Smooth Handoff in Mobile IP

Master's Thesis by

Babak Ayani

2002-05-14

Department of Microelectronics and Information Technology at KTH

Completed at the Department of Electrical Engineering and Computer Science at University of California in Berkeley

Examiner

Professor Gunnar Karlsson, KTH

UC Berkeley Adviser

Professor Jean Walrand

Department of Microelectronics and Information Technology at KTH

Department of EECS University of California in Berkeley

Abstract

With the increasing popularity of devices such as mobile phones and PDAs, there is a higher demand for wireless access to the Internet. Mobile IP was proposed to provide such access. This report gives a brief introduction and some background to Mobile IP and then focuses on handoffs (i.e. when the mobile node moves from one base station to another) in Mobile IP and especially on how to eliminate the loss of packets during such a handoff.

In particular, this report focuses on handoffs when dealing with real time applications such as voice traffic where there is an upper limit on the delay between incoming packets before the quality of the session becomes unacceptably poor.

We propose a solution for how to eliminate the loss of packets and we compare it to other existing solutions. The proposed solution is to have buffers at every base station (foreign agent) where all the incoming packets are saved. If the mobile node moves to another foreign agent, the packets in the buffer for that mobile node will be sent to its new foreign agent where they are delivered to the mobile node. Thus, the packets will not be lost. We will analyze and compare different buffering management solutions to get the best result. Also, the performance of this proposed solution is analyzed during a handoff to verify that its delay does not exceed the above-mentioned upper limit.

Acknowledgements

First of all, I would like to thank my parents for always being there for me and supporting me. Without their support, I would never have made it this far.

Many thanks to my advisor at UC Berkeley, professor Jean Walrand for giving me the opportunity to come and do my thesis at UC Berkeley. I would also like to thank him for discussing problems of my thesis with me and for reviewing my report.

I would like to thank my supervisor at KTH, professor Gunnar Karlsson for his constructive comments and for proof-reading my thesis.

Many thanks to Hector Velayos at the IMIT department in KTH for reading and giving comments on my thesis, for the interesting discussions we had, and for taking his time and answering all my questions. His help was very appreciated.

I would also like to thank Gaurav Agarwal and Rahul Shah for the interesting discussions I had with them about my thesis and for the useful feedback I've received from them. Also, I would like to thank Gaurav for all the help I've got since I came to Berkeley and Rahul for helping me with trying to do simulations in ns.

A special thanks to Shemida Leopando-Arteta for all her help during my stay in Berkeley, and for being a good friend and listening to all my problems.

Finally, I would also like to thank Anuj Puri at the EECS department in Berkeley for answering my questions about Mobile IP and for his feedback.

Table of Contents

1 Introduction

Due to the increasing use of PDAs, portable computers and cellular phones, there has been an increasing demand for wireless Internet access. However, some problems need to be solved before mobile access to the Internet can become widespread. A first problem is caused by the way the Internet Protocol (IP) routes packets to their destination according to IP addresses. These addresses are associated with a fixed network location and would not work in a wireless environment since when the mobile node moves, it will eventually reach a new network, with a new network number and a new IP address [8].

Mobile IP [3] was designed to solve this problem (and other problems associated with mobile networking) by allowing the mobile node to use two IP addresses: a fixed home IP address and a temporary care-of IP address that changes at each new point of attachment (i.e., at each new network that the mobile node visits).

There are however several additional problems that need to be solved to make Mobile IP efficient. One of the major problems is the loss of packets during handoffs (i.e., when the mobile node moves from one base station to another). We will describe this problem in more detail in section 2.3.2. We will in this thesis concentrate on this problem of loss of packets during handoffs and propose a solution for this problem. In particular, this report focuses on handoffs when dealing with voice traffic where there is an upper limit on the delay between incoming packets before the quality of the session becomes unacceptably poor.

This report is organized as follows. In chapter 2, we first give a brief overview of Mobile IP and then discuss problems that this protocol faces. Chapter 3 describes related work that has been done in the same area and their suggested solutions. In chapter 4, we explain what smooth handoffs are, explain different solutions for achieving them and discuss advantages/disadvantages with these solutions. We also propose a solution for eliminating the loss of packets during handoffs. In the next chapter, Chapter 5, we discuss the protocols that are used in smooth handoffs. In Chapter 6, we introduce and discuss two different buffer management schemes. We also have a buffer size analysis and we compare the two different solutions. Chapter 7 presents an analysis of the performance of the proposed handoff scenario. In this chapter, we will also calculate the number of lost packets when a handoff is performed. The conclusions describe some problems we faced during this work and discuss future work that can be done in this area.

2 Mobile IP Overview

In this paper we will concentrate on IP version 4 (IPv4) when discussing Mobile IP. The reason for this is that IPv4 is a more current topic than IPv6 which lies more in the future and we believe that it is therefore more interesting to concentrate on IPv4. The goal of Mobile IP is to provide mobility support for a mobile host connected to the Internet without having to change its IP addresses. The mobile node is usually attached to the Internet by a wireless link. This link may thus have a much lower bandwidth and higher error rate than the wired links in the Internet. It is therefore a goal of Mobile IP to minimize the number of messages sent over the link by which the mobile node is attached to the Internet and to keep these messages as small as possible. Since the mobile nodes are likely to be battery powered, doing so also reduces the power consumption of the mobile node.

2.1 Operations

As long as the mobile node (MN; for details about the terminology, see 11 - the Appendix - at the end of the report) is on its home network, it receives and sends packets according to normal IP mechanisms. The home network has an agent, called the home agent (HA) that maintains information about the location of the MN. The HA also relays packets to the MN when it is outside its home network, as we explain below.

When the MN moves outside its home network, to what is called a foreign network, it obtains a care-off address (COA) in the local address space of the foreign network from an agent in this network called the foreign agent (FA). The Mobile IP protocol can use two types of COA. In the first case, the COA is assigned to the MN. In this case, we say that the COA is co-located. In the second case, the COA is assigned to the FA and is called a foreign agent COA. A foreign agent COA is the IP address of the foreign agent with which the mobile node is registered while a co-located COA is an IP address temporarily assigned with the mobile node. Since it is complicated to deal with both cases, we choose in this work to only deal with foreign agent COAs. Unless otherwise specifically mentioned, when we talk about COA in this report we mean foreign agent COA.

After having obtained a COA, the MN sends a *Registration Request* to its HA informing it about its new location. In the first case (co-located COA), the MN sends this Registration Request directly to the HA. In the second case (FA COA) the MN sends the Registration Request via the FA.

The HA then sends back a *Registration Reply*, acknowledging the registration request. For more details about the registration procedures, see chapter 4. Any node on the Internet (referred to as a correspondent node, CN) sends packets destined for the MN to the MN's home address in its home network. The HA intercepts these packets and tunnels them to the COA of the MN. The end of the tunnel is the FA if the MN has a FA COA or the MN itself if it is a co-located COA.

A more detailed description of Mobile IP can be found in[3].

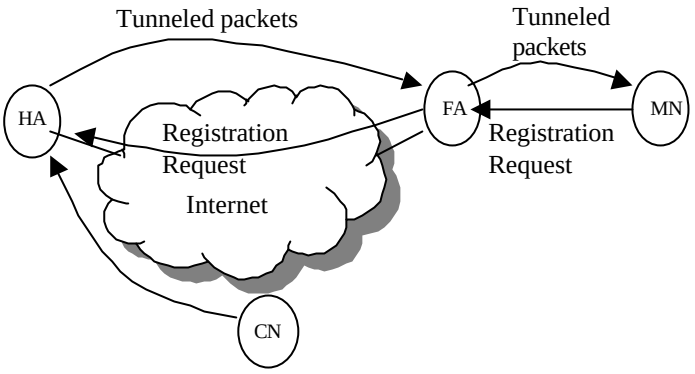

Figure 2-1 MN moving to a foreign network in Mobile IP.

2.2 Procedures

The Mobile IP proposal can be thought of as the cooperation of three major subsystems[10]:

- **Discovery of the Care-of-Address**: The discovery mechanism when the MN finds its new IP address outside its home network as it moves along in the Internet.
- **Registration with the HA**: Once the MN knows the IP address at its new point of attachment, it has to register that IP address with its Home Agent.
- **Tunneling to the Care-of-Address**: The delivery of datagrams to the mobile node when the MN is away from home.

We provide a brief description of these three subsystems below.

2.2.1 Discovering the Care-of Address

Home Agents and Foreign Agents broadcast agent advertisements [4] at regular intervals. These agent advertisements make the HAs and FAs known to a mobile node. If a MN can no longer hear agent advertisements from a FA that previously had offered a COA to the MN, the MN presumes that the FA is no longer within its range and starts searching for a new COA. There are now two possibilities for the MN: it can either register with an existing FA or search for a new one.

2.2.2 Registering the Care-of Address

As we explained above, once the MN gets a new COA it has to inform its HA by sending a *Registration Request.* The HA then denies or approves the request. and sends a *Registration Reply* back to the MN. If the request is approved, the HA updates its cache bindings. If the

request is denied for some reason (for example if the authentication failed or if the HA does not have enough resources), this is also stated in the *Registration Reply*. Obviously, the matter of authentication is very important here. We do not want a malicious host to pretend being a MN, sending a phony Registration Request and having all the datagrams for a MN sent somewhere else. We discuss security in Mobile IP in chapters 5.4 and 5.5.

2.2.3 Tunneling to the Care-of Address

The HA uses IP-in-IP encapsulation [11] to tunnel the datagrams it receives to the MN. This means that the HA inserts a new IP header (also called the tunnel header) in front of the received datagram's IP header (see Figure 2-2). In this IP header, the MN's COA is the destination IP address and the HA is the source address. As explained above, the end of the tunnel can either be the FA (if the MN has a FA COA) or the MN (if it is a co-located COA). Once received by the other end of the tunnel, the tunnel header is removed and the original datagram is delivered to the MN (in the case of FA COA).

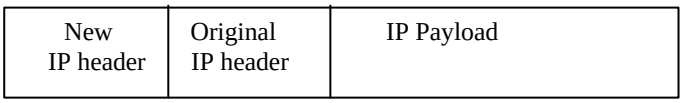

Figure 2-2 IP-within-IP Encapsulation.

2.3 Problems Facing Mobile IP

There are several problems with Mobile IP . Two major problems are the routing inefficiency problem and the loss of packets during handovers. We will talk about these two problems in the following two subsections.

2.3.1 Routing Inefficiency

In Mobile IP, when a CN wants to send a packet to a MN, it first sends the packet to the HA of the MN. The HA then sends the packet to the FA which delivers the packet to the MN. When the CN is very close to the current location of the MN and the HA is very far away, this procedure results in a packet path that is much longer than necessary (see Figure 2-3). It is desirable to overcome this problem by some sort of routing optimization and there have been several suggestions for how to achieve this routing optimization. Some of these suggestions are overviewed in chapter 3. However, we do not focus on this problem in this report.

2.3.2 Loss of Packets during handover

When the MN moves, it will eventually get out of its present FA's reach and have to register with a new FA, as we explained earlier. When the MN is outside its old FA's reach and before the HA has received the MN's new COA, the HA sends packets destined for the MN to its old

FA and thus the packets are most probably lost. This is what “loss of packets during handoff” means.

This loss of packets is the problem that we focus on in this work. We first briefly describe other work within this area to solve this problem and then state our proposal to overcome the handoff problem. Finally, we analyze the performance of the suggested solution.

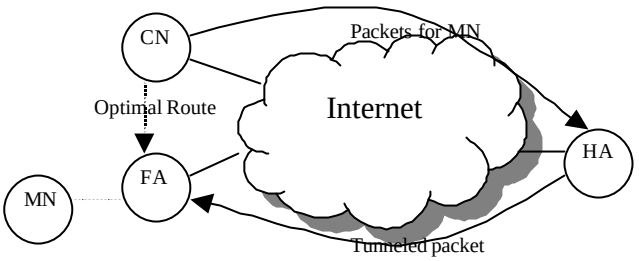

Figure 2-3 Routing Inefficiency Problem.

3 Related Work

This chapter briefly describes related work that has been done in this area.

3.1 Routing Optimization

Mobile IP route optimization [12] was designed to provide a solution for the problems mentioned in section 2.3. The solution is as follows.

Any host on the Internet that wishes to participate maintains a *binding cache*. When the HA receives a packet for a MN that is away, the HA sends a *binding update* message to the source of the packet (the CN), informing it of the MN's current COA. The source then updates its binding cache and sends other eventual packets to the MN *directly* to it without bypassing the HA. This way, we can avoid the triangular routing problem.

FAs can also make use of the binding updates to reduce packet loss during a handoff. When a MN changes its COA from one FA to another, the new FA may send a *Binding Update message* to the old FA, informing it of the MN's new location. The old FA then updates its binding cache and re-tunnels any incoming packets for the MN to its new COA. This process is called a *smooth handoff*. However, according to this scheme, packets that arrive at the old FA before it has received a Binding Update message are still lost.

