On Inter-network Handover Performance using Mobile IPv6

Rajiv Chakravorty† Pablo Vidales‡ Leo Patanapongpibul‡ Kavitha Subramanian† Ian Pratt† Jon Crowcroft†

†Systems Research Group,
University of Cambridge Computer Laboratory,
William Gates Bldg., 55 JJ Thompson Avenue,
{*Firstname.Lastname*}@*cl.cam.ac.uk*

‡Laboratory for Communication Engineering,
University of Cambridge Engineering Department,
William Gates Bldg., 58 JJ Thompson Avenue,
{*pav25,lbp22*}@*cam.ac.uk*

***Abstract*— In this paper, we report on our practical experiences using the Mobile IPv6 protocol in an integrated LAN-WLAN-GPRS testbed. Through detailed analysis from packet traces of inter-network (vertical) handovers conducted over the testbed, we examine the performance of transport protocols such as TCP during such handoffs. The considerable differences in link-layer characteristics between the different networks reveal a number of performance issues. We show how inter-network handovers can have significant impact on TCP performance, causing connections to timeout and leading to severe underperformance. We highlight several such problems that seem inherent in providing *transparent* mobility in a heterogeneous environment.**

From practical experimentations over our testbed, we discuss how some of these problems can be addressed through exploiting knowledge of the link-layer conditions to improve the handover process. We propose and evaluate schemes such as Fast Router Advertisements (RA), RA Caching, Binding Update (BU) simulcasting to aid Mobile IPv6 protocol during inter-network handovers. We show how a proxy installed in GPRS network for smart buffer management, can improve TCP performance during handovers involving GPRS, and also use of *soft* handovers to improve TCP performance. We demonstrate the benefits from each of these schemes, and conclude with our experiences using Mobile IPv6 to successfully migrate TCP connections during inter-network handovers in an overlay environment.

I. Introduction

There is a strong growth in mobile Internet access, fuelled by the increasing popularity of *WiFi* (*i.e.* IEEE 802.11b-based WLANs), and the worldwide deployment of wide-area wireless networks such as GPRS and third generation wireless (3G). Multi-mode devices (e.g. WLAN-GPRS cards) are becoming increasingly affordable, and thus a growing number of mobile devices such as laptops, PDAs and handhelds are equipped to connect to multiple networks.

Mobile IPv6 can play a key role in integrating these different link-layer technologies, with the promise of enabling *transparent* mobility through use of a unified network layer. When mobile users migrate from the coverage of one network access point to another, they are said to perform a handoff. Most handoffs occur between access points of the same network technology and are termed *horizontal handoffs*. Handoffs between different access points belonging to different networks (e.g. WLAN to GPRS) are referred to as *vertical handoffs*, and pose a significantly greater challenge.

Transparent mobility aims to enable mobile users to seamlessly move across networks, wired as well as wireless, with minimal disruption to packet flows. A mechanism that can enable this has to exhibit a low handoff latency; incur little or no data loss (even in highly mobile environments); scale to large inter-networks; adapt different applications to the networks environments; and finally act as a conjuncture between heterogeneous environments and technologies without compromising on key issues related to security and reliability.

In reality this is difficult to achieve. Wireless (and wired) technologies offer links that have widely varying link characteristics. Current generation cellular networks such as GPRS and 3G offer bandwidths that are much higher than those of their predecessors, however, they are still significantly lower than WLANs. These wide-variabilities in link-layer characteristics can pose a serious impediment to providing transparent mobility in an overlay environment.

A. Research Contributions

In this paper, we investigate the extent to which Mobile IPv6 can be used to successfully migrate TCP connections during inter-network handovers. To understand the performance issues in inter-network handovers, we first characterize a handover process in Mobile IPv6 in two main steps – a handoff *decision* and *execution*. We discuss how a handoff decision is completely independent from its execution, with execution contributing towards overall handover latency. In this paper, we will focus on the handoff execution process using Mobile IPv6 by breaking it into three components – detection, configuration and registration, each of which contributes to the overall handover latency.

We describe our implementation of a fully-integrated Mobile IPv6 based LAN–WLAN–GPRS testbed. By using a testbed, consisting of the world's two most widely deployed wireless data networks – local-area wireless network (WLANs) and wide-area wireless (GPRS) – we analyse what happens when multi-mode mobile devices perform vertical handoffs using Mobile IPv6. We closely examine the handover process itself, and its effects on TCP, and also give reasons for its under-performance.

We show how TCP performance during inter-network handover can be improved using schemes that can help minimize handoff execution latency components during handovers. We experimentally evaluate schemes that improve vertical handovers – Fast Router Advertisements (RAs), RA Caching, and Binding Update simulcating in Mobile IPv6, smart buffer management using TCP proxy involving GPRS networks, and *soft* handovers that improves TCP performance dramatically. We believe this to be the first work that has attempted to practically evaluate the performance of inter-network handovers using Mobile IPv6 in this environment.

B. Paper Outline

The rest of the paper is structured as follows. The next section gives a brief overview on Mobile IPv6. Section III characterises the inter-network handoff process in Mobile IPv6, while Section IV describes a fully-integrated Mobile IPv6 based LAN–WLAN–GPRS testbed. Section V present results from vertical handovers, while Section VI proposes and evaluates the efficacy of several schemes that improve vertical handover performance. Section VII describes related work available in the literature, and the last section concludes the paper.

II. Mobile IPv6 Overview

Perkins introduced Mobile IP (MIPv4) [5], which has now become a principle driver to enabling host mobility, mainly due to the continous expansion of the mobile Internet, and specifically due to its compatibility with IP. In more recent developments, Mobile IPv6 (MIPv6) has been proposed, that overcomes several shortcomings in MIPv4, offering a larger address space, route optimisation, and improved security [6].

Unlike MIPv4 (without route optimization), MIPv6 is inherently optimized by using a direct notification mechanism to the nodes that know and route packets to the mobile node's new location. Every mobile node (MN) has a home network and is identified by a home IP address on that network. The 128-bit IPv6 address consists of a 64-bit routing prefix, which is used for routing the packets to the right network, and a 64-bit interface identifier, which identifies the specific node on the network and can essentially be arbitrary. Thus, IP addresses in MIPv6 can identify either a node or a location on the network, or even both.

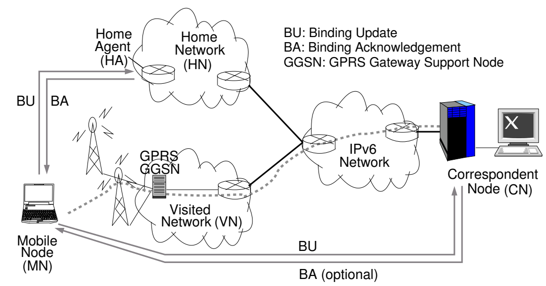

Fig. 1. Mobile IPv6 Overview.

A router in the home network called a *home agent*, acts as the mobile node's trusted agent (since both share security associations) and forwards IP packets between the mobile's *correspondent nodes (CN)* and its current location, identified by the care-of address (CoA). The MIPv6 protocol also includes a location management mechanism using Binding Updates (BU). When a mobile node changes it current address, it can send BUs to its correspondents as well as home agent to notify them about the new location so that they can communicate directly (figure 1). The mobile may also be triggered to send a BU if it receives a packet from a new correspondent via the home agent. All nodes, whether mobile or stationary are recommended to understand the Binding Request option, enabling the packets destined to the MN to be efficiently routed without going through the HA.

