MIPv6 Experimental Evaluation using Overlay Networks

Pablo Vidales [a] Carlos Jesús Bernardos [b] Ignacio Soto [b]
David Cottingham [c] Javier Baliosian [d] Jon Crowcroft [c]

[a] *Strategic Research, Deutsche Telekom Laboratories*
Ernst-Reuter-Platz 7, D - 10587 Berlin (Germany)

[b] *Universidad Carlos III de Madrid*
Avda. Universidad 30, 28911 Leganés (Spain)

[c] *Computer Laboratory, University of Cambridge*
William Gates Building, 15 JJ Thomson Avenue, Cambridge CB3 0FD (UK)

[d] *Network Management Research Centre, Ericsson R&D Ireland*
Ericsson Software Campus, Athlone, Co Westmeath, Ireland.

Abstract

The commercial deployment of Mobile IPv6 has been hastened by the concepts of ***Integrated Wireless Networks*** and ***Overlay Networks***, which are present in the notion of the forthcoming generation of wireless communications. Individual wireless access networks show limitations that can be overcome through the integration of different technologies into a single unified platform (i.e., 4G systems). This paper summarises practical experiments performed to evaluate the impact of inter-networking (i.e. vertical handovers) on the Network and Transport layers. Based on our observations, we propose and evaluate a number of inter-technology handover optimisation techniques, e.g., Router Advertisements frequency values, Binding Update simulcasting, Router Advertisement caching, and Soft Handovers. The paper concludes with the description of a policy-based mobility support middleware (PROTON) that hides 4G networking complexities from mobile users, provides informed handover-related decisions, and enables the application of different vertical handover methods and optimisations according to context.

Key words: Mobile IPv6, 4G, Overlay Networks, Handover Optimisations.

Email addresses: `Pablo.Vidales@telekom.de` (Pablo Vidales), `cjbc@it.uc3m.es` (Carlos Jesús Bernardos), `isoto@it.uc3m.es` (Ignacio Soto), `David.Cottingham@cl.cam.ac.uk` (David Cottingham), `javier.baliosian@ericsson.com` (Javier Baliosian), `Jon.Crowcroft@cl.cam.ac.uk` (Jon Crowcroft).

1 Introduction

The Internet Protocol (IP) was developed in the early 1980s with the aim of supporting connectivity within research networks, as part of *catenet* [1]. However, in the last decade IP has become the leading network-layer protocol. It is the basic tool for a plethora of client/server and peer-to-peer networks; it predominates in both wired and wireless worlds, and now the current scale of deployment is straining many aspects of its twenty five-year old design. To overcome the limitations inherent in the original IP design, IPv6 has been proposed as the new protocol that will provide a firmer base for the continued growth of today's inter-networks.

The Internet has experienced an enormous growth in recent years and the number of users accessing services on-the-move has also grown exponentially. Every mobile device is potentially capable of accessing IP services, Wi-Fi networks are becoming ubiquitous; the growth in the hotspot market is being accompanied by similar growth in other wireless technologies (e.g., Bluetooth, UltraWideband, and satellite), posing the urgent need for a larger address space and an adequate support for mobility.

Whereas the main thrust of IPv6 is to increase the addressing space, it also provides important functions to enable mobility (e.g., scaling and ease-of-configuration), mainly derived from the larger address space. IPv4 has difficulties managing mobile terminals for several reasons such as address configuration and location management.

To drive the evolution in the mobile world, Mobile IPv6 (MIPv6) [2] was proposed as a protocol that exploits the added features in IPv6 to extend it and enable micro and macro-mobility. With the introduction of IPv6, many of the disadvantages of the previous version of the mobility support (i.e. Mobile IPv4 [3]) were eliminated. However, neither Mobile IPv4 (MIPv4) nor Mobile IPv6 were intended to support seamless roaming in heterogeneous environments.

The rapidly growing demand for "anywhere, anytime" high-speed access to IP-based services is becoming one of the major challenges for mobile networks. As the demand for mobility increases, mobile terminals need to roam freely across heterogeneous networks, posing the challenge of network integration into an All-IP ubiquitous access platform [4]. Mobile IPv6 stands as the *de facto* solution for mobility management in next-generation systems. Highlights of networking evolution are shown in Figure 1; we consider IP and Mobile IP as primary players in the unfolding of networking in past, present, and future communication systems.

Our work targets networking issues in future 4G communication systems. In particular, we are interested in the impact of vertical handovers and terminal mobility on the performance of the TCP/IP stack. Our work aims to:

(1) **Build** a convenient environment to evaluate MIPv6, emulating the networking environment of 4G systems,
(2) **Measure** the latency when the terminal is roaming between heterogeneous networks,
(3) **Characterise** the vertical handover latency, identifying its main components,
(4) **Identify** the main performance problems during vertical handovers and their impact on the TCP/IP stack,
(5) **Explore** different optimisation methods that decrease the overall latency, and evaluate each of them using the University of Cambridge Computer Laboratory testbed,
(6) **Demonstrate** the efficiency of different optimisations, and the need for a high-level mobility support middleware to enable informed decisions and drive the handover process.

In this paper we present a detailed evaluation of Mobile IPv6, due to its role as the standard for mobility management in next-generation networks. A brief introduction to the basic concepts of Mobile IPv6, and the most important improvements to this protocol are mentioned in Section 2. In Section 3, we describe the tightly-integrated MIPv6-based testbed used during the experiments. Then we describe the testing environment, explain test conditions and scenarios, in Section 4. The vertical handover latency study is summarised in Section 5 and latency partition detailed in Section 6. We explore different handover optimisations, with the results and evaluations presented in Section 7. A high-level policy-based middleware, *PROTON*, is described in Section 8, as a plausible solution to handle context related complexities and enable informed handover decisions. Related work is presented in Section 9, while Section 10 concludes this paper.

2 MIPv6-based Management

In the last decade, some protocols for the dynamic assignment of IP addresses to nodes joining a network segment (e.g., DHCP [5]) have been designed and deployed, but these solutions provide portability and not transparent mobility. By portability we mean the terminal mobility support that allows a host to change its location, but which requires stopping and restarting its upper layer connections (e.g., TCP). In contrast, transparent mobility allows a terminal to move between different networks without dropping any connection. IPv6 [6] provides portability because of its mechanisms for automatic address configuration, but not transparent mobility. Mobile IPv4 [3] and Mobile IPv6 (MIPv6) [2] are the protocols defined to provide support for reachability and transparent mobility in future networks.

The latency due to a handover using basic MIPv6 is proportional to the round-trip time necessary for a Binding Update message (BU) to reach either the Mobile Node's Home Agent (HA) or a Correspondent Node. This is further analysed in

Section 6.

Latency on MIPv6-enabled links can be very high, especially for interactive applications that have real-time requirements. Therefore, the research community is working on mechanisms that decrease this latency as much as possible, at least to levels that support real-time applications in a seamless manner. Two of the most significant proposals are Fast Handovers for Mobile IPv6 (FMIPv6) [7] and Hierarchical Mobile IPv6 (HMIPv6) [8].

FMIPv6 [7] aims to decrease the total latency to almost only the Layer 2 handover time. This approach has been shown to perform well in intra-technology (i.e. horizontal) handovers [9], [10].

The HMIPv6 approach is designed to reduce the degree of signalling required and to improve handover speed for mobile connections by managing local mobility in a more efficient way. Previous work [11], [12] has analysed which of these approaches (i.e. FMIPv6 and HMIPv6) performs better, the conclusion being that a combined approach would be optimal. However, given the implementation complexity that this would require, the FMIPv6 optimisation by itself is good enough (this has been experimentally evaluated in [10]).

Future 4G systems, in which heterogeneity will be the rule instead of the exception, present some challenging characteristics when performing vertical (also known as inter-technology) handovers:

- Bandwidth, delay, and packet losses can be very disparate among the different candidate access network technologies (e.g., Ethernet, IEEE 802.11, GPRS, UMTS, Bluetooth, etc.)
- There are certain technologies that are usually globally available (e.g., GPRS, UMTS), whereas there are others that are only present in certain locations (e.g., WLAN, Bluetooth). This fact dictates an overlay-based model, where "moving up" (handing off from a locally available access technology to a globally available network) is not equivalent to "moving down" (roaming from a globally available network to a locally available technology). This needs to be taken into account while in horizontal handovers the situation does not arise.

Therefore, the vertical handover process is affected by many different parameters, and the solutions needed to improve it must handle this complexity. Moreover, optimisation methods that have been tested for homogeneous environments do not necessary work well in vertical handovers, and we should not assume that everything in the horizontal scenario brings benefits to the heterogeneous case.

This paper aims to clarify this situation; we thoroughly evaluate MIPv6 and catalogue the main effects of mobility on the TCP/IP stack. During this process, we look at possible handover scenarios and optimisations with the intention of defining the best combination and its scope. Finally, we suggest particular optimisation

methods, which can be applied under a certain set of conditions. Subsequently, we remark on the need for a middleware to support mobility and handle these intrinsic complexities in 4G networks.

3 Computer Laboratory MIPv6-based Testbed Architecture

To emulate a next generation (4G) integrated networking environment, our experimental testbed setup consists of a loosely-coupled, Mobile IPv6-based GPRS-WLAN-LAN testbed as shown in Figure 2(a). The cellular GPRS network infrastructure currently in use is Vodafone UK's production GPRS network. The WLAN access points (APs) are IEEE 802.11b APs. Our testbed has been operational since March 2003, its GPRS infrastructure comprises base stations (BSs) that are linked to the SGSN (Serving GPRS Support Node) which is then connected to a GGSN (Gateway GPRS Support node). In the current Vodafone configuration, both the SGSN and the GGSN are co-located in a single CGSN (Combined GPRS Support Node). A well provisioned virtual private network (VPN) connects the Lab network to that of the Vodafone's backbone via an IPSec tunnel over the public Internet. A separate "operator-type" RADIUS server is provisioned to authenticate GPRS mobile users/terminals and also assign IP addresses.

For access to the 4G integrated network, mobile nodes (e.g., laptops) connect to the local WLAN network and also simultaneously to GPRS via a Phone/PCCard modem. The Mobile Node's MIPv6 implementation is based on that developed by the MediaPoli project [13], chosen for its completeness and open source nature. We brokered a semi-permanent IPv6 subnet from BTExact's IPv6 Network, which connects us to the 6BONE. Using this address space, we are able to allocate static IPv6 addresses to all our IPv6 enabled mobile nodes. A router in the lab acts an IPv6/IPv4 tunnel end-point to BTExact's IPv6 network. This router is also an IPv6 Access Router for the lab's fixed-internal IPv6-enabled network and for internal WLANs. Routing in the Lab has been configured such that all GPRS/WLAN user traffic going to and from mobile clients is allowed to pass through the internal router, enabling us to perform traffic monitoring.

