Baltzer Journals November 13, 1997

Fast and Scalable Wireless Handoffs in Support of Mobile Internet Audio

Ramón Cáceres
AT&T Labs
180 Park Avenue
Florham Park, NJ 07932, USA
ramon@research.att.com

Venkata N. Padmanabhan
Computer Science Division
University of California
Berkeley, CA 94720, USA
padmanab@cs.berkeley.edu

Future internetworks will include large numbers of portable devices moving among small wireless cells. We propose a hierarchical mobility management scheme for such networks. Our scheme exploits locality in user mobility to restrict handoff processing to the vicinity of a mobile node. It thus reduces handoff latency and the load on the internetwork. Our design is based on the Internet Protocol (IP) and is compatible with the Mobile IP standard.

We also present experimental results for the lowest level of the hierarchy. We implemented our local handoff mechanism on Unix-based portable computers and base stations, and evaluated its performance on a WaveLAN network. These experiments show that our handoffs are fast enough to avoid noticeable disruptions in interactive audio traffic. For example, our handoff protocol completes less than 10 milliseconds after a mobile node initiates it. Our mechanism also recovers from packet losses suffered during the transition from one cell to another. This work helps extend Internet telephony and teleconferencing to mobile devices that communicate over wireless networks.

1 Introduction

Future internetworks will include networks of small wireless cells populated by large numbers of portable devices. Laptop computers and cellular telephones have proven their utility, while continuing advances in miniaturization promise increasingly functional portable devices. Networks of small wireless cells offer high aggregate bandwidth, support low-powered mobile transceivers, and provide accurate location information. In these networks, users will often carry devices across cell boundaries in the midst of data transfers. A *handoff* mechanism is needed to maintain connectivity as devices move, while minimizing disruption to ongoing transfers. This mechanism should exhibit low latency, incur little or no data loss, and scale to a large internetwork.

The Mobile IP standard [22] specifies a general handoff protocol for the Internet, but does not fully meet these goals. Mobile IP can handle both local-area and wide-area movement in both wired and wireless networks. However, it requires that a mobile node's home network be notified of every change of location. The route optimization extensions to Mobile IP [15] further require that every new location be registered with hosts that are actively communicating with the mobile node. These location updates incur the latency of traveling to the possibly distant home network and communicating hosts. They also add traffic to the wide-area portion of the internetwork. As currently defined, therefore, Mobile IP does not extend well to large numbers of portable devices moving frequently between small cells.

In this paper, we propose a scalable mobility management scheme for wireless internetworks. We employ a hierarchy to exploit the geographic locality inherent in user mobility patterns. The lowest level of the hierarchy optimizes the common case of movement between adjacent cells in the same subnetwork. It operates in the area near the mobile node and handles such motion without involving the mobile node's home network. The higher levels of the hierarchy rely on a more general mechanism like Mobile IP to handle the less frequent movement between subnetworks or administrative domains. Our scheme restricts mobility management traffic to the immediate area surrounding the mobile node, and thus reduces handoff latency and the load on the internetwork.

Our handoff scheme applies to the Internet. It is based on IP [26] and integrates seamlessly with Mobile IP. Our base stations are network-layer routers, in contrast to the link-layer bridges commonly used by existing local-area wireless networks. These base stations use the *gratuitous* and *proxy* features of the Address Resolution Protocol (ARP) [24] to maintain the illusion that wireless hosts reside on a wired link.

An IP-based routing solution offers two advantages over the link-layer bridging approach. First, a base station can filter traffic based on IP multicast groups. It can participate in the Internet Group Management Protocol (IGMP) [8] and forward only that multicast traffic for which there is an interested receiver. Such filtering is especially important given the limited bandwidth of wireless links. Second, a base station can differentiate between packet types based on the *Type of Service* field in current IP headers, or the *Flow ID* field in next-generation IP headers [9]. Careful packet scheduling is especially important at the interface between a wired link and a slower wireless link.

This paper also presents the design and implementation of a fast and reliable handoff mechanism for the lowest level of our hierarchy. We use a lightweight handoff protocol between base stations and mobile nodes to achieve low latency. We also use retransmission buffers in base stations and mobile nodes to recover from packet losses incurred during the transition between cells.

Our handoff mechanism aims to maintain the quality of active traffic streams, in particular voice traffic generated by Internet telephony and teleconferencing applications [18] [29]. Although people do not typically read text or watch video while they move, they often talk while they walk or drive. Consider, for example, the increasing popularity of cellular and cordless phones. Our handoff mechanism should thus satisfy the stringent requirements that interactive speech places on communication delay, jitter, and loss. We believe that a mechanism that works well for interactive speech will also work well for less demanding applications like stored-audio playback, slow-scan video transmission, and reliable data transfer.

We measured the performance of our handoff implementation using a WaveLAN local-area wireless network and Unix-based mobile nodes and base stations. A handoff completes less than 10 milliseconds after a mobile node initiates it based on beacons received from base stations. We also explored the tradeoff between more frequent beacons and smaller retransmission buffers. Our experiments show that a 100-millisecond beacon period, together with a 4-packet buffer per active application-layer conversation, provide good quality of service to packet audio applications. Our results hold even when cells do not overlap enough to allow a handoff to complete before a mobile node loses connectivity with its previous base station.

The rest of this paper is organized as follows. Section 2 surveys related work. Section 3 presents our mobility management hierarchy. Section 4 describes our implementation of the lowest level of the hierarchy. Section 5 reports the experimental performance of the implementation. Section 6 points out some areas for future work, and Section 7 concludes the paper.

2 Related Work

The problem of excessive mobility management traffic has been recognized in cellular telephone networks and proposed Personal Communication Services (PCS) networks. Meier-Hellstern et al. [20] calculate that cellular telephone networks carry many times more signaling traffic than wired telephone networks (4 to 11 times with their sample parameters) because of mobility management operations. They also predict that PCS networks will in turn carry several times more signaling traffic than cellular networks (3 to 4 times with their sample parameters) because of the smaller cell size and higher device density.

Hierarchical mobility management schemes have been proposed to reduce signaling load in these connection-oriented networks. The Global System for Mobile Communications (GSM) [30] and IS-41 [12] cellular standards use Home Location Registers (HLR) and Visitor Location Registers (VLR) to implement mobile registration and tracking. Xie and Goodman [38] propose using a gateway VLR to limit mobility management traffic to the metropolitan area where a mobile currently resides. Jain and Lin [14] propose using an anchor VLR and a chain of forwarding pointers to reduce the number of location updates that travel to the HLR.

Hierarchical and low-latency handoff schemes have also been proposed for connection-oriented data networks, particularly for wireless Asynchronous Transfer Mode (ATM) networks. Acampora et al. [1] propose building a virtual connection tree covering base stations in a local area, with virtual circuits pre-established from the root of the tree to each base station. A handoff thus involves only switching to an already established virtual circuit. Eng et al. [11] use a route chaining technique to extend virtual circuits to the mobile's new location. They can later collapse these chains into a new virtual circuit to obtain a more efficient route. Agrawal et al. [2] propose another scheme based on extending and collapsing virtual circuits. Toh [34] treats groups of adjacent wireless cells as a cluster, and handles movement within a cluster in a single ATM switch that is connected to all the base stations in the cluster. He handles movement between clusters by re-routing virtual circuits at the last switch on the route common to both clusters.

There have been proposals for hierarchical and low-latency handoff in connectionless networks as well. DeSimone and Nanda [10] describe the scheme used in Cellular Digital Packet Data (CDPD), in which groups of base stations are connected to a single Mobile Data Intermediate Station (MDIS). Roaming between base stations in a group is handled locally by the MDIS.

