Mobile Voice Over IP (MVOIP): An Application-level Protocol

Ayorkor Mills-Tettey
Ayorkor.Mills-Tettey@alum.dartmouth.org
Student Honors Thesis
Advisor: David Kotz
Dartmouth College Computer Science

Technical Report TR2001-390

June 1, 2001

Abstract

Current Voice over Internet Protocol (VOIP) protocols require participating hosts to have fixed IP addresses for the duration of a VOIP call. When using a wireless-enabled host, such as a tablet computer on an 802.11 wireless network, it is possible for a participant in a VOIP call to roam around the network, moving from one subnet to another and needing to change IP addresses. This address change creates the need for mobility support in VOIP applications.

We present the design of Mobile Voice over IP (MVOIP), an application-level protocol that enables such mobility in a VOIP application based on the ITU H.323 protocol stack. An MVOIP application uses hints from the surrounding network to determine that it has switched subnets. It then initiates a hand-off procedure that comprises pausing its current calls, obtaining a valid IP address for the current subnet, and reconnecting to the remote party with whom it was in a call. Testing the system shows that on a Windows 2000 platform there is a perceivable delay in the hand-off process, most of which is spent in the Windows API for obtaining DHCP addresses. Despite this bottleneck, MVOIP works well on a wireless network.

This project is supported by the Cisco University Research Program, by Lucent Technologies, by Microsoft Research, and by DURIP contract number F49620-00-1-0212.

1 Introduction

Voice over Internet Protocol (VOIP) transmits real-time data, such as voice or video, over IP networks. Voice over IP offers some benefits over the traditional Public Switched Telephone Network (PSTN) that make it an attractive and promising area of technology for businesses and consumers alike. Using IP networks makes it possible to transmit real-time data as well as traditional non-real-time data over the same network, enabling the development of a rich variety of consumer services including video conferencing, voice-enabled electronic commerce, and entertainment. In addition, using packet-based networks such as IP networks promises to be more efficient than using connection-oriented networks such as the PSTN, as network resources are not tied up unnecessarily during each point-to-point call.

There are two major proposed protocol stacks for Voice over IP. The H.323 protocol stack [1], proposed by the International Telecommunications Union (ITU), appears to be more widely used, and existing applications based on H.323 include Microsoft NetMeeting, eRing, and MediaRingTalk. The Session Initiation Protocol [6], proposed by the IETF, is a simpler but currently less widely used protocol.

Current Voice-over-IP solutions require the hosts engaged in a call to have fixed IP addresses. When either host is a portable computer on a wireless local-area network, such as an 802.11 network, mobility might result in the host needing to change IP address as it moves to a different subnet on the network. Currently, this mobility forces the termination of the VOIP call. Mobile Voice over IP (MVOIP), presented in this paper, provides a mechanism to enable a VOIP call to continue even as the hosts engaged in the call move across a wireless network and need to change IP addresses.

We begin with some background describing the H.323 standard and protocol stack in the following section. We then describe the design and implementation of the MVOIP system in Sections 3 and 4 respectively, leading to a discussion, in Section 5, of the performance of the system. The sixth section discusses related work, and the seventh ends with conclusions and future work.

2 Background

MVOIP is based on the ITU-T[1] H.323 Multimedia Standard, which is the basis for most existing VOIP solutions. We begin with a description of the H.323 standard and its associated protocol stack, continue with a brief overview of 802.11 wireless networks, and finally describe the mobility problem that MVOIP addresses.

2.1 The H.323 Multimedia Standard

The purpose of ITU-T's proposed H.323 Packet-based Multimedia Communication Systems Standard is described well by the standard's original title: "H.323: Visual Telephone Systems and Equipment for Local Area Networks Which Provide a Non-guaranteed Quality of Service." As this title suggests, the standard is broad in scope and defines the mechanism by which real-time information (video, in addition to audio), within the constraint of a real-time call or conference between two or more parties, can be transmitted over packet-based networks that do not provide a guaranteed quality of service.

The H.323 standard defines several terms [13]:

Call: A point-to-point multimedia communication between two H.323 endpoints, that begins with the call set-up procedure and ends with the call termination procedure.

Endpoint/Terminal: An entity that can call and be called, and that generates and/or terminates streams of information.

Gatekeeper: An entity that provides functions including address translation, authorization and authentication of terminals and gateways; bandwidth management; accounting; and billing. It is an optional part of an H.323 *zone*, but must be used by endpoints if it exists.

Gateway*:* An entity that connects two dissimilar networks, e.g., an H.323 network and the PSTN.

Multi-point Control Unit (MCU): An entity that provides support for conferences between three or more H.323 endpoints.

H.323 Zone: A collection of all terminals, gateways, and MCU's managed by one gatekeeper.

[1] ITU Telecommunication Standardization Sector (ITU-T)

Gatekeepers, Gateways and Multi-point Control Units are not essential to the functioning of the H.323 standard on a single local area network supporting only point-to-point calls. Thus, in the interest of simplicity, we did not consider them in the initial design and implementation of MVOIP, and shall not discuss them further in this paper, except under Future Work.

H.323 supports five different types of information streams between endpoints: Audio, Video, Data, Communications Control, and Call Control [13]. Audio and Video streams are processed using audio and video *codecs* respectively, and are transmitted and controlled using the Real Time Transport Protocol (RTP) and the Real Time Control Protocol (RTCP) [21] operating over an unreliable transport such as UDP. The data channel supports applications such as file exchange, database access, and electronic whiteboards, all standardized data conferencing applications defined in the ITU-T T.120 series of recommendations. The H.245 standard provides communications control by defining how endpoints negotiate channel usage and other specifications. The Q.931 signaling protocol defines call control functions that are used for call establishment and termination. The latter two protocols both operate over a reliable data transport such as TCP.

Figure 1 illustrates the relationship between the five streams of information and their related protocols. UDP and TCP can be used as the unreliable and reliable transports respectively for VOIP.

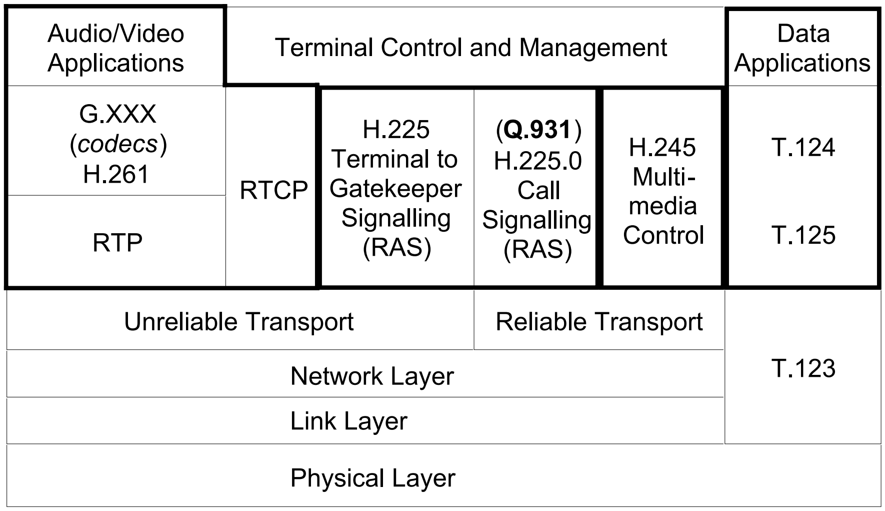

Figure 1 - The H.323 protocol stack (redrawn from [13])

Message flow in a typical H.323 call begins with the exchange of Q.931 call-establishment messages. The H.245 communications control protocol is then used to exchange and negotiate capabilities, and to establish and open channels for the exchange of real-time data. An audio *codec* digitizes and encodes the voice signal, and finally, RTP is used to transmit and receive the media stream. To reduce bandwidth utilization, a host may enable *silence detection* in the audio codec. When silence detection is on, the codec detects quiet periods in the media stream and communicates this to the RTP channel, which sends no packets of data during these silences. Figure 2a illustrates the sequence of messages for call establishment that results in the call represented by Figure 2b. A single ongoing H.323 call thus consists of concurrent signaling, control and media channels open between the two communicating endpoints.

