To appear, Kluwer International Journal on Wireless Communication Systems, 1996.

Handoffs in Cellular Wireless Networks: The Daedalus Implementation and Experience

Srinivasan Seshan, Hari Balakrishnan, and Randy H. Katz
{ss,hari,randy}@CS.Berkeley.EDU
Computer Science Division, Department of Electrical Engineering and Computer Science,
University of California at Berkeley, Berkeley, CA 94720-1776.

Abstract

Network protocols in cellular wireless data networks must update routes as a mobile host moves between cells. These routing updates combined with some associated state changes are called handoffs. Most current handoff schemes in wireless networks result in data loss or large variations in packet delivery times. Unfortunately, many applications, such as real-time multimedia applications and reliable transport protocols, adapt to long term estimates of end-to-end delay and loss. Violations and rapid fluctuations of these estimates caused by handoff processing often result in degraded performance. For example, loss during handoff adversely affects TCP performance [CI94], and high packet loss and variable delays result in poor real-time multimedia performance. In this paper, we describe a multicast-based protocol that eliminates data loss and incurs negligible delays during a handoff. The basic technique of the algorithm is to anticipate a handoff using wireless network information in the form of received signal strengths and to multicast data destined for the mobile host to nearby base stations in advance. This routing, combined with intelligent buffering techniques at the base stations, enables very rapid routing updates and eliminates data loss without the use of explicit data forwarding. We have implemented this protocol using IP Multicast and Mobile IP-like routing. In our implementation, handoffs typically take between 8 and 15 ms to complete and result in no data loss.

1. Introduction

Wireless data networks are usually composed of a wired, packet-switched, backbone network and one or more wireless (e.g., cellular radio or infrared) hops connecting mobile hosts to the wired part. The wireless part is organized into geographically-defined cells, with a control point called a *base station* (BS) for each of these cells (see Figure 1). The base stations are on the wired network and provide a gateway for communication between the wireless infrastructure and the backbone interconnect. As a mobile host (MH) travels between wireless cells, the task of routing data between the wired network and the MH must be transferred to the new cell's base station. This process, known as a *handoff*, must maintain end-to-end connectivity in the dynamically reconfigured network topology.

Although much has been written about cellular handoffs in wireless data networks, there appear to be few actual implementations in operational wireless testbeds. Fewer still are the number of implementations that achieve latencies of the order of a few tens of milliseconds or less with negligible associated data loss. In this paper, we present the design, implementation and performance evaluation of such a system and describe our experience with it. The testbed is part of the Daedalus project at Berkeley [Dae95] and is based on PC (*i*486 and Pentium) base stations and IBM ThinkPad mobile hosts communicating over a 2 Mbit/s AT&T WaveLAN.

The main goals of our work are twofold: to design and implement a handoff mechanism that achieves very low latencies with little disruption in traffic at the receiver, and to do this with as little data loss as possible. We aim for latencies of 30-40 ms or less and no lost packets during a handoff. These numbers are motivated by our desire to cause no disruption to continuous media streams (at 30 frames per second of full motion video, 30-40 ms is about one complete frame; in practice, wireless networks and mobile hosts rarely support such a high frame rate, so the penalty is usually less than a frame) and to cause no degradation in the performance of reliable end-to-end protocols like TCP that invoke congestion control procedures in response to perceived data loss.

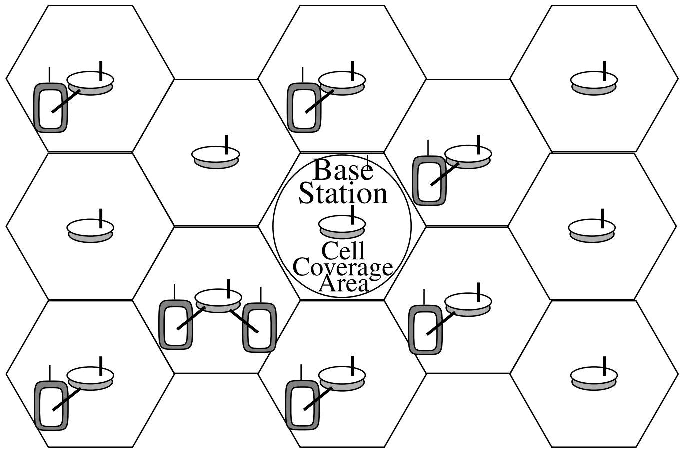

Figure 1. Arrangement of a cellular wireless network.

We achieve these goals by using multicast to arrange for nearby base stations to also receive packets destined for the mobile host and intelligently buffer a small number of most recent ones. After a handoff is completed, the new base station has in its cache the most recent packets and it can therefore ensure that the mobile host gets consistent performance with little data loss. Our measurements show that handoffs typically complete in between 8 and 15 ms in the common case of local handoffs between base stations close to each other. This low latency manifests itself in several ways to higher-level protocols and applications. For instance, when coupled with the snoop protocol [BSK95], it ensures that the sender of a TCP stream does not incorrectly invoke any congestion control procedures and cause a reduction in end-to-end throughput. Application performance is consistent across handoffs — there are hardly any noticeable glitches in video streams sent at close to the peak wireless bandwidth over UDP/IP.

The rest of this paper describes how we achieve our goals and is organized as follows. In Section 2, we describe the design of the handoff algorithm and compare it with the IETF Mobile IP proposal. We describe our implementation in Section 3 and the results of several of our experiments in Section 4. We also describe our experience with the protocol in this section. We discuss our future plans and conclude with a summary in Section 5.

2. The Design of the Handoff Protocol

In this section, we describe the design of our mobile routing protocol designed to achieve low-latency handoffs and highlight the differences between it and the IETF Mobile IP proposal [MJP95]. In the process, we show how our design can be viewed as being mostly compatible with the way routing is done in Mobile IP and how the differences help achieve low handoff latencies and negligible data loss.

We first describe the limitations of the standard IP routing mechanisms in a mobile environment. Each host in an IP network has a unique address that is used to route packets to it. Routers in an IP network forward each packet based on the destination address in the packet header and a routing table containing the next network hop in the route to all different IP addresses. Since the IP-based Internet has an hierarchical structure and hosts usually remain in fixed locations, the routing tables on most routers are small and are updated infrequently. However, these characteristics are not true in networks with mobile hosts. Using the standard routing protocols in this environment will require the routers to store routes to each mobile host and update them whenever a handoff occurs. Such a scheme will suffer from data loss during a handoff unless data is explicitly forwarded from the old to the new router. In addition, this is not a scalable solution, since the size of a routing table increases linearly with the number of mobile hosts and the frequency of updates to it is proportional to the frequency of handoffs in the system.

In most proposed solutions, including the IETF Mobile IP proposal, packets traversing the network during a handoff are lost or experience unusually long delays. Each mobile host in the IETF Mobile IP system is assigned a home network. When the route to a mobile host changes, the protocol notifies a special machine on the home network,

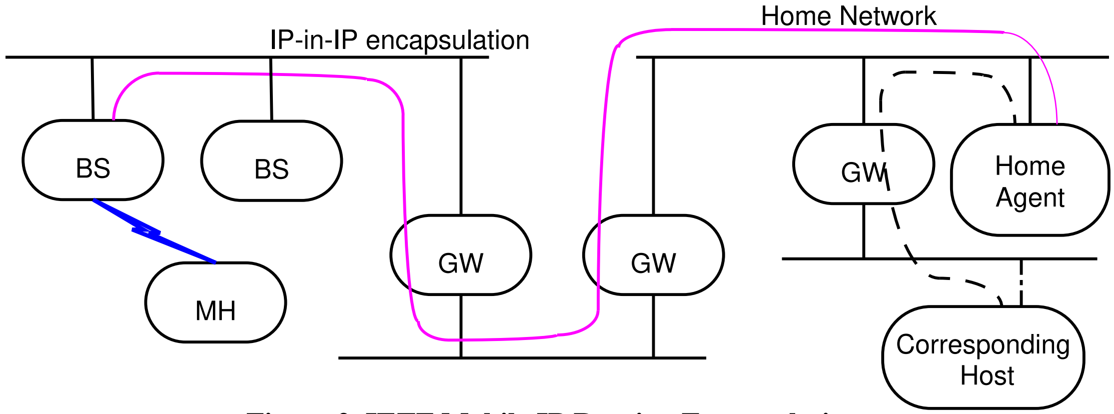

Figure 2. IETF Mobile IP Routing Encapsulation.

called the *home agent*, of the new route. Since routing updates must be sent to the home network, the duration of the update is proportional to the network distance or round trip time between the mobile host and its home. As a result, Mobile IP does not efficiently support continuous mobility far from the home network. While a route is being updated, the protocol delivers packets en route to the mobile host to the incorrect location. The last hop of the previous route for these packets will either drop the packets or forward them to the mobile host's current location. If the packets are dropped, the handoff will result in a significant amount of data loss. Alternatively, forwarding data after the routing stabilizes will cause the data to be delivered after a long additional delay, with an increased likelihood that packets will be delivered out-of-order.