Seshan [32] proposes a scheme in which a mobile's home agent encapsulates data destined for the mobile in multicast packets, and sends these packets to multiple base stations in close vicinity of the mobile. While only one base station actively forwards packets to the mobile, the others buffer recent packets and can quickly forward them to the mobile should a handoff occur. This scheme trades off increased use of buffer space in the base stations for reduced handoff latency and packet losses. The use of multicast relieves the home agent of detailed knowledge of the mobile's current location, but incurs the complexity of managing multicast groups as mobiles move. Some of the above schemes attempt to anticipate handoffs by using measurements of signal strength and knowledge of previous mobility patterns.

In this paper, we introduce a hierarchical handoff scheme for connectionless networks, in particular the Internet. We keep the common case of local handoffs simple to achieve low latency and high scalability. Our handoff mechanism does not multicast data, extend routes, or anticipate handoffs. Rather, it employs what we consider to be a minimal handoff protocol, and uses small retransmission buffers to recover from packet losses during handoffs. We aim to show that our handoffs meet the stated latency, reliability, and scalability goals without undue

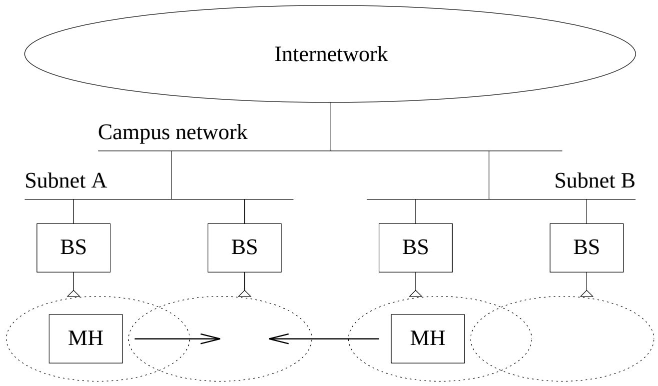

Figure 1: *Hierarchical organization of a wireless internetwork, including wired links, wireless cells, base stations, and mobile hosts.*

complexity.

3 Mobility Management Architecture

Wireless internetworks are typically organized as shown in Figure 1. Mobile hosts (MHs) communicate with base stations (BSs) over wireless links. BSs act as gateways between the wireless and wired links. As an MH moves, it may leave the wireless coverage area of one BS and enter that of another, prompting a handoff between neighboring BSs. Such handoffs could be frequent, especially in indoor environments where cells could be only a few meters in diameter.

Before a handoff, the MH may have been exchanging traffic with other hosts in the internetwork. This traffic could include real-time data, such as packet audio, and non-real-time data, such as background file transfers. Handoffs should complete quickly and incur little or no packet loss to avoid disrupting these traffic streams. Handoffs should also avoid loading the internetwork with excessive control traffic.

The proposed Mobile IP standard [22] requires that, whenever an MH changes the IP subnet to which it is attached, it send a location update message to a *home agent* in its home subnet. This message carries a *care-of address* in the new subnet where the MH can be reached. The home agent intercepts traffic for the mobile host that arrives in the home subnet, and forwards the traffic to the care-of address. The care-of address identifies a *foreign agent* in the foreign subnet. An MH's foreign agent can reside in the MH itself, or in a separate node that in turn forwards traffic to the MH. The proposed route optimization extensions to Mobile IP [15] further requires that location update messages be sent to correspondent hosts, that is, to hosts actively communicating with the MH.

We believe that using Mobile IP to handle all movement in a large internetwork would not meet the stated latency, reliability, and scalability goals. As just discussed, Mobile IP handoffs involve the exchange of control messages between a mobile host, its home agent, and its correspondent hosts. We make three observations. One, handoffs can incur long delays since these hosts and agents may be separated by many hops in a wide-area internetwork. Two, data in transit to the MH may be lost while the handoff completes and the new routes to the MH converge. Three, frequent handoffs by large numbers of mobile devices could add significant load to the internetwork. These observations motivate the hierarchical organization we describe next.

3.1 Hierarchical Mobility Management

Studies indicate there is significant geographic locality in user mobility patterns [17] [34]. For example, a large fraction of business professionals are frequently away from their desks, but spend most of that time within their own office buildings [17]. We should make handoffs fast and efficient in this common case.

Furthermore, when a mobile user is visiting a foreign administrative domain, there is little need to expose motion within that domain to the home agent or to correspondent hosts in other domains. We therefore argue that mobility management within an administrative or security domain should be separate from global mobility management. We propose a hierarchical mobility management scheme that separates three cases:

1. Local mobility.
2. Mobility within an administrative domain.
3. Global mobility.

The local mobility case handles movement between base stations on the same subnet and connected by a fast wired local-area network, such as an Ethernet, bridged Ethernet, or switched Ethernet. A number of such base stations and their associated wireless cells could cover a typical office building. This protocol confines its messages to the local subnet, and operates transparently to protocols at higher levels in the hierarchy.

The second level in the hierarchy is motivated by situations where there is mobility across subnets within the same administrative domain, for example, between buildings on a campus. There is again little need to expose this type of motion to the home domain. Mobile IP, however, requires that the home agent be notified of every change of subnet.

To handle motion within an administrative domain, we propose extending Mobile IP to include a hierarchy of foreign agents. Consider the case of a campus. Each subnet that a mobile host could visit would have one or more *subnet foreign agents*, just as in Mobile IP. There would also be a campus-wide *domain foreign agent*. The subnet foreign agents would include the address of the domain foreign agent in their agent advertisement messages. Mobile hosts that have been enhanced to understand hierarchical foreign agents would use this address as their care-of-address, which remains unchanged as long as they stay within the domain. The subnet foreign agents would also forward to the domain foreign agent any agent solicitation messages they receive. The domain foreign agent would maintain per-mobile host routing entries and update them whenever a mobile moves across subnets within its domain. Maintaining these per-host entries need not be a problem because suitable data structures exist that allow modern routers to handle several tens of thousands of entries efficiently (e.g., [33]).

The need for such hierarchies of foreign agents is a topic of discussion in the Mobile IP Working Group of the Internet Engineering Task Force. Similar hierarchies have been recently and independently proposed by Johnson [16] and Perkins [23]. An earlier proposal for hierarchical location updates was made by Aziz [3].

The third level in the hierarchy handles global mobility, that is, movement across administrative domains. In such cases, it will likely be necessary to inform the home domain of the movement because of security, billing, and other considerations. The interface between administrative domains, for example between the home domain and the domain foreign agent in the mobile host's visited domain, as well as between the home domain and correspondent hosts in other domains, remains exactly as in current Mobile IP. Therefore, global mobility can be handled by the unmodified Mobile IP protocol.

We believe that our mobility management architecture is general enough to handle a large class of mobility patterns in a scalable and efficient way. If we wish to remain compatible with the current Mobile IP standard, we can use a two-level hierarchy with the local mobility protocol layered below unmodified Mobile IP. In the rest of this paper, we discuss the design, implementation and performance of such a two-level hierarchy.

3.2 Local Handoff Protocol

Our local handoff protocol is designed to be simple and fast in order to attain good performance in the common case of handoff between base stations on the same subnet. Figure 2 depicts a mobile host moving between two wireless cells, while Figure 3 shows the associated message exchange. All nodes involved in the protocol have IP addresses in the same subnet.