The mobile node and its home agent can share a preconfigured security association for encrypted and authenticated communication. However, since binding updates to the correspondents are not authenticated, the latest MIPv6 specification [6] make use of the "return routability procedure" to verify the authenticity of the mobile node establishing a new binding entry with the correspondent node. Other mobility signalling and security features are already part of the MIPv6 protocol as header extensions [6].

III. Characterising the Handoff Process

In this section, we investigate the intricacies of Mobile IPv6 handoffs as typically applied to hybrid wireless networks.

Wireless networks such as GPRS and WLANs can be arranged as an overlay on the basis of their offered coverage; such an arrangement is known as a wireless overlay network [1]. For instance, GPRS or 3G can provide country-wide or continent-wide coverage, while 802.11b-based WLANs provides only local-area wireless coverage. However, wireless links in the wide-area offer substantially lower bandwidths. Depending upon the coverage offered a mobile device may opt to vertically handoff between these networks.

When a mobile device performs a handoff from the network providing faster but smaller coverage (e.g. WLANs), to a new network higher up in the overlay (e.g. GPRS), the handoff is called an *upward* vertical handoff. On the other hand, if the new network to which the mobile device handoffs provides only local coverage, it is known as a *downward* vertical handoff. As we shall see, the direction of the handoff is also important to enable appropriate adaptation during vertical handovers in wireless overlay networks.

We characterize handovers in two main steps: a *handoff decision* process and a *handoff execution* process.

A. A Handoff Decision

Handoff decision is the ability to decide *when* to perform a handoff. This can be performed either by the mobile host (mobile-controlled handoffs), or the network (network-controlled handoffs), or even jointly in cooperation by both (mobile-assisted handoffs) [7]. However, unlike horizontal handoffs, a vertical handoff decision involves an *if* as well as *when* decision.

Whereas in a horizontal handoff a mobile device may generally handoff while moving from the coverage of one access point to another (based on signal strength), the same may not necessarily hold for vertical handoffs. A decision to vertically handoff may depend on several issues relating to the network to which it is already connected and to the one which it is going to handoff. For instance, the decision to perform mobile-controlled handoffs may be made by a vertical handoff *agent*, sitting in the mobile device based on policies such as network bandwidth, load, coverage, cost, security, QoS, or even user preference (see, [2]).

User preference is important when performing vertical handoffs. For instance, if the new network to which a mobile device performs a handoff does not offer security, the user may still decide to use the old network. Depending upon coverage, a user may wish to use a secure and expensive link for his official e-mail traffic (e.g. using GPRS), but may still opt for a cheaper link to access web information (e.g. WLAN).

In a wireless overlay network, handoffs may be *anticipated* or *unanticipated*. Anticipated handoffs are usually used to optimize horizontal handoffs, for example in WLANs [17], based on link-layer (L2) triggers during movements of a mobile device. These L2 triggers indicate coverage status of the new network. But when applied to vertical handoffs, L2 triggers can indicate to a mobile device or an access router (depending upon whether it is mobile or network controlled handoff) that it is now under the coverage of the new network (e.g. WLAN) and it *may* wish to execute the handoff. Note that all upward vertical handoffs can be anticipated, which means that a mobile device will always want to handoff to a network higher in the overlay, based on an L2 trigger (e.g. switching to GPRS after receiving an 'L2 down' event from a WLAN).

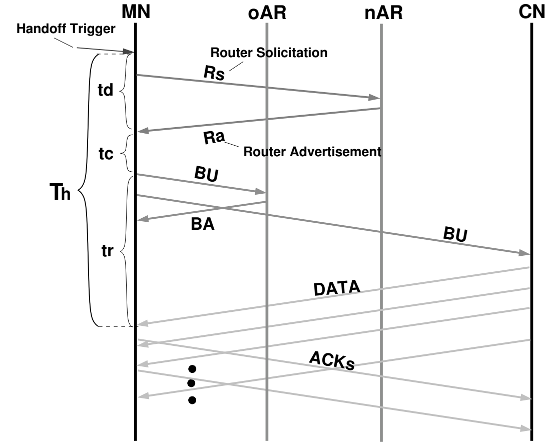

Fig. 2. Partitioning the Handoff Latency in Mobile IPv6.

In constrast, downward vertical handoffs can be anticipated or unanticipated. If anticipated, it can be based on L2 triggers from the new network (e.g., 'L2 up' from WLAN). On the other hand, unanticipated downward vertical handoffs would typically assume that a mobile device is already under the coverage of the new network, but is yet to *execute* the handoff. For instance, a user may prefer to postpone a handoff based on requirements of his applications, and may execute a handoff later, being already aware of the coverage status of the new network.

B. Handoff Execution Process

Handoff execution assumes that the mobile device is under the coverage of the network it is going to handoff to. Three steps characterize the handoff execution process:

Detection Time (t_d). It is the time from when a mobile node is under the coverage of a new wireless access network to the instant it receives a router advertisement from the new access router. When the mobile is under the coverage of the new network, it can detect this coverage using (1) trigger-based router solicitation or, (2) wait to receive a router advertisement from an access router in the visited network.

In figure 2, we show a mobile node (MN) pro-actively sending a router solicitation (Rs). In response, it receives a router advertisement (Ra) from the access router in the visited network. Using such a scheme, a multimode device can immediately receive a Ra to configure its new CoA. However, this requires a trigger for generating Rs

either at the link-layer or from the user. Alternatively, it may wait for an Ra. According to the latest specification, this router advertisement interval can be a minimum of 30ms to a maximum of 70ms [6].

Configuration Time (t_c). This is the interval from the time a mobile device receives a router advertisement, to the time it takes to update its routing table, and assign its interface with a new CoA address based on the prefix of the access router available from the router advertisement.

Registration Time (t_r). This is the time required to send a binding update to the home agent and correspondent node, and receive the first packet from the correspondent node assuming that a binding acknowledgment from the home agent was received beforehand. Note that MIPv6 does not specify waiting for a binding acknowledgement from a correspondent, as it is optional, hence, we only consider the case when a mobile node receives a packet from the correspondent.

Based on its original definition used in [3], we define vertical handoff latency for a mobile host as the amount of time to initiate disconnection from the old network access point to receive the first packet from the new network access point. Thus, from the discussion above, the total handoff latency (T_h) can be given by (see figure 2):

$$T_h = t_d + t_c + t_r$$

The handoff equation suggests that schemes for optimizing vertical handoff latency would essentially involve optimizing t_d and t_r, since t_c typically depends upon the computing capability of the mobile device. The detection time t_d can be minimized by increasing router advertisement frequency, or by using link-layer triggers from the respective network in order to solicit router advertisement from access routers in the visited network. We will show the tradeoffs involved in optimising the detection time.

The t_r component, however, is a function of the latency of the link to which the mobile node will handoff to. For instance, upward vertical handoffs (e.g., WLAN→GPRS) would typically involve updates being sent over GPRS. As link latencies over GPRS are high when compared to WLANs, the overall handoff time in this case may become significant. In section V, we will show how registration times during vertical handovers can be influenced by the other factors besides link RTT.