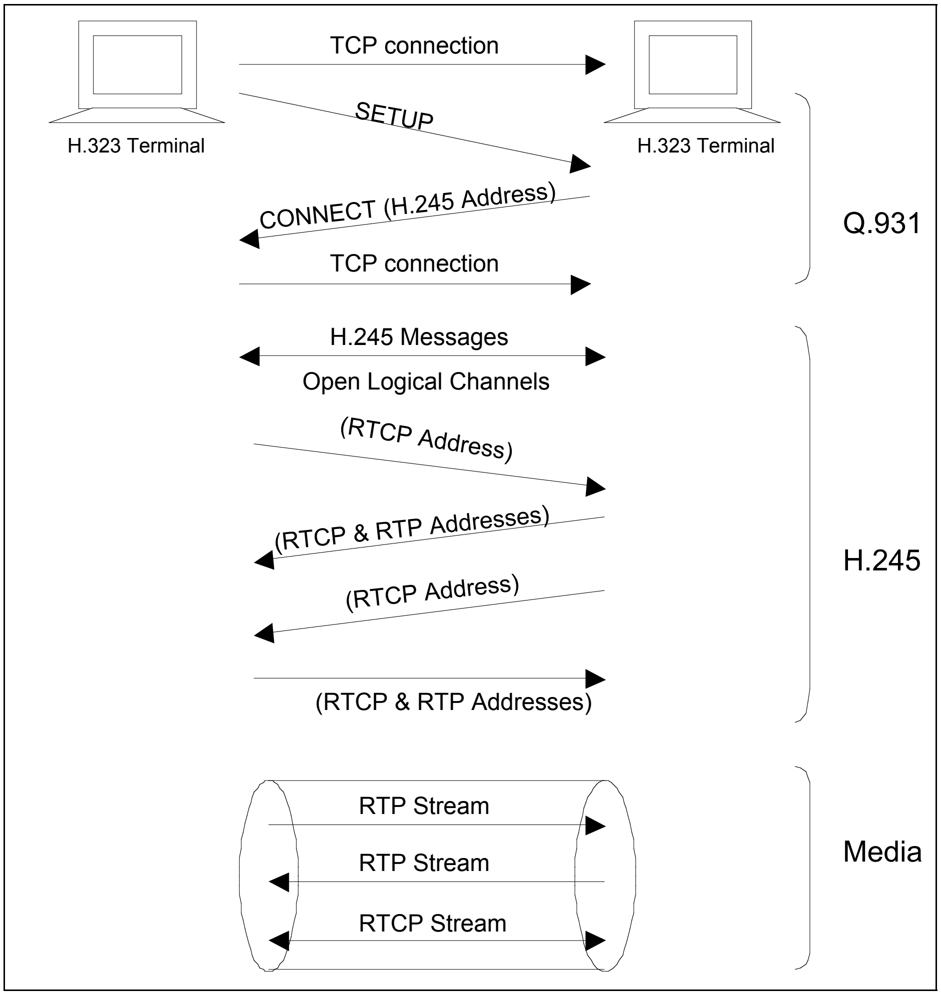

Figure 2a - H.323 message flow during call establishment (redrawn from [13])

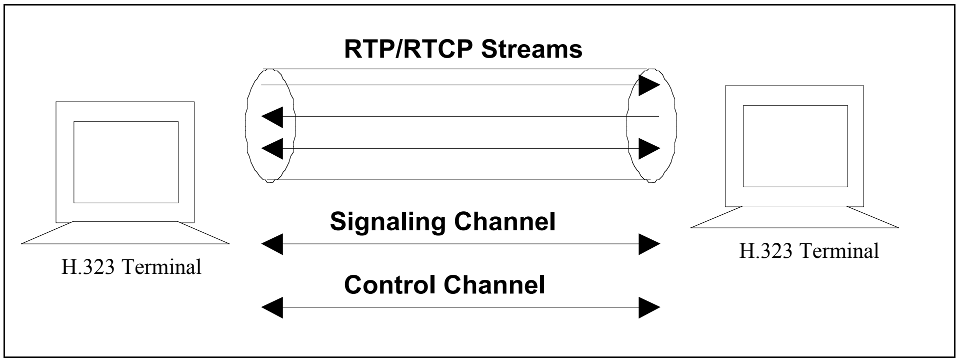

Figure 2b – Communication streams in an ongoing H.323 Call

2.2 802.11 Wireless Local Area Networks (WLANs)

In an 802.11 WLAN [15], a *station*, or *network interface card* (NIC), is the component that connects to the wireless medium. A *Basic Service Set* (BSS) is a set of stations that communicate with each other. When the BSS includes an *Access Point* that connects it to the wired LAN infrastructure, it is referred to as an *infrastructure BSS.* An infrastructure BSS makes it possible for hosts equipped with wireless network interface cards to be part of the IP network built upon the existing wired LAN.

2.3 The Mobility Problem

Consider a host (an H.323 endpoint) that is equipped with an 802.11 wireless network interface card, and is within an 802.11 infrastructure BSS. This host is able to participate in an H.323 call. To take full advantage of the mobility afforded by the wireless network, the mobile host should be able to roam to any point on the wireless LAN, or indeed the Internet, while still maintaining its H.323 call. The call should remain intact regardless of whether the host roams between different 802.11 access points (provided it is always within range of at least one access point), or whether it happens to cross a subnet boundary in the network as it roams. For such roaming to be supported, three levels of mobility are necessary:

- The host must be able to switch from one access point to another without having to reset or otherwise reconfigure its network connections.

- When the host moves into a new subnet, it must be able to detect this change and acquire an IP address for that subnet, which it will use in all its communications while on that subnet.
- The VOIP application must be able to handle changing IP addresses and continue to function normally before and after the change without human intervention.

Currently, the only level of mobility that is supported is the first, which is built into the implementation of the 802.11 wireless LAN. Most DHCP-enabled computers will not automatically acquire a new IP address via DHCP when they switch subnets unless the host is shut down or put to sleep during the switching phase, in which case the computer automatically attempts to renew its IP address when it is restarted or woken up. Finally, no current implementations of H.323 handle changing IP address of call participants during the call. These latter two points thus define the problem to be solved by MVOIP.

3 Design of Mobile Voice over IP

In designing a scheme to handle VOIP mobility at the application level, we sought to create a system that minimizes the number of steps for hand-off by limiting interactions during the hand-off process to communications between the two clients, the access point, and the DHCP server. No other servers or agents are involved in the hand-off process. When MVOIP is used in conjunction with a directory service such as that implemented in [8], updates to the directory service occur after the hand-off is complete and the call has been resumed, thus allowing the call to be resumed as quickly as possible.

3.1 Overview

The first step in handling mobility in a VOIP application is determining that a subnet change has indeed occurred, after which it is necessary to complete a hand-off process that enables the call to be continued. The process is outlined below, first from the perspective of the mobile host, and then from the perspective of the non-mobile host. In this discussion, “mobile” is used to refer to a host that is roaming across the network and is just about to cross a subnet boundary, is in the process of crossing a subnet boundary, or has just finished crossing a subnet boundary. The remote party with whom the mobile host is in a call is referred to as "non-mobile". Note that when both hosts are capable of motion, the classification of a host as "mobile" or "non-mobile" is dynamic over the course of a call, but we assume that the two communicating hosts are not simultaneously "mobile", in other words, the two hosts cannot simultaneously cross subnet boundaries. MVOIP breaks down if both hosts switch subnets at the same instant (see Section 4.4).

From the perspective of the mobile host, MVOIP involves the following steps:

0. **Discover:** In the background, the host periodically checks to see whether it is has changed subnets. When it determines that a subnet change has occurred, it initiates the "hand-off" process comprising steps 1-5 below.
1. **Mobility Alert:** As soon as the mobile host determines that it has crossed subnet boundaries, it sends an MVOIP "Mobility Alert" message to the remote party alerting it to the subnet change.
2. **Pause:** It then pauses its calls to the remote party, and pauses all H.323 Listeners (the entities that listen for incoming calls).
3. **DHCP Renew:** It obtains a new IP address valid for the new subnet, from the DHCP server.
4. **IP Update:** It sends an "IP Update" message to the remote party, reporting its new IP address.
5. **Re-connect:** It finally re-establishes its H.323 connections with the remote party, and un-pauses the H.323 Listeners. The call can then continue.

From the perspective of the non-mobile host, MVOIP involves the following steps:

1. **Listen:** The host listens for Mobility Alerts or IP Updates from the endpoint with whom it is in a call.
2. **Mobility Alert:** When the non-mobile host receives a Mobility Alert from a remote party, it pauses its call to that party.
3. **IP Update:** When it receives an IP update from the remote party, it re-establishes its connections to that party.

Figure 3 illustrates the MVOIP hand-off process.

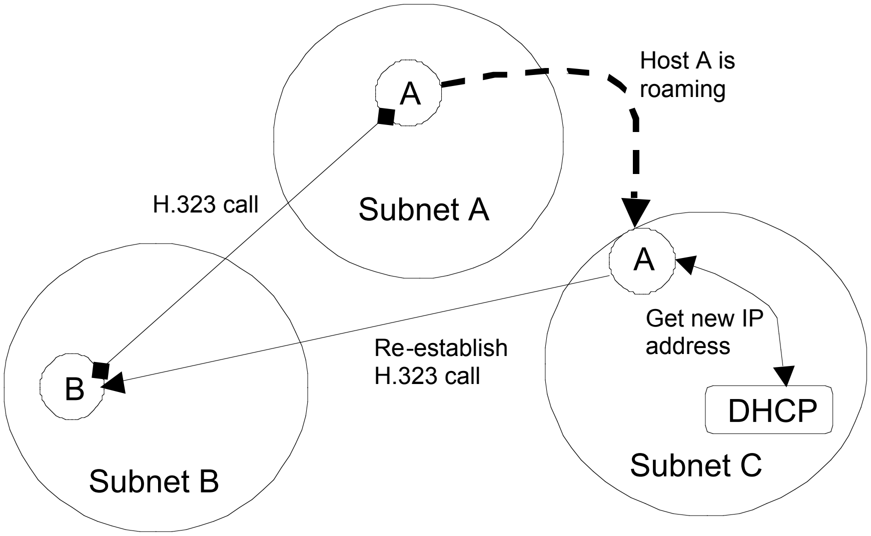

Figure 3 - Changing subnets in MVOIP

3.2 The "Mobility Alert" and "IP Update" messages

Participants in an MVOIP call communicate about mobility using the "Mobility Alert" and "IP Update" messages.