Since the GPRS cellular network currently operates only on IPv4, we use a SIT (Simple Internet Translation) to tunnel all IPv6 traffic as IPv4 packets between the Mobile Node and a machine providing IPv6-enabled Access Router functionality on behalf of the GPRS network. Ideally, the GGSN in the GPRS network would provide this functionality directly, but using the tunnel incurs only minor overhead, and represents the current status of IPv4-to-IPv6 migration.

The testbed integrates several independent IP-based networks, including three IEEE 802.11b sub-networks, Vodafone's GSM/GPRS network, and the Lab's local area network. This setup allows us to do experimental analysis of intra-network and

inter-network handovers.

4 Experimental Environment

The testbed was operated under the following conditions:

(1) The movement detection was performed based on Neighbour Discovery (i.e. router advertisements). In this case, the Mobile Node waits to receive a Router Advertisement (RA) after it arrives at the visited network.
(2) In the case of the vertical handover, when the new interface has a lower priority, compared to the previous one, the Mobile Node waits until the old access router is unreachable and then generates a Router Solicitation (RS) message.
(3) Unless stated otherwise, all access routers are set to multicast RAs, according to the recommended parameters and values specified in [14] (the effects of having these values are analysed in Section 7.1)
(4) For all the cases considered in these tests, the multimode mobile device had all of its interfaces (e.g., LAN, WLAN and GPRS) powered on and listening to their specific networks.
(5) The Soft Handover and RA caching optimisations follow the "make-before-break" philosophy, meaning that at least one handover-related process starts before breaking the connection with the previous Access Router.
(6) The RA frequency and BU simulcasting methods follow the "break-before-make" philosophy. The Mobile Node (MN) waits until the current Access Router is unreachable, and then it begins the handover process.
(7) The results presented in Section 7.4 (i.e. soft handovers) consider the MN listening on both interfaces simultaneously, while performing the vertical handover. In contrast, a *hard handover* occurs when the MN listens exclusively on one interface – expecting packet losses.
(8) All hosts run Red Hat 7.1, with Linux 2.4.16 and the MNs run MIPL 0.9.1. Although this release of MIPL does not fully comply with current Mobile IPv6 standard (RFC 3775 [2]), we argue that results not involving Route Optimisation (that is, MN operating in Bidirectional Tunnel mode) would be the same than with a more up-to-date Mobile IPv6 implementation. On the other hand, if Route Optimisation mode was used, then the results we present in this paper can be seen as a lower limit on the handover latency, since the time necessary for the Return Routability procedure to take place is not taken into account. A good approximation to the results that would be obtained when using Route Optimisation could be obtained by adding up 1.5 RTTs to the experimentally obtained handover time.

    For these experiments we used the workstation and the laptop with an Orinoco wireless silver card and a Sierra Wireless Aircard 750 for IEEE 802.11b and GPRS connections, respectively (see Figure 2(b)).

For the tests conducted, handovers were forced by filtering incoming traffic at the Network Layer using IP6tables-based filters. This allowed us to simulate that the current Access Router the MN was connected to was no longer reachable (for example, because of mobility reasons, the signal level perceived by the MN became very low), triggering in that way a handover. This mechanism enables handovers to be automatically performed without requiring the MN to physically move.

For testing handovers, data transfers were initiated by the multimode device over the Visited Network Access Router. During this data transfer (e.g., ICMP packets or HTTP), we conducted the vertical handover to a different Visited Network, i.e. a different Radio Access Technology (RAT). In the testbed, we cause all traffic to pass through an intermediate router, and monitor the traffic (using tcpdump [1]) to/from the Correspondent Node, allowing us to collect traces at this point for further analysis.

5 Experiments to evaluate MIPv6

To proceed with the performance analysis in heterogeneous environments, we first conducted experiments to evaluate MIPv6 during vertical handovers. These tests aim to measure the impact of mobility on the TCP/IP stack, therefore we completed trials to analyse effects on Layer 3 and Layer 4 for the following scenarios: GPRS-to-WLAN, WLAN-to-GPRS, GPRS-to-LAN, LAN-to-GPRS, WLAN-to-LAN, and LAN-to-WLAN. We explain the most relevant results from the study in this section:

- Measurements of MIPv6 performance during vertical handovers at the Network Layer for every possible scenario, using ICMP as a test case protocol.
- The most popular test scenario in heterogeneous environments is the handover between WLAN and GPRS networks, for which we have included results showing the UDP over MIPv6 handover latency.
- MIPv6 Transport Layer measurements to determine the effects of mobility on TCP, using HTTP as a test case application.
- Packet overhead added by different protocols, considering routing between the MN and the Correspondent Node (CN) and between mobile nodes, in the different test cases.

[1] `http://www.tcpdump.org/`

5.1 Mobile IPv6 Network Layer performance (IP)

To calculate the latency at the Network Layer based on packet loss, we generate ICMPv6 traffic between the Correspondent Node and the Mobile Node. The Correspondent Node sends small ICMPv6 packets (104 bytes) every 200ms [2]. Thus, the handover latency is the product of the number of packets lost multiplied by the time interval between the packets (see Figure 3(a)).

Results shown in Figure 3(a) suggest that Mobile IPv6 protocol was designed for mobility management within the same technology. When dealing with heterogeneous handovers, MIPv6 does not perform as expected and in most scenarios latency exceeds acceptable limits (not even close to support for real-time applications). We should mention that the values presented can have a precision error of 200ms due to the experimental setup, thus handover latency from IEEE 802.11b to GPRS is between 2200ms and 2400ms. In fact, the mean handover latency for this scenario is 2325ms.

5.2 Mobile IPv6 Transport Layer performance (UDP)

In addition to study the MIPv6 performance at the Network Layer it is also interesting to investigate how current transport protocols are affected by mobility in heterogeneous environments. This section discusses related results collected during some experiments that were performed using the CL testbed.

Several tests were done using MGEN [3] in order to analyse UDP effects on MIPv6. During the experiments, the receiver (i.e. Mobile Node) collects transmitted packets and creates logs, which are then analysed to extract statistical data (using TRPR [4]) such as perceived latency at the Mobile Node.

Figure 4 shows the latency at the Mobile Node, as well as the disparity in the access RTT for WLAN and (cellular) GPRS technologies. In this scenario, the Correspondent Node starts sending UDP packets (packet size 80 bytes, every 50ms [5]). After 22s we can observe that a handover occurred, and 57 packets are lost – this implies a handover latency of 2850ms, which is a value very close to the Network Layer latency between the same pair of technologies (2325ms as shown in Figure 3(a). This supports the hypothesis that the difference between network and transport la-

[2] Since the obtained handover latencies are significantly higher than 200ms, there is not requirement to send data faster in order to obtain good results.
[3] `http://mgen.pf.itd.nrl.navy.mil/`
[4] `http://proteantools.pf.itd.nrl.navy.mil/trpr.html`
[5] These values were chosen so the packet size is similar to that used by VoIP applications, and in order that the packet frequency provides us with a high enough accuracy.

tencies is the *adaptation component*, which is related to the TCP congestion control mechanism and will be explained in the next subsection.

5.3 Mobile IPv6 impact on the Transport Layer (TCP)

We evaluated the impact of vertical handovers on the Transport Layer by collecting traces whilst downloading a file from the CN to the mobile terminal. We include results for every possible scenario using the CL testbed. The values in Figure 3(b) show the total delay when the MN moves between two access points that belong to different technologies. Using the complete picture offered by Figure 3, we can observe some early patterns in the behaviour of MIPv6, when dealing with heterogeneous technologies.

When the MN moves "up" in the model (from a faster network to a slower one with higher RTTs) the latency is smaller, e.g., to handoff from WLAN to GPRS takes 5.5s, whereas when the terminal moves to a lower layer (i.e. to a faster access technology), the disruptions last for longer (the handover from GPRS to WLAN takes about 7.4s). This relation is also true for the latency values at the Network Layer shown in Figure 3(a). This will be explained later in 6.2.

The values are larger for the TCP latency in every scenario, compared to the network-layer latency. This is due to the residual TCP back-off time or the interval for which the TCP flow remains (exponentially) backed-off even after the IP-level handover. We consider this delay part of what we termed *adaptation component*, that simultaneously affects the handover latency at Layer 4 (Section 6 explains this concept).

5.4 Packet Overhead

In this section, we briefly introduce an analytical study of the protocol overhead introduced by Mobile IPv6, by comparing it with the overhead present in IPv4 and plain IPv6.

Mobile IPv6 adds an extra IPv6 header (40 bytes) to every packet sent/received by the MN in the MN-CN portion of the path followed by the data packets. If Route Optimisation support is enabled in an ongoing communication between a MN and a CN, a 24-byte overhead (due to the addition of a Home Address Destination Option in the packets sent by the MN or a Type II Routing Header in the packets sent by the CN) is added instead. For the case of two MNs that are communicating, a 48-byte overhead is present, because both the Home Address Destination Option and the Routing Header are present in every packet of the communication.

Figure 5 shows the packet overhead for different types of IP payloads graphically:

- **TCP 40 bytes.** Mostly TCP packets which carry TCP acknowledgements, but no payload. This is the minimum TCP packet size.
- **TCP 552 bytes.** This packet-size (or 576 bytes) is used by those TCP implementations that do not use path MTU discovery.
- **TCP 1500 bytes.** This is the maximum Ethernet payload size.

  Because a large proportion of the TCP traffic is generated by bulk transfer applications, such as HTTP and FTP, the majority of the packets seen in the Internet are one of the previous sizes [15].
- **UDP VoIP GSM**. UDP-RTP[6] packets carrying VoIP payload using the GSM codec (33 bytes per packet).
- **UDP VoIP G723.1**. UDP-RTP packets carrying VoIP payload using the G723.1 codec (20 bytes per packet).
- **UDP VoIP G711**. UDP-RTP packets carrying VoIP payload using the G711 codec (240 bytes per packet).
- **UDP VoIP LPC10**. UDP-RTP packets carrying VoIP payload using the LPC10 codec (7 bytes per packet).

We analyse UDP-RTP VoIP packets because VoIP has been one of the emerging services in the recent years and also because the packet overhead is a critical parameter in this application (even when using IPv4).

Figure 5 shows that even for 552-byte TCP packets, the overhead is not negligible (about 14-16%) when Mobile IPv6 is used. The use of Mobile IPv6 has a great impact on the packet overhead in VoIP packets – sometimes even tripling it compared with the IPv4 case. Therefore, special care should be taken in the design of the network (i.e. resources, queue management, QoS mechanisms, etc.), especially if applications request QoS guarantees. Thus, we need to take note of the added overhead to support mobility in 4G networks; current deployments (designed for IPv4 traffic) could be insufficient for carrying the new IPv6-mobility-enabled traffic.

6 Vertical handover latency characterisation

A handover process between two disparate wireless technologies is usually divided into three main stages: (1) network discovery, (2) network selection, and (3) handover execution. However, for the case of the transport-layer handover using TCP (i.e. connection oriented), we decided to add a fourth phase to the handover process. Derived from the results collected from experimental work done in the CL testbed, we concluded that after the handover execution there is a period of time during which the mobile host needs to adapt to the new link characteristics. We called this the *adaptation component* (t_a) that affects the handover latency depending on the characterisation old and the new networks (see Figure 6).