High delay jitter and variation in data loss result in unacceptable performance for many standard real-time multimedia applications [MJ95] and reliable protocols such as TCP/IP. Both multimedia applications and reliable protocols adapt to long term end-to-end estimates of delay and packet loss between the data source and mobile host. However, they do not perform well when rapid variations in network characteristics occur, causing high fluctuations in these estimates. In order for these applications and protocols to achieve good performance, the protocol for routing data to mobile hosts must provide communication with consistent delay and negligible data loss.

IP routing support for a mobile host can be split into the separate problems of routing data from the mobile host and routing data to the mobile host. Fortunately, packets from mobile hosts to corresponding hosts (CHs) can use the normal IP routing system and the proposed mobile IP routing protocols can concentrate on the delivery of packets *to* the mobile host. In most mobile IP proposals, there are two basic stages in the routing of a packet to a mobile host. In the first stage the packet is delivered to the mobile host's home agent. The home agent usually resides on the network associated with the home address of the mobile host or advertises a route to the mobile host. It either maintains the current location of the mobile host or knows how to retrieve it. In the second stage of the routing, the home agent forwards the packet to the mobile host's current location. The various methods to do this use encapsulation, loose source routing or dynamic addresses to deliver the packet to the mobile host. The IETF proposal uses IP-in-IP encapsulation to forward the packet. Figure 2 shows the routing used by the IETF Mobile IP protocol.

Our mobile routing protocol uses the same first stage as the IETF Mobile IP protocol. However, it differs significantly from Mobile IP in the second stage in order to achieve low-latency handoff and to reduce packet loss and delay variation during handoff. Handoffs in our system use IP multicast [Dee91] and intelligent buffering in nearby base stations to eliminate data loss and provide consistent performance [KMS+93, Ses95]. We describe these ideas in the rest of this section.

In our scheme, there are three basic parts to the routing of packets to a mobile host. The first part handles the delivery of a packet to the home agent (HA). The second responsibility of the routing system is to determine the physical location of the mobile host. Finally, the routing system must support the delivery of packets from the home agent to the mobile host. These steps are shown in Figure 3.

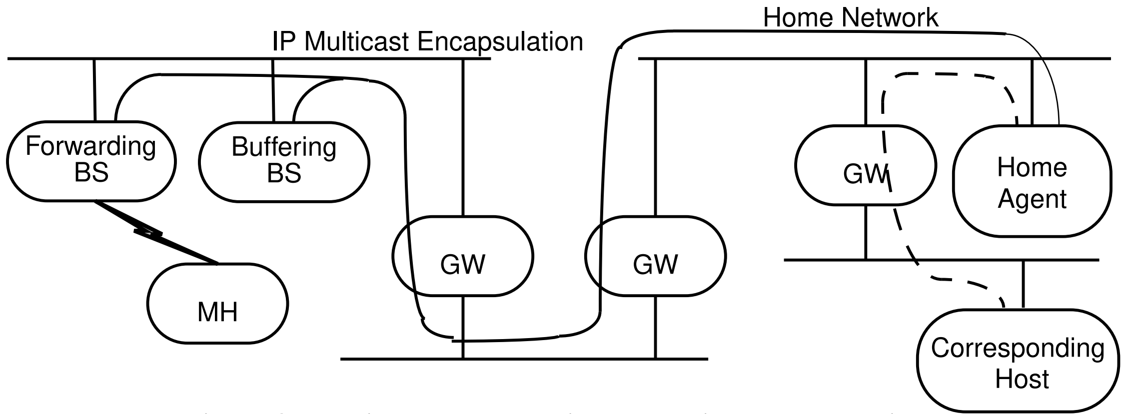

Figure 3. Multicast-based mobile IP Routing Encapsulation.

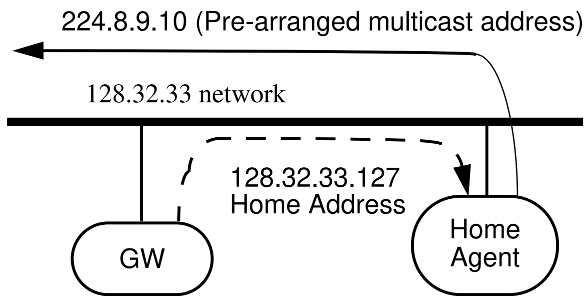

Figure 4. Home Agent Encapsulation.

Each MH is assigned a long-term IP address, called the home address, associated with its home location. The MH's home agent (HA) intercepts any packets transmitted to this home address using the proxy ARP mechanism. The HA is responsible for forwarding these packets to one or more base stations in the vicinity of the MH. In our implementation, we use IP multicast to perform this delivery. Each MH is also assigned a temporary IP multicast address. The home agent encapsulates packets destined for the MH and forwards them to its associated multicast group. The members of this multicast group include the base stations in the vicinity of the mobile host, but the mobile host itself does not join the group. An example of this interception and encapsulation is shown in Figure 4.

The second task of the routing system is to determine the current location of the MH. Each BS periodically broadcasts a beacon message to all MHs in its range. Each MH keeps track of the recent beacons it has received to approximate its current location and motion. The MH uses statistics such as the received signal strength of the beacons and communication quality to identify which BSs are nearby. The MH also determines which wireless network cell it should join as well as which cells it is likely to handoff to in the near future. Based on this determination, the MH configures the routing between the HA and the various BSs.

The delivery of packets from the HA to the BS utilizes the dynamic routing provided by IP multicast. The BS responsible for the cell containing the MH joins the IP multicast group. Each packet transmitted from the HA to the multicast group is forwarded by this BS, the primary, to the MH. At any instant of time, there is at most one primary BS in the system for a given mobile host. In addition, BSs that are identified as likely handoff targets are asked to join the multicast group by the MH. These BSs do not forward the packets from the multicast group to the wireless network. Instead, they buffer the last few packets transmitted from the HA. Typically, handoffs are to cells whose BSs in this manner have been primed for a MH. When a MH enters such a cell, the new primary BS begins transmitting packets from its buffer of packets. The data "in-flight" from the CH is delivered directly from the new BS without having it forwarded from the previous BS. Therefore, this handoff scheme has minimal data loss during handoff and incurs no additional delays due to data transfer. The other advantage of using multicast is that the routing to the MH does not have to be changed at the HA, since the multicast addressing itself provides the level of

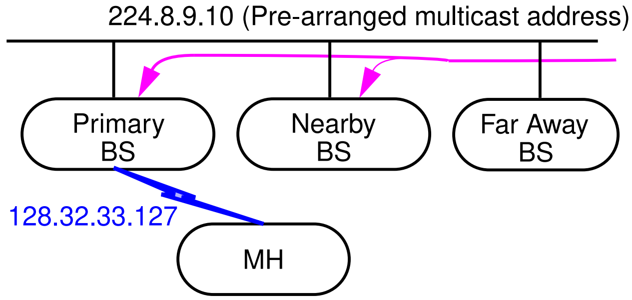

Figure 5. Home Agent to Mobile Host Routing.

indirection necessary for fast route updates. The routing of data between the HA and the MH is shown in Figure 5. In this figure, the multicast address corresponding to the MH is 224.8.9.10, and it's home address is 128.32.33.127.