3.2 FA Buffering

Perkins and Wang propose in [6] a solution for the problem mentioned in section 2.3.2 (loss of packets during a handoff). The solution is as follows: In addition to the smooth handoff scheme, the FAs should have a buffering mechanism. Besides decapsulating packets and delivering them to the MN, the FA should also buffer these packets. When it receives a Binding Update message, the FA re-tunnels the buffered packets to the MN's new FA. With this solution, packet loss during a handoff can be completely eliminated unless the MN takes too much time to find a new FA (after it loses contact with the old one), in which case the buffer at the previous FA may overflow. A major side effect of this buffering scheme is however the duplication of packets. The new FA may get packets from the old FA that the MN has already received from the CN (while at the old FA).

To prevent these duplications, Perkins and Wang propose that when the MN receives an IP datagram, it buffers the pair of the source address and the identification (originally used for IP fragmentation) field of the datagram given in the IP header. When the MN requests a smooth handoff, it includes the source address and the identification field of the packets it has already received in the Binding Update message. The previous FA then uses these buffered pairs to drop those buffered packets that have already been sent to the MN and only sends the rest of the packets.

3.3 Hierarchical FAs

In order to have faster handoffs and also decrease the registration overhead that increases traffic in the Internet (this is specially apparent in base stations with small cells where frequent handoffs occur) a hierarchical FA management (see Figure 3-1) was proposed in the same report[6]. The FAs in a domain are organized into a hierarchy (or more precisely, a tree of FAs) to handle local movement of MNs inside the domain. A FA includes in its Agent Advertisements a vector of COA, which are the IP addresses of all of its FA ancestors as well as its own.

When a MN arrives at a new FA, it registers with its HA that FA as well as all the FA's ancestors.

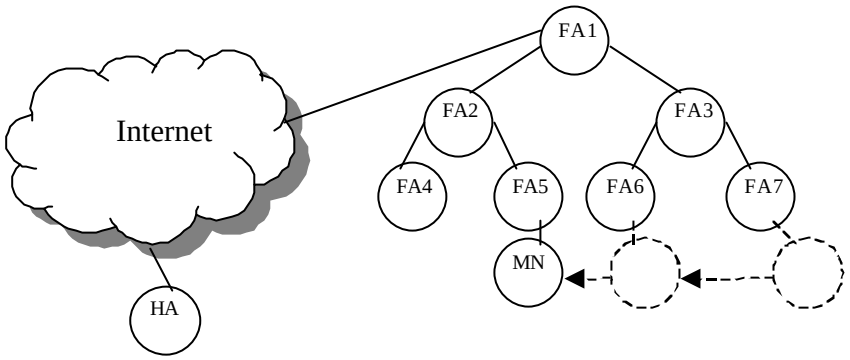

Figure 3-1 Hierarchical FAs.

When a packet for the MN arrives at its home network, the HA tunnels it to the root of the FA hierarchy. When the root FA (FA1 in Figure 3-1) receives such a packet, it re-tunnels it to its next lower level FA. Finally the lowest level FA delivers the packet to the MN.

When a handoff occurs, the MN compares the new vector of care-of addresses with the old one. It chooses the lowest level FA that appears in both vectors and sends a Registration Request to that FA. The higher-level FAs are not informed of this movement since it does not concern them. For Example, In Figure 3-1, when the MN moves from FA7 to FA6, FA3 is the target of the registration request, and FA1, without knowledge of this movement, still correctly re-tunnels packets to FA3. In the meantime, the HA has no knowledge of these local movements (it still tunnels packets for the MN to FA1 as usual) and none of these registrations reaches the HA. Thus, registration overhead in the network is reduced.

There are several other reports that also look into hierarchical mobility management [9]and[17]. These papers focus on the handoffs in a hierarchical scheme.

A. Stephane et al [17] look into and compare the two different scenarios when we have intra domain handoffs (handoffs within the same domain) and inter domain handoffs (handoffs between two different domains).

4 Smooth Handoffs

In this section, we look into different solutions for achieving smooth handoffs and discuss problems and advantages/disadvantages with the different solutions. Finally we describe our proposal for achieving smooth handoffs.

4.1 What is a Smooth Handoff?

When the MN moves, it may get out of its present FA's reach and have to register with a new FA. This change of FA is referred to as a handoff. During the time that the MN is outside its old FA's reach and before the HA has received the MN's new COA, the HA will send packets destined for the MN to its old FA and thus the packets are most probably lost.

Smooth handoff deals with how to minimize and hopefully totally eliminate the loss of packets during a handoff.

4.2 Different Solutions for achieving Smooth Handoffs

There have been several proposals for how to achieve smooth handoffs. This chapter deals with different solutions to achieve these smooth handoffs.

4.2.1 Multicasting Packets

One way to have smooth handoffs is to multicast the packets. Besides sending the data to the FA where the MN is at the moment, the data is also sent to all adjacent FAs. The other FAs will buffer incoming packets and can quickly forward them to the MN if a handoff occurs. This solution is especially useful when we have Real-time services such as sending audio (Internet telephony, video conferencing etc) when there is a constraint on the delay between subsequent packets. The downside with this solution is that it uses additional network resources and memory space by sending data to *all* adjacent FAs and buffering the data there. Another problem is how to know which FAs are adjacent each other in order to know to which FAs a packet should be sent. We will discuss advantages and disadvantages with multicasting packets in section 4.4.1.

4.2.2 Buffering at the HA/FA

Two other solutions are buffering of all incoming packets at the HA or at the FAs. When the MN moves to a new FA, the new FA sends an update message to the HA or old FA (depending on whether the buffering is at the HA or the old FA) telling it to send incoming packets for the MN to it's new location. Packet loss during a handoff can then be completely eliminated, unless the

MN takes too long time to find a new FA after it loses contact with its previous FA, in which case the buffer at the previous FA or the HA may overflow.

4.3 Problems with Buffering at the FA/HA

A major side effect of this proposed buffering scheme is the duplication of packets. The new FA may get packets from the old FA/HA that the MN has already received from the CN (while at the old FA). Modern implementations of TCP [1] assume that duplicated TCP acknowledgements are caused by lost data packets and will in that case invoke some sort of congestion control mechanisms[5]. Because of this, duplicated acknowledgement packets due to handovers may cause upper layer protocols like TCP to slow down more than necessary , degrade performance [1] and also waste network resources. This is obviously not wanted and we would like to avoid this.

We could overcome this problem of acknowledgement packet duplication by having the MN send an acknowledgement to the FA/HA for every packet that it receives whereby the FA/HA would delete that packet from its buffer. In this case, the only packets that are left in the buffer are the ones that have not been delivered to different MNs (remember that several MNs usually are connected to one FA/HA). When the old FA/HA receives an update message for a certain MN, it simply sends all buffered packets intended for that MN to the new FA. These acknowledgements are however sent on the link layer level and are outside the scope of this work.

Another solution could be to use some sort of sequence numbering on all the packets and save that numbering in the buffer together with other information. We discuss the information that needs to be saved in the buffers in section 6.3.2. The MN could then send the sequence numbers of all the packets that it has received together with the Binding Update message, and the old FA would then discard those packets and just send the other packets that it didn't receive a sequence number for. The Identification and fragment offset fields of the IP header could for example be used as the sequence number for the saved packets.

The disadvantages of this scheme are several: to start with, the MN now needs to keep track of the sequence numbers of all the packets that it has received (remember that the packets can arrive out of order at the MN, so it is not enough to keep the sequence number of just the last received packet.). Also, since the packets are not deleted from the buffer when the MN receives and acknowledges them, the buffer becomes full faster and there are more unnecessary packets in the buffer. This could be a major drawback since new arriving packets could be discarded because the buffer is full with packets that have already been delivered to the MN and are not really needed in the buffer. This could also happen to other packets in the buffer that have not been delivered, depending on the chosen algorithm in case the buffer is full. Considering these points, we believe the best solution would be to have the MN send a link layer acknowledgement for all the packets that it receives.

As Perkins and Wang mention in[6] "whether and how much packet loss can be avoided depends on how quickly an MH finds a new FA, and how many packets are buffered at the previous FA. This in turn depends on how frequently FAs send out beacons, or agent advertisements, and how long the MH stays out of range of *any* FA. A large buffer at an FA can tolerate less frequent beacons and longer period of loss of contact. On the other hand, more frequent beacons take up

more wireless bandwidth and denser coverage requires more FAs (i.e. more equipment). Balancing these factors is important for achieving optimal smooth handoff." We will look into some of these factors in chapters 6 and 7 .

.

The first question to consider about buffering incoming packets is whether to have the buffering at the HA or at the FAs. Here is a short summary of the advantages and disadvantages of the different schemes:

4.4 Advantages and Disadvantages of the Different Solutions

In this section we discuss the advantages and the disadvantages of the earlier discussed solutions for achieving smooth handoffs.

4.4.1 Advantages/Disadvantages of Multicasting

The disadvantage with multicasting the data to all adjacent FAs is that it uses additional network resources and memory space by sending data to *all* adjacent FAs and buffering the data there. Another disadvantage is the problem of how the new FA is supposed to know which was the last packet received by the MN at the old FA. Yet another problem is how to know which are adjacent Fas.

The advantage is that the packets will already be at the new FA when the MN moves so that the handoff will be much faster.

4.4.2 Advantages/Disadvantages of Buffering at the FA

The IP distance (distance that packets on the IP level have to travel; this is not the same as the physical distance) between the old FA and the new FA should probably be less than the one between the new FA and the HA. Thus, the packets will have to travel less, the delay will be less and there will be less traffic in the network if the buffering is at the FAs.

This is an important reason for having the buffering at the FAs. The major problem with buffering at the FAs is security and authentication between the new and old FA. How do you validate the authenticity of the new FA so that a malicious node can't pretend to be serving the MN and have the old FA send all the buffered packets to it instead of to the new FA? This is a very important issue that needs to be looked upon. We discuss authentication and security during handoffs in chapter 5.5. Another disadvantage is that the FA might not have enough resources that are needed for buffering packets for all MNs connected to it at different times. For example, the FA might be able to serve and buffer a maximum of 100 MNs. At a certain time, maybe 110 MNs are in the coverage area of that FA and want to register with that FA. The FA will then not be able to provide service for all these MNs and some incoming packets might not be buffered due to lack of resources and thus be discarded. The HA on the other hand has to guarantee that it will provide service to all MNs in its domain and thus, that problem would not occur if the buffering is at the HA instead.

If the buffering is at the HA, the packets are saved in a centralized way instead of having them distributed among several FAs in a decentralized fashion. The disadvantage of having a centralized system is that should the HA crash for some reason all the information gets lost. Also, the HA could become a bottleneck if the traffic is very heavy.

4.4.3 Advantages/disadvantages with buffering at the HA

One advantage with buffering at the HA is that there is no need for sending a binding update message to the old FA. In this scheme, the binding update message can be included in the registration request that is sent to the HA. Thus, instead of sending two messages (one update message to the old FA and one registration request to the HA) it is enough to send only one message. This is however only a very small advantage. More importantly, the authentication problem is much easier solved here since the HA and MN already have a way of authenticating with each other (see [3] pages 65-66) and can use that. A major disadvantage of buffering at the HAs is when the MN is sending acknowledgements for received packets. These acknowledgements have to go through the Internet all the way back to the HA. If the acknowledgements are sent on the link layer, this is impossible (link layer acknowledgements can't be sent through the Internet to the HA). Also, there would be a triangular path since the *FA* sends data to the MN, but the acknowledgement is instead sent to the *HA*.

Another major disadvantage of having the buffering at the HA is that the distance between the HA and the new FA is probably longer than the distance between the old FA and the new FA (see 4.4.2).

All in all, considering the arguments mentioned above, we believe that it is better to buffer at the FAs than at the HAs. Also, in the literature ragarding buffering that I have read, the researchers who have studied this question shared this opinion [6]-[7].

4.5 The Smooth Handoff Scenario

In our handoff scenario, we are assuming that there are no "dead zones" between two adjacent FAs, which means that the MN is always within the cell of at least one FA. However there can be some overlaps between the cells that adjacent FAs cover which means that the MN could receive agent advertisements from several FAs. See section 7.1 for different solutions on which FA to use in such a case.

Let's assume that the solution is based on signal strength and that there is some overlap between the cells of adjacent FAs. As the MN starts moving it notices that the signals it is receiving from the FA are getting weaker. The MN then starts looking for another FA. When it receives agent advertisements from this new FA, the MN initiates a handoff by sending a Registration Request (see section 5 for details about the handoff protocols) to this new FA that forwards it to the MN's HA. The new FA also sends a binding update message to the old FA informing it about the MN's new location and its new COA and asking the old FA to send buffered packets for the MN to this COA (see Figure 4-1). The old FA then sends an acknowledgement for this update message and then *re-tunnels* all the buffered packets to the new FA. The new FA then decapsulates these packets and delivers them to the MN.

When the HA receives the registration request it processes the request and then sends a registration reply back to the FA that relays it to the MN. The registration reply holds information about whether the request was accepted or denied and the reasons for that.