[6] The RTP header is 12 bytes

The stages are therefore:

- *Detection Period (t_d).* It is the time taken by the mobile terminal to discover the available network(s) using link-layer signalling or the Network Layer detection mechanism. The Mobile IPv6 generic movement detection mechanism uses the facilities of IPv6 Neighbour Discovery. When the MN is not sending traffic, it listens to IPv6 router advertisements to determine what networks the terminal is within the coverage of.
- *Configuration Interval (t_c).* This is the interval from the moment a mobile device receives a Router Advertisement, to the time it takes to update the routing table and assign its new Care-of Address based on the received network prefix, including the Duplication Address Detection (DAD) delay. This interval depends on the terminal characteristics (e.g., memory, processing power, etc.)
- *Registration Time (t_r).* This elapses between the delivery of the Binding Update to the Home Agent and correspondent nodes, and the reception of the first packet at the new interface – with the new Care-of Address as the destination address. This time component increases if the mobile terminal is configured to wait for the Binding Acknowledgement sent by the Correspondent Node, as mentioned in Section 11.7.2 of [2].
- *Adaptation Time (t_a).* When dealing with vertical handovers at the transport level, we need to consider t_a in the total handover latency. This delay only happens when the mobile host adapts the connection to the new technology at the Transport Layer, adjusting the TCP state machine parameters (e.g., congestion window size, timeout timers, etc.) due to the heterogeneous nature of the technologies and the disparity in the link characteristics. Thus for the case of TCP transmissions, the transport-layer latency (T_t) is equivalent to the network-layer latency (T_n) plus the adaptation component (t_a), as shown in Figure 7.

The definition of *handover latency* in Mobile IPv6 is limited to the period of time when the Mobile Node is unable to receive IPv6 packets both due to the link (i.e. Layer 2) switching delay and IP protocol operations (i.e. network-layer handover), as defined in [2]. However, this paper analyses the latencies in both UDP/MIPv6 and TCP/MIPv6 for transport-layer handovers. For the UDP case, the latency is equivalent to the network-layer latency, however, for TCP transmissions we extend the term of handover latency to consider the t_a component.

In this study handover latency is divided into: network-layer handover latency and TCP handover latency. The former matches the definition included in [2], and the latter is defined as the period of time when the Mobile Node is unable to receive IPv6 packets with a stable pattern (meaning that the throughput is close to the link's normal behaviour) due to the link switching delay, the IP protocol operations, and the adjustments performed at the transport layer due to huge disparities in link

characteristics.

It is important to mention that we analysed the case of TCP handover because studies have shown that 85% of the traffic in the Internet is generated by TCP connections [15]. Therefore, proposals to minimise TCP handover latency during vertical handovers are essential for future seamless roaming – one possible approach is to reduce t_a, in order to improve the overall latency.

6.1 Analytical representation of latency partition

As mentioned above, we have characterised the impact of vertical handovers in the Network Layer using ICMPv6 protocol, and the effects at the Transport Layer when using UDP and TCP as transport protocols. We consider the cases of the GPRS-WLAN-LAN testbed based on current standard wireless technologies to outline an experimental study that determines the impact of each individual component in the overall handover latency.

The overall latency is found by summing the delays to discover the new network (t_d), to build the Binding Update message using the prefix from the new access router (t_c), and to register the recently formed Care-of Address (CoA) with the Home Agent and correspondent nodes (t_r), as shown in Equation 1. The network discovery delay depends on the movement detection mechanism, but if the generic L3 Neighbour Discovery based mechanism is used by the MN, this time depends mostly on the Router Advertisement frequency and also on the policy that the MN uses to consider a router unreachable. Generally – if the Lazy Cell Switching algorithm is used –, the MN may use the Advertisement Interval (if included in the Router Advertisements received by the MN) field as an indication of the frequency with which the current default router is sending these messages. Therefore, if during this time interval (which indicates the maximum time between two consecutive RAs) the MN does not receive any new RA from the current default router (i.e. at least one RA has been lost), the MN can use the event of losing a certain number – m – of RAs as a possible L3 handover indication (in MIPL, for example, a figure of 2 lost RAs is used, triggering the MN to send Router Solicitation messages on all the available interfaces to discover new reachable routers). It should be noted that with additional support, such as the Complete Prefix List mechanism [16] or the Detecting Network Attachment (DNA) protocol [17], the detection time may be reduced, improving the overall performance (the reliability of the detection mechanism is also improved).

The network-layer latency is given by:

$$T_n = t_d + t_c + t_r \tag{1}$$

where

$$t_d \text{ is a random variable between } 0 \text{ and } m \times RA_{int_{MAX}} \tag{2}$$

$$t_c = t_{DAD_{LL}} + t_{DR} + t_{CoA} + t_{DAD_{NL}} \tag{3}$$

$$t_r = t_{RR} + t_{BU} \tag{4}$$

m = *number of consecutive missed RAs, used by the MN to trigger the movement detection*,

$RAint_{MAX}$ = *Maximum RA interval (i.e. time between two consecutive Router Advertisements)*,

$t_{DAD_{LL}}$ = *Time taken for Duplicate Address Detection for link local address,*

t_{DR} = *Time taken for Default Router configuration,*

t_{CoA} = *Time taken for configuring the new CoA,*

$t_{DAD_{NL}}$ = *Time taken for Duplicate Address Detection for CoA,*

t_{RR} = *Time taken for entire Return Routability procedure (= 1.5 RTTs in the best case scenario - i.e. no packet losses)*

t_{BU} = *Time taken for update of the binding in the HA (= 1 RTT from the MN to the HA using the NAR) and this can be done simultaneously while updating the CNs (an optimisation to minimise this delay is explained in Section 7.3)*

The configuration time depends on the terminal characteristics, however, some methods have been proposed to reduce this delay such as avoiding the DAD mechanism to minimise disruptions when roaming between access routers [18]. The last component (t_r) is dictated by the RTT of the network used for the registration process, as shown in Equation 4.

6.2 Experimental latency partition

We have evaluated the impact that hard (i.e. only one active interface at the same time) handovers have on the network and transport layers. To demonstrate latency

partitioning, we consider the cases of GPRS-WLAN-GPRS and GPRS-LAN-GPRS because these are the most common scenarios. The values in Table 1 show the latency partition for vertical handovers.

We observe that the raw latency values included in the tables are too high to even dream of using MIPv6 (or its implementation, MIPL) with real time applications for which the tolerated delays are less than 100ms. For example, the handover latency from a hotspot to the cellular system takes around 6.8 seconds to complete, and in the scenario where the terminal moves from a cellular system to an IEEE 802.11b access point there is has an overall delay of 3.8 seconds.

As t_d is a random variable, that depends on the RA frequency, we can see that the values for the standard deviation in every case are very large – around 40% and 50% of the mean value. Ideally, when the mobile host moves from a high-RTT and low-bandwidth network (e.g. GPRS) to a low-RTT and high-bandwidth access technology (e.g. WLAN), the registration time should be smaller than the opposite case – moving from a lower layer to an upper layer – as RA frequencies are higher in the lower networks and also because the time required for the signalling to be completed is lower. However, we can note from Table 1 that this is not necessary true for all the cases.

The registration time is basically related to the RTT of the network; since WLAN offers links that have low RTTs, then t_r for GPRS to WLAN handover should be smaller than for its counterpart (i.e. WLAN to GPRS). The measured values show the opposite. This is mainly because two reasons:

- **High buffering effects.** The Mobile IPv6 implementation used in these tests (MIPL) tries to send the BU to the CNs piggybacked in a data packet. Therefore, the BU is sent in the first data packet after the movement detection (or after a certain maximum amount of time, to avoid delaying the Binding Update so much). Because of the high buffering in the GPRS GGSN, the MN perceives an inflated RTT, and hence, an increased Retransmission Timeout (RTO). Consequently, the registration process can complete only after the source eventually retransmits with a high value of the RTO. This leads to a very high latency when moving from a hotspot to the cellular network (see Figure 8), affecting the t_r delay.
- **Effects of applied handoff mechanisms**. There is also an additional reason that affects the overall handover latency, that is related to the Movement Detection mechanism specified in Section 11.5 of the MIPv6 RFC document [2], and directly affects the t_d delay. The specification does not include policies to determine when the mobile node needs to perform Router Discovery, due the loss of connectivity through the current network. However, it does enumerate certain indications that should be considered (e.g. Advertisement Interval field) to detect unreachability of the current Access Router.

For the case of MIPL [13], which is the MIPv6 implementation used in the CL testbed, there are different policies according to certain conditions when deciding the handover. The Movement Detection mechanism included in MIPL covers the following situations:

- *No movement detection*. If the detected Router Advertisement comes from the current router, we infer that the MN is still within the coverage of the current Access Router and there is no other network available. Hence, the MN does not perform any handover action.
- *Horizontal Handover.* This occurs when the incoming RA is received through the same interface – and it is not sent by the current Access Router, which implies that both the new and old access routers are within the same layer (i.e. wireless technology). Under these conditions, MIPL executes the handover using *Eager Cell Switching* (ECS) algorithm. The RA received is processed immediately, followed by the configuration and registration procedures.
- *Vertical Handover.* The incoming RA comes from a different Access Router, and it is received on a different interface from the one being used. This means that the new and old routers of attachment belong to different access technologies. In this case MIPL checks if the new network interface (NIC) is preferred over the old interface. If the new one is preferred, then the RA is processed, and if not, MIPL triggers a kind of *Lazy Cell Switching* (LCS) algorithm for vertical handovers.

We need to detail the LCS algorithm implemented in MIPL in order to understand its effect on the handover latency. If the interface that receives the RA from the new Access Router (i.e. recently available network) is different from the one being used, then the terminal waits until the current network is unreachable. As shown in Figure 9(b), the MN detects that the old Access Router is not reachable when it does not receive RAs for a period of time equal to twice the RA interval ($2 * RA_{interval}$). The MN sends a Router Solicitation after 8s because the RA interval is configured to 4000ms for the example shown in Figure 9.

For this implementation of the LCS (i.e. MIPL), the current network has higher priority than any other technology available, without considering the link characteristics. Thus, if no other policy is taken before the handover execution, a slower network can have a higher priority than a faster technology. In this case, the mobile node applies the LCS algorithm instead of ECS when moving from the cellular system to an available hotspot, resulting in a bigger t_r as shown in Table 1.

We can better observe the importance of applying the appropriate handover execution method (in MIPL either ECS or LCS) in Figure 9. When we applied ECS, whilst performing a downward (to handoff from a global coverage to a local coverage access technology) handover, the total latency was approximately 0.5s, whereas for the case where LCS was used (i.e. when the slower interface had a higher preference value), the handover process took approximately 7.5s in total.