Handoffs are mobile host-initiated and occur when the MH discovers a BS with a stronger signal than the current one. The sequence of events and messages during a typical handoff are shown in Figure 6. In the figure, the MH is moving from BS1's cell to BS2's cell. The handoff processing begins when it receives a new beacon measurement from BS2. Based on the beacon samples and some hysteresis, the MH decides that BS1 should be buffering packets and that BS2 should be forwarding packets. Using hysteresis ensures that very frequent and unnecessary handoffs do not occur. The heuristics used to accomplish this are described in Section 3.5.2. After comparing the desired state for each of the BSs in the area with their current state, the MH transmits a set of control messages to the various BSs. These control messages request each BS either to begin or end forwarding and buffering of packets. At most one BS is in the forwarding state at any instant of time for a given MH. While these requests are being delivered and processed, packets continue to arrive from BS1. The list of the last few packets received by the MH is included in the control message that activates forwarding on BS2. This allows the new primary BS, BS2, to determine which packets in its buffer have already been delivered to the MH by the previous forwarding BS. After synchronizing its buffer, the BS2 begins forwarding the remaining packets from the buffer and the multicast group to the MH.

Using multicast to establish the routing in advance greatly reduces the actual duration and amount of data loss of handoff. This allows handoff to complete without affecting the performance of data transfers using TCP or other higher-level protocols. An important difference between the IETF Mobile IP proposal and our scheme is that we encapsulate packets intended for a MH using an IP multicast address, while Mobile IP uses a unicast address. However, the MH does not join the multicast group and directly receive these multicast packets. Instead, the MH controls which base stations in its vicinity may join the group and receive packets on its behalf, and arranges for one of the participation base stations to forward decapsulated packets to it. Intelligent buffering policies at the base stations help achieve low latencies and low data loss while keeping these buffers small in size. Mobile IP is not designed to support continuous mobility while receiving data from various sources. Nor is it particularly well-suited for roaming far from the home network, since all route updates need to be made at the home agent every time the MH changes cells. Our multicast-based approach enables these features by consuming more resources in the wired network than Mobile IP in the form of bandwidth (for multicast) and buffer space. We believe this to be an acceptable trade-off, given the state of current technology and future trends towards rapidly increasing wired network bandwidth.

Other proposals to reduce the latency of handoff include VC Trees [AN94] and group-based routing [GS94]. The VC Tree approach splits the wireless network into areas called neighboring mobile access regions. In each region, a single fixed switching node acts as the root of a tree. This root node is responsible for keeping track of the location of all mobile hosts within its region and routing ATM cells to them. This solution does not attempt to eliminate data loss. In addition, this approach creates several scaling problems associated with the size of access regions. Small regions result in a high rate of costly inter-region handoffs and large regions place a significant load on the root nodes. The group-based routing proposal approaches the same problem by creating a three-layer hierarchy of

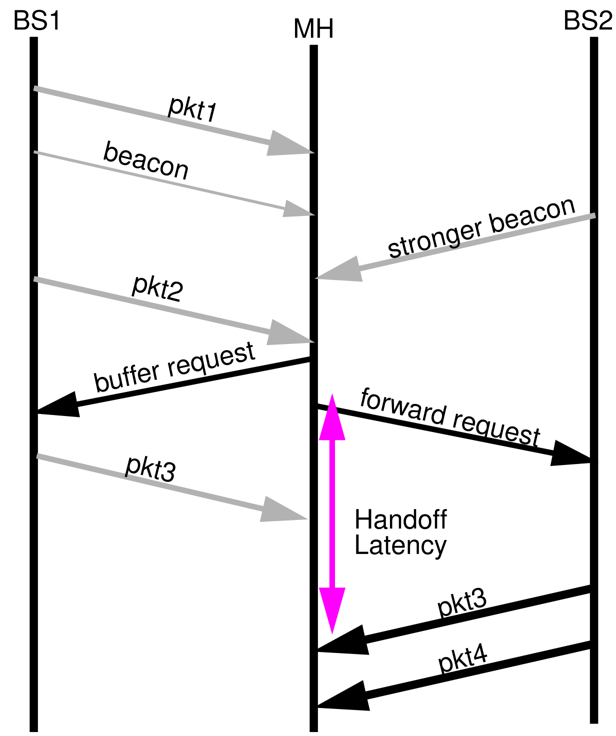

Figure 6. Typical handoff messaging.

mobile hosts, base stations and supervisor hosts. In this system, the various base stations are statically assigned to different multicast groups. The supervisor and mobile hosts negotiate to transmit packets to the multicast group associated with the destination's current location. Unfortunately, this system forces MHs to store a significant amount of state associated with other mobile hosts in the nearby area. In addition, the update and search mechanisms used by the system are not scalable. Both the VC Tree and group-based routing systems statically define regions where handoffs occur smoothly. Handoffs between these regions incur greater overhead. These regions should be dynamically defined to adapt to the individual user movement patterns as well as wireless network characteristics. Our approach avoids these drawbacks while providing the desired handoff performance.

3. Implementation

In this section, we describe the details of the implementation of the mobile routing protocol that meets the dual goals of low latency and negligible data loss.

3.1 System Architecture and Overview

We have implemented the routing protocol on a testbed as part of the Daedalus project at Berkeley. The implementation platform consists of IBM ThinkPad laptops and PC (*i*486 and Pentium) base stations running BSD/OS 2.0 from BSDI, communicating over an AT&T WaveLAN. The WaveLAN radio is a 915MHz, wireless, direct sequence spread spectrum local area network interface [ATT]. It provides a shared, ethernet-like link with a raw signaling bandwidth of 2 Mbit/s. Since it is a broadcast-based network, the 2 Mbit/s bandwidth is shared by all machines within radio transmission range. The WaveLAN network interface card provides a mechanism to retrieve the radio signal strength, silence level and signal quality observed during the reception of a packet.

The network topology of the system is shown in Figure 7. There are currently several PCs and mobile hosts in our experimental system. One fixed location host is the source of data during our experiments and another is a home agent for the mobile hosts. In our experiments, the mobile host handoffs are between two PC base stations on an

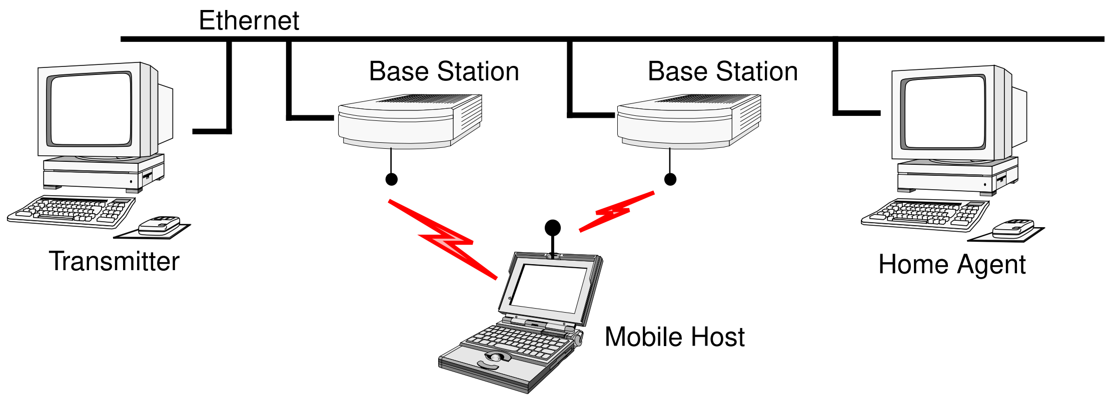

Figure 7. Network topology for experiments.

single Ethernet. The base stations were also moved to separate Ethernets for some experiments to evaluate the performance of handoffs when the base stations are not close to each other. The implementation currently supports data transfers to and from mobile hosts and smooth handoffs based on the design presented in Section 2.

The implementation consists of four modules: the *encapsulator*, the *decapsulator*, the *beaconing system*, and the *route analyzer*. The encapsulator at the home agent and decapsulators at the base stations perform the actual routing of packets between the corresponding host and the mobile host. The beaconing system is used to identify the current location of the mobile host. The route analyzer runs on the mobile host and uses the information provided by the beaconing system to configure the routing of packets. We describe these various modules in more detail in the following sections.