Each BS broadcasts *beacons* over its wireless link. Each beacon carries the

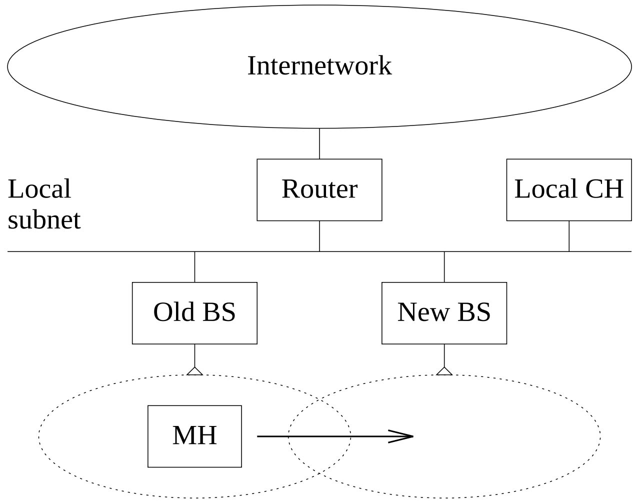

Figure 2: *A local handoff in which a mobile host moves between adjacent wireless cells. All nodes in the diagram have addresses in the same subnetwork.*

address of the BS that sent it. When an MH is in the overlap region of two cells, it will hear beacons from both BSs. Based on a metric such as wireless signal strength, the MH may decide to initiate a handoff from its current BS to a new BS. The ensuing message exchange and related processing are as follows:

1. The MH sends a `Greet` message to the new BS, conveying its own address as well as that of the old BS. It also makes the new BS its default gateway.

2. The new BS creates a routing table entry for the MH so that it can forward packets to the MH. It also responds with a `Greet Ack` message. When the MH receives this message, if it has packets recently sent to the old BS in its retransmission buffer, it sends them to new BS.

3. The new BS sends a `Notify` message over the wired link to the old BS to inform the old BS that the MH has moved. This message conveys the address of the new BS.

4. The old BS deletes its routing table entry for the MH. If it has packets recently sent to the MH in its retransmission buffer, it sends them to the new BS for forwarding to the MH. It also returns a `Notify Ack` message to the new BS.

5. The new BS broadcasts a `Redirect` message on the wired link to notify any interested nodes on that link that the MH has moved. Interested nodes include the router connecting the subnet to the wider internetwork and any

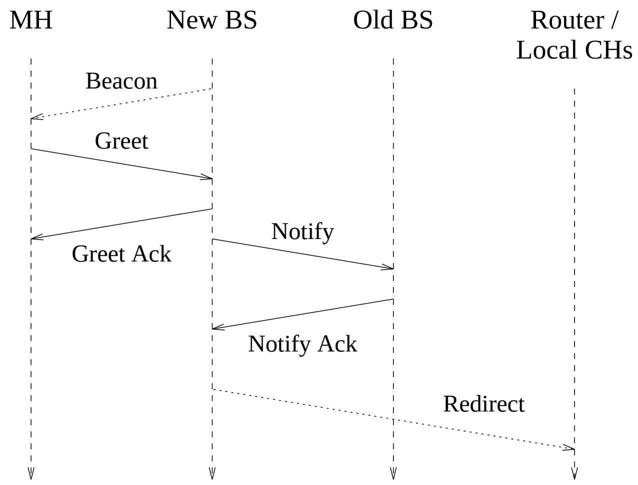

Figure 3: *Message exchange during a local handoff. Time flows downwards. Solid arrows denote point-to-point messages. Dotted arrows denote broadcast messages.*

local correspondent hosts. `Redirect` uses the standard ARP protocol so that no new functionality is required of interested nodes.

There are a number of points to note regarding the above protocol. First, the MH initiates the handoff. This is natural since it is in a good position to judge the quality of connectivity to different base stations. However, the amount of processing done by the MH is minimal — its involvement ends when it receives the `Greet Ack` message. The base stations do the bulk of the handoff processing, in keeping with the fact that they are part of the more resourceful wired infrastructure.

Second, retransmitting buffered packets from the old BS and from the MH is optional, but improves performance when cells do not overlap enough to allow handoffs to complete before the MH loses contact with its old BS. In general, some time elapses after the MH leaves the old cell but before the old BS and the MH realize it. As a result, the last few packets sent by the old BS over the wireless medium to the MH and vice versa may be lost. Retransmitting buffered packets after a handoff is intended to recover from these packet losses.

Third, base stations use a combination of *gratuitous* and *proxy* ARP to maintain the illusion that mobile nodes reside on the wired link (or the collection of transparently bridged wired links). A BS acts as a proxy ARP agent for any MHs in its cell. Any node on the wired link wanting to communicate with an MH issues an ARP request to obtain the link-layer address of the MH. The current BS for that MH responds with its own link-layer address on behalf of the MH, and the requesting node saves the information in its ARP cache. Subsequent packets for the MH are thus sent to the BS, which forwards them to the MH.

The new BS begins to act as a proxy ARP agent for the MH when it receives the `Greet` message from the MH. The old BS ceases to act as a proxy ARP agent for the MH once it receives the `Notify` message from the new BS. To complete the illusion that the MHs are on the wired link, the base stations also copy relevant broadcast messages between the wired and wireless links. Examples of these messages include Mobile IP *agent advertisements* and *agent solicitations*.

Fourth, the `Redirect` message shortens a 2-hop path into a 1-hop path, and prevents chains of forwarding pointers from forming. `Redirect` takes the form of a gratuitous proxy ARP packet. When an MH registers with a BS, the BS broadcasts over the wired link an ARP packet that maps the MH's network-level address to the BS's link-layer address. The ARP specification requires nodes to update their ARP caches with the information in new ARP broadcasts [24]. Therefore, any nodes with existing ARP entries for the MH that point to the old BS will update themselves to point to the new BS. They thus will send subsequent packets for the MH directly to the new BS.

There is no need for an explicit acknowledgement to the `Redirect` message because it is not essential for ensuring that data reaches the MH after a handoff. If the `Redirect` is lost, nodes with existing ARP entries for the MH will continue to send data to the old BS. The old BS will have been reliably notified of the handoff. It will forward the data to the new BS after issuing an ARP request for the MH, to which the new BS will respond. Alternatively, the new BS can issue two or three consecutive `Redirect` messages to increase the likelihood that the information reaches all relevant hosts.

Fifth, we note that ARP constitutes a robust and efficient mechanism for local mobility management. ARP is robust because it relies only on soft state. For example, it uses timeouts to clear cache entries and maintains itself without manual intervention. ARP is efficient because it uses broadcasting to update cache entries at multiple nodes with a single message. We have not, however, addressed the security weaknesses of ARP. Any host in a local area network can impersonate another host at the data link level by acting as a proxy ARP agent. Our scheme suffers from the same problem, in effect assuming that hosts on local area networks are trustworthy. Our handoff mechanism is no more and no less secure that ARP itself.

Finally, our mechanism extends cleanly to next-generation IP networks. In these networks, the functionality of ARP will be available through the *neighbor discovery* features of IPv6 [21].

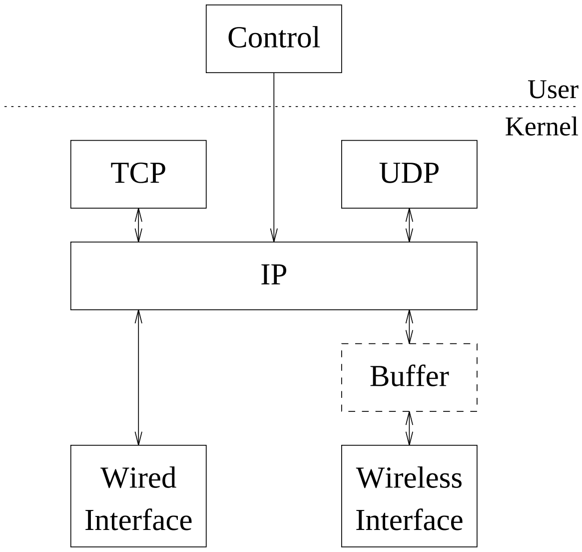

Figure 4: *Software modules in the Streams implementation at a base station.*

4 Implementation

We implemented our local handoff mechanism on a Solaris 2.4 software platform. All protocol elements reside in the Unix kernel under the Streams framework, while some control software resides in a user-level program at base stations. We pursued a kernel implementation to avoid the latency of user-kernel crossings. In keeping with the design of our protocol, the bulk of the implementation is on base stations and only a small portion is on mobile hosts. We discuss these two components separately.