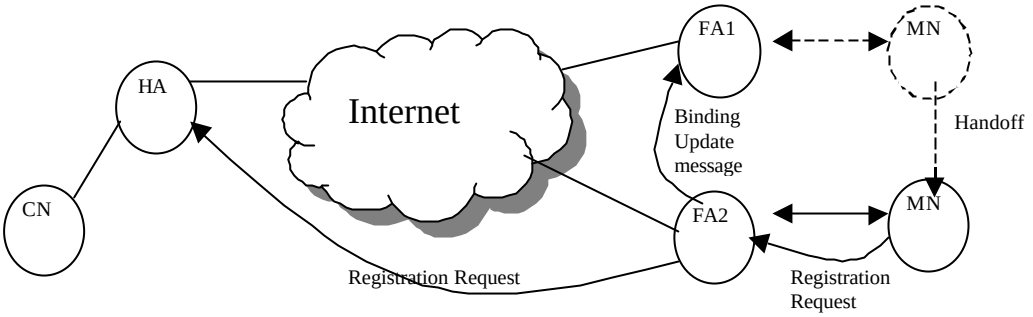

Figure 4-1 The Smooth Handoff Scenario.

5 The Proposed Handoff Protocols

There are five messages that are exchanged to complete a handoff. These messages are the following:

1. **Agent Advertisement:** Sent by FAs and received by MNs.
2. **Registration Request:** From the MN to the HA via the FA.
3. **Registration Reply:** From the HA to the MN via the FA in reply to a registration request.
4. **Binding Update:** From the new FA to the old FA notifying the old FA of the MN's new location (its new COA) and telling it to send buffered packets for the MN to that address.
5. **Binding Acknowledgement:** Sent from old FA to new FA to acknowledge the binding update message.

We will discuss these messages in more detail in this chapter. Of these five messages, the three first ones are exactly the same as proposed in the Mobile IP protocol specification [3] and the last two are my proposals. The only addition to the three first protocols is that the registration request should also contain the MN's old FA IP address in its extensions. If the MN had a shared secret with its previous FA, this authentication key should be sent in the registration request as a mobile-foreign extension. These two pieces of information need to be sent from the MN to its new FA so that the new FA knows where to send the Binding Update message and to be able to authenticate with the old FA. However, this information is not needed to be sent to the HA. The FA could either remove these two fields before forwarding the registration request to the HA, or just leave it as it is in which case they would just be ignored by the HA. Fore more details about authentication and other security concerns, see sections 5.4 and 5.5.

Before discussing the different protocols in detail, we need to answer the following two questions:

Question 1. What information does the MN need to carry with it when it moves from one FA to another?

First of all, the MN needs to know the IP address of its HA in order to know who to register with every time it moves. The MN needs to carry the address of the FA that it was previously attached to so that the new FA learns where to send the Binding Update message. The MN also needs to carry its home address. This information is included in the registration request that is sent from the MN to the HA via the new FA. Also, the previous FA needs to know the home address of the MN so that it knows which packets from the buffer to send to the new FA. This information should be included in the binding update message sent from the new FA to the old FA.

Finally, depending on what authentication scheme was used between the MN and the old FA and what solution is chosen for authentication between the new FA and the old FA, the MN might want to pass the authentication key it was sharing with the old FA to its new FA. The new FA can then use this authentication key to authenticate with the old FA. Specifically, the new FA sends this authentication key to the old FA that checks whether it matches with its own authentication key.

Question 2. What information does the new FA need from the old FA?

Besides the data in the buffer, the MN carries all other information that the new FA needs about it, so there is no important information that the new FA needs from the old FA.

We now describe the above-mentioned protocols. Since the three first protocols are identical or almost identical to the Mobile IP specification protocols, we do not go into detail describing those but refer instead to[3].

5.1 Agent Advertisements

This protocol is as mentioned earlier exactly the same as proposed in the Mobile IP RFC[3].

Mobility agents (HAs and FAs) transmit advertisements to advertise their services on a link. MNs use these advertisements to determine their current location in the Internet. An Agent Advertisement is an ICMP Router Advertisement [4] that has been extended to also carry a mobility Agent Advertisement Extension which is shown in Figure 5-2. The length field is equal to (6 + 4*N) where N is the number of care of addresses advertised. The Sequence Number fieled contains the number of Agent Advertisement messages sent since the agent was initialized. The Registration Lifetime contains the "longest lifetime measured in seconds that this agent is willing to accept in any Registration Request". The code part in the protocol includes among others information on whether the mobility agent is a FA or a HA and if the FA is busy and will not accept registrations from additional MNs. For more information, see[3].

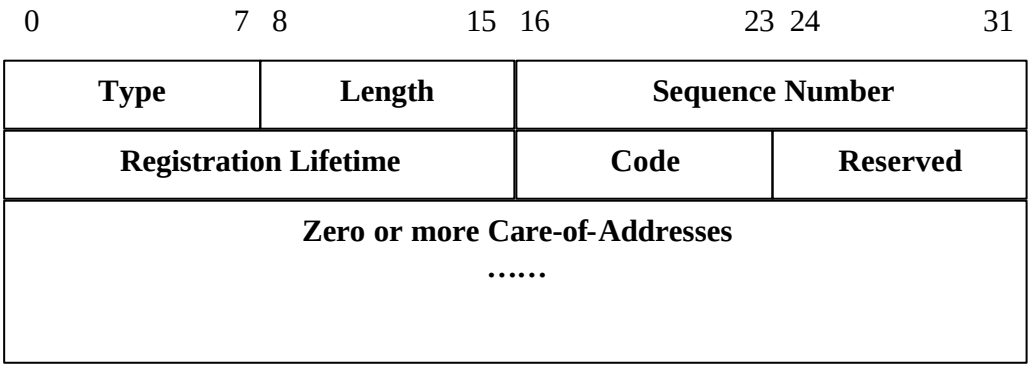

Figure 5-1 Mobility Agent Advertisement Extension.

5.2 Registration Request

As mentioned earlier, this protocol is the same as proposed in the Mobile IP RFC[3].

The Registration Request and the Registration Reply are both sent to UDP port 434. The overall structure of these registration messages is shown in Figure 5-2 where the Mobile IP message header looks like either Figure 5-3 (if it's a registration request) or Figure 5-4 (registration reply).

Figure 5-2 Data structure of a registration message.

The registration process is almost the same whether the MN has a FA COA or a co-located COA. In the former case, the MN basically sends the request to the FA which relays the request to the HA. In the latter case, the MN sends its request directly to the HA, using its co-located COA as the source IP address of the request.

Figure 5-3 shows the protocol format for a registration request according to the Mobile IP specification.

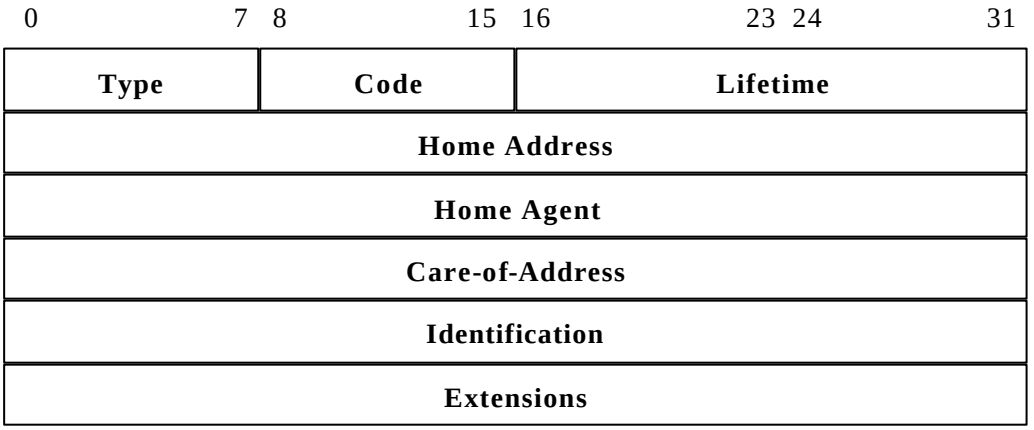

Figure 5-3 Registration Request Format.

The code part contains among other things information about whether the MN is using a co-located COA or a FA COA. The lifetime contains the number of seconds remaining before the registration is considered expired. A value of zero indicates de-registration and a value of 0xffff indicates infinity. The home address contains the fixed home IP address of the MN. Home Agent is the IP address of the MN's home agent. The identification is a 64-bit number constructed by the MN, used for matching registration requests with registration replies. For more details, see[3]. The Extension part should include the old FA address of the MN (the IP address of the MN's old FA). It could also include the Authentication key that the MN was sharing with its previous FA. It is sent from the MN to the new FA for authentication purposes between the new FA and the old FA. The information in these two fields are of no interest to the HA and will therefore be ignored by the HA.

5.3 Registration Reply

Figure 5-4 shows the protocol format for a registration reply according to the Mobile IP specification. The code field contains a value indicating the result of the registration request (see below). If the Code field indicates that the registration was accepted, the Lifetime field is set to the number of seconds remaining before the registration expires. If the code field indicates that the registration was denied, the content of the lifetime field is unspecified and should be ignored by the MN on reception. The other fields are the same as in the registration request protocol.

The following values are defined for use within the Code field:

5.3.1 Registration Successful

0 registration accepted
1 registration accepted, but simultaneous mobility bindings unsupported

5.3.2 Registration Denied by the FA

64 Reason Unspecified
65 Administratively Prohibited
66 Insufficient Resources
67 MN failed Authentication
68 HA failed Authentication
69 Requested Lifetime too long
70 Poorly formed Request
71 Poorly formed Reply
72 Requested encapsulation unavailable
73 Requested Van Jacobson compression unavailable
80 Home network unreachable (ICMP error received)
81 Home agent host unreachable (ICMP error received)
82 Home agent port unreachable (ICMP error received)
88 Home agent unreachable (other ICMP error received)

5.3.3 Registration Denied by the HA

128 Reason Unspecified
129 Administratively Prohibited
130 Insufficient Resources
131 MN failed authentication
132 FA failed authentication
133 Registration Identification mismatch
134 Poorly formed Request
135 Too many simultaneous mobility bindings
136 Unknown home agent address

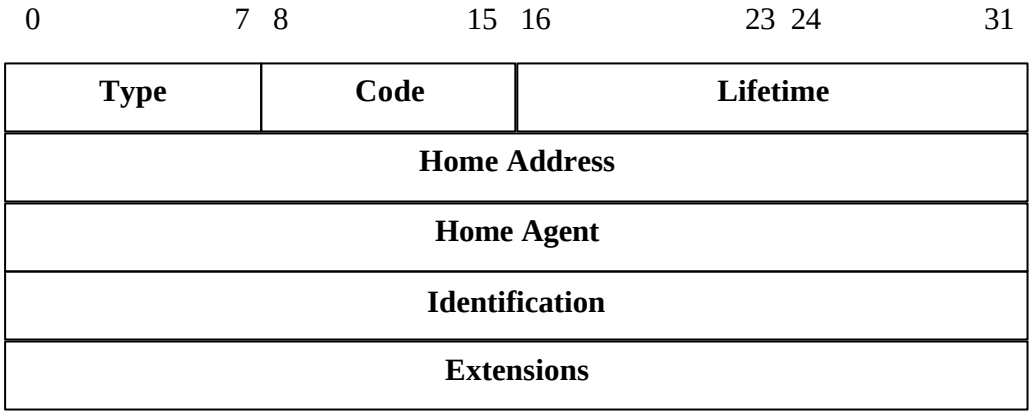

Figure 5-4 Registration Reply Format.

5.4 Security During the Registration Procedure

As mentioned earlier, the registration in Mobile IP must be secure so that false registrations can be detected and rejected. Otherwise, a malicious host could for example pretend being the MN, send a fake Registration Request and have all the datagrams for that MN sent somewhere else.

In order to solve the authentication problems, as C. Perkins mentions in [8] "each MN and HA must share a security association and be able to use Message Digest 5 with 128-bit keys to create unforgeable digital signatures for registration requests [14]". This security association (also called an authenticator) is included in the authentication extensions (see explanation further down). Further on, as mentioned in [8] 'the signature is computed by performing MD5's one-way hash algorithm over all the data within the registration message header and the extensions that precede the signature. To secure the registration request, each request must contain unique data so that two different registrations will in practical terms never have the same MD5 hash."

The method to do this is to include a unique value along with the registration request (in the identification field) that changes with every new registration. There are two ways of making the identification field unique. The first way is to use a timestamp. In that case, every new registration has a later timestamp and thus differs from previous timestamps. The other way is to generate a random number (a nonce) and insert it into the identification field of the registration request. With enough bits of randomness (usually 32 bits are used), it is very unlikely that two independently chosen values for the identification field will be the same.

There are three authentication extensions that are defined for use with Mobile IP. These authentication extensions are included at the end of the registration requests and registration replies. They are the following:

- **The mobile-home authentication extension:** this extension is required in all registration requests and replies.
- **The mobile-foreign authentication extension:** This extension may be included in the registration requests and replies in cases when a mobility security association exists between the MN and the FA.

- **The foreign-home authentication extension:** This extension may be included in the registration requests and replies in cases when a mobility security association exists between the FA and the HA.

The format for these extensions can be seen in Figure 5-5.