We made several modifications to the network modules of the operating system to support the mobile routing protocols. These modifications are described in the following sections. In addition, we added special support for high priority packets by implementing the low delay IP type-of-service option (IP TOS) in BSD/OS 2.0. This was done by adding a special low delay packet queue for each network interface, especially for the WaveLAN. When sending data to the network interface card (NIC), the operating system transmits data from the low delay queue before processing the normal queue.

3.2 The Encapsulator

The first stage of routing between a corresponding host and a mobile host delivers the packet to an entity that understands mobile routing, the home agent. When a mobile host leaves its normal home location, it initializes the home agent encapsulation by specifying a pre-defined multicast address corresponding to it. The home agent then intercepts all packets destined for the mobile host and forwards them to this IP multicast address. The home agent in our system is implemented using encapsulation code added to the IP routing modules, and a set of calls to configure the encapsulation and interception of packets. The processing of packets at the home agent is shown in Figure 8.

When a mobile host is away from its home location, the home agent uses the proxy ARP mechanism to respond to all ARP requests for the MH. However, the home agent returns its own ethernet address to the transmitter of the ARP request. This ensures that all packets destined for the mobile host are successfully intercepted by the home agent.

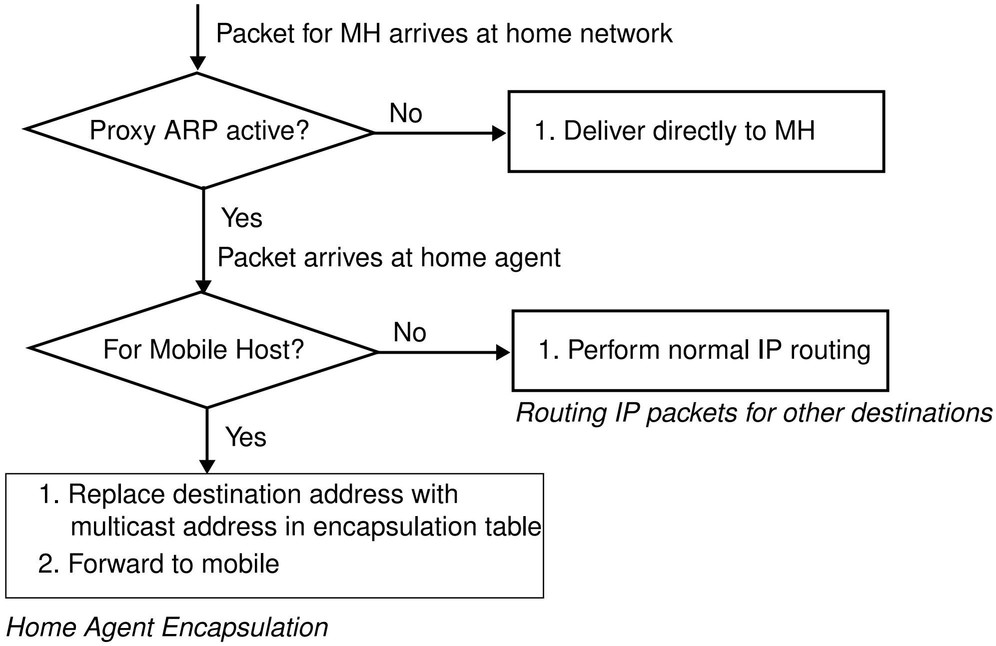

Figure 8. Flowchart for the encapsulator.

At the home agent, the IP routing code receives all packets intended for the mobile host. At this point, the home agent must identify that the packet is destined for a mobile host and encapsulate the packet using the multicast address. The following changes to the kernel of the home agent enable this:

1. The home agent kernel contains an encapsulation table containing information about each mobile host for which it is responsible. Each entry in the list contains the IP address of the mobile host, the IP Multicast address to use while encapsulating packets for that mobile host, and the time-to-live (TTL) for the encapsulated packets.

2. Two newly added IP-level socket options, `ADD_ENCAP` and `DEL_ENCAP`, control the encapsulation. The home address of the mobile host and the IP Multicast group to forward the packet to are inputs to both socket options. The `ADD_ENCAP` option also requires the TTL to use on the IP Multicast group as an input. When a mobile host leaves its home address, the home agent joins the IP Multicast group corresponding to the mobile host and uses the `ADD_ENCAP` socket option to add an entry to its encapsulation table. This initializes the encapsulation and forwarding. If the mobile host returns to its home network, the home agent uses the `DEL_ENCAP` socket option to delete an entry from the table and has the base station leave the associated IP Multicast group. This socket option call combined with the deletion of the associated proxy ARP entry allows the mobile host to receive packets directly at its home address.

3. The mobile encapsulation module uses the information contained in the encapsulation table and the destination address of the packet to encapsulate and forward packets. This module encapsulates the original packets with an IP header containing the IP Multicast address associated with the mobile host. Finally, the home agent uses the normal IP packet output routines to forward the packet.

4. Once the packets are sent encapsulated with the multicast address, standard IP Multicast routing delivers them to the base stations subscribed to the appropriate multicast groups. The route analyzer requests only the base station that is currently forwarding packets to the mobile host and base stations that are likely targets for handoff to listen on the mobile host's multicast group. The mobile host itself does not join the multicast group.

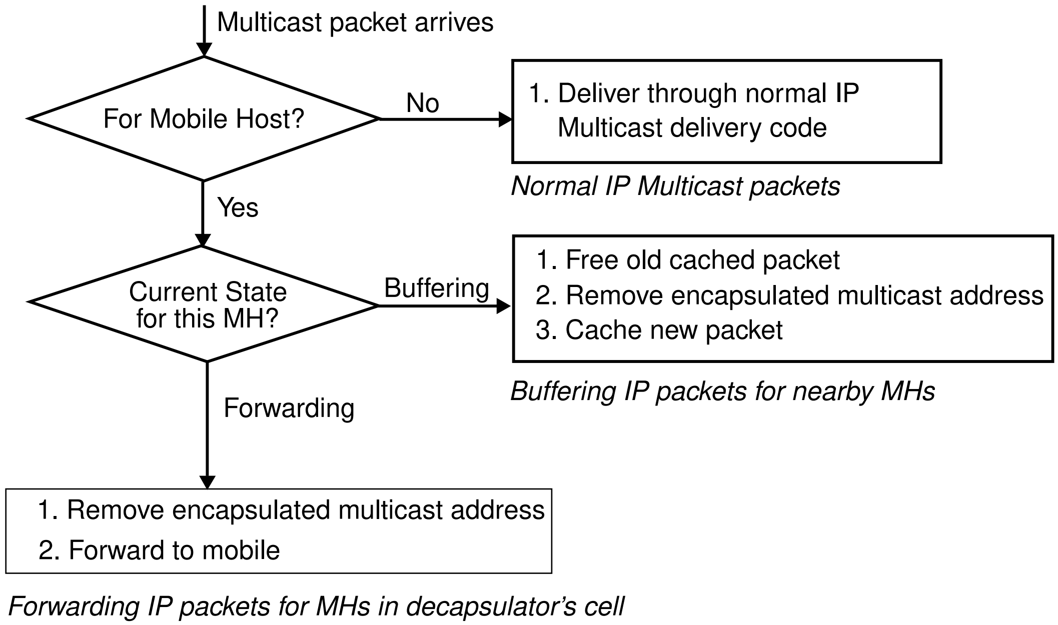

Figure 9. Flowchart for the decapsulator.

3.3 The Decapsulator

Packet routing from the corresponding host to the mobile host is performed by decapsulators that run at each base station. The route analyzer on the mobile host requests one or more decapsulators in its vicinity to receive packets on its behalf. The requested base stations join the IP Multicast group associated with the mobile host and receive packets intended for the mobile host. The route analyzer chooses a single base station in its area to be the current forwarding base station (the primary base station). The decapsulator on the primary base station decapsulates and forwards packets across the wireless link to the mobile host. The other base stations that receive packets for the mobile host do not forward them on; instead, they buffer the last few packets received. The decapsulator is implemented in two major sections: a set of changes to the IP routing code and a user-level daemon that receives and processes requests from the mobile hosts.