IV. Mobile IPv6-based LAN-WLAN-GPRS Testbed Implementation

In this section, we describe our experimental testbed. As part of the Cambridge Open Mobile Systems (COMS) project [16], we have implemented a Mobile-IPv6 based LAN-WLAN-GPRS testbed (figure 3). In this testbed, 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 located at different locations of the William Gates Building housing the University of Cambridge Computer Laboratory and the Laboratory for Communication Engineering.

The 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 SGSN and GGSN node is 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 wireless testbed, mobile nodes (e.g. laptops) connect to the local WLAN network and also simultaneously to GPRS via a PCCard modem. The mobile node's MIPv6 implementation is based on that developed by the MediaPoli project [14], chosen for its completeness and open source nature.

Additionally for our testbed, we have brokered a semi-permanent IPv6 subnet from BTExact's IPv6 Network, which connects us to the 6BONE [15]. Using the address space, we are able to allocate static IPv6 addresses to all our IPv6 enabled mobile nodes. A router in the lab acts a IPv6/IPv4 tunnel end-point to the BTExact's IPv6 network (shown in figure 3). This router is also an IPv6 access router (Home Agent) for the lab's fixed-internal IPv6-enabled network and also for internal WLANs.

Special arrangements has been made to support routing GPRS/WLAN traffic through the internal network. Routing in the Lab has been configured such that all GPRS/WLAN user traffic going to and from mobile clients are allowed to pass through the internal router, enabling us to perform traffic monitoring. The arrangement assists us in analysing traffic traces to accurately replay the TCP connection timelines, as we show in next section, during vertical handoffs.

Since the GPRS cellular network currently operates only on IPv4, we use a SIT (Simple Internet Translation) to tunnel all IPv6 packets 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.

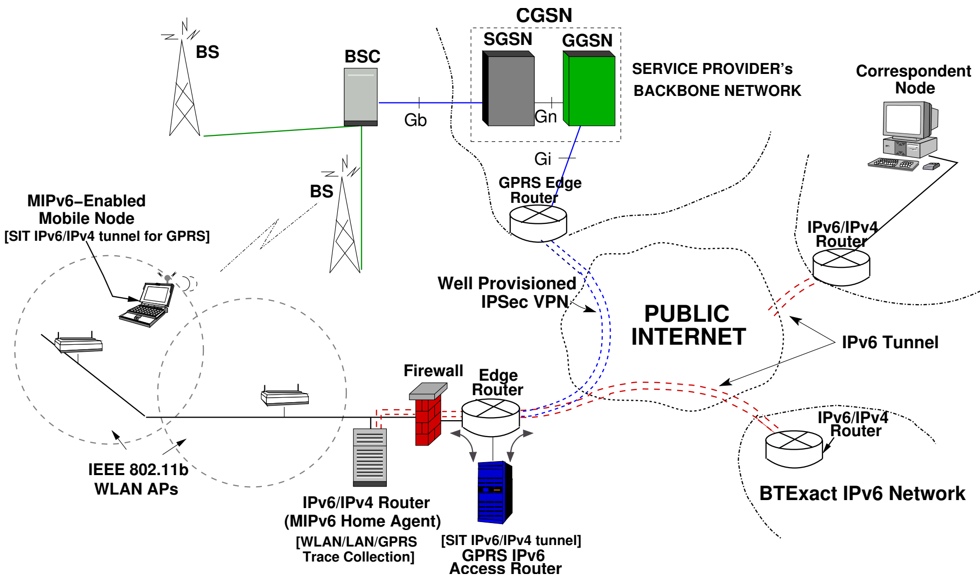

Fig. 3. Mobile-IPv6 based LAN-WLAN-GPRS Testbed Implementation.

V. Testbed Measurements and Analysis of Vertical Handoffs

For the tests we conducted, handoffs were forced from WLAN (and LAN) to GPRS and vice versa. For testing handovers, file downloads were initiated by the multimode mobile device (mobile node) over WLAN from an internal web-server (correspondent node), and then forced an inter-network handoff to GPRS and back again to WLAN.

We operated the test-bed under following conditions:

- Network discovery for the mobile node was performed based on router advertisements and not based on layer-2 triggers, which means the mobile node doesn't generate a router solicitation (R_s), but instead, waits for the first router advertisement (R_a) from the access router of the visited network.
- Unless otherwise specified, all access routers including the home agent are set to multicast router advertisements in accordance to the recommended values specified by the neighbour discovery protocol (RFC 2461 [4]).
- For all cases, a vertical handoff assumes that the multimode mobile device has all of its network interfaces (LAN/WLAN/GPRS) powered on simultaneously to reduce the initialization time. This does not necessarily mean all interfaces are linked to their respective networks.
- All hosts in these tests run Linux 2.4.16. For GPRS we make use of a Motorola T260 GPRS mobile, which is a "3+1" (3 downlink, 1 uplink channels) handset.

In the testbed, we allow all traffic to pass through an intermediate router, and simultaneously monitor the traffic (using `tcpdump`) being sourced from the web-server to the mobile device during all active data sessions. As mentioned earlier, the internal router is also the IPv6 access router for the WLAN, and there is a separate GPRS access router (logically co-located to the GGSN), that acts as an access router for the GPRS network. These routers were set to advertise router advertisements randomly between 3 to 10 seconds as typically restricted by [4].

A. Evaluation of Vertical Handovers

We consider two different cases in these tests for vertical handoffs: the first case between LAN↔GPRS, while the second one between WLAN↔ GPRS.

For each of these tests, we initiated a file transfer about 25MB size from the mobile node in WLAN (and LAN), and forced a handoff to GPRS, and again a handoff back to WLAN (and LAN). We collected the `tcpdump` traces of the handoff in an internal router as well as in the mobile client. Since the latest version of `tcptrace` for analysing TCP traces does not provide support for MIPv6 so we had to add it (`tcptrace+` [18]).

Figure 4 shows a vertical handoff between LAN→GPRS and back again from GPRS→LAN. As evident from the close-up plots of the vertical handoff in figure 4, we find LAN→GPRS handoff to take around 6s, and also about 6s to handoff for the GPRS→LAN case.

Within this interval, as evident from the LAN→GPRS handoff close-up (top-right) of figure 4, the TCP data session at the source (web server) times out and backs-off exponentially. It retransmits 4 times before it finally receives an ACK from the mobile device, but this time from the GPRS link piggybacking a binding update sent directly to