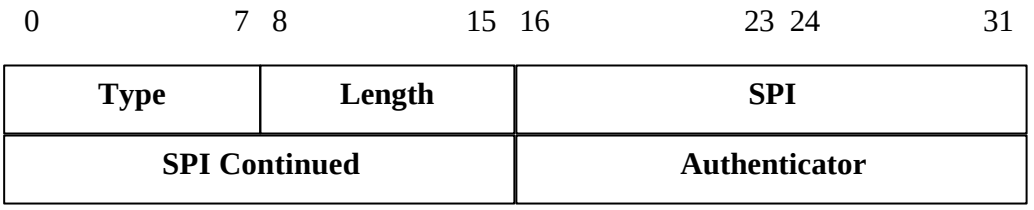

Figure 5-5 Mobile IP Authentication Extension.

All these three authentication extensions have similar formats. The only difference is that they have different type numbers.

5.5 Security During the Binding Update Procedure

Whenever a binding update message is transmitted, it has to be accompanied by an authentication extension. However, the authentication process is a little bit trickier in the case of handoffs than during the registration procedure. The reason for this is that the MN and the FA might not share any special secret that can be used to build a security association (which then could be used for authentication purposes).

Even without a shared secret, the new FA has to persuade the old FA that the binding update message has not been forged. The overall procedure for the authentication process is like this:

- The FA uses agent advertisement flags and extensions to provide information about the style of the security that it is prepared to offer the MN.
- The MN then chooses one of theses available services.
- The FA responds to the MN's request and if necessary, it cooperates with the MN to provide a smooth handoff operation and to obtain a registration key from the HA.

If a security association exists between the MN and the old FA, the MN can create a registration key by picking a random number and encoding it using their shared secret. The MN would then send this registration key to the new FA (in the registration request) which would then include it in the binding update message. In this case, the registration request has to include a mobile-foreign authentication extension (see chapter 5.4).

However, in most cases, a security association will not exist between the MN and the FAs. In those cases, the MN will instead rely on the HA to pick and provide for a registration key to be used by the MN and the FA. This can be done in two ways. If the FA and the HA share a security

association, the FA can request the HA to encrypt a certain selected registration key using their security association and send back the result to the FA as part of the registration reply. The HA will also notify the MN of this registration key value by using the mobility security association that always exists between a MN and its HA.

On the other hand, if the FA does not have a security association with the HA but instead has a public key, it can send this public key to the HA in the registration request, and more or less achieve the same result as explained above.

Lastly, if the FA does not have a public key, or a security association with either the MN or the HA, there is still a chance of a *Diffie-Hellman* key exchange. I will not in detail explain how this works but will instead refer to[15].

5.6 Binding Update Message

Figure 5-6 shows the protocol format for a binding update message. As explained earlier, it is sent from the MN's new FA to its old FA notifying the old FA of the MN's new location (COA) and telling it to send buffered packets for the MN to that address. The Authentication extension is for authentication between the new FA and the old FA and can for example contain the SPI and Authenticator that the MN shared with its old FA (which the MN will send to its new FA in the Registration Request). For more details about this, see section 5.5. A possible problem could occur if there is no connection between the old FA and new FA (e.g. there is a firewall). How do you solve that? However, a requirement for this work is that *there is* a connection between the old FA and the new FA. This problem is therefore outside the scope of this work.

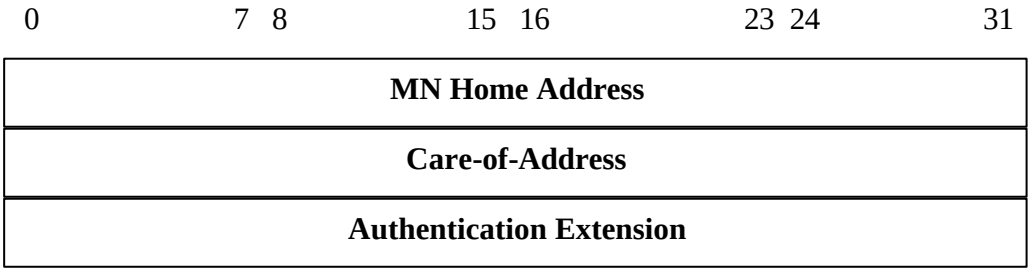

Figure 5-6 Binding Update Message Format.

5.7 Binding Acknowledgement Message

Figure 5-7 shows the protocol format for a binding acknowledgement message. It is sent in response to a binding update message so that the sender of the binding update message will know that the old FA received the update message. The code part of the protocol will contain the result of the update. The reason why an acknowledgement to a binding update should be compulsory is that otherwise, if the new FA sends an update message and does not get any buffered packets

sent back to it, how will it know if this is because there were no packets for that MN in the old FA's buffer or because the binding update message somehow was lost and never received by the old FA? If a binding acknowledgement is not received after a certain time, the new FA should re-send a binding update message until it receives an acknowledgement.

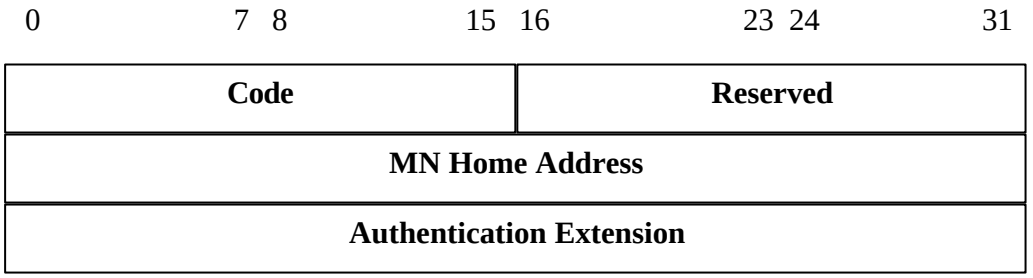

Figure 5-7 Binding Acknowledgement Message Format.

6 The Proposed Buffer Management Schemes

There are several issues that need to be considered regarding how the buffering of incoming data at the FAs should be managed. In this section we will look into these issues in more detail. The following are some of the matters that need to be worked on:

- How do you save (buffer) the incoming packets? For example, what information besides the actual data needs to be cached? Should incoming packets for different MNs be cached separately or all together at the same place?

- What kind of algorithm or solution should be used in case the buffer becomes full? Should the incoming packets just be discarded or should they be switched with a packet in the buffer? And in that case, what algorithm should be used (FIFO, priority, randomly etc.)?

- If several MNs are connected to one FA, how does the FA decide which node to serve? Does it allocate a certain time slice to each MN and then send all packets in the buffer for that MN to it? Or does the FA take one packet at a time from the beginning of the buffer (in the case of having only one buffer) and send it?

- If the FA takes a packet from the buffer and tries to send it to a MN but the MN has moved to another cell (i.e. a handoff has occurred), what happens?

- What do you do if a packet is sent for a MN but no acknowledgement is received? This could happen due to several different reasons: one reason could be that the packet is lost on its way to the MN and never reaches the MN. Another reason could be that the packet arrives at the MN but the acknowledgement is lost and never reaches the FA. Yet another reason could be that the MN moves to another cell which results in that the packet never reaches the MN.

- If an acknowledgement is not received, should the packet be retransmitted after a certain maximum roundtrip time or not?

6.1 Implementation of Wireless Networking in Real Devices

Before we start discussing the different possible buffering schemes, we need to gain a little bit more knowledge about how networking (and specially wireless networking) is implemented in real devices. In the following section, we will discuss this in more detail.

6.1.1 Network Adaptors

According to [20], normally when a host (in our case the FA) receives an IP packet, the packet is sent to a lower layer in the network architecture (such as the link layer) whereby the packet is no longer available at the IP layer. It is then up to the lower layers to send the packet to the network link where it is sent to its destination (a MN in our case). See [20] for more details about this.

Each host is connected to the network via a network adaptor. The network adaptor usually sits on the system's I/O bus and delivers data between the host's memory and its network link. Figure 6.1 (copied from [20], page 71) shows this.

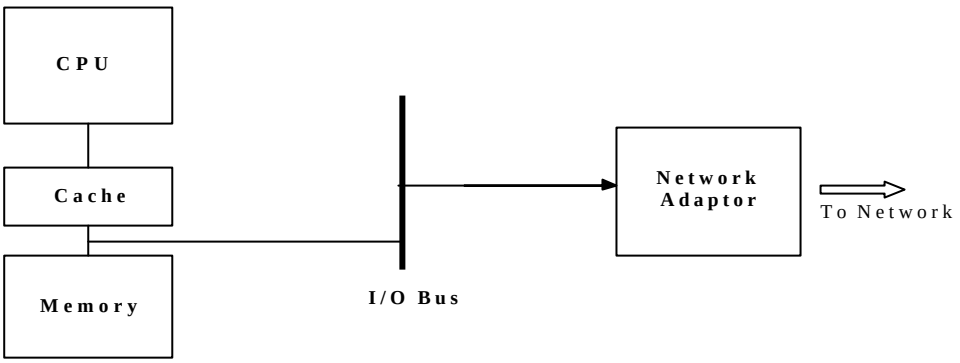

Figure 6-1 The architecture of a host containing a network adaptor.

The normal behavior is that whenever a host wants to send packets on its network, it first sends the packets to the network adaptor (and the packet is thereby no longer available at the IP level), which then sends the packets on the network link. If the network adaptor cannot send a frame/packet (e.g. when the MN that the frame is for has moved to another cell) it will notify the upper level which is the LLC (Logical Link Control) and it is up to the behavior of the LLC to check if the frame will be sent again or not. It is important to notice that there are two different memory spaces involved here: one in the host and one in the network adaptor. If a packet is moved from the host memory to the adaptor memory without saving a copy in the host's memory (which is normally the case), the packet is basically lost when transmission fails. This would mean that packets would be lost during handoffs in our case.

To solve this problem, a solution is for the FA to save a copy of the packets sent to the network adaptor in the host to allow retransmission and forwarding to a new FA. These copied packets are deleted from the buffer in the host when the network adaptor confirms that the packet was transmitted successfully. For more information about the implementation details, see [20] and [24].

Another thing that is important to pay attention to is that the network device is a serial device, it deals with one message at a time (even though it can store several messages internally). This implies in our case that when a packet has been sent on the network link, no other packet can be sent before either an acknowledgement for that packet has been received or the packet just sent is declared lost (there are several mechanisms to detect that a transmitted packet was lost and they mainly depend on the link layer technology).

6.2 Possible Solutions

There are two ways of saving the incoming data packets: they can either be saved all at the same buffer or we can have one buffer for every MN. In the following sections we will discuss advantages/disadvantages and other issues concerning these two solutions. We will discuss issues about the buffering management scheme for both cases. We will also discuss which of these two solutions that is the fastest, talk about parameters that affect the buffer size and possible solutions if the buffer(s) is/are full. Finally, we will give one example for each solution of what happens starting when a packet arrives at the FA and ending when the MN receives the packet and sends an acknowledgement for it to the FA.

6.3 Having only one Buffer

Considering that up to 50 or 100 MNs can be and usually are attached to one single FA, it seems much easier to implement and maintain only *one* buffer instead of having a buffer for each MN. The question is however which solution that is the fastest, i.e. has the shortest lookup time and which one that utilizes the buffer/buffers more efficiently. We will look into this question in section 6.7. In the following sections, we will discuss other issues about having only one buffer at the FA.

6.3.1 The Buffer Size

The question of how big the buffer should be is a pretty complex question that depends on a lot of factors. It depends on how many clock cycles it takes for the FA to take one packet and send it, the lifetime of the packets and the size of the packets. It also depends on the rate by which packets arrive at the FA which in turn depends on how many MNs are connected to the FA, how frequently packets for the MNs are sent in average, and how many MNs that are receiving data packets at one time.

Another factor is how often the MNs move to another cell. This will tell us how often the FA receives binding update messages in which case the FA has to interrupt sending packets and instead look through its buffer. There are a lot of other factors influencing how fast the buffer becomes full. We will discuss these factors and other issues concerning the buffer size in section 6.6.

6.3.2 Buffering of Incoming Packets

If we only have *one* buffer at every FA where all the incoming packets are saved, we need a way to distinguish between different MNs. The way to do this is by the MNs' home addresses. This address can be accessed from the inner (the original) IP header where it is the destination address of the packet.

Also, a certain lifetime must be chosen for every FA or possibly all FAs in a certain domain. All incoming packets for that FA (or all FAs in that domain) would then have this lifetime. When the lifetime for a packet expires, the packet is removed from the buffer. For this to work, every packet will get a timestamp when they arrive. Also, the FA must have a timer.

Here is an example to illustrate how this works: assume that the lifetime for a certain FA is chosen to 2 seconds. The FA will then search its buffer in even intervals (e.g. every 5 ms.) and subtract the timestamp of every packet from the current time. If this value is equal or greater than 2 seconds, the packet is deleted from the buffer and can be replaced by another (incoming) packet.

Figure 6-2 shows how packets are buffered at the FA. For every packet we have a timestamp and then the de-capsulated Mobile IP packet.

Figure 6-2 How packets are cached in the buffer.

6.3.3 Sending Packets to the MN

The packets in the buffer are served in a FIFO order. That means that the FA takes packets from the top/beginning of the buffer and sends them to the right MN. Arriving packets are always saved at the end of the buffer (called buffer1). After a packet has been sent, the FA takes the next packet from the top of buffer1 and sends it. If a link layer acknowledgement for a sent packet is not received after a certain maximum roundtrip time (which is technology dependent), the packet should be retransmitted.