Several additions to the IP routing module perform the functions of decapsulating, forwarding and buffering packets. This module receives all packets that arrive at the base station and have an IP Multicast address. The decapsulator scans all multicast packets to identify the ones that are destined for a registered mobile host. It then processes the packet based on the current state of decapsulation (either primary forwarding or nearby buffering) for the mobile host. Figure 8 shows the processing of multicast packets at the decapsulators.

To perform the necessary tasks, we made the following changes to the kernel of the base stations:

1. The base station kernel contains a decapsulation table containing information about each mobile host for which it is currently responsible. An entry for a mobile host in the list contains its associated IP Multicast address, its home IP address, and current state of decapsulation from this base station. The decapsulator on a mobile host's primary base station is in the forwarding state. The other base stations that have been requested to join the multicast group associated with the mobile host have decapsulators in the buffering state. In this state, a decapsulator caches the last 12 packets it has received that are destined for the mobile host. We tuned the implementation to buffer 12 packets based on our measurements of handoff times and throughput rates over the WaveLAN. At a data transfer rate of 1.4 Mbit/s, 12 packets corresponds to about 100 ms of data for the largest packets (about 1440 bytes). Thus, 12 packets translates to a time duration that is about a factor of three higher than our latency goal of 30 ms, thereby ensuring that we hardly ever incur data loss during a handoff.

2. A set of newly added IP-level socket options, ADD_DECAP, DEL_DECAP, ENA_DECAP and DIS_DECAP, control the decapsulation. The information passed to these socket options include the IP Multicast group and the home IP address associated with a mobile host. When a mobile host enters a base station's vicinity, the decapsulator uses the ADD_DECAP socket option to add an entry to decapsulation table and to join the base station to the MH's IP Multicast group. The initial state of decapsulation is to only buffer packets for a mobile host. When the mobile host enters the base station's cell (i.e., it chooses the decapsulator as its primary forwarder), it uses the ENA_DECAP call to change the state of decapsulation from buffering to forwarding. During the change to forwarding state, the base station forwards to the mobile host any packets that were stored while the decapsulator was in buffering mode and have not yet been delivered to the mobile host. This eliminates any loss of packets en route to the mobile during handoff. To identify which packets to transmit from the buffer, the MH passes the IP IDs of the last three packets received by it and packets after these are transmitted. This simple synchronization hint works well in practice and helps prevent duplicate packets from arriving at the MH in most circumstances. Once the mobile host leaves the cell, the decapsulator returns to the buffering state using the DIS_DECAP call. Finally, DEL_DECAP deletes the decapsulation entry from the table and has the base station leave the associated IP Multicast group.

3. The decapsulation module uses the information contained in the decapsulation table and the destination multicast address to perform the final delivery of a packet. The base station must have a host route entry in its routing table for the home IP address of the mobile host. This entry forces the packet to be routed across the wireless network to the actual mobile host. We accomplish this by assigning each MH a temporary IP address in the region's wireless network and setting up a route entry for the home address through this temporary address. This allows the base station to use the normal IP packet output routines to forward the packet to the mobile host.

A user-level daemon, `decapd`, performs many of the control functions needed in the decapsulator. As soon as a mobile host comes within range of a base station, the route analyzer on the mobile host creates a TCP control connection between the route analyzer and the decapsulation daemon. When the TCP control connection is created, the `decapd` process adds a routing table entry between mobile host's home and temporary address. `Decapd` accepts four different forms of requests on these connections. The requests map directly to the four different socket options possible on a base station. The route analyzer uses the information provide by the beaconing system to decide which base stations should be forwarding packets and which base stations should be buffering packets. It communicates these decisions to the decapsulators across these TCP control connections. By processing these packets a decapsulator puts the base station into the state desired by the mobile hosts.

3.4 The Beaconing System

The beaconing system runs on the mobile hosts and base stations. Its primary responsibility is to provide the route analyzer with information about the location of the mobile host. The beaconing system communicates to the route analyzer the IP address and communication quality of each base station that can be heard by the mobile host. The route analyzer uses this information to determine which base stations should listen on the multicast group associated with a mobile host.

The WaveLAN NIC measures the signal strength, the background noise, and the signal quality of the received packet at the time of reception. These measurements are used to determine the best target base station for a handoff. However, these measurements must be retrieved upon the reception of a packet from the NIC. To simplify the collection of this data, we added a process to the base station that periodically sends beacon packets. At the mobile host, we added a set of calls to help identify these beacon packets and retrieve the corresponding signal measurements.

Two new `ioctl()` calls, ADD_BEACON and DEL_BEACON, facilitate the retrieval of radio measurements from the WaveLAN NIC at the mobile host. The information passed to these calls include the IP address and WaveLAN link layer address of a base station. The ADD_BEACON `ioctl` adds the passed address to a table of base stations in range, while the DEL_BEACON call removes an entry from this list. When a WaveLAN beacon packet arrives

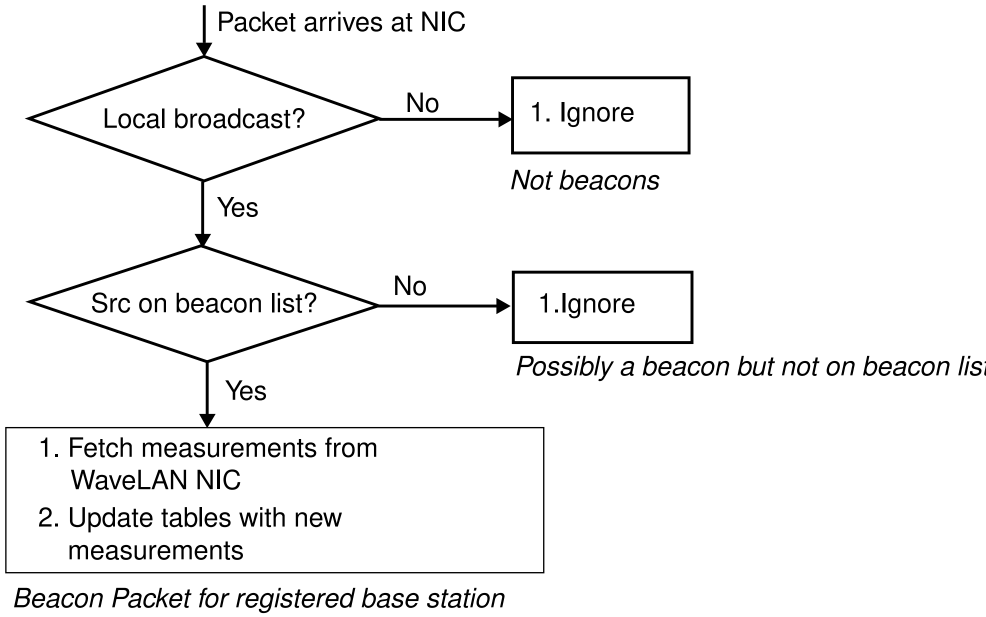

Figure 11. Flowchart for beacon processing.

from one of the base stations in the table, the device driver retrieves the signal measurements from the NIC and stores it. The device driver currently stores the last three samples of each base station. Figure 11 shows the filtering and retrieval of the beacon signal measurements.

The beacon packets sent from a base station contain the IP addresses for its wired and wireless interfaces and a time stamp. These beacons are transmitted over UDP/IP to the WaveLAN broadcast address at a frequency of one beacon every second (this is a configurable parameter). At this beaconing rate, we observe approximately a 10% degradation in aggregate bandwidth across the WaveLAN.

At the mobile host, the route analyzer uses the `ADD_BEACON` and `DEL_BEACON` ioctls to update the list of base stations that are in range of the mobile host. This allows the mobile host to have relatively fresh information about the communication quality to each of the nearby base stations. These radio measurements are retrieved at user level using the BSD kernel memory interface, `kvm`. We wrote a simple library that mapped the entire table of base stations and measurements into the memory of a user-level program. The library also provides support for retrieving the sample of a specific base station.

3.5 The Route Analyzer

The route analyzer uses the information and functionality provided by the decapsulators and beaconing system to determine and control the route of packets to the mobile host. There are two major parts to the route analyzer: a module that interacts with the beaconing system and a module that interacts with the decapsulators. The following sections describe these modules in detail.