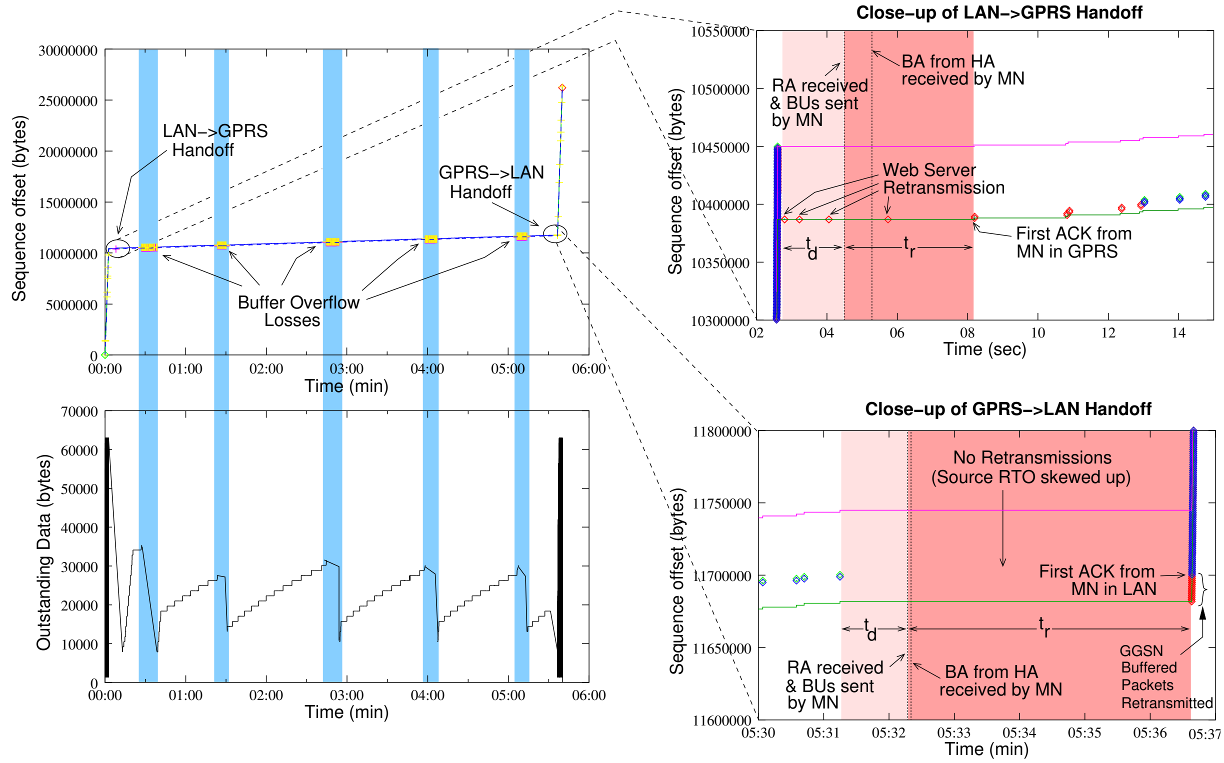

Fig. 4. Practical experience with Mobile IPv6 based vertical handoffs in a tightly-integrated LAN-WLAN-GPRS testbed. Left half of the figure 4 (top-down) show the sequence-time and outstanding data plots (using tcptrace+ [18]) in vertical handoffs that involved transfer of a 25MB file over LAN, with a vertical handoff to GPRS and then a reverse handoff back to LAN. Right half (top) of figure 4 show the close-up of LAN→GPRS handoff, while bottom-tight show the same for GPRS→LAN handoff.

the web server. The transmission between web-server and the mobile device resumes soon after it starts using the GPRS link.

For the second case shown in figure 5, a handoff was conducted between WLAN→GPRS and also back from GPRS→WLAN. We find that it takes around 4s to handoff from WLAN→GPRS, and in this case the TCP data session backs-off at the server end, to re-transmit 3 times before an ACK piggybacked along with the binding update is available from GPRS. The handoff from GPRS→WLAN takes about 7s, which is also quite similar to the handoff latency observed for the LAN→GPRS case.

Notice that the right-bottom close-up plots in figure 4 and 5 for both GPRS→LAN and GPRS→WLAN handoff does not indicate any TCP retransmissions from the source (web-server) for a substantial duration. This is not suprising, as the phenomenon can be explained based on the amount of excess buffering offered by the current GPRS networks [12].

Most GPRS networks provide a substantial amount of buffering (per-mobile node) in their GGSN nodes, observed at up to 200KB [12]. However, the UK's Vodafone GPRS network has recently been reconfigured to reduce the allocation per-mobile to about 30KB. Long-lived TCP sessions will progressively increase their congestion window until they exceed this threshold, experience loss and then recover using fast re-transmit (halving their congestion window).

However, 30KB is rather more than the bandwidth delay product of the GPRS downlink. The resultant queuing at the GGSN leads to the source's RTO (TCP's Retransmission Timeout value) becoming inflated. Thus, as we can see from figure 4 and 5 (right-bottom) the source experiences a substantial handoff latency. In fact, the first packet after the handoff is available to the mobile host only after the web-server has timed-out to retransmit, and that it eventually retransmits all in-flight packets that were actually lost during the handoff process over GPRS. Notice again that the extent of this packet loss is proportional to the buffering caused in the GPRS GGSN at the time of the handoff, which unfortunately, is very high for any long-lived TCP session over GPRS.

This is certainly not what is expected, and it clearly shows how excess buffering in GPRS can have dire consequences for TCP flows during a vertical handoff, from GPRS→WLAN as well as GPRS→LAN. However, once the source times-out, it retransmits aggressively to rapidly increase its congestion window soon after it starts receiving packets from WLAN (and LAN), so as to quickly normalize its RTO values. The bottom-right close-up plots of figure 4 and 5 show how the sequence trace shoots-up

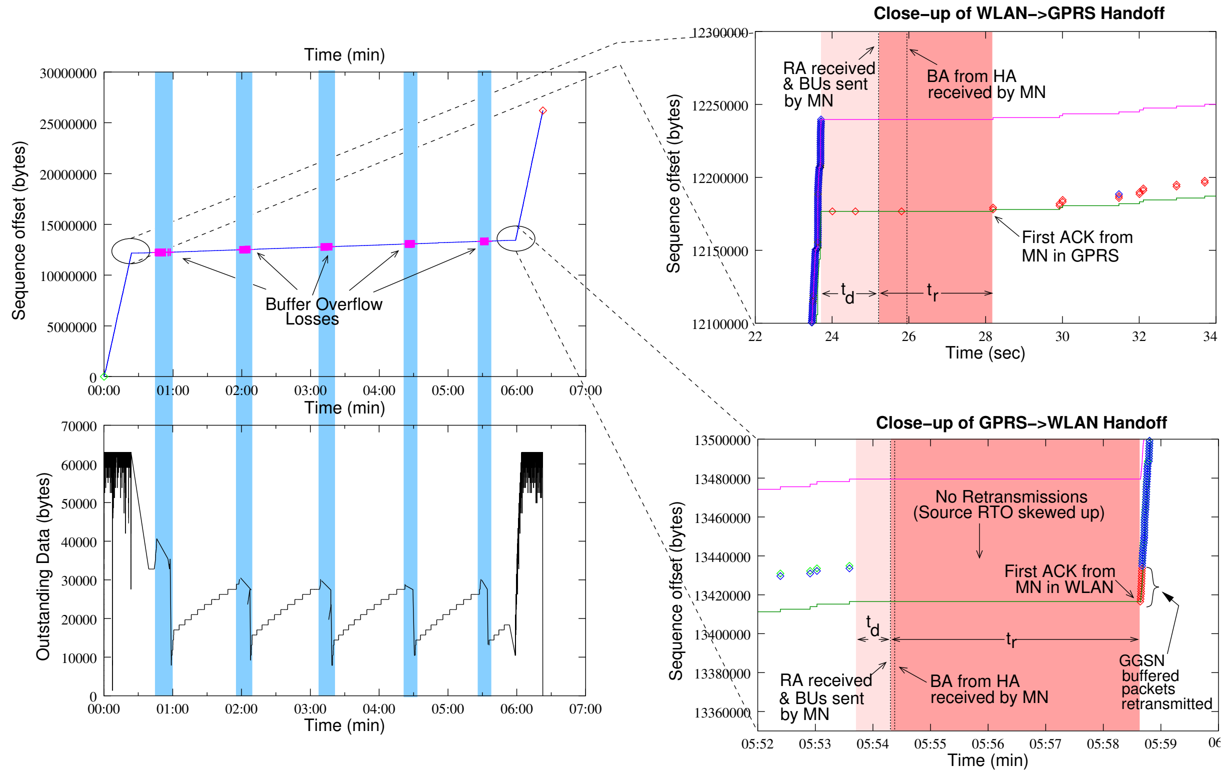

Fig. 5. Left half in figure 5 shows (top-down) the sequence offset, and outstanding data plots (using tcptrace+ [18]) in a vertical handoff that involved transfer of a 25MB file over WLAN, with a vertical handoff to GPRS and then a reverse handoff back to LAN. Right half (top) of figure 4 show the close-up of WLAN→GPRS handoff, while bottom-tight show the same for GPRS→WLAN handoff.

soon after the handoff to LAN and WLAN respectively.