The Mobility Alert message serves as a courteous notification to the remote party that a subnet change is occurring. The mobile host may send the Mobility Alert to prevent the call from being abandoned if the broken connection is detected before the IP Update message arrives. It can also be used as a prompt for the remote application to display a useful message to the user during the hand-off process. For implementation reasons discussed in Section 4, the Mobility Alert is a useful but optional message and its receipt by the non-mobile party is not essential to the correct functioning of the hand-off procedure.

The IP Update message is a notification of the mobile host's new IP address after a subnet change. It prompts the non-mobile host to continue its call with the mobile host by accepting the incoming connection from the new IP address.

MVOIP messages have the following fields:

<Version> An MVOIP version number, currently 1.0.

<Type> The type of message, set to 0 for a “Mobility Alert” and 1 for an “IP Update” message.

<Call ID> The H.323 identifier for the call in question.

<Source> The former IP address (now invalid) of the mobile host that is the sender of the Mobility Alert or IP Update.

<New IP> For Mobility Alert messages, this field is null. For IP Update messages, it contains the new IP address of the remote party (i.e. the sender).

<Destination> The IP address of the recipient of the messages, i.e., the non-mobile party.

<Timestamp> The time in seconds since the beginning of the call, at the instant when the mobile host sends the message.

<Update Number> For Mobility Alerts, this field is null. For IP Updates, it contains the number of IP Updates including the current message that have been sent so far, starting at 1 for the first subnet change.

3.3 MVOIP Security

There are several security concerns in MVOIP. The most obvious threat is that of impersonation: a malicious third party could masquerade as the mobile host and send IP Update messages to the non-mobile host, which would result in the current call being redirected to an unintended location. Alternatively, a third party could intercept the IP Update messages, modify and resend them to redirect the call to a desired location. It could also eavesdrop on the IP Update messages sent to the non-mobile host from the mobile host, and use these to track the movement of the mobile host, thus compromising its privacy. We can mitigate these threats with the following scheme for security in MVOIP. The scheme depends on the existence of a shared secret or key between the two hosts in the call. This key can be established via an authenticated directory or key-distribution service, or in any other reasonable way, and is used to encrypt the MVOIP messages.

The <Timestamp> and <Update Number> fields are used to enforce security in MVOIP. The <Timestamp> is the length of time in seconds since the beginning of the call in question. Including this field introduces the requirement that the two communicating hosts have precise, but not necessarily synchronized clocks. Having precise clocks such that the perceived time offset between the two clocks does not change by more than a second over the duration of a call is sufficient to ensure that the two hosts share a common perception of the length of time since the beginning of the call. The <Timestamp> field is included in every Mobility Alert or IP Update message that a mobile host sends out. The <Update number> is set to null for Mobility Alert messages. For IP Update messages, it is the number of IP Update messages, including the current one, that mobile host has sent to the non-mobile host since the beginning of the current call. It starts with a value of 1 for the first subnet change, and increments by 1 for every subsequent IP Update message. After the mobile client constructs the MVOIP messages (Mobility Alerts and IP Updates) it encrypts them using the shared secret key before sending them over the network.

When the non-mobile host receives a Mobility Alert message, it decrypts it using the secret key for the current active call, and checks the timestamp field. If the timestamp is off by more than two seconds, or if an IP Update message has been received from the mobile node in the last two seconds, the Mobility Alert message is discarded. Checking the timestamp prevents replay attacks in which a third party can capture a Mobility Alert message sent by the mobile host, and re-send it later in an attempt to disrupt the call by causing the non-mobile node to pause the call. If the attacker resends the message as soon as it is captured, i.e., when the mobile node has not yet sent an IP Update message, the replay attack has no adverse effect on the call, and the non-mobile host can simply ignore the duplicate Mobility Alert message. Note that because all messages are encrypted, an attacker cannot compromise a call by constructing and sending a fake Mobility Alert message.

When the non-mobile host receives an IP Update message, it decrypts it using the secret key, and checks the <Update number> field against the expected update number. If the numbers are not equal, it discards the IP Update because it is indicative of a replay attack. As in the case of Mobility Alert messages, encryption prevents the attacker from constructing and sending fake IP Update messages and also from intercepting and modifying genuine IP Update messages. Because IP Update messages are sent within IP packets that contain the source and destination addresses in the clear, a third party that "sniffs" close enough to the non-mobile host would still be able to track the movement of a mobile host by looking at the source address of incoming TCP connections on the non-mobile host's listening MVOIP port. This is a shortcoming of the security scheme, but we judge the risk from this shortcoming to be minimal, since it requires the attacker to be able to sniff on the network of the party that the mobile host is in a conversation with, and in most cases, it is impossible to know prior to the beginning of the call, who this party is going to be or where in the world they are located.

3.4 Discovering IP Addresses of Mobile Hosts

A host wishing to call another host must be able to discover the IP address of the host to be called. Because hosts are mobile and can change IP addresses at anytime, MVOIP needs to be used in conjunction with a dynamic directory infrastructure that will always contain the most current IP address of the hosts using the service. Using this directory system, a mobile host can always be reached with incoming calls. The directory infrastructure implemented by Ammar Khalid [8] meets MVOIP's needs in this regard.

4 Implementation of MVOIP

The MVOIP protocol is platform independent and can be implemented for any Voice over IP application. The current implementation of MVOIP is based on the OpenH323 project's open-source implementation of the H.323 standard [2]. Our implementation works on a DHCP-enabled Windows 2000 platform that, to be a mobile host, must be equipped with a Lucent "Gold" 802.11-compliant network interface card and must be within range of an 802.11 wireless network that supports the Dynamic Host Configuration Protocol (DHCP). Any Windows 2000 platform connected to the Internet can be a non-mobile MVOIP host. The current implementation does not include the security measures described in Section 3.3 above.

4.1 Identifying subnet changes (“Step 0” on the mobile host’s end)

MVOIP uses hints from the network to determine when the mobile host crosses a subnet boundary. A mobile host’s subnet is determined by the subnet of the access point to which it is currently associated. Because each wireless access point has only one interface to the wired LAN, it is necessary for the mobile host to switch access points to switch subnets. A mobile host’s first hint of a possible subnet change is thus the discovery that it is associated to a new access point. Since each subnet in a local area network may be equipped with several wireless access points, however, switching access points is not a deterministic test of having switched subnets. As the results in section 5 show, there is a perceivable delay during the hand-off process, so it is important to be as accurate as possible in determining subnet changes to reduce the occurrence of unnecessary hand-offs. If the host could query the associated access point for its subnet address, it would know for certain whether it had crossed subnet boundaries. Because the link layer is below the IP layer in the TCP/IP protocol stack, however, information about the IP or subnet address of the access point is not available at the link layer using current implementations of 802.11 (at least in Lucent and Cisco implementations). The host must thus use additional hints to increase its accuracy in determining when it crosses a subnet boundary.

A further hint that is used by MVOIP is the time elapsed since the last packet of RTP data was received from the remote endpoint. If the remote endpoint has *silence detection* turned off on its audio codec, then the mobile endpoint will receive packets of audio data at regular time intervals for as long as its connection to the remote party is valid. The receive time interval depends on the audio codec used, and in our case, using the GSM 06.10 codec, it is approximately 80ms. If the host encounters a new access point and has not received an RTP packet from the remote party within a time period 20 ms greater than the average receive interval, the host initiates hand-off. As shown in the Results section, this hint is less effective when the remote host enables silence detection in its audio codec and is in a normal conversation where it is silent for approximately half of the time. One possible solution, although we did not implement it, is to have the remote end send periodic “heartbeat” RTP packets even when silence detection is on.

In MVOIP, we implement mobility detection as a background thread that continuously polls the wireless network interface card to determine if an access point change has occurred. The polling frequency used is 1 second. When an access point change does occur, MVOIP queries the RTP channel to determine the time elapsed since the last packet was received, and it initiates the hand-off procedure if the delay exceeds a fixed threshold determined by the average receive time interval.

4.2 MVOIP "Mobility Alert" and "IP Update" Messages

As soon as a subnet change is detected, the mobile host sends the "Mobility Alert" message, as a UDP packet, to the non-mobile host. UDP is used because by the time a subnet change is detected, the IP address of the mobile host is no longer valid for the subnet it has entered. It is thus not possible to establish a TCP connection with the remote party. Because UDP does not guarantee packet delivery, and also because ingress filters in network routers may in some cases discard packets originating from invalid IP addresses, the implementation of the hand-off procedure does not depend on the receipt of the Mobility Alert message. Still, if the message does make it through, it can be a helpful hint to the non-mobile host.

The mobile hosts sends the "IP Update" message via TCP because the receipt of this message is essential for the correct continuation of the call, and by the time this message is sent, the mobile host has acquired a valid IP address and is capable of establishing a TCP connection.

4.3 Pausing and Re-establishing the H.323 Call

As described in the Background (Section 2), an H.323 call consists of several channels of information flow between the communicating parties. These include a Q.931 signaling channel, an H.245 control channel, and several channels carrying the real-time data and its associated control messages. In the OpenH323 implementation of H.323, each of these channels of information is a separate thread of execution within the VOIP application. To pause and re-establish an H.323 call, each of these channels must be paused, reset or reconfigured where necessary, and re-opened individually. For TCP channels, this involves establishing a new TCP connection using the updated IP address. For UDP channels, it is sufficient to reset the sockets used for communication and update the metadata concerning the remote party. Although new underlying network connections need to be established between the communicating parties for each channel of information flow between them, at no point during the hand-off are the abstractions of the communication channels between the parties shut down or terminated. As such, to re-establish the call, the parties do not need to repeat the H.323 call signaling process, nor do they need to re-negotiate capabilities or negotiate the opening of any logical channels, because these capabilities are already in effect and are remembered, along with the call reference number, during and after the hand-off process. The latency of the hand-off procedure is hereby kept to a minimum.