3.5.1 Information Retrieval Module

The information retrieval module is responsible for identifying which base stations are nearby and for retrieving any information the wireless network can provide about these base stations. This module is the collector of "hints" about user mobility available from the system. In the WaveLAN-based system, these hints come from the signal measurements available from the NIC. The beaconing system provides a mechanism to filter and retrieve these measurements. The information retrieval module chooses which measurements to filter and retrieve.

At the mobile host, the information retrieval process receives all packets destined for the beacon service UDP port.

When a beacon packet arrives, the process attempts to read the radio measurements associated with the transmitting base station, maintained by the kernel. If the WaveLAN device driver has the base station in its list of nearby base stations, the process retrieves the beacon measurements. Otherwise, the process performs an `ADD_BEACON` `ioctl` to register this base station as one that is in range of the mobile host. It also creates a TCP control connection to this base station for use by the route control module. If a beacon has not been received from a base station for a short amount of time, currently tuned to ten seconds, the module decides that the base station has gone out of range of the mobile host and uses the `DEL_BEACON` `ioctl` to remove the base station from its list. This allows the mobile host to maintain relatively fresh information about the communication quality to each of the nearby base stations. In addition, it also provides the mobile host with a reliable communication channel to each base station in range.

3.5.2 Route Control Module

The route control module uses the data gathered by the information retrieval module to identify the mobile host's connectivity. It determines which base station it is best for the mobile host to route data through as well as which base stations the mobile host is likely to handoff to in the near future. It uses this information to configure the routing of the nearby base stations.

Each time a beacon arrives at the mobile host, the information retrieval module provides the route control module with a new list of nearby base stations. This list is sorted in order of decreasing signal strength. Normally, the route control module chooses the base station with the best signal strength as the primary. It uses hysteresis to avoid changing primary base stations too frequently. The strongest signal must be at least 15% better than the current primary base station's signal before the route module initiates a handoff. The choice of this threshold was empirically determined based on our experience with the system. The module also identifies the nearby base stations with the strongest signals as likely targets for handoff. These base stations are requested to buffer packets for the mobile host. The module has three different policies that can be used to determine the number of buffering base stations. The first scheme requests all base stations in range to buffer packets for the mobile host. Another policy has a fixed number of base stations buffering at all times. The last policy adapts the number of buffering base stations to the rate of the mobile host's motion. The module estimates the motion of the mobile host using the total magnitude of signal measurement changes in the past few seconds. There are three definable thresholds that categorize the motion as 'none', 'slow' or 'fast'. For each class of motion a different number of base stations buffer packets for the mobile host. The choice of policy should be based on several factors including: (i) the characteristics of user movement, (ii) the availability of backbone network resources for multicast and buffering, and (iii) the cost of multicast and the desired "hit-rate" of handing-off to base stations that have routing already set up.

Once the route control module has determined the state of decapsulation at the various BSs, it must communicate these decisions to the decapsulators. The module sends a control packet to each base station that needs to change state across a TCP control connection. In order to deliver this packet quickly, the module transmits this packet with the low delay IP TOS flag set. This message contains the new state the base station should be in and the IP IDs of the last three packets received by the mobile host to help the new primary base station synchronize its buffers. At each base station, the `decapd` process reads these messages and performs the appropriate decapsulation socket options. The state changes at the different base station comprise the handoff of a mobile host from one cell to another and set up the routing desired by the mobile host.

4. Performance Analysis

In this section we present the results of several experiments with the handoff protocol and describe our experiences with using it over the WaveLAN wireless testbed. Our experimental setup is the same as described in Section 3. There are two base stations, a data source and a home agent on the wired backbone, and a mobile sink connected over a WaveLAN link. In order to perform controlled experiments, we force handoffs to occur at predetermined instants of time. We are interested in measuring the overall end-to-end throughput (and degradation from the maximum), the handoff latency, the number of dropped packets, the network utilization and the buffer requirements at the base stations. While the overall end-to-end throughput is what applications perceive, understanding end-to-end effects requires a closer analysis of what happens to individual packets during a handoff. This motivates our choice of the handoff latency and number of lost packets as performance metrics. We describe the results of using both UDP and TCP as higher-level transport protocols.

The rest of this section is organized as follows. We first present our experimental results and explain them. We then analyze some of the overheads in the protocol, and conclude with a description of some problems we observed with the WaveLAN MAC layer.

4.1 Experimental Results

We first measure the maximum throughput in the absence of handoffs with the beaconing system turned on from the two base stations. We then measure the throughput when handoffs are done between two nearby buffering base stations at different periodic intervals of time. For one of these runs, we draw a timechart of the precise messaging and events that occur in the process. These are all done with TCP as the higher-level protocol. After this, we measure the throughput, number of lost packets, number of duplicate packets, and latency when handoffs are done between base stations that don't buffer packets in advance; we will also vary the number of network hops between the base stations.

We first formally define the term *handoff latency*. The main objective in measuring this parameter is to determine the disruption in the arrival of data seen at the receiver because of a handoff. We define the difference in time between the arrival of the first new packet from the new base station and the time at which the decapsulation request was sent to the new base station from the mobile host as the handoff latency. This is a more accurate definition than simply the difference in time between the arrival of the first new packet from the new base station and the arrival of the last packet from the old base station, since it accurately captures the response time to a "start forwarding" request. In addition, the latter definition leads to very low measured latencies when data rates across the WaveLAN are high (> 1.25 Mbits/s), since the WaveLAN is a shared medium and packets enqueued at the link layer before the handoff continue to arrive from the old BS even after a handoff has been completed.

In the absence of handoffs, the peak achieved throughput is 1.45 Mb/s. This is about 10% less than the peak point-to-point wireless link throughput achieved over TCP or UDP with no other traffic present. Packet traces indicate that WaveLAN MAC-level effects cause this degradation; we will discuss this issue at the end of this section.

Table 1 shows the throughput achieved (with standard deviation) for TCP transfers to a mobile host with handoffs

Time Between Handoffs (sec)	Throughput (Mbits/s)	Standard Deviation (Mbit/s)
1	1.42	.011
2	1.43	.016
3	1.43	.012
5	1.43	.014
8	1.44	.012
10	1.43	.012
∞	1.45	.011

TABLE 1. Throughput achieved for TCP transfers to mobile host at various handoff frequencies.

between two buffering base stations occurring at frequencies ranging from 1 ms to 10 ms. Each entry in the table is the average of 20 transfers of 8 MBytes of data. In this experiment, handoffs were forced between the base stations at regular intervals of time. The end-to-end throughput degrades by less than 1% even at a very frequent handoff rate of one handoff per second, demonstrating that our protocol achieves good end-to-end performance without imposing high overheads.

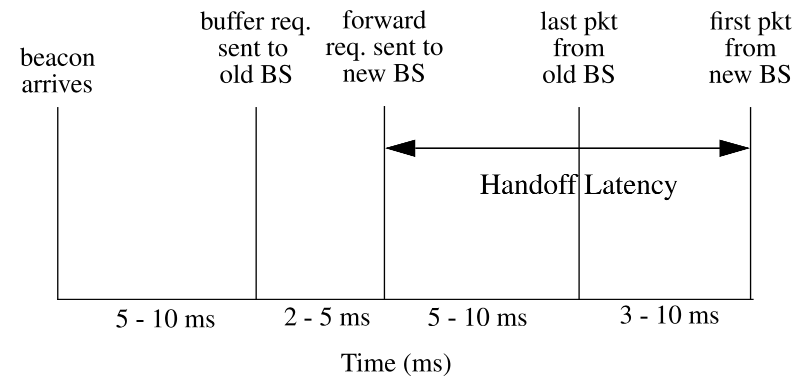

Figure 12. Timing of handoff events in the common case of adjacent, buffering base stations.

We define the *distance* between two base stations to be the number of network hops traversed by IP multicast packets between them. This number could be larger than the number of hops non-multicast packets take due to the presence of tunnels [Dee91] in the multicast route. We will analyze the performance of our protocol as a function of the distance between the old and new base stations.