B. Partitioning the Handoff Latency

Analysis of the GPRS↔WLAN and GPRS↔WLAN vertical handoffs from figure 4 and 5 show only the total vertical handoff latency. However, we can also partition overall handoff latency into its components. In this section, we perform handoff tests for over 10 trials in order to partition the handoff latency.

As discussed earlier, total handoff latency is sum of the detection time (t_d), configuration time (t_c), and registration time (t_r). t_d depends on the router advertisement frequency. In these tests, however, we set the router advertisement frequency to vary randomly between 1s (minimum) to 3s (maximum). Ideally, the values specified in [6] could be used.

Table I and II give the handoff latency partition. For the WLAN→GPRS case, we find a mean t_d of around 808ms, while GPRS→WLAN gives a t_d of about 2241ms. These values for t_d, can show high variability due to the frequency of the router advertisements.

Registration time (t_r) is a function of the link-layer characteristics; hence, these values should be typically higher when executing upward vertical handoffs (e.g. WLAN→GPRS) as binding updates are propagated over the GPRS links that have high RTTs.

However, table I and II show almost similar registration times (t_r) for the WLAN→GPRS as well as the LAN→GPRS case, when compared to the GPRS→WLAN and GPRS→LAN case respectively. Ideally, since WLAN and LAN offer links that have low RTTs, the registration times during GPRS→WLAN and GPRS→LAN handoff should have been lower.

Nevertheless, as we have explained earlier, because of the excess buffering in the GPRS GGSN, the source perceives inflated RTTs, and hence, a skewed-up RTO. Consequently, in many cases that we have observed, the registration process could finish only when the source had to eventually retransmit with a high value of RTO. As such, this results in high vertical handoff latency for both GPRS→WLAN and GPRS→LAN handoff case.

Also notice that the standard deviation in the total handoff latency for both GPRS→WLAN and GPRS→LAN are higher than those for WLAN→GPRS and the LAN→GPRS case. This variation in the handoff latency for GPRS→WLAN and GPRS→LAN case, is due to the variability as seen in the amount of excess buffering at the GPRS GGSN during handoffs as discussed earlier. The impact of this is that it can give high variability in the RTTs perceived by the source (web-server), consequently leading to significant variations in its RTO calculation, and hence, variable handoff latencies.

WLAN↔GPRS Handoff	WLAN→GPRS				GPRS→WLAN			
(the split in ms)	Min	Mean	Max	Std. Dev.	Min	Mean	Max	Std. Dev.
Detection Time (t_d)	200	808	1148	304	739	2241	3803	919
Configuration Time (t_c)	0.853	0.870	0.890	0.009	0.380	1.062	1.186	0.233
Registration Time (t_r)	2339	2997	3649	395	2585	4654	7639	1611
Total Handoff Latency (T_h)	3323	3806	4438	310	5322	6896	8833	1118

TABLE I
HANDOFF LATENCY PARTITION (IN MS) FOR WLAN↔GPRS TAKEN OVER 10 RUNS

LAN↔GPRS Handoff	LAN→GPRS				GPRS→LAN			
(the split in ms)	Min	Mean	Max	Std. Dev.	Min	Mean	Max	Std. Dev.
Detection Time (t_d)	347	1168	2070	497	0.734	2058	3257	987
Configuration Time (t_c)	0.868	0.967	1.187	0.116	0.681	0.779	0.925	0.091
Registration Time (t_r)	2299	3308	4759	632	2357	4467	7183	1387
Total Handoff Latency (T_h)	2805	4996	5107	561	4011	6525	8196	1176

TABLE II
LATENCY PARTITION (IN MS) FOR LAN↔GPRS HANDOFFS OVER 10 RUNS

The configuration time (t_c), which depends upon the host computing capability and state of the interface (for e.g. up, down, suspend) give considerably low values. Other factors also contribute to the handoff latency, for example, tunneling IPv6 packets over an IPv4 network adds to the some overhead.

VI. IMPROVING HANDOVER PERFORMANCE

In this section, we attempt to improve handover performance by focussing on the detection time (t_d) and the registration time (t_r), and finally using 'soft' handovers.

A. *Fast Router Advertisements*

While the neighbour discovery protocol RFC 2461 [4] specifies a random router advertisement interval between 3s and 10s, this interval is large given the impact detection interval can have on overall handoff latency. By reducing the router advertisement interval, we can improve the detection time and overall handoff latency.

Its is interesting to note that the latest IETF draft on Mobile IPv6 takes this issue into account, and specifies a much shorter interval between 30ms (`MinRtrAdvInterval`) to 70ms (`MinRtrAdvInterval`) for access routers using MIPv6[6]. However, increasing RA interval should also take into account the resultant overhead caused, as there will now be a trade-off involved; increasing RA frequency can result in substantial overhead, especially over 'long-thin' links such as GPRS.

In order to evaluate the effects of RA frequency on the handoff detection time, in both WLAN→GPRS and GPRS→WLAN, we modified the Linux IPv6 RA deamon (`radvd+` [18]) to support different RA interval values including one specified by the latest Mobile IPv6 draft [6].

Table III shows the effect of varying RA interval on mean handoff detection time (t_d) taken over 5 runs. In these tests, we started file downloads and then forced handoffs between GPRS↔WLAN, keeping the RA frequency in one link constant, while varying the other and vice versa. We collected `tcpdump` traces at all network interfaces at the client-side and also at the internal router. What one would expect is that when the RA interval is reduced, the detection time spent waiting for the RA to be heard would also reduce. Based on the average values, we do find that as we increase the RA frequency in WLAN, the mean detection time reduces.

The case is somewhat different for GPRS. As we increase the RA frequency in GPRS, the mean detection time does not show substantial improvement when compared to WLANs. Though the best case values are still encouraging, analysis from `tcpdump` traces in many cases show RAs being excessively delayed, and then quickly arriving in bunches in GPRS along with other TCP data packets. This phenomenon is not unusual, as RA packets can oftentimes experience highly variable delays, as we have already shown from GPRS link characterization [13]. High and variable delays in GPRS is due to the link-layer [13], and also to some extent due to the lack of proper buffer management strategy in current GPRS networks.