4.1 Base Station

Figure 4 shows the organization of the software components in a base station. The three components that we created or modified were `Control`, `IP`, and `Buffer`. We discuss each of these in turn.

Control Program

The control program performs initialization and administration functions. It needs to pass information in only one direction to the IP kernel module, which is available as `/dev/ip` in the file system name space. Accordingly, we defined several new `ioctl` types, and use them to pass messages from the control program to the IP module. New `ioctl` types are defined for the following functions:

1. Set the beacon period (in milliseconds) and the destination address for the beacons (typically a broadcast address). These values can be changed dynamically.

2. Start and stop beaconing from this base station.

3. Set the number of packets to be buffered by the buffer module. This value can also be changed dynamically.

IP Module

A large portion of the handoff implementation resides in the IP module. This functionality includes generating and processing handoff protocol messages, processing the above `ioctl` types, and interacting with the buffer module.

Handoff protocol messages take the form of ICMP packets [27]. We defined new ICMP types for the following messages: `Beacon`, `Greet`, `Greet Ack`, `Notify`, and `Notify Ack`. The size of our ICMP packets is 42 bytes (the sum of the lengths of the ICMP, IP, and WaveLAN headers). We perform routing changes in response to `Greet` and `Notify` messages by calling kernel functions that manipulate the IP routing table.

We periodically send beacons using a timer mechanism available in the Solaris kernel. This periodicity provides a convenient way of ensuring the reliability of the protocol messages. Every beacon interval, we retransmit any unacknowledged protocol messages. This retransmission interval may need to be shorter than the beacon period if frequent packet losses are found to adversely affect handoff performance. We did not encounter this problem in our testbed.

Most of the `ioctl`s mentioned above are also processed by the IP module. Those in the first category only require setting some variables. Starting the beaconing process requires creating data structures to hold the various protocol messages, and setting the beacon timer. Stopping the beaconing process involves disabling the timer and other cleanup operations. The `ioctl` to set the size of the buffer is the only `ioctl` that is passed on to the buffer module.

Buffer Module

The buffer module is a standard Streams module created by us. It is optional in the sense that handoffs would work without it. It is there to enhance application performance by recovering from packet losses suffered during handoffs, especially when neighboring cells have little or no overlap.

The buffer module is inserted and removed from the system using a modified version of the `ifconfig` program, following the approach described by Wakeman et al. [36]. After the base station boots, `ifconfig` with the `unplumb` option is used to tear down any streams associated with the wireless interface. Then the modified `ifconfig` with the `plumb` option is used to rebuild these streams with the buffer module included below IP.

The buffer module resides below IP and above the wireless interface driver. Every IP packet sent over the wireless network passes through the buffer module. The module adds the packet to its buffer, using the `dupmsg` Streams routine, before passing the packet on to the wireless network driver. `dupmsg` increments a reference count for the Streams buffer containing the packet and returns a pointer to the buffer, but does not make a new copy of the packet. Buffering is thus fast since it incurs only a few pointer and counter manipulations. Pointers to all buffered packets are currently placed in a single FIFO queue. We may use a more sophisticated queueing discipline in the future.

Upon receiving a `Notify` message during a handoff, the IP module at the old BS sends a Streams message to the buffer module asking it to retransmit all buffered packets originally destined to the departing MH. The buffer module sends these packets back to IP, which routes them to the new BS as called for by the new route for the MH established during the handoff. IP is not aware that these are retransmitted packets.

4.2 Mobile Host

The handoff implementation at the MH is confined to small changes to the IP module, corresponding to the small amount of processing required of the MH by our local handoff mechanism, and an optional buffer module.

When an MH decides to do a handoff, it sends a `Greet` message to the new base station conveying the wired IP address of its current BS. To make this message reliable, the MH checks every time it receives a beacon whether it has received a `Greet Ack` corresponding to the previous `Greet` message. If not, it resends the `Greet` message. All these messages take the form of ICMP packets. When the MH begins a handoff, it modifies its routing table to make the new BS its default gateway. Also, if there are packets in the optional retransmission buffer, it retransmits them after it has received a `Greet Ack` message from the new base station.

We discuss how the MH decides to initiate a handoff when we describe our experimental setup below.

5 Experimental Results

We conducted a number of quantitative and qualitative experiments to evaluate the performance of our local handoff implementation. Our aim was to determine if our handoff mechanism can support interactive voice traffic, even in the extreme case of non-overlapping cells. We first describe the testbed used to conduct the experiments, then discuss the results.

5.1 Experimental Setup

Our configuration resembles that in Figure 2. Our experiments involve four nodes: a mobile host, two base stations, and a correspondent host on the same subnet. The mobile host connects to a 2-Megabits/second (Mbps) WaveLAN wireless network, while the correspondent host connects to a 10-Mbps Ethernet wired network. The base stations connect to both networks. The WaveLAN network was idle except for the traffic generated by our experiments, while the Ethernet was moderately loaded by regular use.

The base stations are Dell desktop PCs with a 90-MHz Pentium processor, an Ethernet adapter, and a WaveLAN card. The mobile host is a Toshiba laptop PC with a 75-MHz Pentium processor and a WaveLAN card. Finally, the correspondent host is a Sun SPARCstation 5 with a 70-MHz SPARC processor and an Ethernet interface. We used two other Pentium PCs to passively monitor the wired and wireless links. All these systems run the Solaris 2.4 (SunOS 5.4) operating system.

WaveLAN is a direct-sequence spread spectrum radio network with a raw bandwidth of 2 Mbps and a range of about 50 meters. In all our experiments, we used WaveLAN PCMCIA cards that operate in the 915-MHz ISM band. Maximum application-to-application throughput between two WaveLAN nodes was approximately 1.2 Mbps, while round-trip times averaged 7 milliseconds (ms).

For the handoff experiments presented in this paper, we simulate motion between two abutting but non-overlapping cells. This cell layout guarantees that the MH loses contact with its old BS before the handoff completes. Packets arriving at the old BS and destined for the MH will be lost between when the MH leaves the old cell and when the handoff completes. This is the worst case scenario for handoffs assuming there are no coverage gaps, i.e. adjacent cells abut on each other. We believe that this is a reasonable assumption in most environments, especially considering that adjacent cells in reality will have some overlap which would reduce the chances of there being dead zones. Moreover, there is little that can be done if a mobile is indeed in a dead zone for long.

The drive towards smaller cells also tends to shrink the overlap region between cells. There is a trend towards smaller cells because they offer advantages in terms of aggregate throughput, power consumed by mobile transceivers, and accuracy of location information. We therefore believe that it is important to experiment with the case where handoffs do not complete before an MH loses contact with its old BS. The scenario we reproduce could also be the common case in infrared networks that require line-of-sight connectivity.

In our testbed, the MH is always in range of both BSs. The BSs are in two different rooms along one hallway, while the MH is in a third room across the hall. The MH continuously listens for beacons and checks to see if the latest beacon received was sent by the current base station. If so, it ignores the beacon. Otherwise, it initiates a handoff to the new base station.

During our tests, we arrange through software for exactly one BS to beacon at any one time, and we trigger handoffs by ceasing to beacon from one BS and starting to beacon from another. This setup allows us to control the instant when handoffs take place and thus to reliably reproduce test conditions. It also does away with the need to physically move test machines during experiments.