From this situation, we can foresee the need for a policy-based system to take well-informed handover-related decisions, and deal with the complexities posed by heterogeneous networks. The handover performance can be improved just by having an *a priori* policy to set the interfaces' preferences according to the link characteristics. For example, the GPRS preference is set to a lower value than the IEEE 802.11x interface due to its higher RTT values. Thus, when a downward handover occurs, the ECS algorithm is applied and not the LCS execution method, improving the vertical handover performance in this scenario.

6.3 Network and transport-layer latency

We have previously described and analysed the different parts that global handover latency is composed of. One important component, that little attention has been paid to thus far, is the *adaptation component* t_a, that is present when we are dealing with TCP connections. We can define t_a to be from the point at which the MN receives the first data packet on the new interface from the CN, to the point where the new interface's throughput first reaches the value of the average throughput it achieves over the lifetime of the connection.

We measured T_n and t_a [7] for the WLAN to GPRS and LAN to GPRS handover types (figures 10(a) and 10(b)). The results obtained [19] show that the adaptation time is a significant fraction of the total handover time, especially in the WLAN to GPRS case. Clearly this increased connection interruption time will have a significant effect on any applications that require even a mediocre quality of service.

We have also investigated downward handovers from GPRS to WLAN and GPRS to LAN. The results in both cases indicate that the adaptation time is negligible, and therefore we do not include the data here. However, we note that TCP performance is affected by reordered and duplicated packets.

7 Impacting MIPv6 latency

As shown in the previous section, the total MIPv6 handover latency is the sum of the IP layer latency (T_n) and the TCP adaptation component (t_a). In this section, we focus on impacting the MIPv6 latency to minimise disruptions in the connectivity while roaming between heterogeneous networks. We discuss the following methods: RA frequency, RA caching, BU simulcasting, and Soft Handover to improve networking performance.

[7] Values are calculated by measuring the quantity of data sent in each 0.1 second interval. This value was chosen as sufficiently large to filter out high frequency variation in the trace, whilst exhibiting the highly dynamic nature of the connection's throughput.

7.1 RA frequency

Higher RA frequency improves performance due to the reduction in t_d, which results in smaller latencies during vertical handovers. There are two methods related to the Neighbour Discovery protocol that can reduce the network discovery time: (1) incrementing the RA frequency or (2) generating an on-demand Router Solicitation.

Related to incrementing the RA frequency, the current Mobile IPv6 specification [2] considers this topic and proposes shorter intervals – 30ms (MinRtrAdvInterval) to 70ms (MinRtrAdvInterval). However, decreasing the RA interval is not the end of the story, as this has a direct impact on the link performance, caused by the added overhead. Thus, there is an obvious trade-off involved, especially over reduced-capacity links such as GPRS networks.

In order to evaluate the RA frequency's impact on vertical handovers, we used a modified version of the Linux IPv6 RA daemon (radvd [8]) to perform experiments using different RA interval values, including the ones specified in [2]. Figure 11 shows the effect of varying RA interval on mean t_d values obtained from over 20 runs. Based on these results, we observed that although increasing the RA frequency reduces the total latency, it is not the best option for low-bandwidth networks such as GPRS.

For example, in the case of WLAN, increasing the RA frequency from 300-400ms (min-max) to 40-70ms represents a marginal increase in the overhead, and the benefits in t_d (as a consequence, in the overall latency) are very significant (obtaining a reduction of 65%), whereas this same situation on the GPRS link has completely different effects. Reducing the RA interval to 40-70ms means using almost 50% of the link to transmit RAs (considering a 36.9kbps link of a '3+1' GPRS phone), however, there is a 61% reduction in the t_d delay. Thus, after this analysis, we suggest for the case of the GPRS overlay the following RA interval values to maintain a convenient trade-off between added overhead and benefits: *MinRtrAdvInterval=500ms* and *MaxRtrAdvInterval=1000ms* (see [20] for details).

7.2 RA caching

This method is oriented to eliminate the discovery time (t_d) from the total latency. The main principle behind this optimisation is that the mobile node caches every incoming RA, and when it needs to perform a vertical handover (specifically upward handovers) because of the loss of coverage, the mobile terminal does not wait

[8] `http://v6web.litech.org/radvd/`

for the next incoming RA from the currently available access network, but instead it uses a previously cached RA when possible, reducing (t_d) to zero.

It is important to mention that this optimisation is only useful when there is an overlap between the coverage of the old and new access routers. The algorithm was originally designed for horizontal handovers between overlapping cells, a detailed description of this version is included in [21]. We extended it to support vertical handovers, however it only reduces t_d when a cached RA exists (i.e. for the upward handover scenario).

The optimisation was implemented as a module, part of a middleware to support complete mobility (see Section 8). In general, the RAs are extracted from the network stack before they reach the MIPv6 module. These RAs are maintained in the RA-cache (according to certain policies such as time-to-live and signal strength). Then, when the MN performs a handover, high-level policies decide if an upward handover will occur and check for the cached RA to process it.

7.3 BU simulcasting

During IP-level handovers, the time required to communicate the new Care-of Address to the Home Agent and correspondent nodes (t_r) is typically limited by the RTT to the CN and the HA. A technique that can be used to minimise the impact of t_r on the overall latency (T_n) is to simulcast the BUs. Thus, the registration time is limited by the RTT of the fastest network and not by the latency of the new network as specified in the Mobile IPv6 specification, (Section 11.7 [2]). The t_r delay is given by: $R_t = T_{RR} + t_{BU} \geq 2.5$RTTs (Figure 12 shows the reduction values for different scenarios, using estimated RTT values).

7.4 Soft Handover

The optimisations discussed thus far have been related to *hard handover*; the MN disconnects (stops listening) from the old interface, and just then starts listening to the new interface. As a result, packets that were already *on the air* – sent by the source (i.e. CN) before it realises that the Mobile Node has moved to another network – or those destined to the old network interface are discarded. These packets need to be retransmitted by the source, leading to a reduction in performance because of vertical handovers. However, vertical handovers can be made "soft" to improve inter-networking.

Traditionally, *soft handovers* (the MN attaches to the new network before breaking the connection with the old access point) have been exploited for link-layer handovers in cellular networks. This paper analyses soft vertical handovers between

highly heterogeneous networks. To enable this study using our testbed, we modified the source code of MIPL [13] so that during the handover process every on-the-air packet destined to the previous interface (i.e. old interface) is read and passed to the application, despite the incongruity between the packet's destination address (old interface's address) and the current terminal's CoA (new interface's address).

Thus, the MN keeps receiving packets from the previous network and meanwhile, it completes the registration process with its new CoA and starts receiving packets through the new interface, which has the CoA assigned. We may think that using soft handovers would enable the seamless roaming. However, the benefits are not always evident due to drastic differences in link characteristics (heterogeneous networks).

Figure 13 shows an example trace collected using a modified version of MIPL, showing the benefits and drawbacks of soft handovers in vertical scenarios; this figure shows the WLAN-GPRS-WLAN case, sending TCP traffic between the CN and the MN. As evident in the plot (centre), performing soft handovers leads to dramatic reductions in handover latency (there is no packet loss during the handover, although packet reordering can occur).

The MN is connected to the WLAN (upper left corner) and initiates a handover to the cellular network. However, the source (CN) continues sending packets destined to the old interface for approximately 1ms (until SND.UNA+SND.WND-SND.NXT reduces to zero). The mobile terminal receives the on-the-air packets through the old interface and responds sending ACKs using the new interface, which is, in this scenario, much slower than the previous one. The CN times out and starts retransmitting these packets, whereas the MN sends SACKs to the source because it is receiving duplicated packets (see lower left corner plot). We can observe that when the MN performs an anticipated upward handover (WLAN-GPRS), the disparity in the link characteristics affects the TCP connection, retransmissions occur, and the benefits of soft handovers are less dramatic.

For the other case (right side), the MN is connected to the GPRS network and starts a handover to the WLAN. Some packets are on-the-air, until the CN realises that the MN has moved – in this case, the registration process delay is very small, compared to the time that it takes for the packets to arrive at the old interface (lower right plot). The source starts retransmitting the on-the-air packets because these have not arrived to the MN due to the RTTs in the GPRS network. Finally, the MN sends some SACKs as it receives delayed on-the-air packets on the old interface, which have been already retransmitted by the source. We can see again that because of the huge differences between networks, the source retransmits packets that are already at the MN, wasting bandwidth and reducing the benefits of soft handovers. We are exploring TCP modifications to reduce the impact of link disparities in soft handovers.

The deployment of this kind of handover mechanism is crucial for the offering of real-time services, such as VoIP – which is one of the most promising services in 4G networks – and video streaming. Although UDP is usually the preferred transport protocol for real-time protocols, due to the use of UDP-restricted firewalls, many applications use today TCP. As an example, Skype supports both transport protocols [22]. Therefore, handover mechanisms that retain on-the-air packets are critical for seamless roaming.

8 High-level Handover Decisions

Individual wireless access networks show limitations that can be overcome through the integration of different technologies into a unified platform (i.e. 4G system). Nevertheless, the integration of heterogeneous networks poses many challenges such as adding complexity to the processes of deciding when to handoff, selecting the best network, and minimising roaming effects using appropriate handover methods. For this purpose we designed PROTON, a solution that supports dynamic decision-making processes related to roaming between heterogeneous technologies. PROTON deploys a formal policy representation model, based on Finite State Transducers, that evaluates policies using information from the context to manage mobiles' behaviour in a transparent manner, hiding 4G systems' complexities. We blend concepts of autonomic computing into the design of the solution and manage to improve user experience in typical 4G scenarios while keeping transparency.

The growth in the popularity of Internet services among mobile users, together with the higher QoS required by novel applications, demands improving resource management capabilities in mobile devices to offer a better user experience. High mobility, seamless roaming, high data access rates and transparent connectivity to services from "any" device are dominant trends in the 4G vision and the basic reasons to think that *autonomic computing* [23] represents a plausible solution for emerging challenges. We sense that taking into account heterogeneity, dynamics, and complexity added in 4G environments, an appropriate support should endeavour to possess these key elements with the intention of offering a complete seamless solution . From these concepts, we integrate the following characteristics in PROTON's design:

- *To be autonomic, a system needs to "know itself".*
- *An autonomic system must configure and reconfigure itself under varying and unpredictable conditions.*
- *An autonomic system never settles for the status quo—it always looks for ways to optimise its workings.*
- *An autonomic system knows its environment and context surrounding its activity, and acts accordingly.*
- *An autonomic system cannot exist in a hermetic environment.*

- *An autonomic system will anticipate the optimised resources needed while keeping its complexity hidden.*

The PROTON architecture enables an autonomic solution by supporting these principles. This architecture and the main processes related to PROTON are summarised in this section. PROTON was fully deployed and evaluated in different scenarios. A detailed description of the system and its full evaluation can be found in [24].

8.1 Architecture

PROTON components are divided into network-side and host-side components. The reason for this is that because of the number of decisions required to fully support the handover process, the raw policy set can get too complex to maintain within a resource-limited mobile device. However, the functionality still being completely based on the mobile host, only the highly demanding pre-processing tasks related to the policy evaluation model are placed in the network, in which computing constraints are much more relaxed.