4.4 Handling Error Situations

There are numerous opportunities for error situations during the hand-off process and these must be handled appropriately.

It is possible that moving out of range of the current access point implies not that the mobile host is switching subnets, but that it is moving completely out of range of the wireless network. In this case, it does not associate to a new access point, and the DHCP process fails after a timeout of about 3 seconds. Or, the DHCP renew process might fail for other reasons, preventing the mobile host from obtaining a new IP address. Finally, it is possible that both parties might change subnets concurrently, making it impossible for them to re-establish their connections since they now do not know how to contact each other. If the mobile host is unable to re-establish its network connections, or to reconnect to the remote party within a specified time-out period, it shuts down all its communication channels and assumes that the call has ended. Similar time-outs are used on a non-mobile host, after which a paused call or a broken connection is assumed to be defunct. The host then gracefully closes all communication channels, ending the call. At this point, the hosts, either automatically or initiated by their human users, might choose to try contacting each other again with a new H.323 call. In the case when both hosts simultaneously cross subnet boundaries and as such no longer know each other’s addresses, a dynamic MVOIP directory system for a host to look up another's current IP address is essential. Ammar Khalid [8] has implemented such a system.

5 Experiments and Results

To measure the performance of MVOIP, we measured the number of RTP packets sent and received over the duration of a call that included one subnet change. Figure 5 plots the number of RTP packets of data sent and received per second by the mobile host for a 1-minute long call. The graph also indicates the "mobility checks" that occurred at 1-second intervals (the location of these points at a value of 10 on the graph has no significance). For this experiment, we used a wireless-enabled laptop as the mobile host roaming from one subnet A to another B, in a call with a non-mobile host stationed in subnet B. The mobile host used a lucent "Gold" WaveLAN card to communicate at 11Mbps to access points connected to the 10Mbps Ethernet network. The non-mobile host was plugged into a 10Mbps Ethernet network. Both ends of the H.323 call ran our modified version of the OpenH323 software on Windows 2000, and had the silence-detection option turned off. In the figure, the hand-off process is clearly observed as a 3.3-second long period during which no packets are sent or received by the mobile party. The spike in the number of packets sent as soon as hand-off is completed is due to buffering of data by the audio codec and the height of this spike is shown by other experiments to be proportional to the size of the jitter buffer (in seconds) chosen for the audio codec.

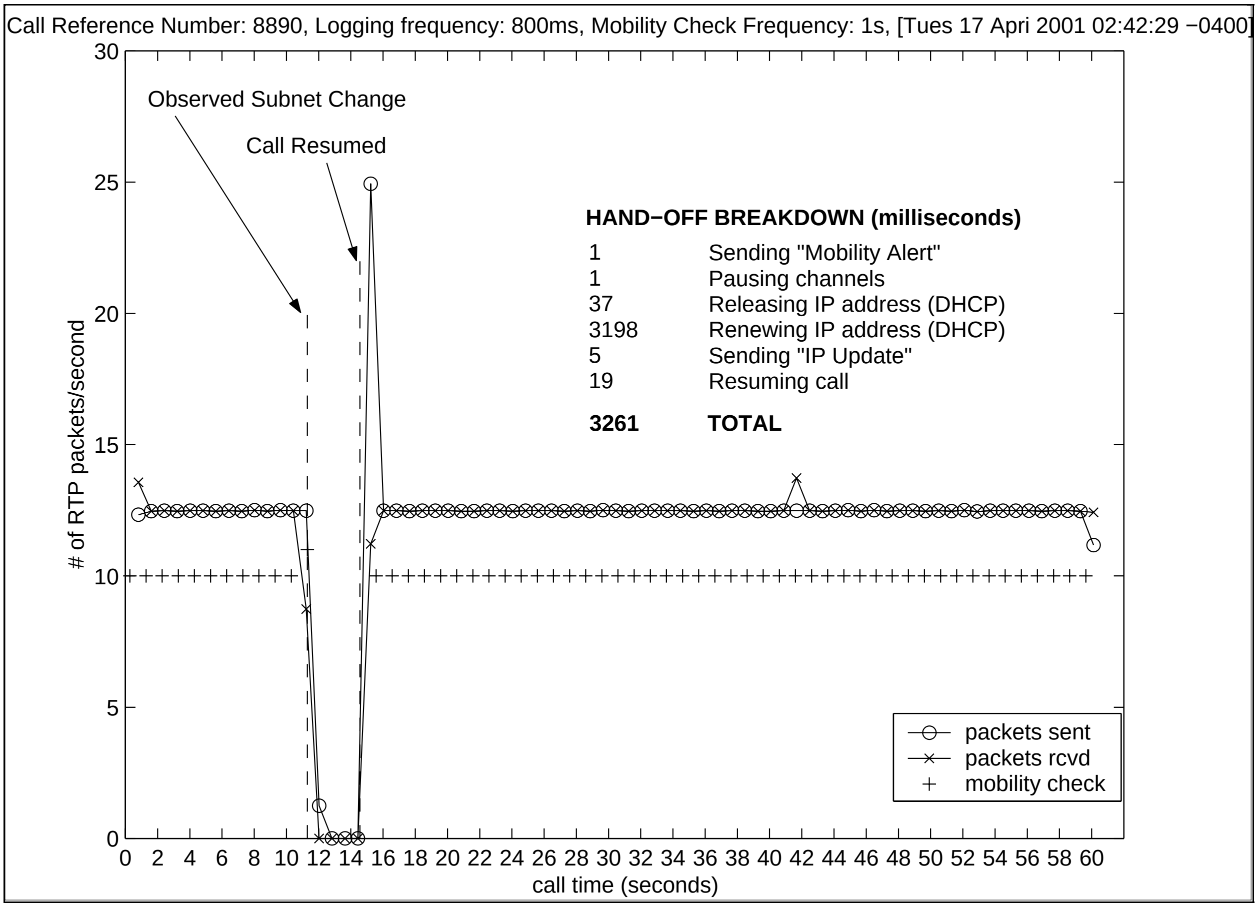

Figure 4 - Packets sent and received by the mobile host during an MVOIP call with a subnet change

For the call illustrated in Figure 4, a breakdown of the hand-off duration into its component steps (i.e., sending the Mobility Alert, pausing the call, releasing the old IP address, obtaining a new IP address, sending the IP Update, and finally resuming the call) shows that the majority of the hand-off time is spent in the Windows 2000 API call to obtain an IP address from the DHCP server. A closer examination of this process reveals that after obtaining an IP address from the DHCP server, the operating system verifies that the IP address acquired is not in use by any other host on the subnet. It does so by broadcasting three Address Resolution Protocol (ARP) messages at approximately 1-second intervals, before returning successfully from the API call if no reply is received.

Figures 5 and 6 summarize the hand-off duration for 30 calls, each with a call length of one minute and comprising one subnet change. Figure 7 illustrates the average total hand-off time for these calls, with an error bar of one standard deviation from the mean. For these 30 calls, the hand-off duration ranged between 3.2 and 9.5 seconds, with more than 50% of the calls having a hand-off duration of less than 3.5 seconds, as illustrated in Figure 6. Figure 5 shows that in all but two of the calls, the second half of the DHCP process, i.e., obtaining a new IP address, is the longest part of the hand-off process for the reason that has already been described. For up to 40% of the calls, however, the first part of the DHCP process, i.e., releasing the old IP address, also becomes a significant bottleneck. Releasing the IP address appears to take either less than 50ms, or almost 3 seconds, with no values between. These observed durations are probably due to built-in time-outs in the DHCP API and they result in the bi-modal nature of the graph in Figure 6. The obvious solution to this problem is to skip the first step of releasing the old IP address, since its lease period will eventually expire, freeing it up for later use by other hosts. In the current implementation, skipping the DHCP release does not yield expected results, however, because the length of time taken by the Windows API call to obtain an IP address depends on whether it has released its old address. In cases where the host has crossed a subnet boundary and has not released its old address, the process of obtaining an IP address may take up to 15 seconds, and often fails due to built-in time-outs. It is thus clear that the lack of control by the programmer over the Windows DHCP client-side interface is detrimental to the performance of MVOIP.