In order to analyze the effectiveness and efficiency of our multicast- and buffering-based handoff scheme, we performed three sets of experiments and measured the handoff latency and number of packets lost during a handoff.

Handoff method	Distance between BSs	Handoff latency	No. of lost packets
Multicast-based with buffering	0 hops	8-15 ms	0
	1 hop	8-15 ms	0
	2 hops	8-15 ms	0
	3 hops	8-15 ms	0
Multicast-based and no buffering	0 hops	8-15 ms	2-3
	1 hop	10-15 ms	3-4
	2 hops	10-15 ms	3-4
	3 hops	10-15 ms	3-4
No multicast and no buffering	0 hops	15-20 ms	2-3
	1 hop	18-23 ms	3-4
	2 hops	20-25 ms	4-5
	3 hops	25-30 ms	4-5

TABLE 2. Results of experiments with various handoff policies.

Table 2 shows the results of these experiments. We now analyze and explain these results.

1. *Multicast routing with buffering base stations:* This is the best case for our handoff scheme. Here, the new base station to which the MH transfers has already joined the multicast group and is receiving and buffering packets for the MH prior to the handoff. In practice, this happens if the MH has a good idea of its location and sets up the state of the decapsulation at the new base station in advance. In this case, we expect the best handoff latencies (relative to the other experiments) and almost no packet loss. In this situation, we observe handoff latencies between 8 and 15 ms and no lost packets. The number of packets transmitted from the cache of the new BS is usually between three and five at high data rates (about 1.4 Mbits/s). Figure 13 shows the timechart of events that occur during a typical handoff between buffering base stations and is independent of the number of hops between base stations. The horizontal axis shows the observed range of times between events.

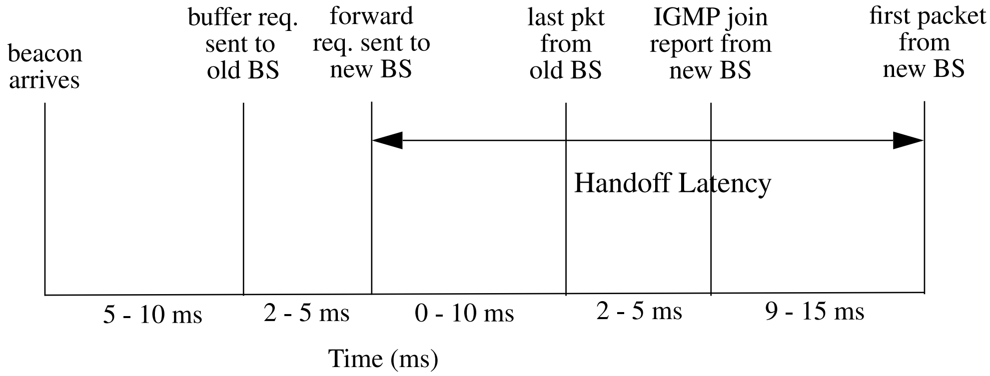

Figure 14. Timing of handoff events for handoffs between non-buffering base stations at distance 3.

2. *Multicast routing with no buffering:* In this case, the new base station joins the multicast group corresponding to the MH but does not buffer packets on its behalf. We did this experiment to isolate the effects of route updates and buffering. In practice, this may happen when the MH decides that requiring the BS to buffer packets for it in advance would unnecessarily consume buffer resources in the network, or if the BS has an admission control policy for mobile hosts. Since the route to the MH through the new BS has already been updated, we observe low latencies, in the range between 10 and 15 ms. However, the number of lost packets is between two and four, depending on the data rate. At the highest data rates of about 1.4 Mbits/s, there are usually three or four lost packets. This is consistent with the number of retransmitted packets in the first case, described above.

3. *No multicast, no buffering:* This is the pathological worst case for the algorithm and could be caused by highly unpredictable movement by the MH. We performed this set of experiments mainly to illustrate the benefits of buffering packets in advance and multicast-based route management. In this case, the number of lost packets and the handoff latency increase with the distance between base stations. Figure 15 shows the timechart of events during a handoff between base stations at a distance of three from each other, where the new base station has not joined the multicast group corresponding to the MH (and has therefore not primed its buffers in advance). We found that each hop corresponds to approximately between 3 and 5 ms of multicast join time in our system, based on the difference in time between the sending of the IGMP group join message by the new BS and the arrival of the first packet there.

4.2 Analysis of Protocol Overheads

There are two basic sources of inefficiency in the handoff protocol: the routing of packets to the base stations and the overhead of buffering packets at several base stations.

The routing protocol uses a home agent and IP Multicast to deliver packets to the mobile host. This introduces two well-known sources of inefficiency:

1. *Triangle Routing*: Since all packets must pass through the home agent, they may not follow the best route between source and destination. When the home agent resides along or near the best route between the mobile host and corresponding host, the route taken by packets to the mobile host is reasonably efficient. However, packets may experience high overheads when the home agent is far from the best route. To avoid this problem, the corresponding host must also be able to transmit packets to the forwarding and buffering base stations of a mobile host. This type of route optimization is difficult as long as we use IP Multicast to deliver packets to multiple base stations.

2. *DVMRP*: We chose to leverage off the widely deployed and successful IP Multicast mechanism to deliver packets to multiple base stations. However, many other mechanisms can be used to perform the necessary delivery of packets. The current IP Multicast implementation uses DVMRP [Dee89], which has some disadvantages including inefficient support for scalable wide area multicast, sparse membership groups and high

overhead of maintaining multicast routing tables. The delivery of packets from the home agent to a few, possibly distant base stations is not well suited under DVMRP-based IP Multicast. Future multicast protocols will be better suited to the task. However, we can also support the necessary delivery using a more limited multicast protocol designed specifically for mobility support.

The second cause of overhead in the protocol are the buffering base stations. The measurements of the routing protocol indicate that packet loss and handoff latencies are significantly lower for handoffs to buffering base stations. However, buffering consumes both base station memory and network bandwidth. Therefore, we must avoid buffering packets at base stations that the mobile host does not visit to improve the scalability of the handoff algorithm. There are several factors that affect the backbone network resource consumption:

1. *Physical layout of base stations*: This determines which base stations can be heard by a mobile host and their relative signal strength. Therefore, this affects the number of base stations that may be buffering packets for a mobile host. If base stations have a high degree of overlap in cell coverage (i.e. they are placed very close together in comparison to the cell radius), a mobile host will have difficulty in identifying the primary targets for handoff. If there is insufficient overlap, holes will appear in the wireless network coverage, adversely affecting handoff performance.

2. *The mobility patterns of users*: This determines the efficacy of the MH's strategy to choose buffering base stations. Certain mobility patterns may need many more buffering base stations for reasonable performance. The trade-off is between network resource consumption and improved handoff performance.

3. *The wired network layout of base stations*: If base stations are located on the same physical, broadcast-based network, the additional network bandwidth used by the multicast is very small. Base stations that tend to buffer packets for the same mobile hosts should be placed close together in the network. This implies that the network of base stations must be designed to mirror the cell topology. In general, this problem is a variant of the graph partitioning problem, which is NP-complete [GJ79]. Some heuristics to address this problem in switch based networks are described in [MS95]. We are working on efficient heuristics to implement this feature in more general situations.

User mobility traces are necessary to tune each of these factors. Unfortunately, our system is too immature to provide these traces yet. However, once the system is tuned the overhead of buffering simplifies to the use of bandwidth (equal to the wireless link bandwidth) on a few wired network hops between nearby base stations and the use of small buffers (12 packets for WaveLAN) in the buffering base stations. Given the much higher bandwidth of wired networks compared to wireless networks and magnitude of the memory required, this overhead is quite reasonable for the handoff performance gains.

Our system has not yet reached the scale where these disadvantages pose a significant problem, but we are working towards improving the scalability of the basic techniques [Ses95]. Overall, our experience with the handoff mechanism presented in this paper has been very positive an encouraging. In the common case, adjacent base stations prime their buffers prior to the actual handoff and this results in very low latencies and consistent performance observed by higher-level protocols like TCP. [BSK95] describes the details of the interactions between the handoff protocol and the snoop agent, and describes how little degradation is observed in end-to-end TCP performance even in the face of frequent handoffs. Applications like real-time video transmissions see no noticeable glitches, since the time for a handoff is less than the time required to transfer and display a frame, and hardly any packets are lost.