Of course, if the lifetime of a packet in the buffer expires before an acknowledgement is received (which could happen because the packet for some reason never was delivered to the MN or that the acknowledgement was lost), the packet will be deleted.

In the cases where packets are dropped, e.g. if the lifetime of a packet expires or if the buffer is full and incoming packets are ignored, it is up to higher layers (like TCP) to notify the sender that the packet never was delivered at its destination, and we will not focus on that problem in this report.

6.3.4 A simple Example

The following is an example to briefly illustrate what happens at the FA when packets arrive and especially when a handoff occurs. When packets for MNs arrive at the FA that they are currently connected to, they are first de-capsulated and then saved at the end of the buffer (as mentioned above). The FA takes packets from the beginning of the buffer and sends them to the corresponding MN. Lets assume that 20 MNs (MN1, MN2, MN3, …. ,MN20) are connected to a foreign agent called FA1. Let's also assume that 5 of these 20 MNs are active at a certain time. The buffer could for example look something like Figure 6-3. In this figure, the packets on the right side are the ones that arrived first.

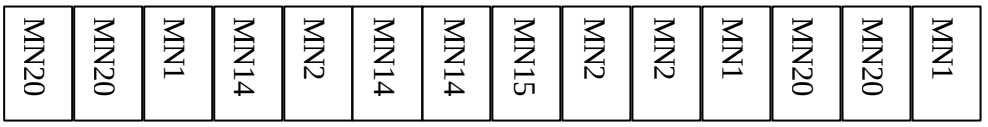

Figure 6-3 Example of a buffer at a FA.

Of course, packets for a certain MN can arrive out of order at the FA. For example, the second packet for MN1 (the fourth one from the right) might have been sent before the first one and needs to be delivered before that one. This reordering is however done at the mobile nodes and is outside the scope of this work.

At time zero when the buffer looks like Figure 6-3, the buffer is full and all incoming packets are ignored and not saved. FA1 takes the first packet (MN1) and sends it to mobile node 1. When the acknowledgement for MN1 is received, it is deleted from buffer1 and the next packet can be sent. Now there is an empty space in buffer1 and the first incoming packet could be saved at the end of buffer1. FA1 now takes MN20 and sends it. If an acknowledgement is not received, MN1 is resent after a certain time (this goes on until MN1's lifetime expires). The same procedure occurs over and over. The interesting thing happens when a handoff occurs and one of the MNs moves to another cell. For example, MN14 moves to another cell after it has received the first packet for it (the seventh packet from the right) but before the second packet. FA1, unaware of this move, sends packets for MN14 as normal. The only difference is that there are no acknowledgements for these packets. Instead, after a while, a binding update message is received from the new FA (called FA2) that MN14 now is connected to. FA1 sends an acknowledgement to this update message and then searches through its buffer for packets for MN14. Next, FA1 encapsulates these packets (meaning that FA1 puts another IP header in front of the datagram with the MN's new COA as the destination address and FA1's IP address as the source address) and sends them to FA2. When FA2 receives the packets, it de-capsulates them and sends them right away to MN14. This means that packets coming from other FAs (i.e. packets that are sent in reply to a binding update message) would have higher priority since they are not saved in the buffer where all the incoming packets are saved (referred to as buffer1), but instead sent right away before packets from buffer1.

This is of course only one solution. Another solution would be to put the packets at the end of FA2's buffer and deliver them to MN14 when the packets' turn comes. A problem then occurs if the buffer is full when the packets for MN14 arrive from FA1 whereby they are dropped. Also, if this solution is chosen, the handoff time will increase and that might disturb the quality and the

performance of data delivery during the handoff. Since we want to minimize the handoff time in our case, we choose the first solution in this work.

Yet another solution would be to buffer packets arriving from other FAs in a special buffer. This buffer (buffer3) should then be given higher priority than buffer1 in order to reduce the probability of losing handoff packets.

6.4 Assigning One Buffer to each MN

In this section we will look more carefully into the solution of having one buffer for every MN at a FA. Figure 6-5 shows what this scheme looks like.

MNs arrive and leave the cell of a certain FA all the time, which means that the number of MNs that are connected to a certain FA is not constant. This rises the question of how many buffers that should be at the FA. If the number of buffers is too small, arriving packets for some MNs are not buffered which is unacceptable. On the other hand, if there are too many buffers, there will be extra costs for maintaining these buffers that are empty and unnecessary resources are wasted. The easiest and probably best solution is to have as many buffers as the maximum number of MNs that the FA is able to serve simultaneously. In this way, all MNs attached to the FA are always guaranteed to be serviced.

Here are some of the things that we will discuss in more detail in this section:

- What to do if a buffer is full.
- When a new packet arrives, how does the FA know in which buffer to save the packet?
- An example on the course of events from when a packet arrives until it is sent to the corresponding MN and an acknowledgement is received.
- The buffering management scheme. This includes discussing questions like for example what to do with packets that have been sent and are waiting for an acknowledgement or what happens when an acknowledgement or a binding update message from another FA is received.

6.4.1 Buffering of Incoming Packets

The FA needs an index for every buffer to be able to know which buffer belongs to which MN and in which buffer an incoming packet should be saved. The home address of each MN is perfect as such an index. Also, we need to have a flag to indicate if the buffer is in use by a MN or if it is available. Whenever a buffer belongs to a MN and is in use the flag is set to one. When the MN moves to another FA, a binding update message is received and all the packets in the buffer belonging to the MN are sent to its new destination thus making the buffer empty. Then the flag is set to zero. This indicates that the buffer is empty and can be used by another MN in case one arrives and wants to register with FA1.

Figure 6-4 shows the index (the MN's home address and the flag bit) followed by the actual buffer for a random MN. Each packet shown in Figure 6-4 would look like Figure 6-2. .

Home Address of MNx	Flag	Packetx	Packet5	Packet4	Packet3	Packet2	Packet1

Figure 6-4 What an index and a buffer looks like at the FA.

When packets arrive at the FA, they are first decapsulated. The FA then does a lookup trying to match the incoming packet's mobile node home address with the buffer having that address as an index. When the right buffer is found, the packet is saved at the end of that buffer. We will discuss disadvantages and advantages with this scheme in section 6.7.

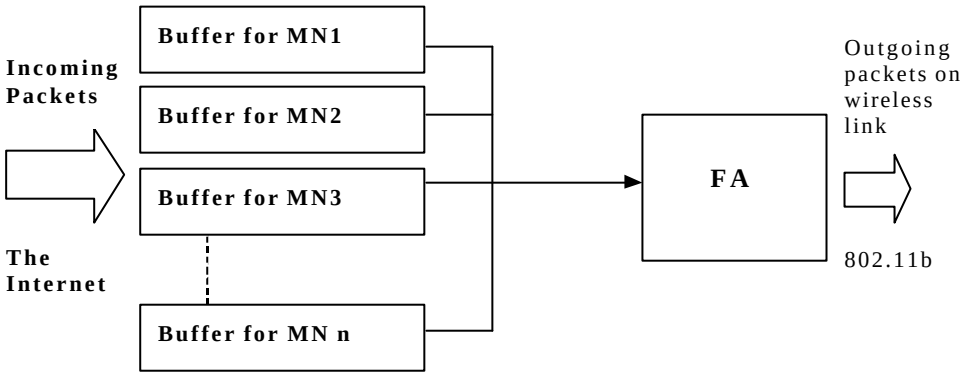

Figure 6-5 The Scheme of having one Buffer for every MN.

6.4.2 A simple Example

The following is an example to briefly illustrate what happens at the FA when packets arrive and especially when a handoff occurs.

As mentioned above, when packets for MNs arrive at the FA (called FA1 in this example) that the MN is currently attached to, they are first decapsulated. The FA then does a lookup trying to

match the incoming packet's mobile node home address with the buffer having that address as an index. When the right buffer is found, the packet is saved at the end of that buffer (this buffer is referred to as buffer1 in this example). Every MN is given a certain time slice. During that time slice, packets are taken one at a time from the beginning of buffer1 belonging to that MN and sent to the MN in question (we call this for MN1 in this example). When an acknowledgement for the sent packet is received, the next packet is sent. At time zero when the buffer looks like Figure 6-6, the buffer is full and all incoming packets are ignored and not saved.

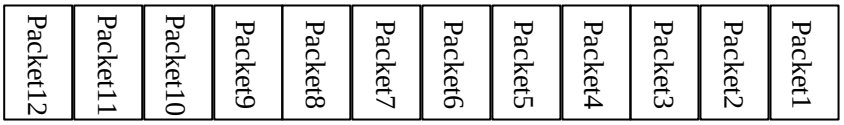

Figure 6-6 Example of what one of the buffers look like.

FA1 takes the first packet (packet1) and sends it to MN1 (and keeps a copy of it). When an acknowledgement for packet1 is received, it is deleted from buffer1. Now there is an empty space in buffer1 and the first incoming packet for MN1 could be saved at the end of that buffer. FA1 now takes the next packet (packet2) and sends it..

If an acknowledgement is not received for a sent packet, the packet is resent after a certain time. The same procedure occurs over and over until the time slice of MN1 is up. Then the FA moves on and serves the next MN, MN2.

The interesting thing happens when a handoff occurs and one of the MNs moves to another cell. For example, MN2 moves to another cell before FA1 starts serving it (or while FA1 is serving it). FA1, unaware of this move, sends packets for MN2 as normal. The only difference is that there are no acknowledgements for these packets. Instead, after a while, a binding update message is received from the new FA (called FA2) stating that MN2 now is connected to FA2. FA1 sends an acknowledgement to this update message and then takes all packets in buffer1 belonging to MN2, encapsulates them and sends them to FA2. When FA2 receives these packets, it de-capsulates them and sends them right away to MN2. After a certain time (see section 6.8), FA1 sets the flag to zero, which indicates that the buffer is empty and can be used by another MN in case one arrives and wants to register with FA1. This procedure goes on and on all the time.

There are other events besides incoming acknowledgements and binding update messages that interrupt the FA from taking packets for the MN that is being served and send those packets to the MN. Below are some of these events that cause an interrupt:

- When a new MN wants to register with the FA.
- When packets are received from another FA in response to a sent binding update message.
- When a packet is sent and no acknowledgement for that packet is received after a certain time. In that case the FA has to interrupt whatever it is doing and re-send the packet.

- When a new packet arrives. The FA has to interrupt whatever it is doing, find the right buffer and save the packet in that buffer.

- When the lifetime of a packet expires and the FA has to delete it.

6.5 Issues When the Buffers are Full

In this section we discuss issues about what to do when the buffer/buffers is/are full. This is a very important question since the loss of packets and the handoff time for a certain buffer size depends on the chosen algorithm here.

The arguments here are valid for both the case when we have only one buffer and the case when there is one buffer for every MN.

As we mentioned earlier, there are several solutions for our problem of what to do when the buffers are full. The easiest solution to implement is to ignore incoming data and not save them in the buffer when the buffer becomes full.

A consequence of this chosen algorithm is that the buffer size does not have an impact on the handoff time in our case: if the buffer is pretty small and becomes full quickly, the incoming packets are just dropped. Since we are dealing with UDP datagrams here (voice traffic) where there is no reliability and no guarantee that the datagrams will make it to their destination, no special action is taken. Thus, a small buffer size will not have an impact on the handoff time. However, it will have an impact on the quality of the voice traffic.

In the case of only one buffer, another consequence of this solution is that it could happen - if the lifetime for the packets is very big - that a certain MN receives a lot of packets at a certain time and occupies the whole buffer. This would then stop the other MNs from receiving data (since incoming packets for the other MNs will be discarded since the buffer is full). In order to prevent this from happening and to divide the space in the buffer somehow equally between the MN's attached to the FA (so that packets for one single MN don't take up all the space in the buffer and starve the other MNs), the lifetime of the saved packets should be chosen with care and should not be too big. Thereby, the packets would be erased after their lifetime has expired, and other data (from for example other MNs) could be saved in the buffer.

6.6 Buffer Size Analysis

In this section we will discuss issues about the buffer size. We will look into the parameters that have an impact on how fast the buffer becomes full, thus affecting how big the buffer size should be. The discussions are made for the case with only one buffer but are also valid for our other case where a buffer is assigned to each MN.

There are two factors that determine how fast a buffer of a certain size becomes full. These factors are called the **arrival rate** ? and the **service rate** µ.

The arrival rate is defined as the number of packets arriving at the FA per time unit. The service rate is the number of packets served by the FA per time unit.

If $\mu > ?$; i.e. the packets are served at a faster rate than they arrive, the buffer would never get full and there would not be a need for a buffer if the packets were arriving at a constant rate. This is however not the case most of the time. Usually the incoming data does not arrive at a constant rate: there could be heavy traffic at certain times and then the traffic could be very slow. Thus, even if $\mu > ?$ at certain times, we still need a buffer.

Figure 6-6 shows the connection between the arrival- and service rate and the buffering scheme at the FA.