The host-side components are organised into a three-layered system: *Context Management layer*, *Policy Management layer*, and *Enforcement layer*, which sit on top of network layer in the protocol stack. The network-side contains the components related to the specification and deployment of the policies.

8.2 Network-side Components

Those components that involve operator's management or high computational cost are located in the network to reduce complexities at the mobile terminal. This is the case of policy definition, storage, and conflict resolution. Network-side components are shown in Figure 14(a).

Policy Editor—To create the system policies, the operator must write them in a high-level policy specification language. We chose Ponder as the high-level language because of its expressiveness and deployed tools. In particular, in PROTON we have used the *Ponder Policy Editor* and its compiler [25] to create the first internal Java representation of the policies.

Policy Repository—The policy repository is implemented using a Light-weight Directory Access Protocol (LDAP) server, which is intended to store system policies in their high-level representation as well as in the internal Java representation.

Policy Translator—This component translates the policies specified in Ponder language [26] into the evaluation model described in Section 8.5.

Conflict Resolution Module (CRM)—The conflict resolution module builds the deterministic Finite State Machine modelling every active policy. The CRM performs two main tasks: (1) it combines the policies among them considering the system constraints and (2) it resolves conflicts among those rules. During this task, all possible static and dynamic conflicts are foreseen. Therefore, the algorithms that are executed have a high computational cost. The main benefit of adding such overhead on the network-side is to avoid heavy tasks in the mobile device (usually a terminal with limited computational power and memory capacity). Furthermore, after resolving conflicts and constructing the deterministic Finite State Machine, the mobile device can react quickly to incoming events.

Model Deployment Module (MDM)—Once all policies and constraints are combined in a TFFST, it is delivered to the mobile device and installed into its *Policy Master* to drive its decisions. This process is carried out sending the Java object via RMI.

8.3 Host-side Components

Context Management Layer (CML)—This layer has two type of components responsible for collecting Networking Context: *Sentinels* and *Retrievers*. The former is responsible for collecting dynamic elements, and the latter manages static elements. There is a responsible object for each context element, and it has individual settings (e.g., polling frequency and local rules) depending on the complexity and dynamics of a particular fragment.

Policy Management Layer (PML)—Responsible for the control and evaluation of the policies to drive the behaviour of the mobile device. It is composed of the following elements:

Policy Master: This component acts as the Policy Decision Point (PDP) in the policy system [27]. It receives events (e.g., Transition-Pedestrian produced by the VelocitySentinel) from the CML, and according to these inputs, it decides the possible actions to execute, which are immediately sent to the Enforcement Layer.

Context-based profile selector: The fact that only a small portion of sensory input is relevant under certain conditions is used to improve the performance of the system. Some inputs can generate special events (i.e. *macro-events*) which are then used by the selector to load a profile that defines a valid subset of policies to evaluate, i.e. the appropriate TFFST . An example of a macro-event is velocity—if host speed is more than 90 km/h the only active policies are those that produce an upward handover as an action. This means that mobile users should never attempt to connect

to a less ubiquitous access technology when moving at very high speeds.

TFFST Repository: The TFFSTs are produced in the network side, as mentioned in Section 8.2, and then deployed into the mobile device where they are kept in the TFFST repository. Thereafter, the selected TFFST and its evaluation are decided according to the events received from the CML.

Enforcement Layer (EL)—Formed by different *Executors* that are the Policy Enforcement Points (PEPs) of the system [27]. They are responsible for performing the actions that result from evaluating the TFFST. The EL connects with the lower layers through a *Control Interface* (CI) that captures incoming router advertisements just before they reach the Mobile IPv6 module—prior to the handover procedure. The CI executes different scripts, which receive the selected interface as a parameter and outline the execution handover method.

Communication protocols—For the CML-PML connection and the communication within the PML, we use a generic asynchronous notification service called *Elvin* [28]. This service was primary designed as a middleware for distributed systems, however, many research projects have used Elvin due to its simplicity. Ponder uses this messaging service in its framework, and we decided to use it in our system as well.

8.4 Policy Specification

PROTON follows the *Event-Condition-Action* (ECA) paradigm where policies are rules that specify actions to be performed in response to predefined conditions, triggered by events. This high-level policy is compiled into an initial Java representation and translated into TFFSTs.

8.5 An Evaluation Model Based on Finite State Transducers

Finite State Automata are classical computational devices used in a variety of large-scale applications. FSTs, in particular, are automata whose transitions are labelled with both an input and an output label. They have been useful in a wide range of fields, but particularly in Natural Language Processing. This discipline makes intensive use of grammatical rules, which are ambiguous by nature, and requires quick decisions based on those rules, in particular in fields such as speech recognition with major performance requirements.

Additionally, Finite State Machine-based solutions are typically light-weight. They can be implemented as arrays of states, transitions, and pointers among them without falling into heavy management structures.

We represent the policies as *deterministic transducers* that are a category of transducers without ambiguities. This means that at any state of such transducers, only one outgoing arc has a label with a given symbol or class of symbols.

Deterministic transducers are computationally interesting because their computation order does not depend on the size of the transducer, but rather only on the length of the input since the computation consists of following the only possible path corresponding to the input and writing consecutive output labels along the path [29].

For representing policies with FSTs we used the model presented in [30], where a more detailed description of it can be found. It is based on a modification of predicate augmented FSTs [31], in which predicates are replaced by a metric representing the relation between a policy and a given event.

A policy has a condition delimiting a "region of events" where a given event can or cannot lie. When such an event is inside two or more overlapping regions a modality conflict may arise. We are concerned about how tautly a condition fits to an event instead of how far from the border it is. Thus, the preferred condition will be that which is the most "taut" around the event under consideration.

In order to quantitatively represent the aforementioned *tautness*, we use the metric called Tautness Function, a real number in the interval $[-1, 1]$ so that the more taut a condition is, the closer its TF is to zero.

8.6 Modelling Policies with TFFSTs

To understand how the entities introduced before are used for modelling policies, we present how *obligations* and *constraints* are expressed. TFFSTs may model authorisations, prohibitions and dispensations as well, but the following two policy types are expressive enough to deal with current PROTON requirements.

Some actions can be conditioned on the occurrence of more than one event. This is the case for the *lazy switching handover method*, in which after initiating the handover (receiving the *NetworkSelected(nic)* event) we need to delay the action—or wait for the *TimerOver(delay)* event. To express an action as a consequence of a set of events (e.g., Rule 2) a transducer such as the one in Figure 15 is built.

In the figures, the symbol "?" represents the TF associated with the *all-event* condition while the "-" symbol represents set subtraction and "ϵ" means a null event. Symbols "$<$" and "$>$" enclosing TFs in the labels express identity between inputs and outputs. The *NetworkSelected event* is indicated using "ns", the *TimeOver event* with "to", and the *Handoff action* uses the "ho" abbreviation. By convention, state

0 is *initial* and a double-circled state is *final*.

```
 Rule 1:

inst oblig /ProtonPolicies/Obligs/LazyHandover {
  on NetworkSelected(nic) → TimerOver(delay);
  subject /ProtonPMAs/HandoverPMA;
  target t = /ProtonTargets/HandoverExecutor;
  do t.handoff(nic);
  when t.isRAreceived(nic);
}
```

8.7 Policy Translation

Policy translation from high-level languages into internal policy evaluation models can be a complex task that needs to be kept simple and performed on an ad-hoc basis in our system. The main challenge of the deployment was the implementation of the TFs associated with the policy conditions. Considering the fact that the PROTON policy model is based on the tools provided by Ponder, the correct approach was to keep its object-oriented approach using target and subject methods to compute TFs.

Thus, when target or subject methods are called to check a "when" clause, a corresponding method is called at the same time to assign a TF value instead of the boolean value that Ponder assigns to the condition. This method should be developed explicitly, enabling the design of different TF computations depending on a specific parameter (for conditions represented by logical combinations of simple conditions, the TF algebra remains valid).

9 Related work

The concepts of wireless overlay networks and vertical handover were introduced in 1996, as part of the BARWAN project at Berkeley [32, 33]. The first overlay networks testbed, the BARWAN testbed, included WaveLAN, Infrared, and Ricochet wireless networks. Obviously, this project, which was the pioneer one in the area of mobile networking, was based on MIPv4.

Other researchers [34, 35] have worked on the evaluation of MIPv4 during intra-technology handovers (i.e. horizontal handovers), for example, the project at Stanford, MosquitoNet [36].

Now, with novel protocols, researchers concentrate on minimising delays in order to enable seamless handovers, needed for supporting real-time applications.

It should be noted that the type of handover (i.e. horizontal or vertical) is a very important factor that should be taken into account when dealing with the optimisation of handover performance. There are solutions defined within the IETF MIPSHOP WG, such as Fast Handovers for Mobile IPv6 [7] and Hierarchical Mobile IPv6 [8], that perform quite well in horizontal scenarios [10], as these protocols were designed with IEEE 802.11x to IEEE 802.11x handover scenarios in mind. On the other hand, the FMIPv6 solution [7] does not apply to GPRS $\Longleftrightarrow$ IEEE 802.11x handover scenarios, because the handover signalling is something to be performed at the very edge of the network (i.e. Access Routers) and conducting FMIPv6 signalling between an AR in the IEEE 802.11x network and the GGSN in the GPRS core network requires the flows to traverse too many hops, making the process less reactive than is desired in a Fast Handover approach. The vertical handover scenario poses additional challenges, because of the presence of heterogeneous networks that may present disparate characteristics.

The *Moby Dick* project [9, 37] proposed and implemented a global end-to-end MIPv6-based architecture to offer QoS in heterogeneous environments. The testbed included UMTS-like TD-CDMA wireless access technology, IEEE 802.11b WLANs, and wired connectivity. The mobility management approach followed by this project is based on Fast Handovers for Mobile IPV6 [7] and results, focused on intra-technology (horizontal) handovers, can be found in [10]. Further work is being done as part of a new initiative: the *Daidalos* project [38]. Nevertheless, the lack of access to a real operator's 3G network is the main difference between these projects and our testbed.

Projects like MIND [39], which is the follow up of the BRAIN (Broadband Radio Access for IP based Networks) IST Project, proposed their own local mobility management optimisation: BCMP (Brain Candidate Mobile Protocol), which combines properties of HMIPv6 and FMIPv6. Performance results are good for the WLAN horizontal scenario [40], although the network complexity in terms of required infrastructure is high. Moreover, the obtained performance results obtained were only for the horizontal handover scenario.

The *Nomad* [41] project terminated in June 2004 [42], having successfully set up a MIPv4-based testbed [43, 44]. While this evaluated seamless roaming between heterogeneous networks based on MIPv4 – assuming the presence of foreign agents in each visited network – they did not analyse the performance of MIPv6 in 4G networks.

10 Conclusions

One of the main challenges in future communication systems is heterogeneity, when mobile devices roam between networks. The diversity in these environments

augments the complexity in every stage of the handover process: network discovery, network selection, execution, and adaptation (see Section 6). In particular, one of the main networking-related problems is the latency of vertical handovers that results from the sum of the partial delays related to the aforementioned stages. Using MIPv6, we have conducted sets of experiments to measure, characterise, and improve latency in 4G environments.