4.3 Problems with the WaveLAN MAC layer

In the process of our experiments with the WaveLAN network, we gained significant experience with the device and were satisfied with the way it worked and the relatively high bandwidth it could support. However, we did discover some flaws in the implementation of the inherently unfair CSMA/CA MAC-layer protocol.

Figure 17 shows the transfer sequence of a 10000 packet, 1.45 Mbits/s TCP transfer with handoffs being performed every 10 seconds. Each packet is 1440 bytes in size. In order to keep the number of points on the graph small, only

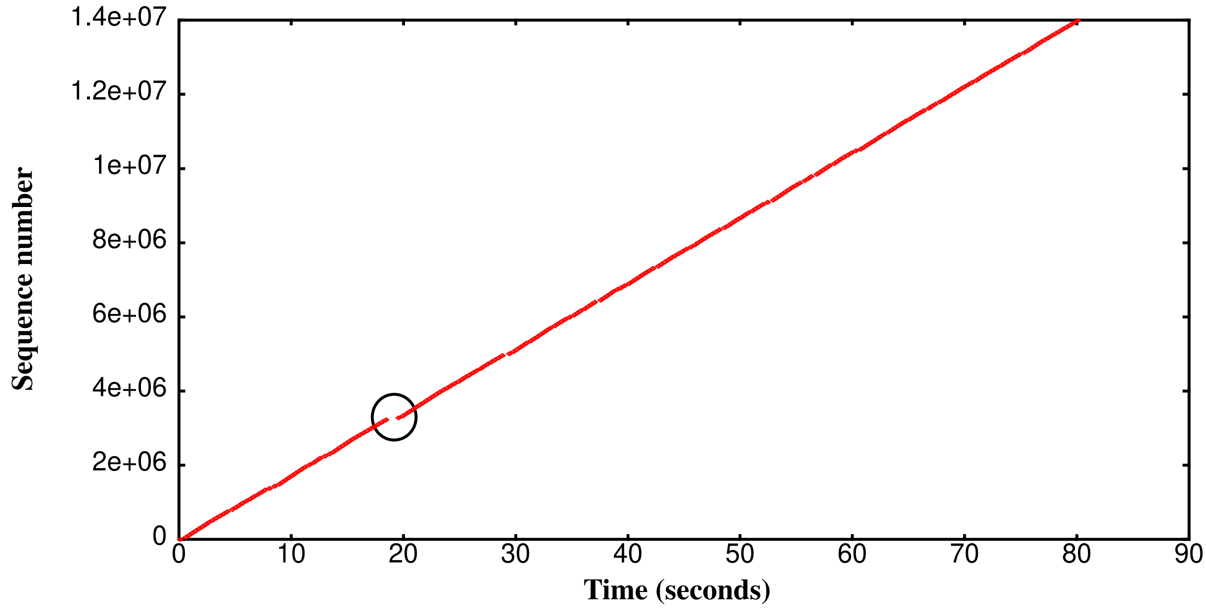

Figure 16. Sequence numbers for transfer to mobile host over WaveLAN with handoffs every 10 seconds.

every twentieth packet sequence number has been plotted. Most of the transfer is smooth and the handoffs are hardly noticeable on the graph (they take less than 15 ms). However, there is one place, marked by the circle, where the handoff latency is about 900 ms. In our experiments, this happens about 10-15% of the time when the transfer is at the highest possible rate.

An investigation of this situation reveals MAC-level effects of the WaveLAN to be largely responsible for this phenomenon. The WaveLAN is a shared medium and adjacent base stations contend for the link based on the CSMA/CA algorithm. The problem is that the implementation of this algorithm on the WaveLAN has some artifacts that exacerbate the inherent unfairness of the scheme. Under heavy load through one of the base stations, the other base station is denied access for much longer than it should even if it has packets to send. Its MAC-layer protocol continually performs exponential backoffs; when it finally becomes the forwarding base station, it has backed off to the point where the link remains idle for several hundred milliseconds.

We were able to reproduce this artifact several times and traced the times at which various packets were being sent through the WaveLAN interface. Recall that the beacons from the base stations are transmitted at a periodic rate of one packet per second, per base station. These beacons should therefore be transmitted and arrive at the mobile host through the WaveLAN interface once per second. Instead, several (between seven and ten) beacons from the new base station arrive all at once at the mobile host, after being queued at the old base station for many seconds (and experiencing the associated Ethernet backoffs). Unfortunately, the wireless link remains idle for between 200 and 950 ms when this happens, causing disruptions beyond the control of the network and higher layers of the protocol stack. This artifact is usually not observed when transfer rates are lower than about 1.1 Mbits/s, strongly suggesting that the MAC-layer of the WaveLAN does not degrade gracefully under heavy, shared load. Fortunately, this phenomenon only occurs a relatively small fraction of the time under heavy load.

5. Summary and Future Work

In this paper, we have presented a handoff protocol that achieves latencies between 8 and 15 ms and no data loss in the common case when handoffs are between base stations that are topologically close to each other. The protocol uses multicast for fast route updates and intelligent buffering at the base stations to achieve this performance. Applications such as real-time video playback and higher-level protocols such as TCP do not notice any disruptions in transmission during a handoff.

We are working on techniques to make this protocol more scalable. This includes a quantitative analysis and measurement of the resource utilization in the wired network and base stations, and techniques to reduce this consumption. As part of the Daedalus project, we are also investigating ways of having multiple wireless communication devices on a mobile host, each with different network characteristics and coverage, and performing handoffs to another interface when the quality of the network deteriorates on a given device. We expect to develop algorithms based on monitoring the status and quality of the communication channel to perform such *vertical handoffs*.

6. References

[AN94] A. S. Acampora and M. Naghshineh. An Architecture and Methodology for Mobile-Executed Handoff in Cellular ATM. *IEEE Journal on Selected Areas in Communications*, 12(8):1365–1375, October 1994.

[ATT] American Telephone and Telegraph. *WaveLAN: PC/AT Card Installation and Operation*.

[BSK95] H. Balakrishnan, S. Seshan, and R.H. Katz. Improving Reliable Transport and Handoff Performance in Cellular Wireless Networks. *ACM Wireless Networks*, 1(4), December 1995.

[CI94] R. Caceres and L. Iftode. Improving the Performance of Reliable Transport Protocols in Mobile Computing Environments. *IEEE JSAC*, 13(5), June 1995.

[Dae95] The Daedalus Project Home Page. http://daedalus.CS.Berkeley.EDU/, 1995.

[Dee89] Steve Deering. *Host Extensions for IP Multicasting*. RFC, SRI International, Menlo Park, CA, Aug 1989. RFC-1112.

[Dee91] S. E. Deering. *Multicast Routing in a Datagram Internetwork*. PhD thesis, Stanford University, December 1991.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability.* W. H. Freeman and Company, New York, 1979.

[GS94] R. Ghai and S. Singh. An Architecture and Communications Protocol for Picocellular Networks. *IEEE Personal Communications Magazine*, 1(3):36–46, 1994.

[KMS+93] K. Keeton, B.A. Mah, S. Seshan, R.H. Katz, and D. Ferrari. Providing Connection-Oriented Service to Mobile Hosts. In *Proc. 1993 USENIX Symp. on Mobile and Location-Independent Computing*, August 1993.

[MJ95] S. McCanne and V. Jacobson. vic: A Flexible Framework for Packet Video. In *Proc. ACM Multimedia '95*, November 1995. to appear.

[MJP95] A. Myles, D.B. Johnson, and C. Perkins. A Mobile Host Protocol Supporting Route Optimization and Authentication. *IEEE Journal on Selected Areas in Communications*, 13(5), June 1995.

[MS95] A. Merchant and B. Sengupta. Assignment of cells to switches in pcs networks. *IEEE/ACM Transactions on Networking*, 3(5):521–526, October 1995.

[Ses95] S. Seshan. *Low Latency Handoffs in Cellular Data Networks*. PhD thesis, University of California at Berkeley, December 1995.