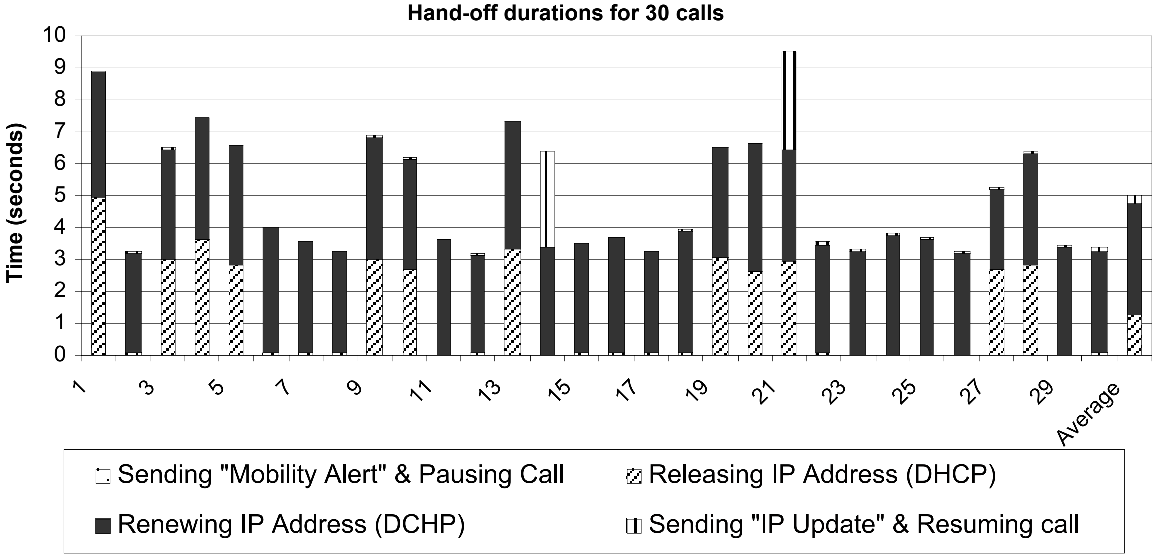

Figure 5 - Breakdown of the hand-off duration in 30 calls with subnet changes

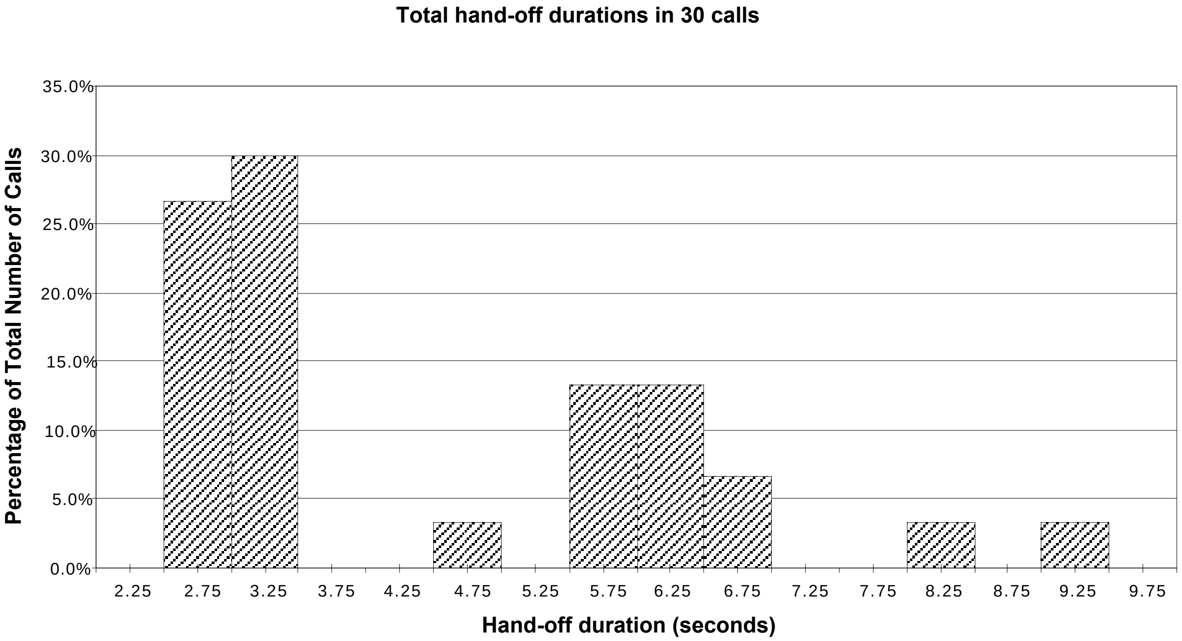

Figure 6 - Distribution of total hand-off duration in 30 calls. Each range is labeled with the center value of the range.

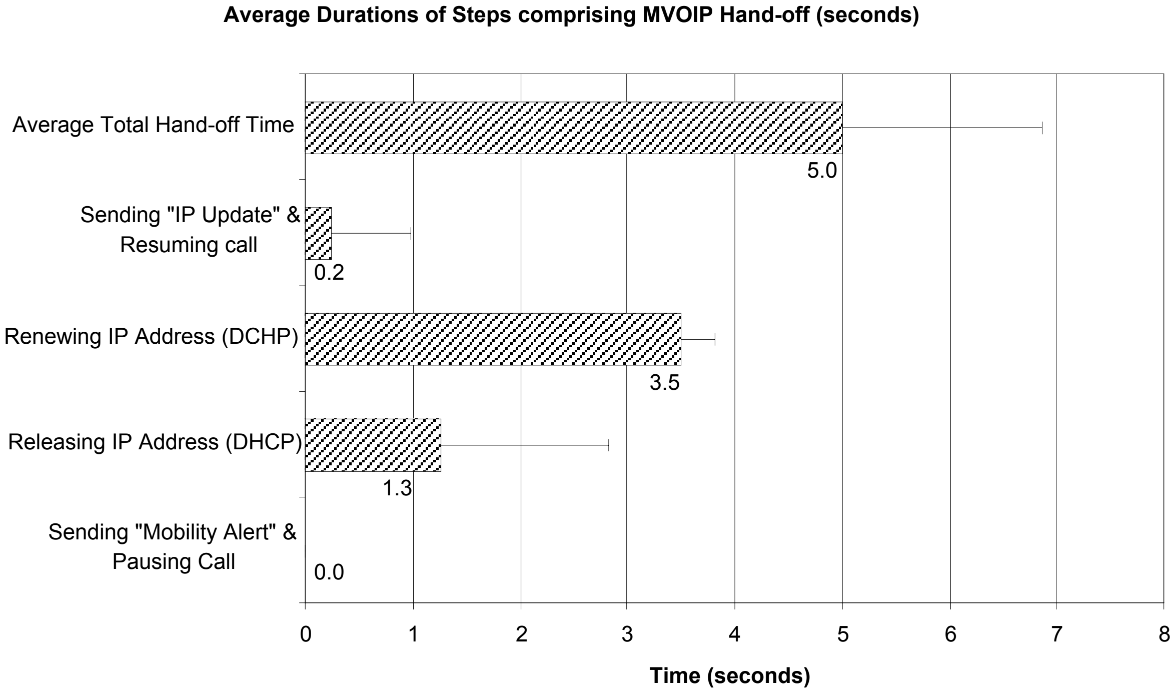

Figure 7 - Average hand-off duration for 30 calls

Figures 4 to 7 show that there is a perceivable period of silence during the hand-off process in an MVOIP call. Hand-off takes approximately the same length of time whenever it is initiated, regardless of whether a subnet change actually occurred. For this reason, it is important that the mobile host initiates the hand-off process only when it is absolutely necessary to do so, that is, when it crosses a subnet boundary. As discussed in section 4.1, the hints used by MVOIP to identify when subnet changes occur do not provide a deterministic indication of a subnet change. Figure 9 compares the effectiveness of the hints currently used by MVOIP to determine subnet changes. In the test scenario that yielded these results, the mobile host roamed back and forth twice across three subnets, thus undergoing a total of eight subnet changes, and up to sixteen access-point changes (see Figure 8). The number of access-point changes the mobile host makes depends on which access points it associates with as it roams, which in turn depends on perceived signal strengths and other factors built into the 802.11 implementation. We have no direct control of these associations, beyond ensuring that mobile host follows the same physical path in each test run.

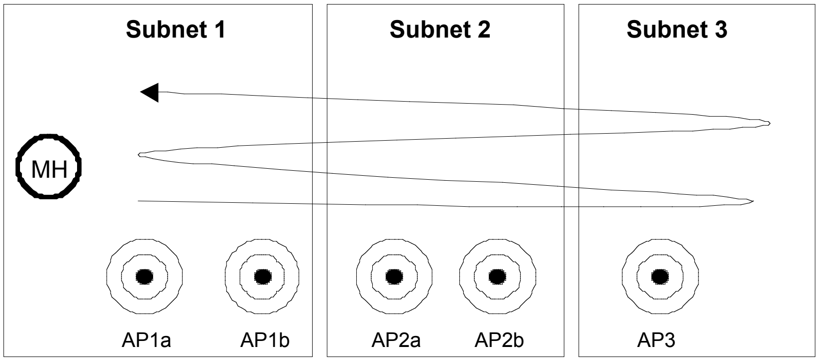

Figure 8 - Test Scenario for the results in Figure 9. The mobile host (MH) roams from subnet 1 to 3 and back again, two times. It thus crosses 8 subnet boundaries and changes access points 16 or fewer times.