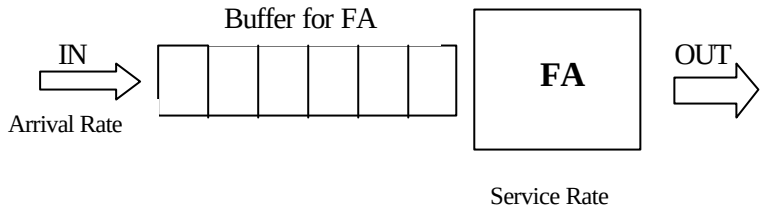

Figure 6-7 The connection between the arrival- and service rate and the buffering scheme at the FA .

One parameter that also affects how fast the buffer becomes full but has nothing to do with neither the arrival rate nor the service rate is the lifetime of the packets. Obviously, if the lifetime is longer, unsent packets will remain longer in the buffer and the buffer becomes full faster.

The following two sections discuss the parameters that affect the arrival rate and the service rate in more detail.

6.6.1 Parameters that Affect the Arrival Rate

The following are the parameters that determine the arrival rate:

- The number of MNs that are connected to the FA.
- The number of the MNs connected to the FA that are active at the same time in average.
- How frequently packets are sent for the MNs. Of course, the rate by which packets are sent for a MN is not constant. At certain times, the traffic is very heavy and sometimes non existent.

6.6.2 Parameters that Affect the Service Rate

Below are the parameters that determine the service rate:

- How many clock cycles it takes for the FA to take one packet from the buffer and put it on one of its wireless channels and send the packet to its destination.
- The size of the packet that is being sent (since the bigger a packet is, the longer it takes to send it).
- Other events that consume the processor's time/resources interrupting it from taking packets from the buffer and sending them. Here are some of these events:
  - When an Acknowledgement for a sent packet is received from a MN. The FA then has to search its buffer and delete that packet from it.
  - When a binding update message is received from another FA The FA then has to search its buffers for packets belonging to the MN that triggered the binding update message and send those packets to the new FA.
  - When a new MN wants to register with the FA.
  - When packets are received from another FA in response to a sent binding update message.
  - When a packet is sent and no acknowledgement for that packet is received after a certain time. In that case the FA has to interrupt whatever it is doing and re-send the packet.
  - When a new packet arrives. The FA then has to interrupt whatever it is doing and save the packet in a buffer. This is even more time consuming in the case when we have a buffer for every MN where the FA has to find the right buffer to save the incoming buffer in.
  - When the lifetime of a packet expires and the FA has to delete it.

6.7 Comparison of the two Solutions

In this section we discuss advantages and disadvantages with our two proposed solutions.

As we mentioned in section 6.4.1, one disadvantage with the solution of having one buffer for every MN is the extra time that is consumed to find the right buffer to save the incoming packets in. This extra time is however very small and can be ignored. The main advantage of this scheme is that when an acknowledgement or a binding update message from another FA is received, it will be much faster to find the right packet/packets. This is because we don't have to search through the whole buffer like we do in our other case. For example, when a binding update message is received, the FA finds out which MN the message belongs to and takes all packets in the buffer belonging to that MN and sends them to the new FA. In the case of only having one buffer, we would have to search through the whole buffer for packets belonging to the MN. If the size of the buffer is rather big, the time it takes to search the buffer is noticeable and would among others deteriorate the handoff time.

In the case of having one buffer for every MN, if separate memory space would be used for the buffer of every MN, the buffer resources would not be used efficiently. Here is an example to illustrate why: if we can have a maximum of 10 MNs connected to a FA and the total number of packets that can be buffered at the FA is 100, each buffer would have place for 10 packets. Let's assume that 5 MNs are active at a certain time. This means that 5 of the buffers (with place for a total of 50 packets) are empty and their buffer space is not used. If the traffic to one of the active

nodes would be heavy and the number of arriving packets would be 20 more than the number of packets served (which means that we would need to have a buffer size of 20 to be able to save all the incoming packets), 10 of the incoming packets would be lost. This happens even though the FA has the resources to save those packets (remember that 5 buffers are empty and their buffer space is not used). In order to avoid this, the buffers in a FA should use a mutual memory space. This would mean that in practical one buffer could have all the memory.

The conclusion of this section is that having one buffer for every MN is faster and gives us a better handoff time but it does not use the buffer spaces efficiently if not a mutual memory space is used for the buffers belonging to the FA. We believe however that the best solution is to only have one buffer at each FA. The arguments for this conclusion is mentioned in chapter 7.3 where we have calculated the number of lost packets during a handoff if no buffer existed.

6.8 Possible Problems and Suggested Solutions

In our test bed, there is still a scenario where there is a chance that packets will be lost during a handoff. The scenario is like the following: after a handoff occurs, some packets may be sent from the HA to the old FA *before* the new registration request is received by the HA. However, these packets might arrive at the old FA *after* the old FA has received a binding update message and already sent buffered packets to the new FA. This is pretty likely since the distance between the new FA and old FA is usually much smaller than the distance between the new FA and the HA. As a consequence, these packets will not be sent to the new FA and are lost (deleted when their lifetime expires).

To avoid this, for a certain time *t* after the binding update message is received and buffered packets for a certain MN (MNx) are sent to the new FA, the FA should forward new incoming packets for MNx to its new FA as soon as they arrive. The time *t* should be chosen so that the registration request with certainty has reached the HA (so that the HA is aware of MNx's new location and incoming packets for MNx are sent to its new FA, thus insuring that the above explained scenario won't happen). *t* should be in the range of about 500ms (or less).

7 Performance Analysis

In this section we will analyze the performance of the Mobile IP handoffs. We are going to calculate the performance of our Smooth Handoff scheme for some Real time audio services (voice traffic in our case). We would like to measure the packet inter-arrival time at the MN (which is the time it takes for subsequent packets to arrive at the MN), especially when a handoff occurs. The analysis is valid for both our suggested buffer management schemes (see chapter 6).

Our test bed looks like Figure 7-1. UDP packets are sent from a correspondent node (CN) over a wired network to the MN via the HA and the FAs (FA1 and then FA2 after the handoff).

Our wireless link is wireless LAN 802.11b with an average bandwidth of 5 Mbps [21] [22] [23]. For our audio source, we choose in our scenario the pulse code modulation (PCM) format as the Internet telephony audio coding format. With this coding format, the packet size of the sent data is 200 bytes and the packets are sent every 20ms from their source [18]. The beacon period (i.e. how frequent agent advertisements are sent) is chosen to be 10 ms. (see section 7.3 below). We will in section 7.3 discuss the beacon period and its significance for the handoff time and the buffer size. The distance between two neighboring FAs is approximately 50m (WLAN 802.11b). The MN has a maximum speed of about 30 m/s (this is for a person in for example a car).

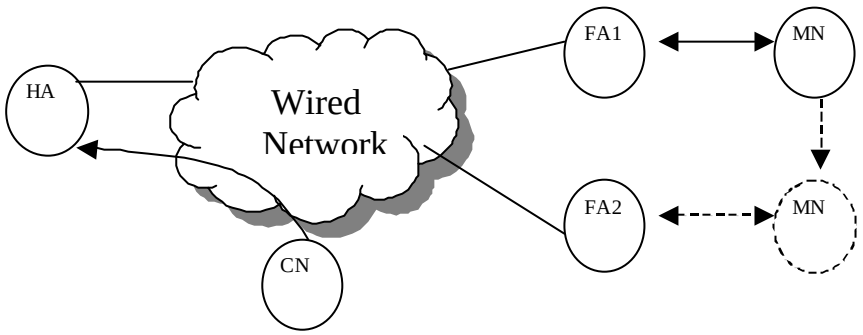

Figure 7-1 The handoff scenario.

As mentioned in section 4.5, in our handoff scenario we are assuming that there are no "dead zones" between two adjacent FAs, meaning that the MN is *always* within the cell of *at least* one FA. There are two cases here that we have to consider:

Case 1. Handoff time in the case where there are cell overlaps.

Case 2. Handoff time in the case where there is no overlap between adjacent cells, i.e. one cell starts where the other ends (as shown in Figure 7-5).

Figure 7-2 shows the case where there are cell overlaps (the dotted circles in Figure 7-2 show the cells covered by each FA). As it is evident from Figure 7-2, this overlap is considered to be a maximum of 10m. Having cell overlaps means that the MN sometimes could receive beacons from several FAs. We discuss in section 7.1 different solutions for which FA to use in such a

case. We will calculate and compare the packet inter-arrival time both for the scenario with overlapping cells and the case where there are no cell overlaps in section 7.2.

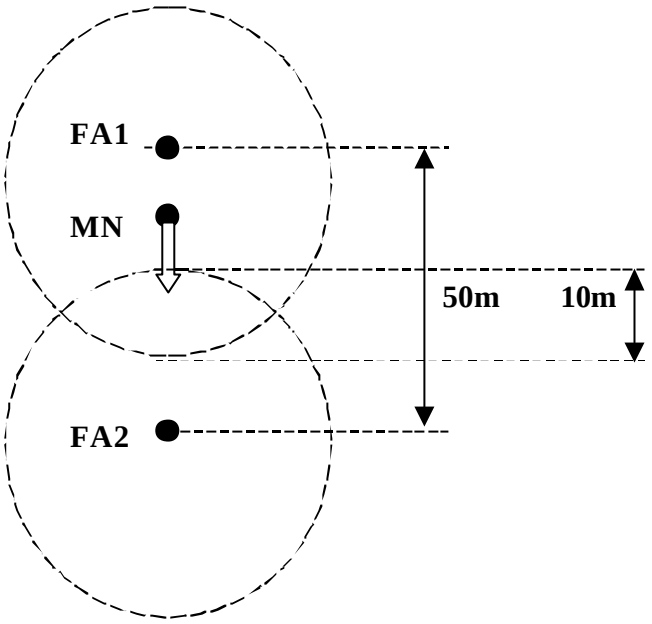

Figure 7-2 Overlap between adjacent cells.

Figure 7-3 shows what happens during a handoff when data is being sent from the CN to the MN in the case of no cell overlaps. Figure 7-4 shows the same procedure but in the case of cells overlapping.

From the beginning, the MN is attached to FA1 and packets for the MN are sent to it through its HA and FA1. Packets are sent from the CN every 20ms. They will arrive at FA1 every 20ms ± ?t (?t is the jitter in our network). We will discuss this jitter a little bit later.

At time t1, the MN moves within the cell of a new FA (FA2). If there is no overlap between adjacent cells (see Figure 7-5), the MN has moved outside the cell of its previous FA (FA1) and is thus not able to receive packets from that FA. If there *are* cell overlaps as shown in Figure 7-2, the MN can still receive packets from its old FA (possibly even after it has sent a registration request). After a certain time t3-t1 (which is the FA detection time; i.e. the time it takes for the MN to receive an agent advertisement from the new FA after the MN has entered the cell of the new FA), the MN sends a registration request to the new FA, FA2. In the case with no cell overlaps, the maximum FA detection time, $t_{detect-max}$ or

$(t3-t1)_{max}$ is the beacon period (which in our case is 10 ms as mentioned above). This means that

$0 < (t3-t1) < 10$ ms.

However, in the case when we have overlaps between adjacent cells, the FA detection time is equal to zero (with the speed we have in our case) since the MN is receiving signals/packets from the old FA when it receives an agent advertisement from the new FA and sends a registration request to it.

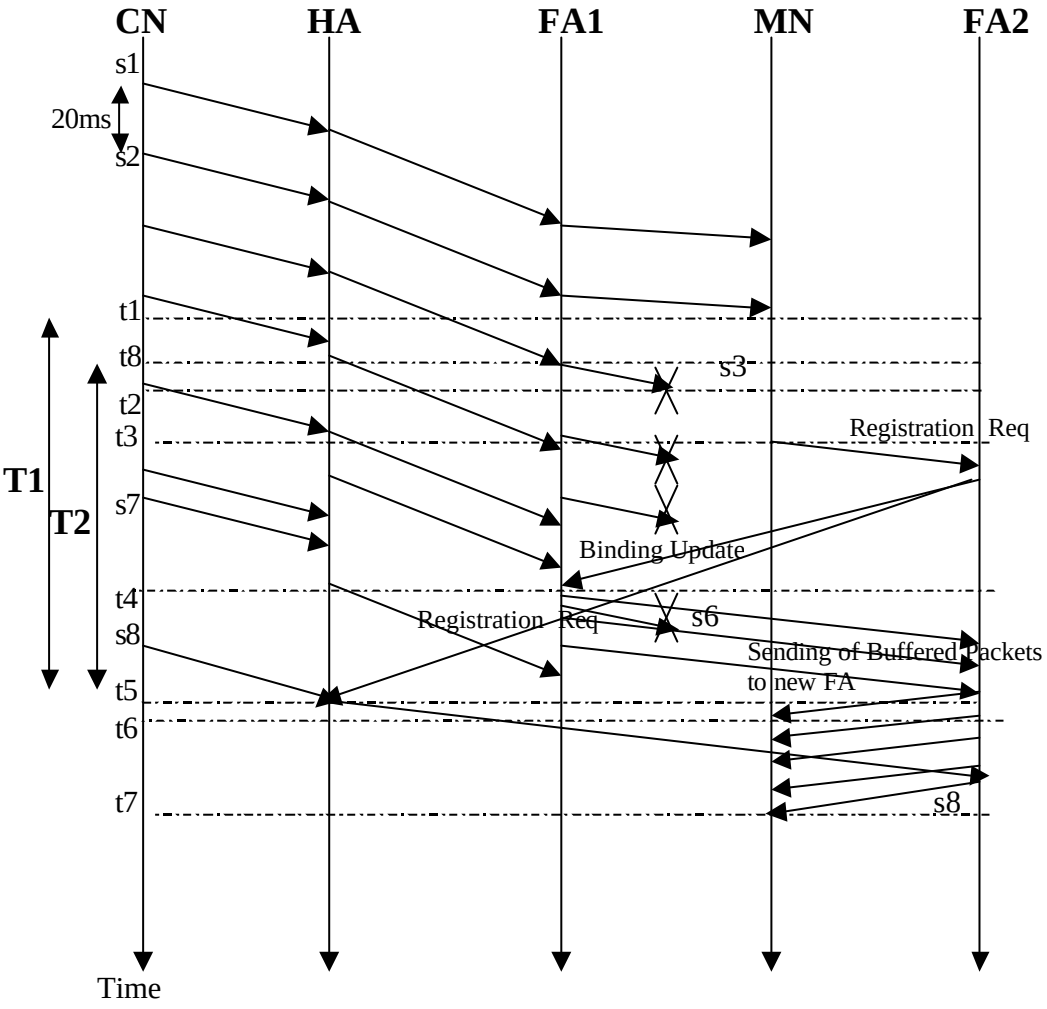

Figure 7-3 Time Sequence graph of the MIP Handoff Scenario in the case of no cell overlaps.