Nevertheless, we have on our MHs a separate handoff implementation that uses signal strength measurements reported by the wireless interface hardware. The software on the MH incorporates hysteresis to avoid thrashing when an MH is in the region of overlap between two cells and signals from the two base stations have similar strengths. These handoffs follow the WaveLAN specifications [37] and interoperate with commercial WaveLAN base stations, or WavePoints. WavePoints are link-layer bridges that communicate over a shared wired link to carry out a handoff. We have verified that our WaveLAN-standard handoffs work when an MH moves away from one WavePoint and towards another.

We are modifying our signal-strength monitoring software so that it triggers the network-layer handoffs proposed in this paper. In particular, the WaveLAN driver on the MH decides when a handoff should take place, then notifies the IP module via a Streams message. The IP module is not the appropriate place to process raw signal strength measurements for a specific wireless network like WaveLAN, since IP should remain independent of any one link technology. Rather, the IP module will receive a generic handoff notification from the underlying wireless interface driver, then initiate the network-layer handoff described earlier.

We note that our experiments primarily involved one-way audio streams and one-way data transfers to the MH. During handoffs, therefore, only the retransmission buffer at the old BS (and not the one on the MH) came into play.

5.2 Handoff Mechanism Performance

The time to complete a handoff has two components: the *rendezvous time* and the *protocol time*. Rendezvous time refers to the period from when the mobile leaves the coverage area of its current base station until it hears a beacon from a new base station. In a wireless network engineered to have no dead zones between cells, the worst case value for this quantity equals the beacon period. In our experiments, we varied the beacon period from 10 ms. to 1 s. and arranged for the rendezvous time to always equal the beacon period.

Protocol time is that required to restore the flow of traffic after the mobile receives the beacon that triggers a handoff. This time includes the exchange of `Greet`, `Greet Ack`, `Notify`, and `Notify Ack` messages to effect routing changes

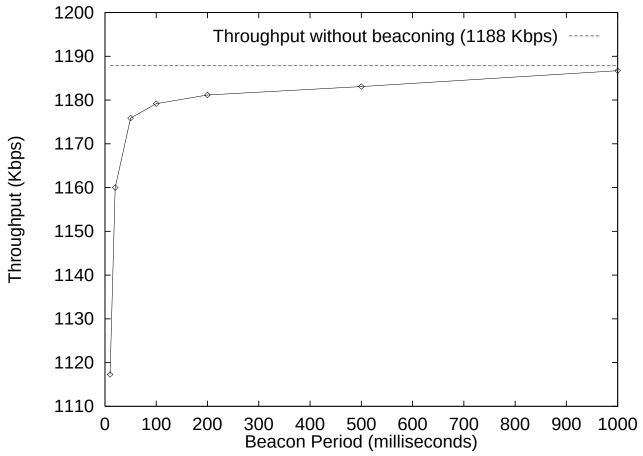

Figure 5: *Impact of beacons on TCP throughput, as measured by a* `ttcp` *receiver. Throughput begins to drop significantly when the beacon period drops below 50 ms.*

at the old and new base stations. The time to send and process the `Redirect` message is not included since those operations are not critical to a correct handoff.

Our measurements show that protocol time is less than 10 ms. In particular, the mean protocol time over a 10-handoff test was 9.33 ms., with a standard deviation of 0.25. Therefore, in our experiments, total handoff time is equal to the beacon period plus approximately 10 ms.

We also measured the impact of beacons on application-layer throughput as measured by the public-domain `ttcp` benchmark [35]. The concern here is that the overhead of frequent beacons on the wireless medium may adversely affect application performance. `ttcp` measures Transmission Control Protocol (TCP) [28] throughput between two hosts by setting up a connection, sending a specified amount of data, and closing the connection. Our `ttcp` experiments involved sending 4 Mbytes of data in 1024-byte segments. We report the throughput measured at the receiver, averaged over 10 experiments in which one base station sent periodic beacons while the other relayed TCP data from the correspondent host to the mobile host.

As shown in in Figure 5, TCP throughput begins to drop noticeably (and by a larger fraction than the beacons' share of the total available bandwidth) when the beacon period drops below 50 ms, but remains high when the period is above 50 ms. For example, throughput remains above 99% of maximum when the period is 100 ms. With a beacon period of 10 ms, the beacon bandwidth is 33.6 Kbps, which is less than 3% of the total. But TCP throughput drops by more than 6% compared to the case with no beaconing. The reason for this disproportionate drop is the increased chance that beacon packets cause the WaveLAN MAC layer

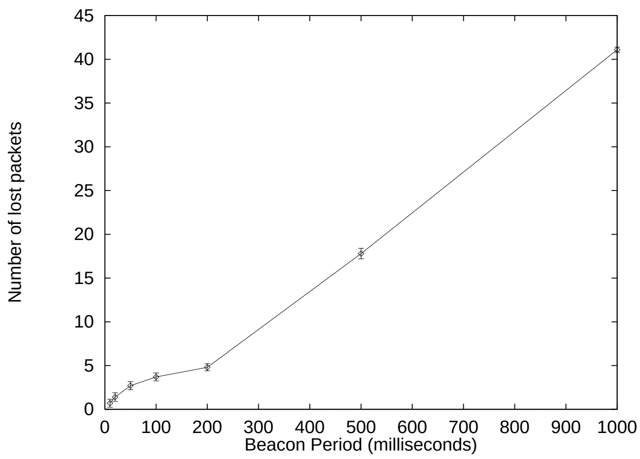

Figure 6: *Number of packets lost due to a handoff, when there is no buffering or retransmission, from a stream of UDP packets that emulates an audio stream.*

of the base station that is transmitting TCP packets to back off. Thus, our conclusion from Figure 5 is that we are free to pursue the latency benefits of low rendezvous times through the use of beacon periods as low as 50 ms.

Finally, we evaluated the performance overhead of the buffer module. Again using the `ttcp` benchmark, we found that there was no appreciable difference between TCP throughput with the buffer module included and TCP throughput without the buffer module, even as the number of packets buffered reached 30. This is not surprising considering that only a few pointer and counter manipulations are involved in buffering each packet. We conclude that interposing the buffer module between IP and the wireless interface in the protocol stack results in negligible *overhead*. We discuss the performance *benefits* of buffering in the presence of handoffs in the following subsections.

5.3 Packet Audio Performance

The main goal of our experiments was to evaluate the effects of handoffs on Internet audio applications. Examples of these applications are `vat` (visual audio tool) [13] and `nevot` (network voice terminal) [31], built on top of the User Datagram Protocol (UDP) [25] and commonly used over the Multicast Backbone (MBone) [18].

We wrote a simple benchmark program, `udpbench`, that sends a stream of UDP packets from one host to another. It reports at the receiving host any lost, duplicated, or out of order packets, along with packet interarrival times. It is these quantities that are of most interest to us in the packet audio case. Throughput is less important since audio traffic is not bandwidth-intensive.

In our quantitative handoff experiments, we used `udpbench` to send data from the local correspondent host to the mobile host. We made `udpbench` emulate the packet stream produced by `vat`. Of the several audio packet formats supported by `vat`, we chose *pcm* as the worst case. *pcm* sends packets with the least interarrival time, 20 ms., and requires the most bandwidth, 78 Kbit/sec. It uses an average packet size of 200 bytes. `vat` assigns application-level sequence numbers to individual audio packets, which allows it to detect and respond appropriately to missing, reordered, or duplicate packets. We use sequence numbers for the same purpose in `udpbench`.

Packet audio applications like `vat` employ a *playout delay* mechanism by which the receiver delays playing out the audio contained in an arriving packet for some amount of time. This mechanism is used to smooth out variations in packet interarrival times, which are common in an internetwork environment. The corresponding playout buffer is sized to match the measured jitter in the network. The value of playout delay also depends on whether `vat` is operating in *conference* mode or in *lecture* mode. The playout delay needs to be smaller in the former case as it involves an interactive conversation. We measured the playout delay for particular MBone sessions that employed `vat`. For example, it was approximately 100 ms. for a local conference and 4–5 s. for a lecture from a distant host. To put this in perspective, human factors studies have shown that the maximum tolerable delay for interactive conversations is approximately 200 ms.