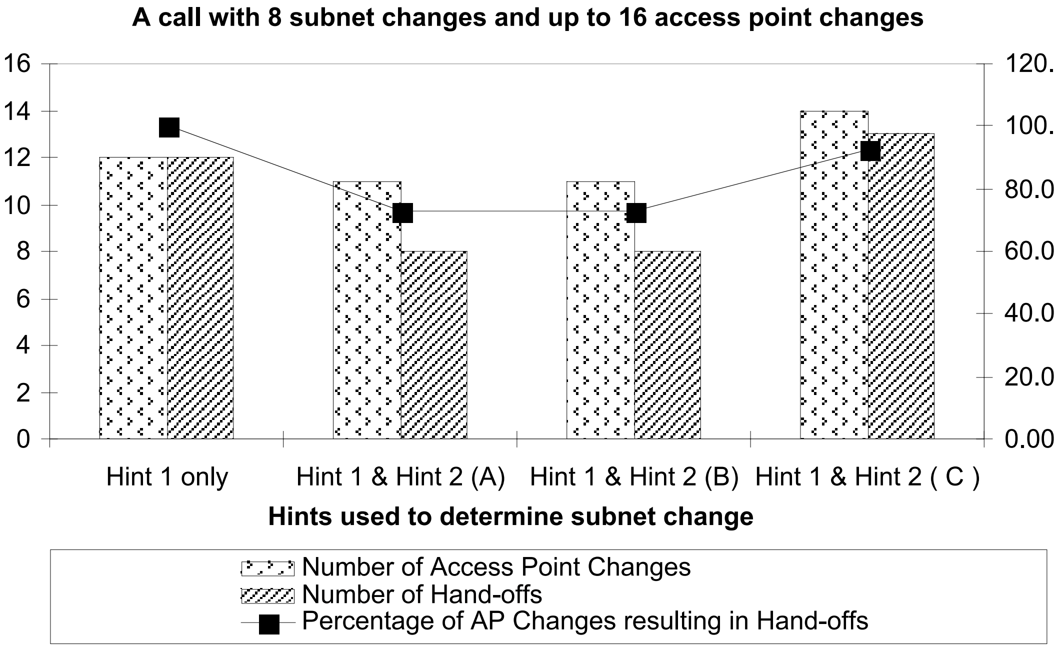

Figure 9 - The effectiveness of hints in determining subnet changes.
'Hint 1' refers to an access point change. 'Hint 2' refers to the amount of time since the mobile host received the last RTP packet from the non-mobile host. In (A), silence detection was turned off on both ends of the call (see Figure 10). In (B), silence detection was enabled only on the non-mobile end and conversation was one-sided, concentrated on the non-mobile end (see Figure 11). In (C), silence detection was enabled on both ends, and conversation was evenly distributed on both ends of the call (see Figure 12). Note that the ideal number of handoffs for the test scenario was 8.

Figure 9 compares the number of access point changes made with the number of hand-offs undertaken, for four different runs of this test scenario. In the first run, the only hint that is used to indicate a subnet change is the access point change, and this results in a hand-off for every access-point association or re-association. In the second run, silence detection is turned off in both hosts (see Figure 10), and the time elapsed since the last RTP packet was received is used as an additional hint to determine a subnet change, as described in Section 4.1. In this case, the mobile host undergoes the minimum number of hand-offs (8), exactly equal to the number of subnet changes made. In the third run, we enable silence detection on the remote host, and use a one-sided conversation of normal-paced human speech where the remote host does all the talking. This conversation is illustrated in detail in Figure 11. In this case, although silence detection is enabled on the remote end, the mobile host makes only the minimum number of hand-offs based on the two mobility hints, because the remote host is not silent for long periods of time. In the fourth and final run, we use a conversation that more closely mirrors a normal conversation such that each host "speaks" in 10-second bursts and is silent for approximately half of the time. Figure 12 illustrates this conversation. In this last case, the use of the second hint is not as effective due to the 10-second long periods of silence from the remote host.

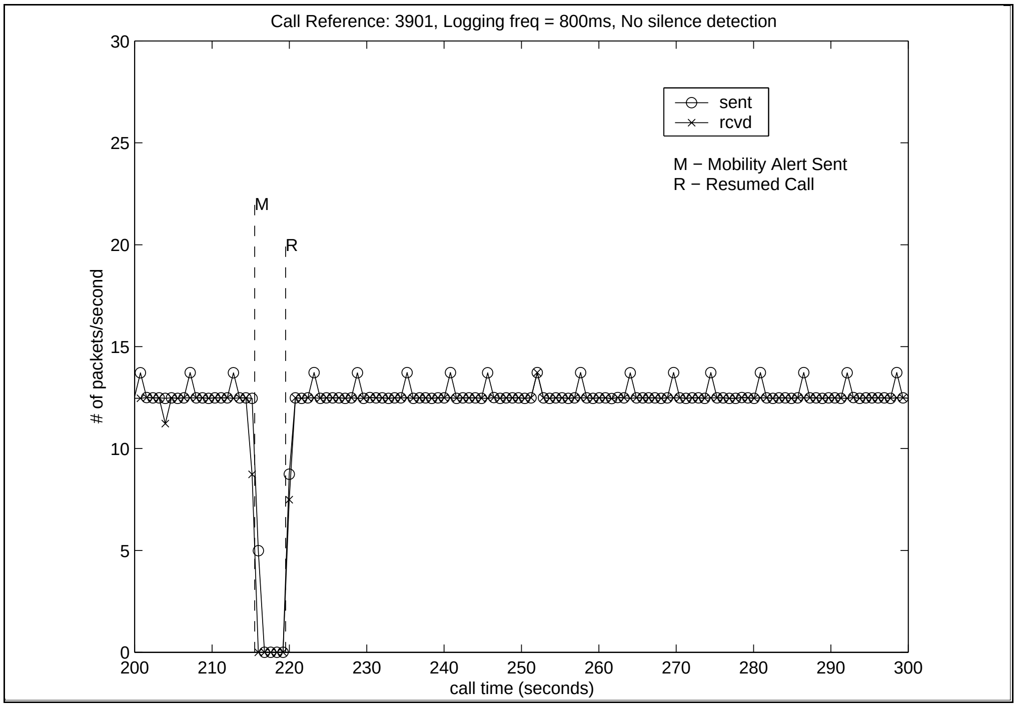

Figure 10 – Packets sent and received by the mobile host during part of a call with silence detection off. (The "spikes" are an artifact of the logging frequency). The hand-off during the call was due to an actual subnet change.

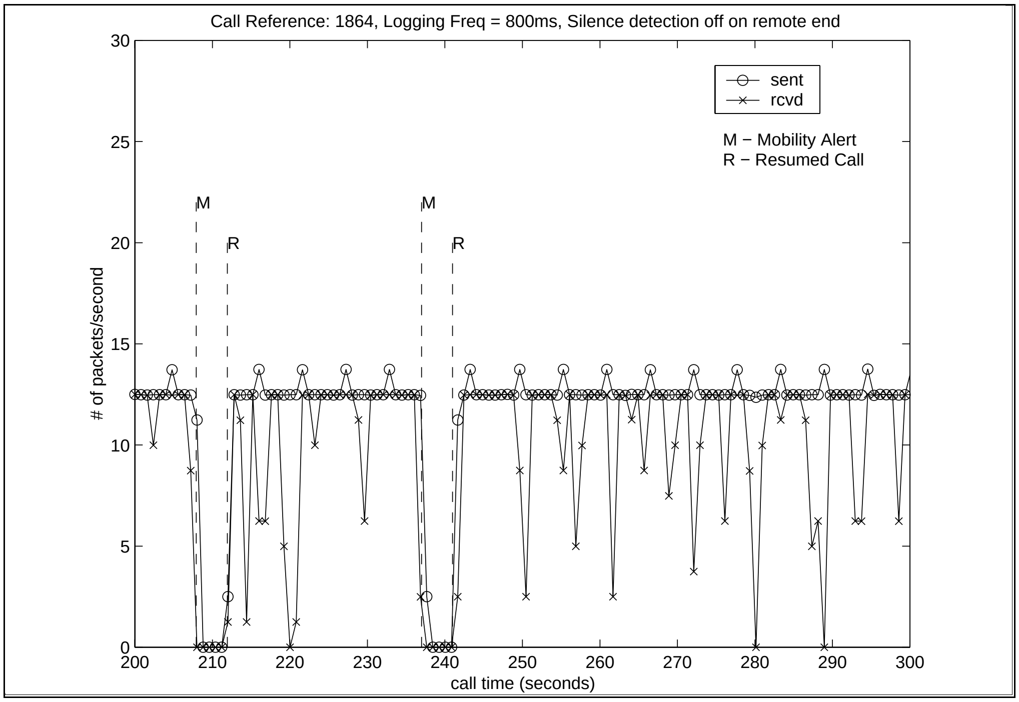

Figure 11 – Packets sent and received by the mobile host during part of a call with most of the conversation coming from the non-mobile host and silence detection enabled on the non-mobile host. The two hand-offs during the call are due to actual subnet changes.

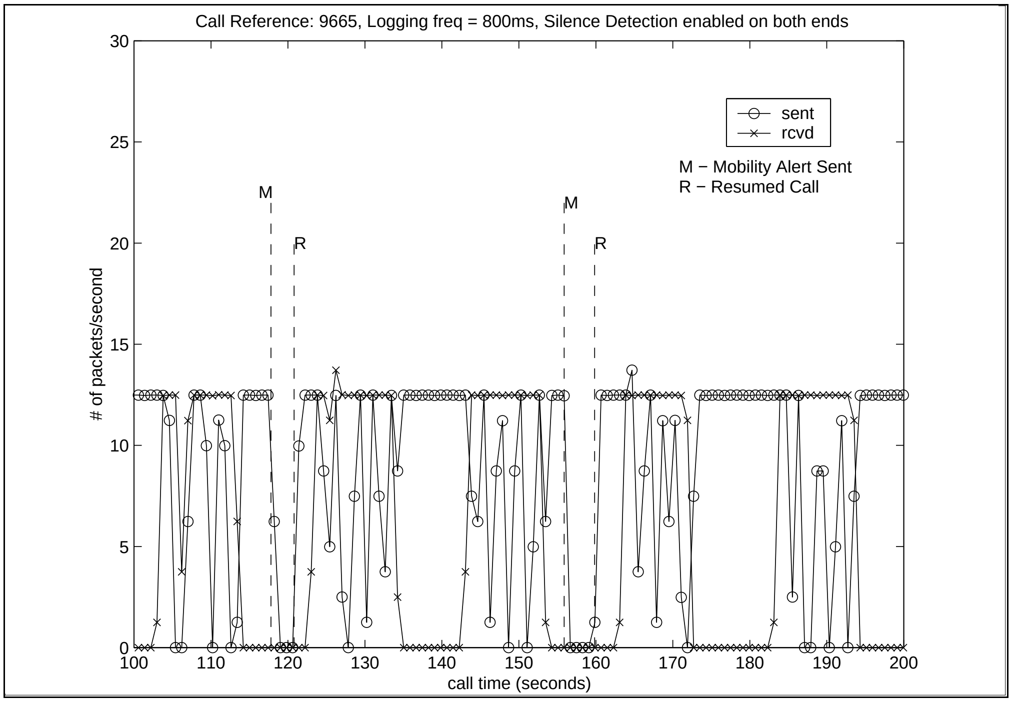

Figure 12 – Rate of packets sent and received by the mobile host during part of a call with conversation balanced between the two hosts and silence detection enabled on both ends. The first hand-off is due to an erroneous determination of a subnet change, and the second is due to an actual subnet change.