In this paper, we have presented our practical analysis of Mobile IPv6 performance focused on the vertical handover latency at the Network and Transport layers, highlighting the main effects of it on the protocol stack. We discussed the means to enable transparent mobility in heterogeneous environments through the reduction of MIPv6 latencies using different optimisation methods. We mentioned that the optimisations described and evaluated in this paper impact different latency components and have an effect only under certain conditions – networking in 4G environments will strongly depend on context, as a result of heterogeneity among access technologies.

We observed that vertical and horizontal handovers are affected in different ways by terminal mobility, thus, not every effect or optimisation is equivalent for both cases. For example, we found that it is not true that when a mobile terminal performs a downward handover, the registration time is smaller. This has two explanations: (1) the high buffering in upper layers (i.e. Vodafone's GSM/GPRS network) and (2) the Movement Detection mechanism included in [2], as was mentioned in Section 6.2.

One of the most interesting and unexpected results is included in Section 7.4. We would have thought that the so-called Soft Handover would always be advantageous for vertical handovers as this is the case for homogeneous environments. However, we have shown that using the current TCP/IP stack soft vertical handovers exhibit a different behaviour due to the drastic changes in link characteristics. Thus, adjustments to TCP protocols would be necessary to improve soft handovers in heterogeneous environments [45].

We need to apply the correct optimisation method in the most appropriate scenario to obtain improved performance. These types of handover-related decisions add complexity to the roaming process that needs to be hidden from the users. Therefore, a middleware to enable informed decisions during the handover process is one of the main drivers towards truly ubiquitous computing.

Performance differences, network heterogeneity, and context dependency are some of the reasons that lead to the need for a policy-based solution where every important condition is considered. PROTON (Section 8) was presented to demonstrate the usefulness of a cross-layer design approach, combined with concepts from autonomic communication and policy-based systems. This client-based middleware enables mobility support for 4G networks, and it can be considered as an early

attempt to apply autonomic communications to future networks.

Future networking environments (4G) will be composed of multiple heterogeneous access networks, highly integrated to offer ubiquitous connectivity to a plethora of IP-based mobile services from different devices. This vision poses the need for a clean policy mechanism (e.g., PROTON) to control the complex relation between service demands and networking resources, based on context.

In conclusion, this paper contributes to the evaluation of MIPv6 as the *de facto* solution for mobility management in future networks. We implemented an integrated testbed to emulate a 4G system and collected real experiences (the most relevant of which are included in this paper) that help us understand the current challenges, and then overcome potential constraints in the deployment of 4G environments.

Acknowledgements

The authors would like to thank Dr. Frank Stajano, Dr. Glenford Mapp, Dr. Marcelo Pias, Dr. Calicrates Policroniades and Dr. Javier Baliosian for their helpful comments that contribute to the improvement of this paper. Also, the authors would like to acknowledge Prof. Andy Hopper for his primordial support for this project since the beginning, back in 2001. Pablo Vidales was sponsored by the Mexican Government through the National Council of Science and Technology (CONACyT) during his studies. Carlos Jes s Bernardos is partially sponsored by the European Union under the E-Next Project FP6-506869.

References

[1] V. Cerf, The Catenet Model for internetworking, DARPA Information Processing Techniques Office, IEN 48 (July 1978).

[2] D. B. Johnson, C. E. Perkins, J. Arkko, RFC 3775: Mobility Support in IPv6 (June 2004).

[3] C. E. Perkins, RFC 3344: IP Mobility Support for IPv4 (2002).

[4] T. Zahariadis, K. Vaxevanakis, C. Tsantilas, N. Zervos, N. Nikolaou, Global Roaming in Next-Generation Networks, IEEE Communications Magazine (February 2002).

[5] R. Droms, J. Bound, B. Volz, T. Lemon, C. Perkins, M. Carney, RFC 3315: Dynamic Host Configuration Protocol for IPv6 (DHCPv6) (July 2003).

[6] S. Deering, R. Hinden, RFC 2640: Internet Protocol, Version 6 (IPv6) Specification (December 1998).

[7] R. Koodli, et al., RFC 4068: Fast Handovers for Mobile IPv6 (July 2005).

[8] H. Soliman, C. Castelluccia, K. E. Malki, L. Bellier, RFC 4140: Hierarchical Mobile IPv6 mobility management (HMIPv6) (August 2005).

[9] A. Cuevas, P. Serrano, J. I. Moreno, C. J. Bernardos, J. Janert, R. L. Aguiar, V. Marques, Usability and Evaluation of a Deployed 4G Network Prototype, Journal of Communications and Networks 7 (2) (2005) 222–230.

[10] C. J. Bernardos, I. Soto, J. I. Moreno, T. Melia, M. Liebsch, R. Schmitz, Mobile Networks Experimental evaluation of a handover optimization solution for multimedia applications in a mobile IPv6 network, European Transactions on Telecommunications 16 (4) (2005) 317–328.

[11] X. P. Costa, H. Hartenstein, A simulation study on the performance of mobile IPv6 in a WLAN-based cellular network, in: 18th International Teletraffic Congress ITC, 2003.

[12] X. P. Costa, R. Schmitz, H. Hartenstein, M. Liebsch, A MIPv6, FMIPv6 and HMIPv6 handover latency study: analytical approach, in: IST Mobile and Wireless Telecommunications Summit 2002, 2002, pp. 100–105.

[13] MIPL, Mobile IP for Linux (MIPL). Developed by HUT Laboratory for Theoretical Computer Science - GO/Core project `http://www.mobile-ipv6.org`.

[14] T. Narten, E. Nordmark, W. Simpson, RFC 2461: Neighbor Discovery for IP Version 6 (IPv6) (December 1998).

[15] S. McCreary, K. Claffy, Trends in wide area IP traffic patterns - A view from Ames Internet Exchange, Tech. rep., CAIDA (2000).

[16] J. Choi, E. Nordmark, DNA with unmodified routers: Prefix list based approach, Internet Engineering Task Force, draft-ietf-dna-cpl-02.txt (work-in-progress) (January 2006).

[17] J. Kempf, S. Narayanan, E. Nordmark, B. Pentland, J. Choi, DNA with unmodified routers: Prefix list based approach, Internet Engineering Task Force, draft-ietf-dna-protocol-00.txt (work-in-progress) (February 2006).

[18] M. Bagnulo, I. Soto, A. Garcia-Martinez, A. Azcorra, Avoiding DAD for Improving Real-Time Communication in MIPv6 Environments, in: Proceedings of the Joint International Workshops on Interactive Distributed Multimedia Systems and Protocols for Multimedia Systems, Vol. 2515, Springer-Verlag, 2002, pp. 73–79.

[19] D. N. Cottingham, P. A. Vidales, Is latency the real enemy in next generation networks?, in: Proceedings ICST CoNWiN, 2005.

[20] R. Chakravorty, P. Vidales, K. Subramanian, I. Pratt, J. Crowcoft, Performance Issues with Vertical Handovers - Experiences from GPRS Cellular and WLAN hotspots Integration, in: Proceedings of The Second IEEE International Conference on Pervasive Computing and Communications (PerCom'04), 2004.

[21] L. Patanapongpibul, G. Mapp, A Client-based Handoff Mechanism for Mobile IPv6 Wireless Networks, in: Proceedings of the 8th IEEE Symposium on Computers and Communications (ISCC), Kiris-Kemer, Turkey, 2003.

[22] S. A. Baset, H. Schulzrinne, An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol, in: IEEE Infocom, 2006.

[23] IBM Research Headquarters (manifesto), Autonomic Computing: IBM's Perspective on the State of Information Technology (October 2001).
URL `http://www.research.ibm.com/autonomic/overview/elements.html`

[24] P. Vidales, J. Baliosian, J. Serrat, G. Mapp, F. Stajano, A. Hopper, Autonomic Systems for Mobility Support in 4G Networks, Journal on Selected Areas in Communications (J-SAC) 23 (12).

[25] Policy Research Group.
URL `http://www-dse.doc.ic.ac.uk`

[26] N. Damianou, N. Dulay, E. Lupu, M. Sloman, The Ponder Policy Specification Language, in: Proceedings of the Second IEEE International Workshop on Policies for Distributed Systems (POLICY 2001), 2001, pp. 18–39.

[27] B. Moore, E. Ellesson, J. Strassner, A. Westerinen, RFC 3060: Policy Core Information Model (February 2001).

[28] G. Fitzpatrick, S. Kaplan, T. Mansfield, D. Arnold, B. Segal, Supporting Public Availability and Accessibility with Elvin: Experiences and Reflections, in: Computer Supported Collaborative Work: the Journal of Collaborative Computing, 2000, pp. 15–51.

[29] E. Roche, Y. Schabes, Finite-State Language Processing, Tech. rep., MIT Press, Cambridge, Massachusetts. (1997).

[30] J. Baliosian, J. Serrat, Finite State Transducers for Policy Evaluation and Conflict Resolution, in: Proceedings of the Fifth IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY 2004), 2004, pp. 250–259.

[31] G. van Noord, D. Gerdemann, Finite State Transducers with Predicates and Identities, Grammars 4 (3) (2001) 263–286.

[32] R. H. Katz, Adaptation and Mobility in Wireless Information Systems, IEEE Personal Communications 1 (1994) 6–17.

[33] M. Stemm, R. H. Katz, Vertical Handoffs in Wireless Overlay Networks, Mobile Networks and Applications 3 (4) (1998) 335–350.

[34] M. Buddhikot, G. Chandranmenon, S. Han, Y. W. Lee, S. Miller, L. Salgarelli, Integration of 802.11 and third-generation wireless data networks, in: Proceedings of IEEE INFOCOM 2003, 2003.

[35] M. Ylianttila, M. Pande, J. Makela, P. Mahonen, Optimization scheme for mobile users performing vertical handoffs between IEEE 802.11 and GPRS/EDGE networks, in: Proceedings of the 13th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, 2002.

[36] K. Lai, M. Roussopoulos, D. Tang, X. Zhao, M. Baker, Experiences with a Mobile Testbed, in: Proceedings of The Second International Conference on Worldwide Computing and its Applications (WWCA'98), 1998.

[37] V. Marques, R. Aguiar, C. Garcia, J. Moreno, C. Beaujean, E. Melin, M. Liebsch, An IP-based QoS architecture for 4G operator scenarios, IEEE Wireless Communications Magazine (June 2003).

[38] EU FP6 Daidalos project, http://www.ist-daidalos.org.

[39] IST-MIND (Mobile IP based Network Developments) project, http://www.ist-mind.org.

[40] E. García, P. Ruiz, et al., MIND Trials Final Report, Project Deliverable (November 2002).

[41] IST-NOMAD project, http://www.ist-nomad.net.