The playout delay mechanism motivates us to investigate the tradeoffs between beacon period and buffer size. On the one hand, small beacon periods result in fast handoffs with little packet loss. They save on retransmission buffers since there is little loss to recover from, but use more bandwidth and processing overhead. On the other hand, large beacon periods result in slow handoffs with significant packet loss. They save on bandwidth and processing overhead, but call for large retransmission buffers. We aim to find a combination of beacon period and buffer size that introduces handoff jitter small enough to be absorbed by the playout buffer, without consuming inordinate amounts of resources.

Figure 6 shows the number of audio packets lost during a single handoff for different values of beacon period when there is no buffering or retransmission. The points on the curve are averages over 10 experiments. The error bars above and below these points mark plus and minus one standard deviation from the mean. For example, for a 100-ms beacon period, 4 audio packets are typically lost and 3 packets are occasionally lost, for a mean of 3.7 and a standard deviation of 0.46.

We also ran a number of qualitative experiments in which we listened through headphones to a `nevot` audio stream flowing from the local correspondent host to the mobile host. These experiments showed that although a single lost packet is barely perceptible by a human listener, 3 or 4 consecutive packet losses are readily noticed and significantly degrade audio quality.

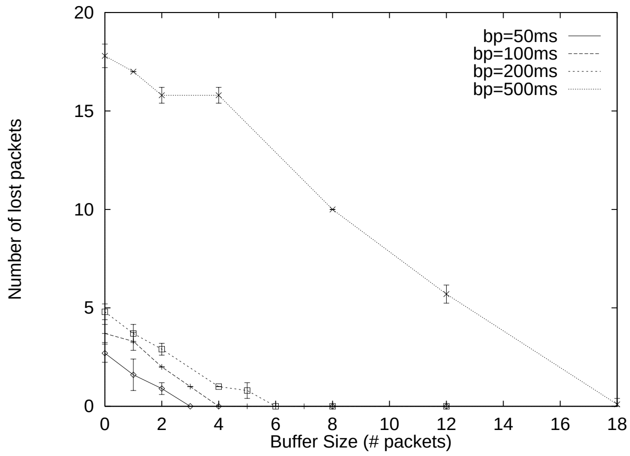

Figure 7: *Number of packets lost due to a handoff, from a stream of UDP packets that emulates an audio stream.* bp *denotes the beacon period.*

Continuing with our quantitative handoff experiments, Figure 7 shows the number of audio packets lost during a single handoff for different values of beacon period and buffer size. Two obvious trends are that losses decrease with decreasing beacon period and increasing buffer size. However, a buffer larger than the actual number of packets lost results in duplicate packets sent to the mobile.

Figure 8 shows how the number of duplicate packets increases with decreasing beacon period and increasing buffer size. These duplicate packets would be correctly discarded through the use of application-level sequence numbers by Internet audio applications such as `vat`, but they waste wireless bandwidth and processing at the mobile host. We need to balance the conflicting goals of minimizing losses while keeping the number of duplicates low. Our experiments also indicate that there is little reordering of packets due to handoffs.

Large retransmission buffers cause further problems not shown in the figures. In our testbed, retransmitting more than 8 packets after a handoff leads to some of the retransmitted packets themselves being lost before they reach the mobile host. This burst of back-to-back packets arrives at the new base station at the wired link speed, but must travel to the mobile host at the slower wireless link speed. As a result, queues at the new base station overflow and packets are lost. In general, we should keep handoff buffers small to reduce memory, processing, and bandwidth requirements.

We conclude from Figures 5–8 and the above discussion that beacon periods of 50–200 ms. and buffer sizes of 2–6 packets are feasible operating points. Our results further allow us to choose particular combinations of beacon period and buffer size that achieve the desired latency and reliability goals.

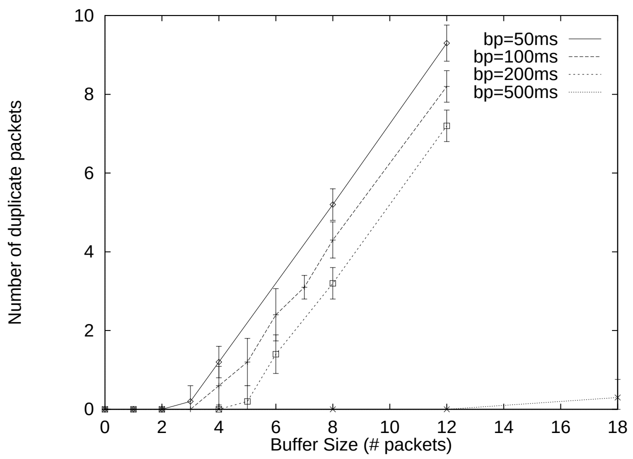

Figure 8: *Number of packets duplicated due to a handoff, from a stream of UDP packets that emulates an audio stream.* bp *denotes the beacon period.*

We now examine the case of 100-ms beacons and a 4-packet buffer to show that our handoff mechanism in fact meets the requirements of interactive voice applications. Figure 9 shows packet interarrival times vs. application-level sequence numbers during one such handoff. The normal interpacket spacing is 20 ms., corresponding to that used by the *pcm* format. The first packet after the handoff, sequence number 56, is delayed by approximately 100 ms. while the handoff completes. The following few packets have interarrival times below 10 ms. These packets were retransmitted back-to-back after the handoff. Finally, interpacket spacing returns to its normal value of 20 ms.

There are several points to note regarding these results. First, the jitter introduced by the handoff is well below the limit imposed by the playout buffer. To avoid interruptions in the audio playback, the playout buffer at the receiver should not empty before the first packet after a handoff arrives. In addition, the playout delay for an interactive conversation should not be more than 200 ms. The maximum interarrival time in Figure 9 is approximately 100 ms., leaving room for other jitter in the wider internetwork.

Second, we note from the sequence numbers that no packets were lost, misordered, or duplicated during the handoff. We have made this situation common by setting the buffer size equal to the expected number of packets lost, which we obtained from Figure 6. However, a single packet is occasionally duplicated due to the variability shown by the error bars in Figure 8. Internet audio applications are programmed to deal with missing, duplicate, and misordered packets, so we consider these anomalies to be acceptable.

Third, qualitative experiments confirm that these handoffs are imperceptible

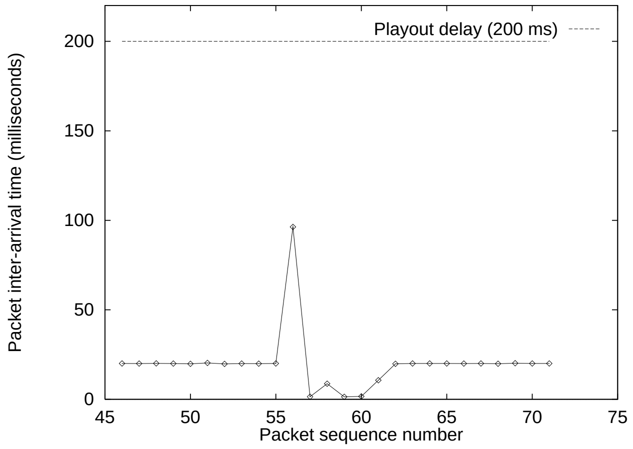

Figure 9: *Packet interarrival times during a handoff, from a stream of UDP packets that emulates an audio stream. All packets arrived in order, without loss or duplication. The jitter added by the handoff is well below the limit imposed by the playout buffer at the receiver. Qualitative experiments confirm that these handoffs are imperceptible to human listeners.*

to human listeners. We listened through headphones to a `nevot` audio stream flowing from the local correspondent host to the mobile host, and did not perceive any effect on the audio stream when the mobile host performed a handoff.