Another interesting aspect is the network overhead caused by the increase in the RA interval in GPRS, which also gives the trade-off between any improvement in handoff latency achieved to the worsening network overhead. RA overhead caused by significantly increasing the RA frequency can lead to substantial overhead in GPRS – an RA interval set at 30-70ms (as per the latest MIPv6 draft [6]) will lead to about 25-50% overhead in terms

RA Interval (MinRtrAdvInterval-MaxRtrAdvInterval)[6]	WLAN→GPRS (Fixed RA in WLAN: 300ms-400ms) Detection Time (t_d) Mean(best-case) (ms)	RA Overhead in GPRS(4KB/s) (BW overhead ratio in %)	GPRS→WLAN (Fixed RA in GPRS: 300ms-400ms) Detection time (t_d) Mean(best-case)(ms)	RA Overhead in WLAN (1.2MB/s) (BW overhead ratio in %)
300ms-400ms	551.33(146.10)	4.75% - 6.33%	234.44(39.92)	0.0157% - 0.0210%
200ms-300ms	360.88(187.16)	6.33% - 9.5%	142.70(69.59)	0.0210% - 0.0317%
100ms-200ms	324.82(41.82)	9.5% - 19%	174.88(95.90)	0.0317% - 0.0633%
40ms-70ms	406.35(44.04)	27.5% - 47.5%	85.91(44.20)	0.0917% - 0.1583%

TABLE III

EFFECT OF MOBILE IPV6 FAST RA ON MEAN HANDOFF DETECTION TIME AND ON THE NETWORK OVERHEAD. PRACTICAL BANDWIDTH (BW) VALUES IN GPRS ('3+1' PHONE APPROX. 4KB/S), 802.11B WLAN (APPROX. 1.2MB/S)

of actual bandwidth (for e.g., when using '3+1' GPRS phone) for 'long-thin' links such as GPRS.

Based on the results from these experiments, we feel that although increasing RA frequency does help improve detection time in WLANs, it may not be the best option for networks such as GPRS. Not only is there a costly trade-off involved due of the additional RA overhead, but also the use of fast RAs is not necessarily a 'guarantee' in reduction of the handoff detection time. Based on such trade-off issues involved, we feel that the RA interval can be set anywhere between 0.5-1s in a GPRS IPv6 access router, which is about half the observed average GPRS RTT, and that also ensures that the resultant RA overhead is negligible.

B. *Client-based Router Advertisement (RA) Caching*

Fast router advertisements as discussed above, minimize the detection time by increasing RA frequency. Waiting for RA to arrive means that certain amount of time will be expended before a mobile host can detect and receive the RA, and then configure its interface with a new CoA. However, it is possible to further improve handoffs by eliminating the detection time altogether.

One useful technique to eliminate t_d is by caching router advertisements. Using RA caching, we can eliminate detection time in L3 handover altogether. For example, during unanticipated handoffs, the decision to handoff typically depends on the user. In this case, a user handing off from GPRS→WLAN may want to complete an ongoing session over GPRS link, and later decide to hand-off to WLAN. In this case, the handoff will not be initiated immediately upon reception of L2 trigger from WLAN, but will wait for a handoff decision triggered by the user. However, any RA received during this period can still be cached, so that when the decision to handoff is taken, the detection time for RA lookup during handoff execution is eliminated, further improving handoff performance.

In the anticipated handoff case, RA caching has only limited benefits. Apparently, benefit of RA caching is available only for the *upward* vertical handoff case. As the network higher in the wireless overlay (e.g., GPRS) is more omnipresent, RAs from GPRS can be cached a priori, and need not wait (or even sought) during handoff (e.g., WLAN→GPRS) leading to complete elimination of the detection interval.

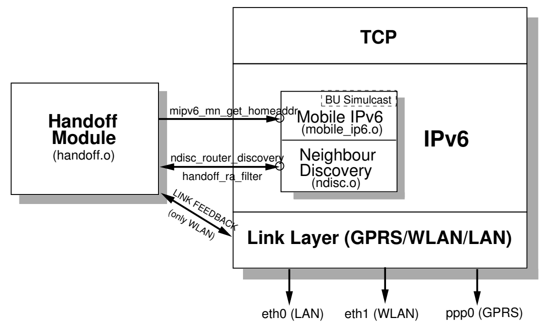

Fig. 6. Mobile Client-based Handoff Module for RA caching.

We have implemented a client-based handoff module in Linux 2.4 for RA caching that completely eliminates the inter-network handover detection time [18]. Using such a handoff module, we have already shown the benefits of eliminating detection time in 802.11b-based horizontal handovers [10]. Here, we extend our handoff module to support inter-network handovers.

Figure 6 shows the handoff module (`handoff.o`), which caches RA from different networks. In this implementation, the handoff module is hooked to the neighbour discovery module (`ndisc.o`) of the IPv6 stack. RAs from different networks are first received by neighbour discovery module of the IPv6 stack, which are then passed to handoff module. The handoff module then checks if RA from the same network is already cached, and makes an update if it has expired. Currently, the handoff module uses a periodic (user configurable) timer, that performs automatic handovers between two different networks (by calling the `ndisc_router_discovery` function of the IPv6 stack). In the current implementation, the handoff module also makes use of the MIPL's Mobile IPv6 module during handovers.

We have evaluated the performance of our handoff module for RA caching. In these tests, we allowed RAs to be cached in WLAN as well as the GPRS network, and then forced handoffs between GPRS↔WLAN periodically, by setting the timer in the handoff module, while simultaneously downloading a file from the server.

Scheme used	Mean Detection Time (t_d in ms)
Mobile IPv6 (Fixed RA: 300-400ms)	551.336ms
Mobile IPv6 (with Handoff Module)	1.21ms

TABLE IV
MEAN DETECTION TIME WITH AND WITHOUT RA CACHING.

Table IV shows the mean detection time with and without the handoff module. We do find that the amount of time it takes to detect an RA from the handoff module cache to be negligible (effectively zero but for processing RA from the cache), typically of the order of few milliseconds, when compared to the mean detection time when not using the handoff module. Thus, RA caching leads to substantial improvement in handoff detection time.

C. Smart Buffer Management using TCP Proxy

As we have shown in the previous section, the high registration times (t_r) during GPRS→WLAN (and GPRS→LAN) handoff is due to the excessive data buffering caused by a long-lived or number of TCP flows in the GPRS GGSN per-mobile, which can lead to inflated RTTs and hence a skewed-up RTO. It is clear that by preventing the source from excessively buffering data in GPRS, one can avoid the source RTO from being skewed-up, and thus, improve TCP's performance.

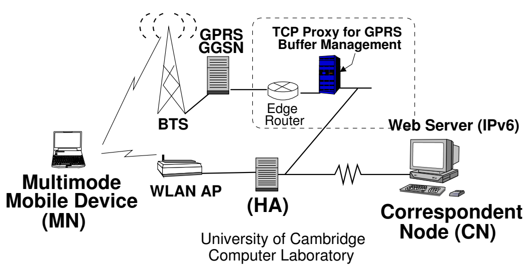

Fig. 7. Experimental Test Bed using TCP proxy.

One way to achieve this, which comes as a significant benefit without modifying the TCP at either fixed and mobile host end-systems, is by using proper buffer management in the GPRS networks. This is possible by installing a transparent TCP enhancing proxy in the GPRS network, that adapts TCP connections that are shipped downwards to the mobile device. This proxy prevents any source from excessively expanding its window to buffer any excess data in GPRS at all.

Fortunately, we are able to reuse a transparent TCP proxy designed to achieve a similar purpose from an earlier piece of our research [13]. The proxy performs *transparent* flow aggregation for TCP flows destined to mobile devices, with receiver window based flow-control with remote hosts to prevent excess queuing in GPRS GGSN from occuring. By limiting the aggregate TCP window to only slightly higher than effective bandwidth-delay product of the downlink, the proxy is able to prevent excess queuing, and thus, is able to avoid source RTT (and RTO) inflation.