Other methods of identifying subnet changes are possible, and can be investigated in future work.

As suggested in Section 4.1, each node could send a periodic “heartbeat” RTP packet even when silence detection is on, to let the other node know that it is still there. Given that we test for subnet changes once per second, it would suffice to send a heartbeat every half second. Using the GSM codec, RTP sends packets every 80ms when silence detection is off. Each RTP packet has a header size of 12 bytes and contains as its payload four GSM frames of 33 bytes each, or a total payload of 132 bytes. The RTP heartbeat packet would simply be an RTP header of 12 bytes with a payload size of zero. Thus, sending an additional “heartbeat” packet every 500ms when silence detection is on would reduce the savings of silence detection by only 1.33%, which is not too high a price to pay for a more accurate detection of subnet changes.

A deterministic test of having switched subnets would be for the mobile host to determine the subnet number of the access point it is currently associated with. Since each host knows the MAC address of its associated access point, the host could obtain the IP address and the subnet mask of the access point using the Reverse Address Resolution Protocol (RARP) [4] from a RARP server. It would then be possible to determine the subnet number. To use this method, each subnet of the network would need a RARP server. Furthermore, using RARP would result in an additional latency during the hand-off period due to the network traffic involved. Because of these considerations, we did not use this method in the current implementation.

Another approach is for each host to download and cache a pre-populated database of access points mapping access-point MAC addresses to subnet numbers for its local-area network, or to have each host populate such a database dynamically for the subnets it frequents, by essentially “learning” about the network over time. The feasibility of the former method depends on the size of the wireless network and the number of access points it contains. Using the Dartmouth College wireless network, which contains 450 access points, as an example, this appears to be a feasible alternative. In the latter alternative, the first time a host encounters a new access point, it would assume a subnet change and would initiate the hand-off process. By examining the new IP address and subnet mask obtained after the hand-off process, the host would determine whether it had actually crossed subnet boundaries and would populate its database accordingly. In this manner, if the host frequents particular parts of the network, the number of erroneous hand-offs in that area of the network would decrease over time.

A final alternative is for the host, after each access point change, to ping or send an ARP query to a known IP address, such as that of the DHCP Server or subnet gateway router, to determine if it has changed subnets. This method, however, requires a timeout period, which may lead to additional handoff latency. There is a tradeoff between handoff latency and handoff accuracy that should be quantified for each of these methods in future work.

The length of time that the hand-off process takes is a bottleneck only because it results in a long period of silence during which the mobile host loses packets of data. If the mobile host could still receive packets destined to its old IP address while it is going through the hand-off process, the hand-off duration would be less critical. Implementing a “soft” rather than a “hard” hand-off between 802.11 wireless access points would make it possible for the mobile host to communicate with two access points simultaneously during the MVOIP hand-off process. It would thus continue receiving audio packets through the old access point while obtaining a new IP address through the new access point, making the hand-off process less noticeable. Having a “soft” hand-off would require modifications to the 802.11 WLAN specification.

6 Related Work

Several schemes have been proposed to handle mobility in IP networks, and a few of these schemes are geared specifically towards optimization for voice traffic.

Mobile IP [16] is a proposal by the Internet Engineering Task Force (IETF) to handle mobility at the IP level. Mobile IP is a forwarding scheme for IP Packets that aims to hide the movement of the mobile host from the upper layer protocols and applications. It characterized by the use of *Home Agents* (HA) and *Foreign Agents* (FA) that handle the routing of IP packets to the mobile host. The network where a host begins becomes its *home network*. When the host moves to a foreign network, it obtains a *care-of-address* (COA) via registration with a Foreign Agent, and it registers with its Home Agent to forward all in-coming packets to the COA using IP-IP encapsulation or "tunneling" (see Figure 13). The care-of-address may be an IP address of an interface on the FA, or it may be an address obtained by DHCP, in which case the address is called a *collocated-care-of-address* (CCOA) and the FA is simply a registration agent. In Route Optimized Mobile IP [17], the remote node with which the mobile node is corresponding (called the *Corresponding Node*) is informed of the current care-of-address of the mobile node to avoid triangle routing caused by forwarding packets through the Home Agent. Fladenmuller and Silva [5] showed that when using TCP over Mobile IP, it takes about 3 seconds for transmission to be reactivated after a hand-off using a care-of-address through a foreign agent. In addition, the performance of TCP is adversely affected due to the slow-start and exponential back-off algorithms built into TCP. The advantage of Mobile IP is that mobility is completely hidden from the higher layers. It is not currently supported by many hosts or routers, however, making its integration into current applications infeasible. In addition, the long delay during hand-off, and the latency caused by forwarding do not make it an optimal solution for voice traffic.

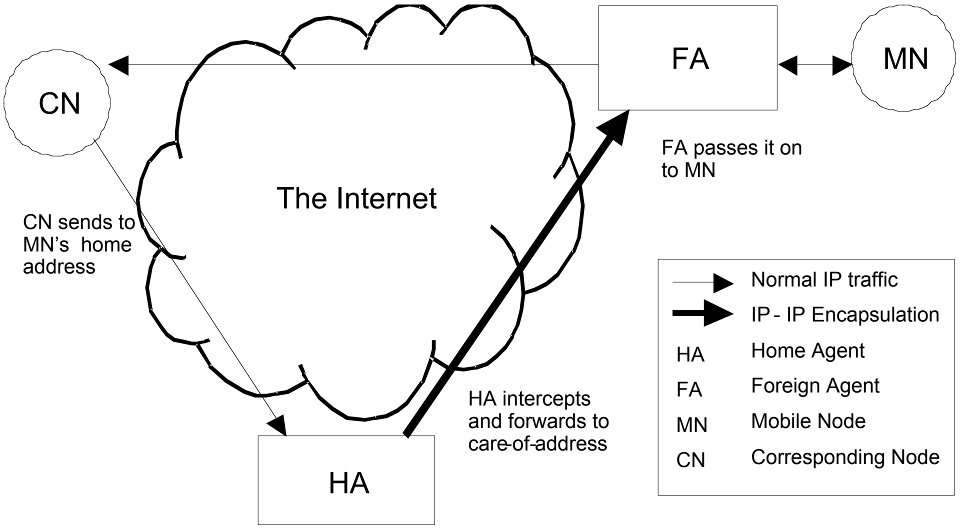

Figure 13 - IP packet forwarding in Mobile IP

Asserting that Mobile IP falls short of satisfying the performance requirements of audio applications, Helmy [7] proposes a multicast-based system for IP mobility support. In this scheme, the corresponding node sends packets to a multicast address to which the mobile node joins from each location visited. This results in a dynamic distribution tree with shortest-path branches reaching all locations visited by the mobile node. Simulation results estimate that the hand-off latency in this system, as measured by the ratio of the number of hops to the Home Agent to the number of hops needed to join the multicast tree, is less than half of that in Mobile IP. The implementation challenges that surround this scheme include the need for the network to support multicast, and for best results, IP version 6.

Ramjee et al. [18] [19] propose a mobility scheme for wireless access networks in which all mobility-related functionality (including mobility between base stations) is handled at the IP layer. Their system uses Mobile IP for macro-mobility between *domains*, but uses another IP-based system, HAWAII, for micro-mobility within domains. A HAWAII domain is an administrative division of the wireless access network and consists of one or more routers and multiple base stations. With a possible size of up to 1000 square kilometers [19], it corresponds more to a wide area wireless network, than to a wireless LAN as we have focused on. HAWAII is an optimized system that avoids many of the delays associated with Mobile IP by allowing registration and de-registration to occur at the base-station level. In this way, when the mobile host moves between base-stations in the same domain, it does not need to send a location update to the home agent but rather updates the routing tables of the base stations and routers on the path to the mobile node using a *forwarding path setup scheme*. The mobile host sends a registration message to its new base station, which in turn sends a HAWAII hand-off message to the previous base station. The previous base station notes the new location of the mobile host and forwards any incoming packets to that location. In addition, it sends HAWAII messages to the router further up in the routing tree, asking it to update its routing entry for the mobile node. The HAWAII system requires base stations to have sophisticated capabilities, making its deployment less straightforward than current wireless networks. The designers of the system have reported simulation results of as little as 5 packets lost per handoff, however, which suggests promise for the scheme (compare this to MVOIP in which approximately 12 packets per second are lost for a hand-off duration of 3 seconds). It is important to note that simulation results cannot always reflect the practical constraints on a system, as the DHCP bottleneck that we discovered in implementing MVOIP clearly illustrates.