Finally, we recall that our experiments simulate the case of adjacent but non-overlapping cells. Cellular networks should avoid large overlap regions to maximize aggregate bandwidth and minimize the number of base stations deployed. At the same time, however, cellular networks should avoid gaps between cells to provide full coverage. In networks with overlapping cells, it is possible to complete a handoff before a mobile node loses contact with its old base station. In those cases, there should be no packets lost during a typical handoff. The speed of our handoffs in fact make such soft handoffs more likely. Therefore, we expect handoffs to introduce less disruption to audio streams in many real networks than in the scenarios we have presented here.

5.4 Reliable Transport Protocol Performance

Fast and reliable handoffs help the performance of other applications besides audio. Important examples are applications that use reliable transport protocols like TCP. We used the `ttcp` benchmark to measure TCP throughput in the presence of handoffs. In our experiments, we triggered 3 handoffs, 6 seconds apart, during the roughly 28-second lifetime of a 4-Mbyte transfer. Again, one base station sent beacons while the other relayed TCP data from the correspondent host to the mobile host.

Our measurements show improvements due to the buffering and retransmission mechanism, as expected. For example, using a 200-ms beacon period, throughput with an 8-packet buffer is 8% higher than throughput without buffering and retransmissions. The reason is that the handoff mechanism locally recovers from packet losses due to handoffs, thus relieving the end-to-end transport protocol from the task. Aside from the fact that the end-to-end mechanism takes more time to become aware of a loss, TCP reacts to losses by initiating congestion control procedures that further reduce throughput [6].

However, our measurements did not show the expected throughput improvements due to lowering the beacon period because of problems with the Media Access Control (MAC) procedures used by WaveLAN. More frequent beacons should increase throughput because they lower the rendezvous time, resulting in fewer packets lost during a handoff. In our testbed, however, there was no noticeable increase in throughput with decreasing beacon period.

We identified the cause as interference with the beaconing process caused by the TCP traffic stream. We observed severe clustering of beacons arriving at the mobile host when a TCP stream from one WaveLAN transmitter competed for the wireless medium with the beacons from another transmitter. Instead of arriving at the regular intervals dictated by the timer at the base station, beacons sometimes arrived in clumps. The long-term average of beacon interarrival times was close to the expected value, but there were large deviations. Seshan [32] has also noted a similar behavior in WaveLAN.

Like other current wireless networks, WaveLAN uses a Carrier Sense Multiple Access/Collision Avoidance (CSMA/CA) scheme that allows a station to obtain an unfair share of the channel. After transmitting each frame, a station that has another frame to send waits for approximately 16 antenna slots. If the medium is still free, the station sends the frame. In contrast, a station that finds the medium busy backs off a random number of slots between 0 and 32, and doubles that range every consecutive time it finds the medium busy.

This scheme makes it possible for a station that has a lot of data to send to repeatedly acquire the channel at the expense of other stations. In our throughput experiments, the base station relaying TCP data sends a continuous stream of packets with near-maximum size. In contrast, the base station that is beaconing attempts to send a periodic sequence of small packets. As a result of the interference suffered by beacons, rendezvous times in our experiments were often much larger than desired, resulting in increased packet losses during handoffs and thus in lower than expected throughput.

We repeated our handoff experiments using a second Ethernet to emulate a wireless network, and verified that the problems just described were absent. In addition to confirming the packet audio results presented earlier, our Ethernet measurements exhibited the expected throughput effects. First, faster beaconing resulted in higher throughput. Second, buffering and retransmitting further improved throughput, particularly for short beacon periods and small buffer sizes.

We conclude that the throughput problems we observed were due to MAC-layer effects and are independent of our handoff mechanism. MAC procedures have been proposed that provide fair access to a shared wireless medium, for example those based on a Request to Send–Clear to Send (RTS–CTS) etiquette [4]. Our results motivate the use of these improved MAC procedures, not only to achieve fairness among application data streams, but also to reduce handoff jitter.

6 Future Work

We can extend the work presented in this paper in a number of ways. First, it would be worthwhile to investigate support for multicasting in base stations. An advantage of base stations that are network-layer routers is that they can more easily filter packets based on IP multicast groups. Multicast groups provide an effective criterion on which to base forwarding decisions, namely whether any mobile hosts in a cell are subscribed to a group. For example, different multicast groups could be used to carry different layers of a video stream encoded with layered coding techniques [19]. A mobile host could subscribe only to the subset of layers appropriate to the bandwidth limitations of its wireless link, and the base station would forward only that subset.

Second, it would be useful to retransmit after a handoff only those packets that were lost during the handoff. The handoff implementation presented in this paper uses its own retransmission buffer. The buffer size is tuned to the number of expected packet losses during a handoff, and the complete buffer is retransmitted after every handoff. As we have seen, this procedure can lead to duplicates.

One alternative is to integrate the handoff mechanism with a reliable link-layer protocol. The buffer maintained by a link-layer Automatic Repeat Request (ARQ) protocol to recover from wireless transmission losses could be reused to recover from handoff losses as well. After a handoff, the ARQ state could be used to retransmit only unacknowledged packets.

Another alternative is to use the *ID* field in the IP header. Base stations can buffer the last few IP packets sent to a mobile, as before. Mobiles keep track of the *ID*s of the last few packets it has received. During a handoff, mobiles send to the new base station that list of *ID*s as part of the `Greet` message. The new base station forwards the list to the old base station as part of the `Notify` message. The old base station then resends only those buffered packets that do not appear in the list of *ID*s. This technique has been used by Seshan [32].

Third, it may be beneficial for the handoff mechanism to differentiate between traffic types. For instance, during a handoff, buffered packets from delay-sensitive traffic streams such as audio could be forwarded before those from other traffic streams such as reliable data transfers. Base stations could differentiate traffic types through implicit information like packet size, or through explicit information like the Type of Service field in current IP headers.

Plans for offering integrated services in the Internet [5] call for routers to apply different scheduling policies to different traffic streams. Next-generation IP headers will contain a *Flow ID* field to identify packets from different traffic streams. The fact that our base stations are IP routers facilitates implementation of these policies at the crucial interface between wired networks and slower wireless networks. Base stations could thus differentiate between traffic types not only during handoffs but also during normal routing.

Finally, a hierarchical mobility management scheme allows the use of small cells at the lowest level without the penalty of expensive handoff processing. Small cells in turn can provide useful location information that may help, for instance, in anticipating and reducing the latency of handoffs at higher levels in the hierarchy.

7 Conclusions

This paper makes two main points. First, it argues that a hierarchical mobility management scheme is necessary for latency and scalability reasons in a world of ubiquitous portable devices that communicate over a large wireless internetwork. Second, it shows that a simple handoff mechanism at the lowest level of the hierarchy can be made fast and reliable enough to support the stringent demands of interactive audio applications.

We have presented the design and implementation of a fast, reliable, and scalable mobility management scheme for wireless internetworks. Our handoffs complete less than 10 milliseconds after a beacon from a base station reaches a mobile node. Our experiments show that a 100-millisecond beaconing period, together with a 4-packet buffer per conversation, provides good quality of service to packet audio applications. Our scheme integrates seamlessly with IP and Mobile IP, and thus applies to the Internet.

The emergence of Internet telephony and teleconferencing has demonstrated that the Internet can provide acceptable quality of service to an important class of real-time applications. This work helps extend these applications to mobile devices that communicate over wireless networks.