[42] P. Philippopoulos, P. Fournogerakis, I. Fikouras, N. Fikouras, C. Görg, NOMAD: Integrated Networks for Seamless and Transparent Service Discovery, in: Proceedings of the IST Mobile Summit 2002, 2002.

[43] K. Kuladinithi, A. Konsgen, S. Aust, N. A. Fikouras, Mobility Management for an Integrated Network Platform, in: Proceedings of the Fourth IEEE Conference on Mobile and Wireless Communication Networks (MWCN 2002), Stockholm, Sweden, 2002.

[44] N. A. Fikouras, A. Udugama, C. Görg, W. Zirwas, J. M. Eichinger, Experimental Evaluation of Load Balancing for Mobile Internet Real-Time Communications, in: Proceedings of the Sixth International Symposium on Wireless Personal Multimedia Communications (WPMC), Yokosuka-Kanagawa, Japan, 2003.

[45] Y. Matsushita, T. Matsuda, M. Yamamoto, TCP Congestion Control with ACK-Pacing for Vertical Handover, in: IEEE Wireless Communications and Networking Conference 2005, New Orleans, USA, 2005.

WLAN ⇒ GPRS	Mean	Std. Dev.	Min.	Max.
Detection time (t_d)	808	320	200	1148
Configuration time (t_c)	1	0	1	1
Registration time (t_r)	2997	416	2339	3649
Total handover latency (t_n)	3806	327	3323	4438

GPRS ⇒ WLAN	Mean	Std. Dev.	Min.	Max.
Detection time (t_d)	2241	968	739	3803
Configuration time (t_c)	1	0	0	1
Registration time (t_r)	4654	1698	2585	7639
Total handover latency (t_n)	6897	1178	5322	8833

LAN ⇒ GPRS	Mean	Std. Dev.	Min.	Max.
Detection time (t_d)	1168	460	347	2070
Configuration time (t_c)	1	0	1	1
Registration time (t_r)	3307	585	2299	4759
Total handover latency (t_n)	4476	520	2806	5107

GPRS ⇒ LAN	Mean	Std. Dev.	Min.	Max.
Detection time (t_d)	2058	1030	1	3257
Configuration time (t_c)	1	0	1	1
Registration time (t_r)	4466	1449	2357	7183
Total handover latency (t_n)	6525	1229	4011	8197

Table 1
Network-layer latency partition for vertical handovers using MIPL. The tables show the average value, in milliseconds, for 40 handover iterations.

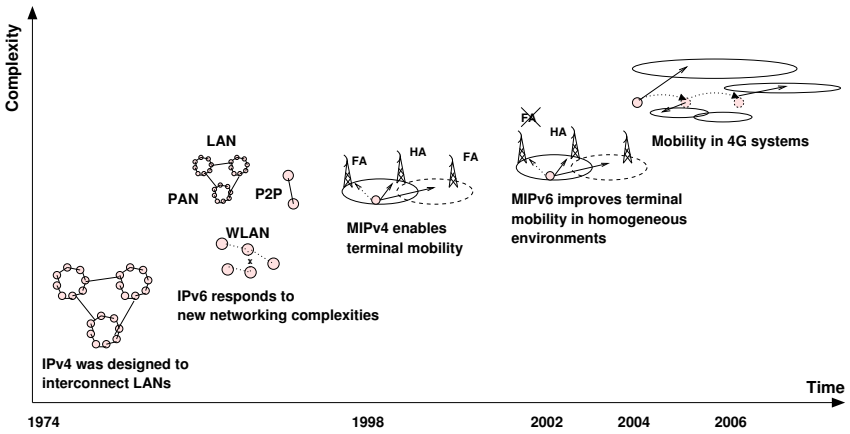
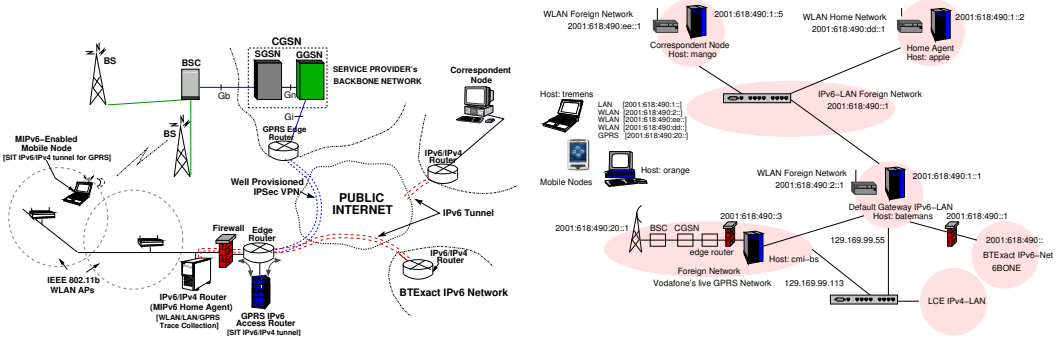
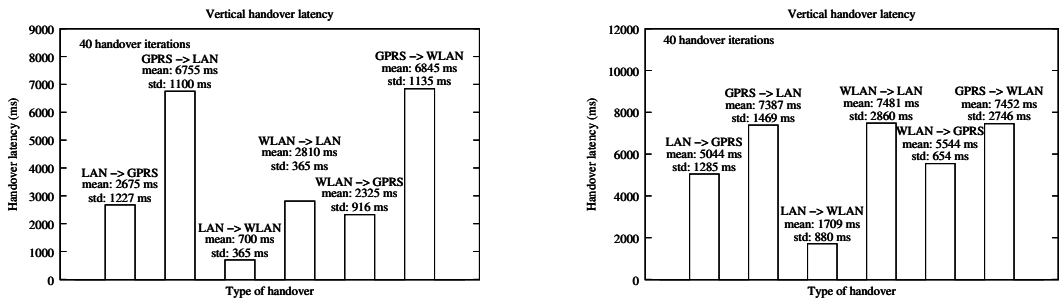


Figure 1. Networking evolution and driver protocols.



(a) Integration view (b) Network-centric view

Figure 2. Experimental setup for 4G systems.



(a) MIPv6 Network Layer vertical handover latency

(b) MIPv6 Transport Layer (TCP) vertical handover latency

Figure 3. Impact of vertical handovers on the TCP/IP stack.

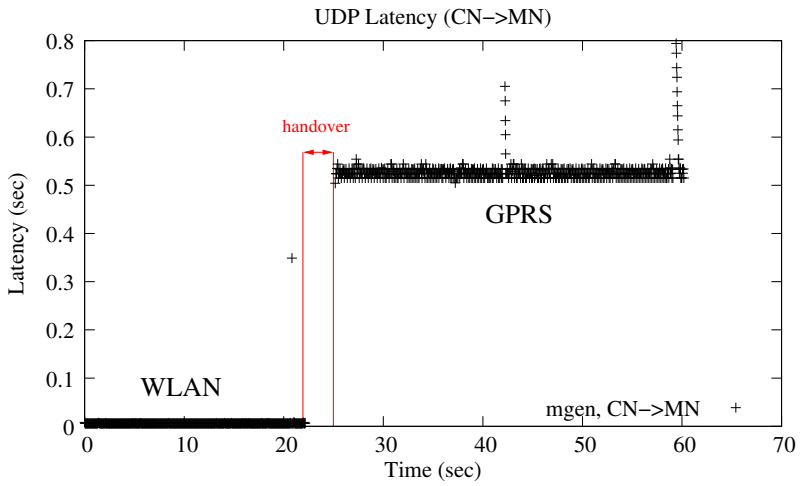


Figure 4. UDP over MIPv6 vertical handover latency.

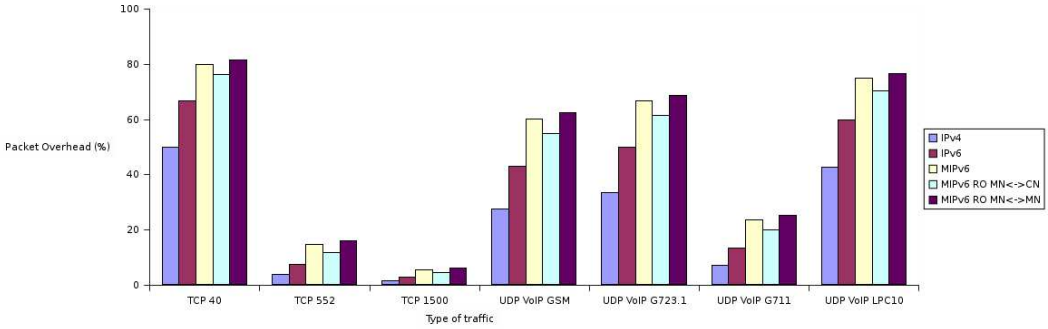


Figure 5. Packet overhead for different types of payloads.

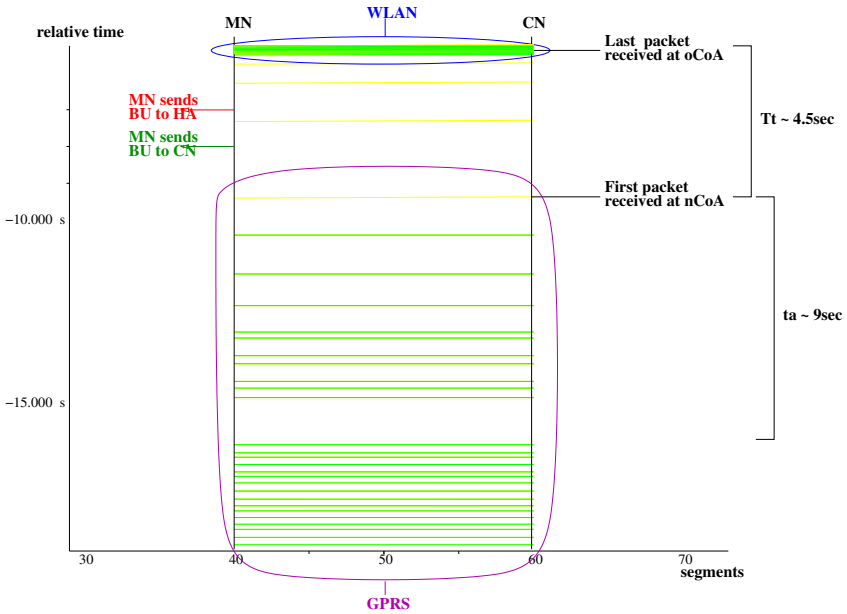


Figure 6. Adaptation component for the test scenario WLAN-to-GPRS

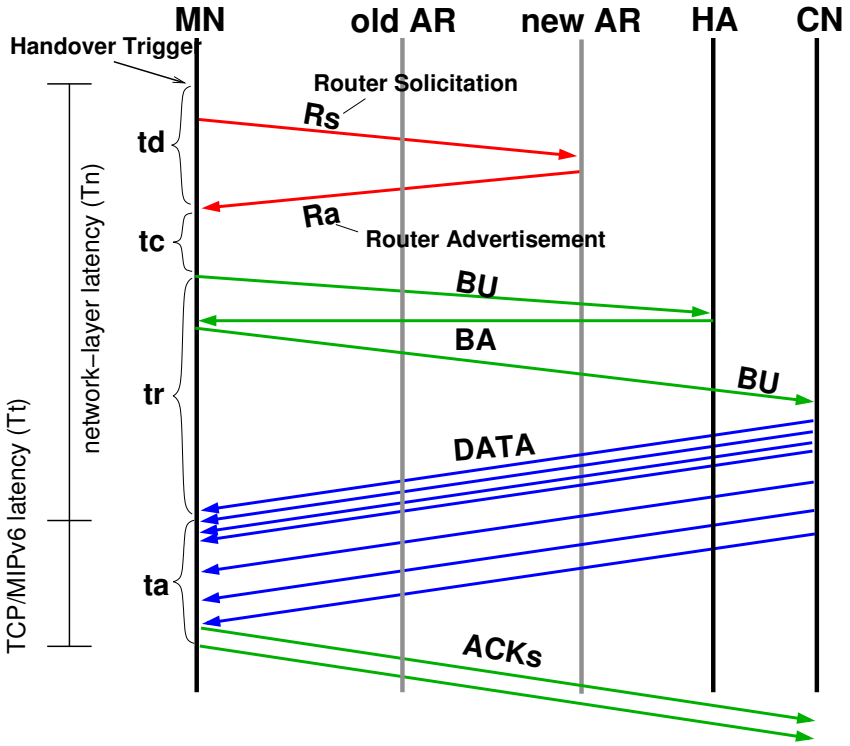


Figure 7. Network and Transport layers' latency partition.