Though our proxy was originally implemented for IPv4, we modified it to understand tunneled IPv6-over-IPv4 traffic. Our proxy runs on Linux `netfilter`, that diverts tunneled IP packets to the user space using Linux `ip_queue` for proxy-specific adaptation.

MIPv6 Scheme used	Mean Registration Time (t_r in secs) (only GPRS→WLAN handovers)
MIPv6 (without proxy)	4.654s
MIPv6 (with only Proxy Installed)	2.12s

TABLE V
MEAN REGISTRATION TIMES WITH AND WITHOUT TCP PROXY.

We evaluated the performance for downward vertical (GPRS→WLAN) handover with our proxy installed in the GPRS network (see figure 7). As usual, we initiated file transfers from a web-server to the mobile client, and compute the mean registration times from over 10 handover runs. As shown in table V, the use of proxy results in substantial improvement in registration times. Using our proxy, we are able to prevent excess packet queuing over GPRS, and avoid TCP window and RTO inflation, to improve TCP performance during downward vertical handoffs.

D. Client-Assisted Simulcast of Binding updates

Registration time (t_r) required to update a network is typically limited by the RTTs to the HA and the CN, whichever is higher, assuming update process is not sequential. One technique that can further optimize this latency, is by ensuring that the BUs are also sent along the faster of the two networks during a handoff. For example, in case of a WLAN→GPRS handoff, the BUs to the HA and the CN are sent using the GPRS link. Unfortunately, sending BUs over GPRS entails high RTT due to the high latency of the GPRS link (shown in figure 8). The registration process in this case entails one GPRS link-RTT, if BUs are simultaneously sent to the HA and CN, which is clearly disadvantageous in terms of performance. However, we could improve performance further by simulcasting BUs over links that are faster, to speed up the registration process. Simulcasting not only optimizes the registration time, but also makes the binding update process more reliable.

Figure 8 shows BU simulcasting for fast registrations. We can find that simulcasting the BU over both the links

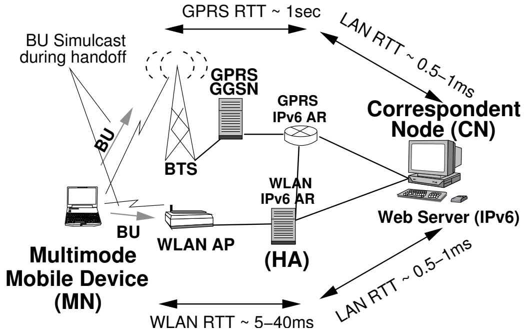

Fig. 8. Simulcasting BU for fast registration.

(GPRS and WLAN), one can achieve a fast registration, not typically limited by the RTT of the GPRS link anymore. Mobile IPv6 [6] offers the opportunity to simulcast BUs for fast registration. In this case, the first BU to CN or HA is sent as usual: the source address in the BU is the new interface address. For simulcasted BUs, the source address has to be modified, to be replaced by the old network interface address, along with the new interface address as a destination option (as `alternate-COA`) [6]. With this mechanism, the CN is able to create a binding entry between the new interface address and home address of the mobile node.

MIPv6 Scheme used	Mean Registration Time (t_r) (only WLAN→GPRS handovers)
Mobile IPv6	2.99s
Mobile IPv6 (with BU Simulcast)	1.36s

TABLE VI
REGISTRATION TIMES WITH AND WITHOUT BU SIMULCAST.

We implemented BU simulcast over MIPL's Mobile IPv6 source code. Implementation required modifications to the MIPL source code, to allow it to simulcast on every vertical-upward (e.g., WLAN→GPRS) handoff. In table VI, we show the mean registration times with and without BU simulcast for over 10 handover runs. We find that BU simulcast is able to achieve much better performance during WLAN→GPRS handovers, being able to perform fast registration using the WLAN link.

E. Soft Handovers with RA Caching

The handoffs discussed thus far have been *hard*; we down (stop listening) on one interface and simultaneously up (start listening from) the other. As a result, packets that were already in-flight, or were destined (and those that already made it) to the previous network interface are, unfortunately, not read. These packets have to be then retransmitted by the source, which leads to underperformance. However, handovers can be made *soft* to improve inter-network handover performance. Traditionally, soft handovers have been successfully exploited for link-layer handovers in cellular networks [7].

We can use a similar scheme for improving performance during vertical handovers. To achieve this, we modified our handoff module (`handoff.o` in figure 6) to support soft handoffs, such that after every handoff, it allows all inflight IP-packets destined to the previous interface to be read, and then be given to the application (here TCP). In order words, it keeps receiving packets from the previous network interface, while at the same time allow complete migration (registration) of IP points of attachment, before starting to send packets from the new interface.

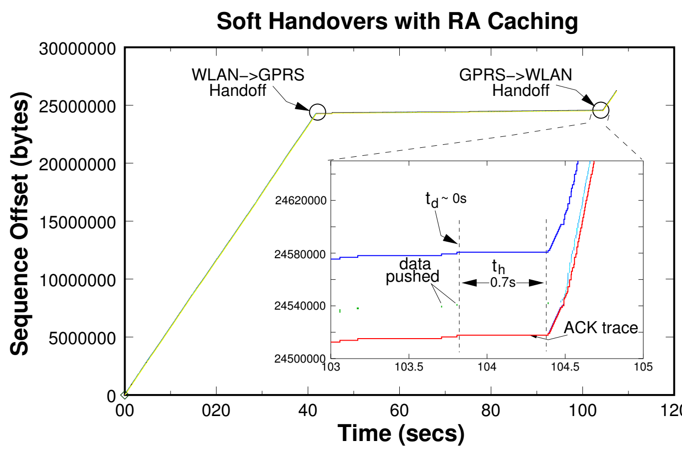

Fig. 9. Soft Handover Performance with RA Caching.

We used this module to evaluate the performance of GPRS↔WLAN handovers. We initiated a file transfer and allowed the soft handoff module to periodically handoff between GPRS↔WLAN. Figure 9 shows one such trace of a handoff using the soft handoff module that also performs RA caching. The close-up plot shows that RA caching is able to completely eliminate the detection time, while using soft handover, the mobile node is able to read all incoming TCP packets from the old interface (GPRS), and then ACK all of them from the new interface (WLAN), which keeps the TCP self-clocking going, and the source transmitting even after the handoff. As is evident from the plot, this leads to a dramatic improvement in TCP performance with only 0.7s required for the overall handoff.

Note, however, that due to the nature of the coverage offered by wireless overlay networks, soft handoffs are typically applicable for the case of downward vertical handoffs (e.g., GPRS→WLAN) because of the coverage offered by the network higher-up in the overlay (e.g., GPRS). Since GPRS coverage is virtually pervasive, packets from GPRS interface can still be read even after a handoff to WLAN, ensuring that inflight packets in GPRS are not lost. On the other hand, for a mobile user moving away from the diminishing coverage of WLANs, efficacy of soft handoffs when applied to upward vertical handoff remains questionable in high mobility environments. As in this case, it is not quite easy to determine if all inflight packets in WLAN could be saved. Our ongoing research explores this further.