When FA2 receives the registration request, it forwards it to the MN's HA. The HA receives this registration request at t5 whereby all new incoming packets for the MN are sent to FA2 instead of FA1 (the first packet doing so is s8 and it arrives at the MN at t7).

FA2 also sends a binding update message to the MN's old FA (FA1), which FA1 receives at time t4. As can be seen from Figure 7-3, none of the packets that arrive at FA1 between t1 and t4 (packets s3 to s6) arrive at the MN. Instead, these packets are buffered at FA1. When the binding update message is received by FA1, FA1 will send all these buffered packets to FA2 whereby they are immediately delivered to the MN (the first one arrives at t6). As mentioned in section 6.8, for a certain time *t* after the binding update message is received and buffered packets for the

MN are sent to FA2, FA1 should forward new, incoming packets for the MN to FA2 as soon as they arrive. This would for example ensure that s7 is sent to FA2 and delivered to the MN.

As mentioned, Figure 7-4 shows what happens during a handoff when data is being sent from the CN to the MN in the case when we have cells overlapping. The only difference between this case and the case of no cell overlaps (as can be seen from the two figures) is that in this case, even after the MN enters a new cell at t1 (and possibly even after it sends a registration request at t3), the MN is still able to receive packets from FA1 (packet s3 and possibly even packets s4-s6 depending on the radio technology of the MN). For more details, see section 7.2.1). This was not the case when the cells didn't overlap.

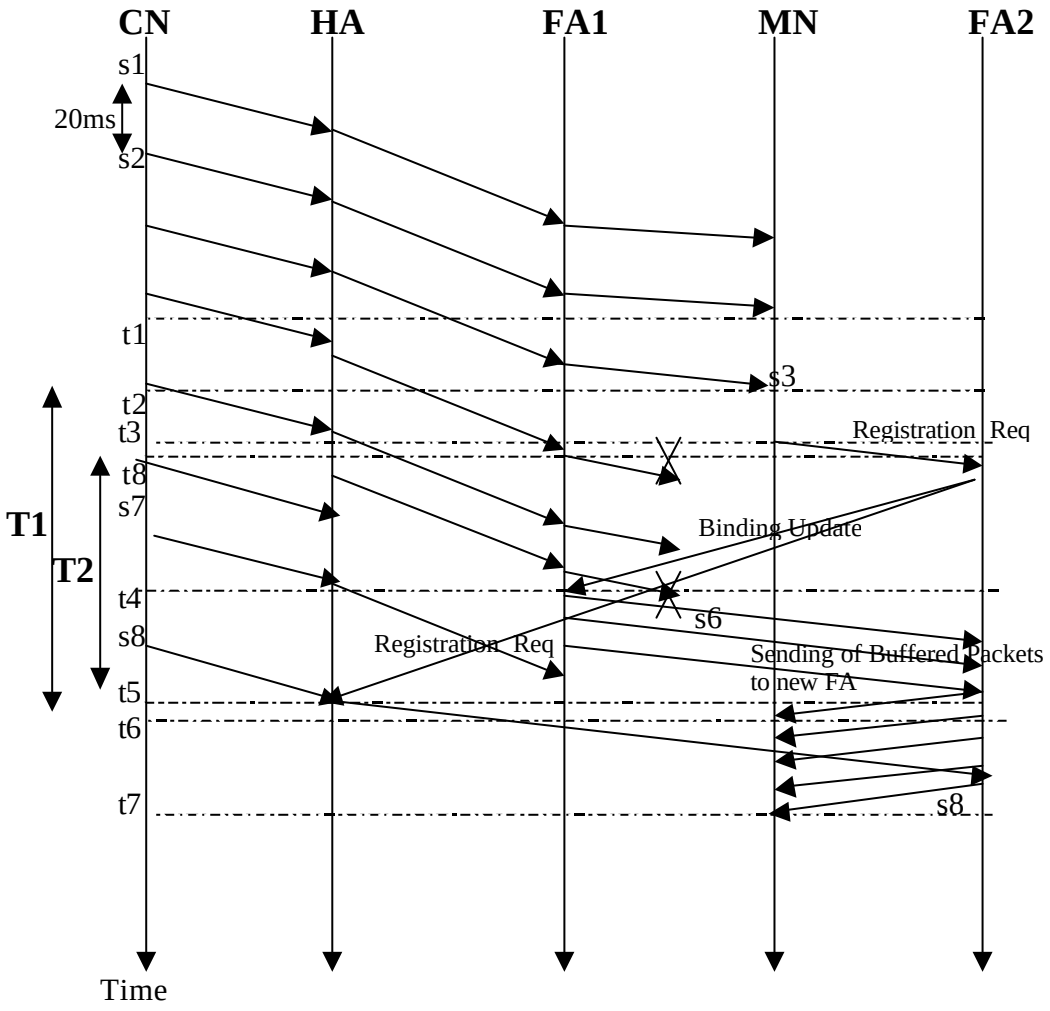

Figure 7-4 Time Sequence graph of the MIP Handoff Scenario when cells overlap.

The time from when the MN receives the last packet from its old FA (before a handoff occurs) until it receives the first packet from its new FA is referred to here as the handoff time and is shown as T1 in Figure 7-3 and Figure 7-4. This is the packet inter-arrival time at the MN during

a handoff and it is very interesting to know what this time is. Various human factor studies have shown that the maximum tolerable delay by the human ear for an interactive conversation is approximately 200ms[18],[25]. This delay is also sometimes referred to as the "mouth-to-ear" delay and is defined as "the time between the moment the sending party has spoken a word and the moment the receiving party has heard the word". If the mouth-to-ear delay is larger than 200 ms, it starts affecting the quality of the interactive conversation noticeably.

The mouth-to-ear delay can be thought of as the combination of the delay at the sending codec, the delay in the wired network plus the delay of sending the packet in the wireless network and the delay at the receiving part's codec. If we choose our wired network to be as shown in figure 7-7 where we have 4 hops between the HA and FA1/FA2, we have the delay in the wired network from the HA to FA1/FA2 and vice versa to be 20 ms (see section 7.3.1for the details).

If we assume the delay at each codec to be 20 ms (see [18]), that would give us that the time from when a packet arrives at a FA until the time when the packet is delivered at the MN should be less than 200 – (20 + 20 + 20) = 140 ms. This time is shown as T2 in figures 7-3 and 7-4 and we will refer to this time as T2 for the rest of this section.

We have concluded that in order to not deteriorate the quality of the voice conversation, T2 has to be less than 140 ms. As can be seen from figures 7-3 and 7-4, T2 is smaller than T1 (which is the handoff time). If we have T1 less or equal to 140 ms, that would guarantee that T2 is less than 140 ms and we would then have reached the QoS limit. We will discuss and calculate the handoff time in section 7.2.

7.1 Movement Detection Algorithms

To decrease the FA detection time and thereby decrease the handoff time, different movement detection algorithms can be used. There are three known movement detection algorithms that we are considering here. For more details about these algorithms, see [2][13][19]. These three methods are explained in the following subsections.

7.1.1 Lazy Cell Switching (LCS)

The LCS method has the main characteristic that after the MN has registered a COA with its HA, it holds on to it - even if the MN is receiving beacons from other FAs - until it "leaves" the cell of the FA offering the COA (by leaving we mean that the MN no longer is able to receive any signals such as agent advertisements or data packets from that FA). The MN decides that it is outside the cell of the FA serving it, when it misses three consecutive agent advertisements from that FA. The MN could for example have a timer and know what the beacon period is, and in that way be able to calculate if it has missed three consecutive agent advertisements. When this happens, the MN starts looking for another FA offering it a new COA by for example sending agent solicitations and listening for agent advertisements.

In other words, assuming that MN movement is rare, this algorithm chooses not to handoff and ignores any newly discovered agents.

7.1.2 Pattern Matching (PM)

In this method the MN compares the subnet prefixes of mobility agents that it is receiving advertisements from to determine new agents and eliminate agents within the same subnet.

Normally, agent advertisements do not contain information about the sending agent's subnet prefix number. Therefore, in order for the MN to use this method, all agents are required to include a prefix-length extension in their agent advertisements. This extension includes the prefix length of the agent's address. If this method indicates that the MN has moved (when a new agent is discovered), the MN will register with the newly discovered agent after the lifetime of its current binding has expired.

This method is useful in multiple agent sub networks where there are several FAs in the same subnet. In the case of single agent sub networks (only one FA in a subnet), which we are dealing with in this report, the PM method's advantages are cancelled and it operates more or less like the LCS method.

7.1.3 Eager Cell Switching (ECS)

The ECS method works in the opposite way of the LCS method. It assumes that mobile nodes change their direction of movement very slowly. That is, if a MN is moving forward in one direction, it is unlikely that it will stop or change its direction. Thus, MNs should handoff immediately upon discovering a new FA. Compared to the LCS method where the MN waits until it has missed three consecutive agent advertisements before it starts looking for a new FA, this method reduces the movement detection time and thereby manages faster Mobile IP handoffs.

7.1.4 Movement Detection based on Signal Strength

Another method for the MN to detect movement is by measuring the strength of the signals it is receiving. If the signals it is receiving from its current FA are getting weaker, the MN should assume that it is moving away from that FA and start looking for another FA (by for example sending agent solicitations). If the MN is receiving multiple beacons, it can decide which one to choose by comparing the strength of the signals.

This method is the best solution and the one that seems most logical. The problem with implementing this method is then how to measure and compare the strength of different signals. However, since it is difficult to use this method in mathematical calculations, I will use the ECS method in the following calculations (since it is – in our scenario - the best one out of the three first mentioned methods).

7.2 Handoff time

As mentioned above, the handoff time is the time from when the MN receives the last packet from its old FA (before a handoff occurs) until it receives the first packet from its new FA. In the

following calculations, I am assuming that when the MN attempts a handoff and sends a registration request, the request is always granted by the new FA. This is however not the case in reality where it could happen that the new FA denies a registration request for different reasons (e.g. not enough resources or authentication denied; see section 5.3.2 for a list of all possible reasons).

As mentioned in the beginning of section 7, there are two different cases that we have to consider:

Case1. Handoff time in the case where there are cell overlaps.

Case 2. Handoff time in the case where there is no overlap between adjacent cells.

We will now calculate the handoff time in both cases in the following subchapters.

7.2.1 Handoff time in the case of cell overlaps

The handoff time in the case of overlapping cells depends on whether the MN is able to receive packets from its old FA after it has sent a registration request or not. This depends on the radio technology of the MN, which decides if the MN is able to tune in more than one channel at a time. In most cases, the MN cannot do this. Also, the latter case should give us a bigger handoff time thus making it more interesting. Because of these mentioned reasons, we will calculate the handoff time assuming that the MN cannot receive packets from its old FA once it tries to register with a new FA and sends a registration request to it.

In that case, the handoff time can in our scenario be thought of as the following:

Handoff time = Handoff initiation time + registration time + binding time +

buffer lookup time + delivery time + processing time (1)

In (1), the **handoff initiation time** is the time from when the MN receives its last packet from the old FA until it tries to register with a new FA and sends a registration request. The **registration time** is the time it takes to send a registration request from the MN to the new FA. The **binding time** is the time it takes to send the binding update message from the new FA to the old FA. The **buffer lookup time** is the lookup time at the old FA's buffer. The **delivery time** is the time it takes to deliver the packets from the old FA to the new FA to the MN and the **processing time** is the time it takes to process the different messages at the two FAs involved.