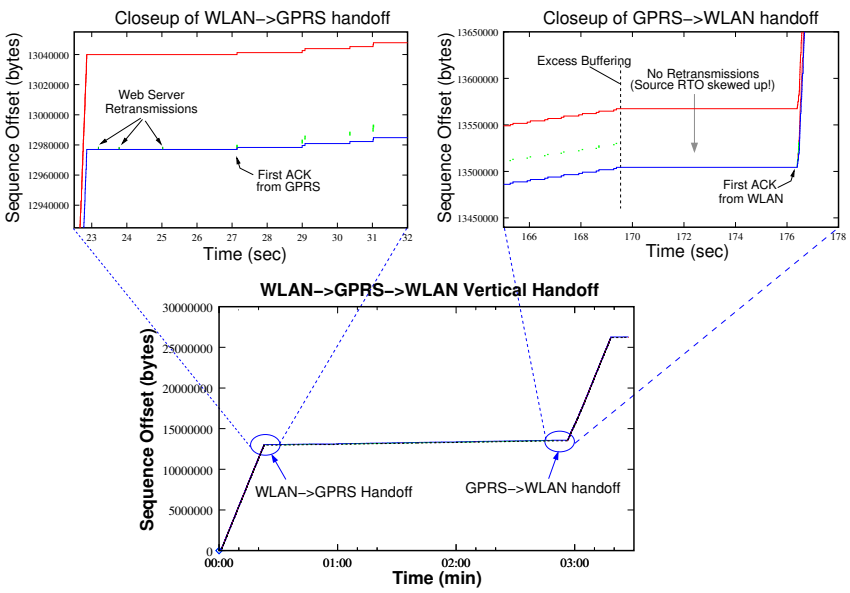
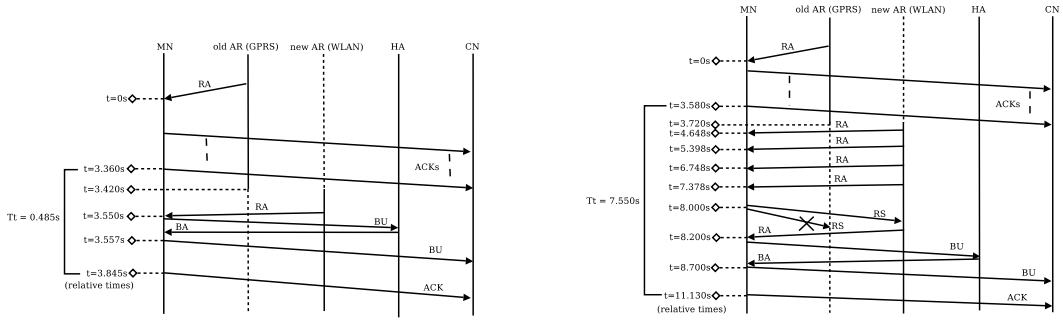


Figure 8. Close-up of a WLAN-GPRS-WLAN handover showing the effects of excess buffering on the latency.



(a) Eager cell switching (b) Lazy cell switching

Figure 9. Effects of the execution method on vertical handover latency.

(a) WLAN to GPRS (b) LAN to GPRS

Figure 10. Total handover time, $T_{handover} = T_n + t_a$

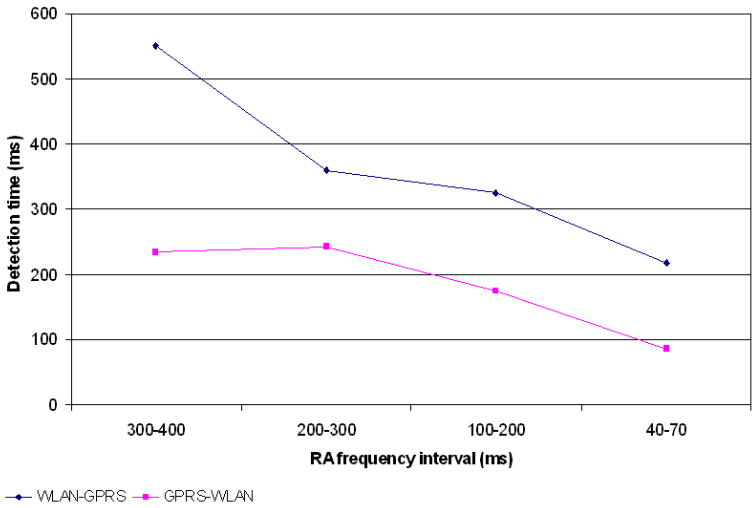


Figure 11. RA frequency variation effects on WLAN and GPRS networks.

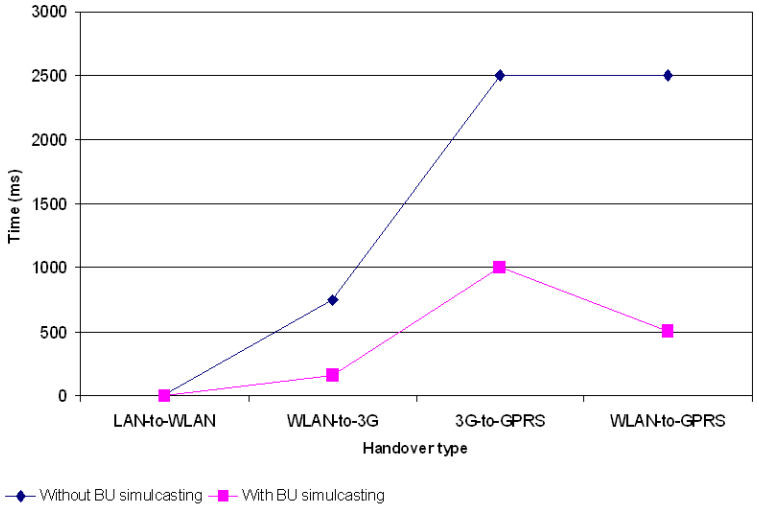


Figure 12. The reduction in (t_d) is calculated pondering the following RTT values: LAN=200ms, WLAN=3ms, 3G=300ms, and GPRS=1000ms.

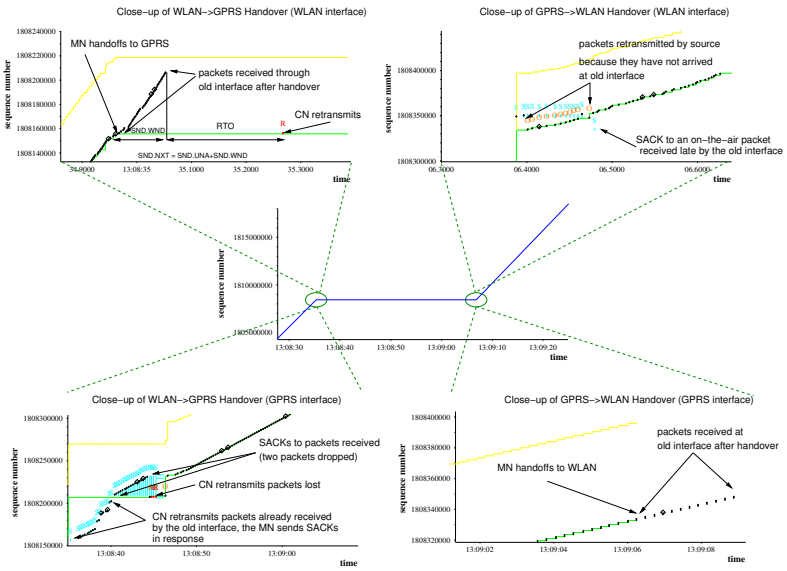
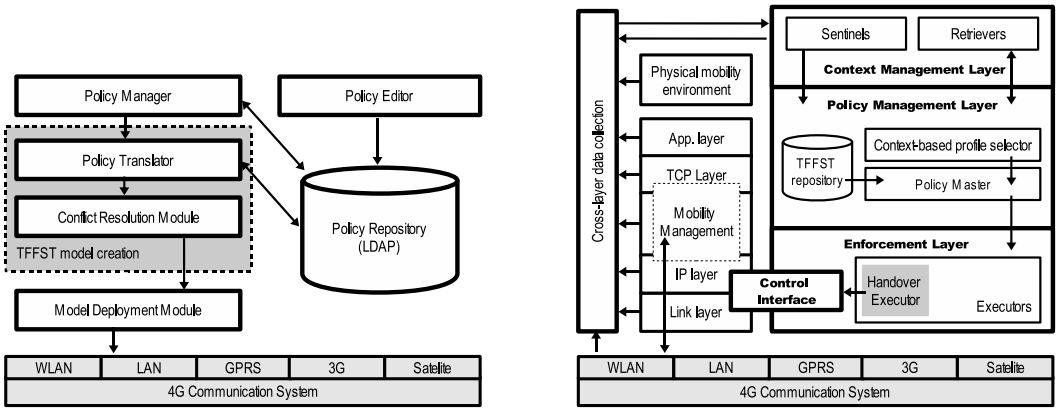


Figure 13. Soft handover between the GPRS network and an IEEE 802.11b access network while downloading a file.



(a) Network-side components (b) Host-side components

Figure 14. PROTON system architecture.

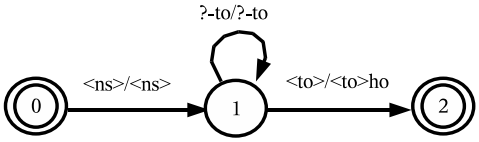


Figure 15. TFFST model for the obligation in Rule 2.