Acknowledgements

Johnathan Reason ran the qualitative audio experiments. Ramachandran Ramjee ported `nevot` to the Toshiba laptops. Herman Chen, Scott Miller, and Thomas Woo helped with Solaris for x86 and with WaveLAN. Robert Baron provided the WaveLAN air interface document.

References

[1] A. S. Acampora and M. Naghshineh, "An Architecture and Methodology for Mobile-Executed Handoff in Cellular ATM Networks," *IEEE Journal of Selected Areas in Communications*, Vol. 12, No. 8, October 1994.

[2] P. Agrawal, P. P. Mishra, and M. B. Srivastava, "Network Architecture for Mobile and Wireless ATM," *Proc. of the 16th IEEE Intl. Conf. on Distributed Computing Systems*, May 1996.

[3] A. Aziz, "A Scalable and Efficient Intra-Domain Tunneling Mobile-IP Scheme," *ACM Computer Communications Review,* January, 1994.

[4] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang, "MACAW: A Medium Access Protocol for Wireless LANs," *Proc. of ACM SIGCOMM '94,* August 1994.

[5] R. Braden, D. Clark, and S. Shenker, "Integrated Services in the Internet Architecture: An Overview," *RFC 1633*, Internet Engineering Task Force, July 1994.

[6] R. Cáceres and L. Iftode, "Improving the Performance of Reliable Transport Protocols in Mobile Computing Environments," *IEEE Journal of Selected Areas in Communications*, Vol. 13, No. 5, June 1995.

[7] A. Conta and S. Deering, "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification," *RFC 1885*, Internet Engineering Task Force, December 1995.

[8] S. Deering, "Host Extensions for IP Multicasting," *RFC 1112*, Internet Engineering Task Force, August 1989.

[9] S. Deering and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification," *RFC 1883*, Internet Engineering Task Force, December 1995.

[10] A. DeSimone, S. Nanda, "Wireless Data: Systems, Standards, Services," *ACM/Baltzer Wireless Networks Journal*, October 1995.

[11] K. Y. Eng, M. Karol, M. Veeraraghavan, E. Ayanoglu, C. Woodworth, and R. A. Valenzuela, "A Wireless Broadband Ad-Hoc ATM Local-Area Network," *ACM/Baltzer Wireless Networks Journal,* May 1995.

[12] "IS-41 Cellular Radio-Telecommunications Intersystems Operations," Electronics Industry Association/Telecommunications Industry Association, January 1993.

[13] V. Jacobson, and S. McCanne, "Visual Audio Tool," Lawrence Berkeley National Laboratory.

[14] R. Jain and Y. B. Lin, "An Auxiliary User Location Strategy Employing Forwarding Pointers to Reduce Network Impacts of PCS," *Record of the 1995 IEEE Intl. Conf. on Communications*, June 1995.

[15] D. B. Johnson and C. Perkins, "Route Optimization in Mobile IP," *Internet Draft*, Internet Engineering Task Force, February 1996. Work in progress.

[16] D. B. Johnson, "Hierarchical Foreign Agents and Regional Registration," *Minutes of the Mobile IP Working Group Meeting*, 35th Internet Engineering Task Force Meeting, March 1996.

[17] G. Kirby, "Locating the User," *Communications International*, October 1995.
[18] M. R. Macedonia, D. P. Brutzman, "MBone Provides Audio and Video across the Internet," *IEEE Computer Magazine*, April 1994.
[19] S. McCanne, V. Jacobson, and M. Vetterli, "Receiver-driven Layered Multicast," *Proc. of ACM SIGCOMM '96,* August 1996.
[20] K. S. Meier-Hellstern, E. Alonso, and D. R. O'Neil, "The Use of GSM to Support High Density Personal Communications," *Record of the 1992 IEEE Intl. Conf. on Communications*, June 1992.
[21] T. Narten, E. Nordmark, and W. Simpson, "Neighbor Discovery for IP Version 6 (IPv6)," *RFC 1970*, Internet Engineering Task Force, August 1996.
[22] C. Perkins, editor, "IP Mobility Support," *RFC 2002*, Internet Engineering Task Force, October 1996.
[23] C. Perkins, "Mobile-IP Local Registration with Hierarchical Foreign Agents," *Internet Draft*, Internet Engineering Task Force, February 1996. Work in progress.
[24] D. C. Plummer, "An Ethernet Address Resolution Protocol," *RFC 826*, Internet Engineering Task Force, November 1982.
[25] J. Postel, "User Datagram Protocol," *RFC 768*, Internet Engineering Task Force, August 1980.
[26] J. Postel, "Internet Protocol," *RFC 791*, Internet Engineering Task Force, September 1981.
[27] J. Postel, "Internet Control Message Protocol," *RFC 792*, Internet Engineering Task Force, September 1981.
[28] J. Postel, "Transmission Control Protocol," *RFC 793*, Internet Engineering Task Force, September 1981.
[29] L. Press, "Net.Speech: Desktop Audio Comes to the Net," *Communications of the ACM*, Vol. 38, No. 10, October 1995.
[30] M. Rahnema, "Overview of the GSM System and Protocol Architecture," *IEEE Communications*, Vol. 31, No. 4, April 1993.
[31] H. Schulzrinne, "Voice Comunication Across the Internet: A Network Voice Terminal," *Technical Report TR-92-50*, Department of Computer Science, University of Massachusetts at Amherst, July 1992.
[32] S. Seshan, "Low-Latency Handoffs for Cellular Data Networks," Ph.D. Thesis, *Technical Report UCB/CSD-96-899,* University of California at Berkeley, March 1996.
[33] K. Sklower, "A Tree-Based Packet Routing Table for Berkeley Unix," *Proc. of the 1991 Winter USENIX Technical Conference*, January 1991.
[34] C. K. Toh, "The Design and Implementation of a Hybrid Handover Protocol for Multimedia Wireless LANs," *Proc. of the 1st Intl. Conf. on Mobile Computing and Networking*, November 1995.
[35] "`ttcp`: Test TCP," U.S. Army Ballistics Research Lab, December 1984.
[36] I. Wakeman, A. Ghosh, and J. Crowcroft, "Implementing Real Time Packet Forwarding Policies Using Streams," *Proc. of the USENIX 1995 Technical Conference*, January 1995.
[37] "WaveLAN Air Interface Data Manual," AT&T Wireless Communications and Networking Division, June 1995.
[38] H. Xie and D. J. Goodman, "Signaling System Architecture Based on Metropolitan Area Networks," *Technical Report WINLAB-TR-39*, Rutgers University, August 1992.

Biographies

Ramón Cáceres is a Senior Technical Staff member at AT&T Labs. He received a B.Eng. in Electrical Engineering from McGill University in 1983, and a M.S. and a Ph.D. in Computer Science from the University of California at Berkeley in 1984 and 1992, respectively. Between 1985 and 1987 he developed data communication products for Pyramid Technology Corp. Between 1992 and 1994 he investigated mobile computing issues at Matsushita Information Technology Lab. In 1994 he joined AT&T Bell Labs, where he pursued research in wireless networking. His current interests include Internet traffic characterization and Internet protocol performance. He he can be reached via electronic mail to ramon@research.att.com.

Venkata N. Padmanabhan is a Ph.D. candidate in Computer Science at the University of California at Berkeley. He received his B.Tech. degree in Computer Science & Engineering from the Indian Institute of Technology, Delhi in 1993 and his M.S. degree in Computer Science from the University of California at Berkeley in 1995. He has done research in the areas of computer networking, mobile computing and operating systems. The focus of his current work is network support for efficient Web access, and data transport over asymmetric networks. He received the best student paper award at the Usenix '95 conference. He is a student member of ACM and the IEEE. He may be reached via e-mail at padmanab@cs.berkeley.edu, and on the Web at http://www.cs.berkeley.edu/~padmanab.