VII. Related Work

Schemes for host mobility can be classified into vertical host mobility (one that involves mobility across different network technology) and horizontal host mobility (mobility within a given type of network). Only few research studies have evaluated vertical host mobility using real test-bed experimentation, and none using Mobile IPv6.

In pioneering research, Berkeley's BARWAN project first evaluated several issues related to vertical handoffs (see, [3], [1]). BARWAN typically builds over a 5-network wireless overlay network. However, their network involved vertical handoffs between Metricom Richochet and WaveLAN [3]. Further, H. J. Wang *et al.* present a policy-enabled handoff system and show the handoff latencies to be around 9s in Metricom and 26s with GSM Cellular, using reverse tunneling to the home agent to avoid packets being dropped at the firewall [2].

Vertical handoffs in BARWAN make use of MIPv4 with a multicast address in the mobile host (the care-of-address) to receive advertisements from potential access points in an overlay. Furthermore, they use fast beaconing and packet/header doublecasting to optimize such handoffs [3]. Nevertheless, a constraint here can be the scale of the solution; managing multicast address is a complex task, and in an environment with hundreds of mobile hosts this can become a limiting factor.

In constrast to the BARWAN project, our research provides insight into the Mobile IPv6 handover process, by closely analysing TCP traces of vertical handovers conducted over a LAN-GPRS-WLAN testbed. In this research, we have focussed on improving TCP performance during such handovers using schemes such as fast RAs, client-based RA caching, smart GPRS buffer management, client-assisted BU simulcasting and by using soft-handovers.

Related research by Stanford's MosquitoNet project make use of two flow-based handoff mechanisms built typically as extensions to Mobile IPv4 [8]. The first mechanism supports multiple packet delivery methods and adaptively selects the most appropriate one to use, relying on the characteristics of each flow. The other, however, enables mobile hosts to make use of multiple network interfaces simultaneously, and controls the selection of the most appropriate network interfaces on ongoing flows. Both approaches are based on traffic flow details that constructs the selection policies.

In a recent study, Milind *et al.* [9] discuss two architectures: a tightly-coupled and a loosely-coupled integration architecture between 3G (CDMA2000) and WLANs. They show the design and implementation of a gateway, called IOTA, that combines a number of different features to loosely integrate both CDMA2000 and IEEE 802.11b networks. Unfortunately, the Simple-IP operation used in their gateway offers only integrated billing and authentication, but no seamless inter-technology handoffs.

VIII. Conclusions

In this paper, we have discussed the use of Mobile IPv6 in wireless overlay networks, and related our practical experience using Mobile IPv6 in this environment. We characterized the Mobile IPv6 handover process for inter-network handovers, and have described the implementation of our LAN-WLAN-GPRS testbed.

By conducting vertical handoffs across different networks, we have analysed packet traces to determine the latencies of the steps in the handoff process, and have examined the effect handoffs have on active TCP flows. We have shown how the disparity in network link characteristics can make a significant difference during vertical handoffs, making them quite different from those of the horizontal handoffs.

We proposed and evaluated schemes to improve handover performance. We demonstrated the trade-offs involved in using fast RAs over different networks, and have shown how RA caching can be used to benefit handoff performance. Furthermore, we evaluated the benefit of installing a smart TCP proxy to improve TCP performance during handovers that involve GPRS. Moreover, we also quantified the benefits of client-assisted BU simulcast to aid and improve the registration process. We have demonstrated how soft handoffs can be used to significantly reduce disruption to TCP connections during handoffs.

Results from these practical experimentations show that Mobile IPv6 can be successfully used to migrate TCP connections during inter-network handovers.

References

[1] R. H. Katz and E. C. Brewer. "The Case for Wireless Overlay Networks". *In Proceedings of SPIE Multimedia and Networking Conference (MMNC'96)*, San Jose, CA, Jan 1996.

[2] H. J. Wang, J. Giese, and R. Katz. "Policy-Enabled Handoffs Across Heterogenous Wireless Networks". In *Proceedings of 2nd IEEE Workshop on Mobile Computing and Applications (WMCSA 1999)*, New Orleans, LA, February 1999.

[3] M. Stemm. "Vertical Handoffs in Wireless Overlay Networks". *ACM Journal of Mobile Networks and Applications (MONET)*, Vol. 3, No 4., 1998, pages 335-350.

[4] T. Narten, E. Nordmark and W. Simpson. "Neighbor Discovery for IP Version 6 (IPv6)". *Request for Comments*, RFC 2461, 1998.

[5] C. E. Perkins. "Mobile IP". *IEEE Communications Magazine*, May 1997.

[6] D. B. Johnson, C. E. Perkins and J. Arkko. "Mobility Support in IPv6" (draft-ieft-mobileip-ipv6-23.txt), May 2003. `http://www.ieft.org`.

[7] N. D. Tripathi, J. H. Reed and H. F. Vanlandingham, "Handoff in Cellular Systems", *IEEE Personal Communications*, December 1998.

[8] X. Zhao, C. Castellucia and M. Baker. “Flexible Network Support for Mobile Hosts”. In *Proceedings of ACM/IEEE MobiCom 98*, Dallas, Texas.

[9] Milind Buddhikot, et al.. “Integration of 802.11 and Third-Generation Wireless Data Networks”. In *Proceedings of INFOCOM 2003*.

[10] P. Vidales, L. Patanapongpibul and R. Chakravorty. “Ubiquitous Networking in Heterogeous Environments”. In *Proceedings of the 8th IEEE Mobile Multimedia Communications (IEEE MoMuC’2003)*, October 2003. (to appear).
`http://www.cl.cam.ac.uk/users/rc277/overlay.html`

[11] R. Chakravorty, P. Vidales, L. Patanapongpibul, I. Pratt and J. Crowcroft. “Vertical Handovers with Mobile IPv6”. University of Cambridge Computer Laboratory– Technical Report, May 2003.
`http://www.cl.cam.ac.uk/users/rc277/overlay.html`

[12] R. Chakravorty and I. Pratt. “Performance Issues with General Packet Radio Service”. in *Journal of Communication and Networks (JCN)*, pages 266-281, Vol. 4, No.2, December 2002. Special Issue on “Evolving from 3G deployment to 4G definition”.
`http://www.cl.cam.ac.uk/users/rc277/gprs.html`

[13] R. Chakravorty, S. Katti, I. Pratt and J. Crowcroft, “Flow Aggregation for Enhanced TCP over Wide-Area Wireless”, in *Proceedings of INFOCOM 2003*. `http://www.cl.cam.ac.uk/users/rc277/gprs.html`

[14] “Mobile IP for Linux (MIPL) Implementation by HUT Telecommunications and Multimedia Lab”, `http://www.mipl.mediapoli.com`.

[15] “6BONE: Testbed for the deployment of IPv6”.
`http://www.6bone.net`

[16] Cambridge Open Mobile Systems (COMS) Project Initiative.
`http://www.cl.cam.ac.uk/Research/SRG/netos/coms/`

[17] N. Montavont and T. Noel. “Handover Management for Mobile Nodes in IPv6 Networks”. *IEEE Communications Magazine*, August 2002.

[18] Vertical Handoff Module Source code for Linux 2.4.16.
Linux IPv6 Router Advertisement Deamon (`radvd+`) for Fast RAs.
`tcptrace+` (tcptrace with Mobile IPv6 support).
`http://www.cl.cam.ac.uk/users/rc277/soft.html`.