ACM MOBICOM'02, Atlanta, USA. pp. 71-82, Sep. 2002.

[13] R. Ludwig, and R. H. Katz, *"The Eifel Algorithm: Making TCP Robust Against Spurious Retransmissions"*, ACM Comput. Commun. Rev., vol. 30, no. 1, pp. 30-36, Jan. 2000.

[14] H. Huang, and J. Cai, *"Adding Network-Layer Intelligence to Mobile Receivers for Solving Spurious TCP Timeout During Vertical Handoff"*, IEEE Network, vol. 20, no.6, pp. 24-31, Nov.-Dec. 2006.

[15] K. Ramakrishnan, S. Floyd, and D. Black, "*The Addition of Explicit Congestion Notification (ECN) to IP*," IETF RFC 3168, Sep 2001