Liao [9] proposes an application-level protocol to handle mobility in Internet telephony applications and as such, of all the systems considered here, his approach is the most similar to MVOIP. The proposed design of the Mobile Internet Telephony Protocol (MITP) involves the use of *Home* and *Foreign Agents*. These agents essentially provide callee tracking and call set-up services for registered hosts, since any caller wishing to reach a mobile host will contact the mobile host's Home Agent. As the mobile host roams, it registers with Foreign Agents and sends *binding updates* to the Home Agent, in a manner similar to Mobile IP. The mobile host handles the hand-off of the ongoing call by the join and departure of a multi-point conference. For example, consider a Host A with address "a1" in a conversation with Host B with address "b". Suppose the call-id for this call is "ab1". Host A then moves to a new subnet where it now has an address "a2". It contacts Host B at address "b", requesting to join the call with id "ab1" using address "a2". After it has joined the call, it sends another message to Host B, requesting that "a1" leave the call. MITP can be used either alone, or in conjunction with the Session Initiation Protocol (SIP), which is an alternative to H.323. It has, however, not been implemented [10].

Park et al [14] propose a hybrid application-level and IP-level scheme for handoff management in H.323 calls. Their system proposes to maintain the signaling (Q.931/H.225) and control (H.245) channels during hand-off using Mobile IP. Because these channels are not sensitive to communication latency, the triangular routing through the Home Agent in Mobile IP does not pose a problem. For the media stream, Mobile IP is not appropriate due to its latency, and they propose to handle hand-off in this case by simply closing the old logical channel and opening a new logical channel directly between the two communicating parties using the Mobile-IP care-of-address. The call-control functions already built into the H.245 protocol will enable the opening and closing of logical channels. This system has not been implemented, but by mathematical analysis, they predict a shorter hand-off delay than Liao's MITP (described above), or the proposals for H.323 Annex H (described below).

Finally, the ITU has been developing an extension to H.323 for mobility support, referred to as H.323 annex H. Several parties, including Motorola [12] and AT&T [20] have made contributions for consideration in this standard. These contributions are a high-level description of required mobility support across all aspects of the H.323 standard, concentrating especially on Gatekeeper discovery and registration. In terms of handing off an ongoing call, which is the main focus of our work, Motorola specifies a hand-off process from one *Wireless Access Unit (WAU)* to another, in which a temporary channel is established between the old and the new WAUs for packet forwarding until the mobile host is fully associated with the new WAU. The Wireless Access Unit is a functional entity that houses and manages the radio transceivers and handles the radio-link protocols with the Mobile Host. It appears to have a similar function to the link-layer 802.11 wireless access point, but has extended capabilities geared specifically to H.323. They do not specify how the hand-off process handles changing IP addresses. AT&T, on the other hand, briefly suggests a hand-off process similar to Liao's MITP, that involves the join and departure of a multi-point conference. Both these proposals are very high-level and broad in scope, and we believe that MVOIP could easily be plugged into either of these proposals, to handle the problem of handoff in an ongoing call.

Several of the mobility schemes discussed thus far are IP-level mobility support systems that require significant changes or advancements in the current capabilities of network elements such as mobile hosts, routers, and base stations, or in the case of Helmy, require multicast support. While the long-term solution to IP mobility may lie in systems that make mobility transparent to higher levels, their infrastructure requirements reduce the feasibility of the current deployment of these schemes. Another challenge in handling mobility transparently to higher-level protocols is the conflicting requirements of these protocols. While real-time applications such as voice are tolerant to data loss but intolerant to latency, other applications require loss-less delivery of data and are less stringent in their latency requirements. This makes it difficult to design an IP-layer scheme for mobility that performs satisfactorily for all higher-level applications. For this reason, we believe that an application-level scheme such as MVOIP is a good approach to support mobility in VOIP applications.

7 Conclusions and Future Work

In this paper we present MVOIP, a scheme for handling mobility in Voice-over-IP applications at the application level. The latency of a hand-off is due mostly to the Windows 2000 implementation of the dynamic host configuration protocol, and the duration of the hand-off period could be significantly reduced by an implementation that gives the programmer more control over the API. With such control, the application could begin to use the IP address returned from the DHCP server while verifying that the address is not in use in the background. This optimistic approach would reduce the hand-off latency significantly.

Despite the DHCP bottleneck, MVOIP is a feasible means of handling mobility that can be integrated with current H.323 implementations and requires no special-purpose hardware or support at the IP level. Because hand-off in MVOIP is strictly between the two communicating clients and does not require the use of specialized agents or servers other than the DHCP server, it is a highly scalable approach to mobility.

Foremost in the future work on MVOIP should be the addition of the security measures described in Section 3.3. In addition, other ways to identify subnet changes should be implemented and their performance compared with the method currently used. MVOIP should also be extended to handle situations where the network connection point changes without necessarily undergoing a subnet change, e.g., when a user switches from an Ethernet to an 802.11 wireless connection. Finally, MVOIP should be extended to support the remaining components of the H.323 standard, most importantly, multipoint conferences and use of the Gatekeeper. An H.323 endpoint communicates with a Gatekeeper using the Registration, Admission and Status (RAS) channel, and in addition, might use Gatekeeper-routed call signaling to make a call to another endpoint. A mobile MVOIP host will thus have to hand-off on its connection to the Gatekeeper, as well as to the non-mobile node.

8 Acknowledgements

I would like to thank my advisor, Professor David Kotz, for his indispensable guidance and support in this project. I would also like to thank Ammar Khalid for his continued input and more importantly, for his work on the directory infrastructure to support mobile services. I am also grateful to Guanling Chen, Wayne Cripps, and Ronald Peterson for their repeated technical assistance.

MVOIP would not have been successful without the support of the people at Lucent and Cisco who furnished the relevant documentation concerning their 802.11 products. Finally, I would like to commend the good work of the OpenH323 project [2] and their commitment to creating an open-source implementation of H.323.

9 References

[1] -----, ITU-T Recommendation H.323 (1998), *Packet-based multimedia communication systems.*

[2] -----, The OpenH323 Project, http://www.openh323.org.

[3] Bill Douskalis, *IP Telephony: The Integration of Robust VOIP Services*, Prentice Hall, 2000.

[4] R. Finlayson, T. Mann, J. Mogul, and M. Theimer, "A Reverse Address Resolution Protocol", RFC 903, June 1984.

[5] Anne Fladenmuller and Ranil De Silva, "The effect of Mobile IP handoffs on the performance of TCP," *Mobile Networks and Applications*, Vol. 4, 1999, pp. 131-135.

[6] M. Handley, H. Schulzrinne, E. Schooler, and J. Rosenberg, "SIP: Session Initiation Protocol", RFC 2543, Mar. 1999.

[7] Ahmed Helmy, "A multicast-based protocol for IP mobility support," *Proceedings of NGC 2000 on Networked group communication*, Nov. 2000, Stanford, pp. 49-58.

[8] Ammar Khalid, "A Directory Infrastructure to Support Mobile Services," Computer Science Senior Honors Thesis, Dartmouth College, May 2001.

[9] Wanjiun Liao, "Mobile Internet telephony protocol: an application layer protocol for mobile Internet telephony services," *IEEE International Conference on Communications ICC '99*, 1999 vol.1, pp: 339 –343.

[10] Wanjiun Liao, e-mail communication, 16 May 2001.

[11] Pete Loshin, *Big Book of IP Telephony RFCs*, Morgan Kaufmann, compiled 2001.

[12] Edgar Martinez, Motorola, “Enhancements to ITU-T Recommendation H.323 to support User and Service Mobility,” Document APC-1646, Oct. 1999.

[13] Mark A. Miller, *Voice over IP: Strategies for the converged Network*, M&T Books, 2000.

[14] DongSeon Park, Wonyong Yoon, and Dongman Lee, “An Efficient Handoff Management for Mobility Support to H.323,” *International Workshop on Mobile Multimedia Communications (MoMuC'00)*, Oct. 2000.

[15] Bob O’Hara and Al Petrick, *IEEE 802.11 Handbook: A Designer’s Companion*, IEEE Press, 1999.

[16] C. Perkins, “IP Mobility Support,” RFC 2002, Oct. 1996.

[17] C. Perkins and D. Johnson, “Route Optimization in Mobile IP,” IETF Internet draft, Feb. 2000.

[18] Ramachandran Ramjee, Thomas F. La Porta, Luca Salgarelli, Sandra Thuel and Kannan Varadhan, “IP-Based Access Network Infrastructure for Next-Generation Wireless Data Networks,” *IEEE Personal Communications*, Aug. 2000, pp. 34-41.

[19] Ramachandran Ramjee, Thomas La Porta, Sandy Thuel, Kannan Varadhan, Luca Salgarelli, “IP micro-mobility support using HAWAII,” IETF Internet draft submission, Jul. 2000, available at http://www.bell-labs.com/user/ramjee/papers/draft-ietf-mobileip-hawaii-01.txt.

[20] Radhika R. Roy, AT&T, “H.323 Mobility Architecture and Protocol for Terminal, User and Service Mobility,” Document APC-1652, Oct. 1999.

[21] H. Shulzrinne, S. Casner, R. Frederick, V. Jacobson, “RTP: A Transport Protocol for Real-Time Applications”, RFC 1889, Jan. 1996.