As mentioned above, the handoff initiation time is the time from when the MN receives its last packet until it decides to perform a handoff and sends a registration request. This time obviously depends on the movement detection algorithm that has been chosen (see section 7.1). If we choose to perform a handoff as soon as another FA is discovered (Eager Cell Switching, section 7.1.3), the handoff initiation time could in the worst case be considered to be just below the time it takes for the MN to receive two consecutive packets (i.e. the MN receives a packet and just before it is about to receive the next packet, it receives a beacon from a new FA whereby the MN

initiates a handoff and sends a registration request to this new FA). This time is equal to the delay between two consecutive packets sent from the source (which is 20ms in our case) plus the jitter in the wired network before they arrive at the MN (referred to as ?t in the beginning of section 7). If we assume this jitter to be 5ms in a worst-case scenario, the maximum handoff initiation time would be 25ms.

In order to calculate the registration time we need to know the size of the registration request message. The size of the registration request messages is difficult to calculate because they depend on if the messages have any extensions and how big they are. Without the extensions, the registration request is 5*32 (see figure 5-3) =160 bits. Then we have to add the IP header (20 bytes = 20 * 8 = 160 bits) and the UDP header (8 bytes = 64 bits). This gives us a total of 384 bits. So it is pretty realistic to assume 400 bits for this message. Same thing for the binding update message. If the wireless bandwidth is 5 Mbps and we assume that the registration requests and the binding update messages are 400 bits, the registration time and the binding times are $400/5*10^6$ = 80 μs each. The distances do not matter here since they are very small (less than 50m) and the speed by which the messages are sent are very fast (close to the speed of light). According to [6], the overhead costs for registration, encapsulation and decapsulation are 1.8 ms, 270 μs and 160 μs respectively. We assume another 200 μs for the processing of the binding update message at the old FA.

The buffer lookup time obviously depends on how many instructions the FAs can perform per time unit. It also depends on what kind of buffer management scheme we are having. If we choose the case of having only one buffer - which is the slower of our two proposed solutions -, we can calculate the buffer lookup time like the following:

We assume the clock frequency of the FAs to be 1GHz, the lookup time to take 10 clock cycles and the buffer to contain a maximum of 100 packets (we calculate in chapter 7.3 that there would be 3 packets in the buffer for every active MN. If we have around 35 active MNs which is a reasonable assumption, that would give us a buffer size of around 100 packets).

The lookup time for one buffer would then be

$T_{lookup} = 100*10/(1*10^9) = 1$ μs.

The delivery time can be calculated the same way that the registration and the binding times were calculated. According to our calculations in section 7.3.1, in a worst-case scenario, 3packets are found in the buffer for the MN every time. With our packet size equal to 200 bytes = 200*8 =1600 bits, the delivery time is

$T_{delivery} = (3*200*8)/ 5*10^6 = 4800/5*10^6 = 0.96$ ms.

This will give us a total handoff time equal to

Handoff time = 25 ms + 80 μs + 1.8 ms + 80 μs + 200 μs + 1 μs + 270 μs + 0.96 ms + 160 μs = 28.551ms

In a worst-case scenario, we could imagine that the last packet before the handoff (s2 in Figure 7-4) was lost and never received by the MN. This would add another 25 ms to the handoff time and would give us a total handoff time of 53.552 ms. This value is by far less than 140 ms and this shows that our scheme can be used for voice traffic applications in the case of overlapping cells. Now we have to find if it also is good enough in the case of no cell overlaps.

7.2.2 Handoff time in the case with no overlap

Figure 7-5 shows the case where there is no overlap between adjacent cells. The handoff time can even here be considered to be as in (1):

Handoff time = Handoff initiation time + registration time + binding time + buffer lookup time + delivery time + processing time

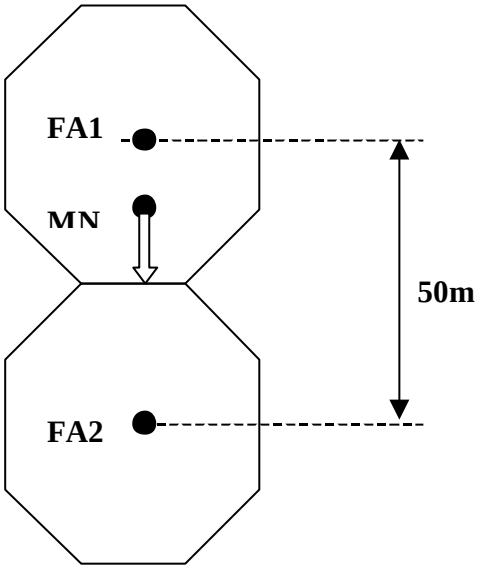

Figure 7-5 No overlap between adjacent cells.

Of these values, the only one that differs from the case with cell overlaps is the handoff initiation time. The handoff initiation time is as mentioned earlier the time from when the MN receives its last packet until it decides to perform a handoff and sends a registration request. The maximum time it takes the MN to receive an agent advertisement and send a registration request after it enters the new cell is the beacon period. This would be if the MN enters the new cell just after a beacon was sent and has to wait a full beacon period until the next beacon signal is sent. This time is 10 ms.

In addition to that, we can imagine in a worst-case scenario that the MN enters the new cell just before it was about to receive a packet in its old cell (i.e. the MN receives a packet and just before it is about to receive the next packet, it enters the new cell and is unable to receive packets from its old FA). The time that passes from when the MN receives its last packet until it enters the new cell is then equal to the delay between two consecutive packets sent from the source (20 ms in our case) plus the delay in the wired network before they arrive at the MN (which we assumed to be 5 ms in chapter 7.2.1). This gives us a total of 25 ms and a total handoff initiation time of 35 ms.

Also, in a worst-case scenario, we could imagine that the last packet before the handoff (s3 in Figure 7-4) was lost and never received by the MN. This would add another 25 ms to the handoff time.

With the other numbers just like in chapter 7.2.1, we would get a total handoff time of:

Handoff time = 35 ms + 25 ms + 80 µs + 1.8 ms + 80 µs + 200 µs + 1 µs + 270 µs + 0.96 ms + 160 µs = 63.551 ms

Although this time is (as expected) bigger than the one in our first case, it is still less than 140 ms which was the upper limit. This shows that our handoff scheme can under our assumptions be used for real time services without any problems even in the case where the cells do not overlap.

7.3 Loss of Packets during Handoffs

In this section, we are going to do an analysis on the number of packets that would be lost during a handoff if we had no buffer at the FAs. The result here is very interesting when deciding on the buffer size and which buffer management solution (see chapter 6) to choose.

7.3.1 The Beacon Period

A factor that affects loss of packets during handoffs very much is the beacon period from the FA (i.e. how often the FAs send agent advertisements). This is evident from the calculations we made in chapter 7.2 for the handoff time. Because of this, to start with, we need to find a reasonable value for the beacon period.

The best approach for achieving the beacon period is to decide that the bandwidth the beacon period consumes shouldn't be more than a certain amount of the total bandwidth (which in our case is 5 Mbps; see beginning of chapter 7).

The bandwidth of the beacon period can be calculated by using the formula

Bandwidth = size/time (2)

Where size is the size of the agent advertisements plus the headers included in the packets (see Figures 5-1 and 5-2) and time is the beacon period.

As can be seen from Figure 5-1, the size of the agent advertisements depends on the number of Care-of-Addresses the FA is advertising. If we assume that the FAs in average advertise 5 COAs it would give us a total size of

Size = 8 + (5*4) + 8 + 20 = 56 bytes = 448 bits, where the two last values are for the UDP and the IP header respectively. Figure 7-6 shows the bandwidth used for this size of the FA advertisement as a function of the time between advertisements.

If we here decide that the bandwidth the beacon period consumes should be one percent of the total bandwidth of 5Mbps, that would give us that the bandwidth of the beacon period should be

$0.01 * 5*10^6$ bps = $5*10^4$ bps. The beacon period for this bandwidth is according to (2) equal to 448/50000 = 9 ms. This can also be seen in Figure 7-6.

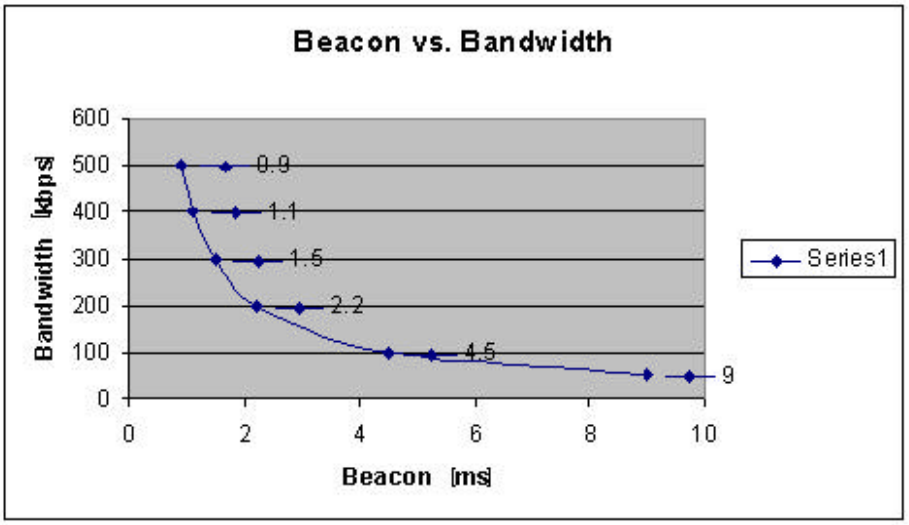

Figure 7-6 The bandwidth as a function of the beacon period

We will now calculate the number of lost packets for this bandwidth and for the same scenario mentioned in the beginning of chapter 7 and shown in Figure 7-7. As shown in figure 7-7, we assume that there are 4 hops in the wired network between the HA and FA1 /FA2. We also assume that the propagation delay between each hop is 5 ms. and that there is no jitter in the network. These assumptions, together with what we mentioned earlier that the packets are sent from the HA every 20 ms gives us that the packets arrive at FA1 every 20 ms and that the propagation delay in the network from the HA to FA1 and from FA2 to the HA is 20 ms (this is a mean value just like all the other values I have used in this section).

With the help of these facts, we will now calculate the worst case time from the point when the MN receives its last packet from FA1 before a handoff until the time when the HA finds out about the handoff and starts sending packets to the MN's new destination (FA2). This time is referred to as T in the calculations below. T divided by the value for how often packets are sent from the HA (20 ms. in our case) will give us the total number of lost packets during a handoff.

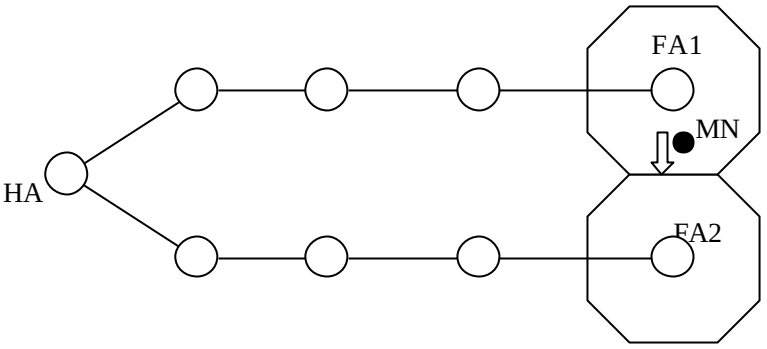

Figure 7-7 Our wired network consists of 4 hops between the HA and FA1 and FA2.

$T = t_{FA1\text{-}MN} + t_{beacon} + t_{MN\text{-}FA2} + t_{FA2\text{-}HA}$;

Here, $t_{FA1\text{-}MN}$ is the time from when the MN receives the last packet from FA1 until it leaves FA1's cell. This value is in the worst case scenario 20 ms. in our case. t_{beacon} is the time from when the MN enters the cell of FA2 until it receives a agent advertisement from FA2. This time is in the worst case scenario the beacon period which we calculated to be 9 ms. above. $t_{MN\text{-}FA2}$ is the time from when the MN sends a registration request until it is received by FA2. $t_{FA2\text{-}HA}$ is the time from when FA2 forwards the registration request until it is received by the HA. We calculated this time above to be 20 ms in our scenario.

Considering that the cell of each FA is very small in our case (about 50 m; see beginning of chapter 7), $t_{MN\text{-}FA2}$ will be much smaller than the other values and can be disregarded.

All this will give us that

T = 20 ms + 9 ms + 20 ms = 49 ms

The total number of lost packets during a handoff will then be 49 / 20 = 2,45 = 3 packets

The above conducted calculations imply that in average, only 3 packets need to be buffered when a MN performs a handoff. The conclusion of this is that out of our two compared solutions for buffering that we discussed in chapter 6, the one with having one buffer for every MN is unnecessary since there are only a few packets (3 in our scenario) that will be buffered. We suggest therefore to choose the other solution and have only one buffer for all MNs at each FA. Another argument for choosing this solution is that the MNs usually move around a lot (i.e. handoffs occur pretty regularly). This would slow and reduce the performance of the solution of having one buffer for each MN even more since it takes a certain amount of time to reserve and "give" a buffer to the new MN each time a new MN tries to register with a FA.

8 Conclusions

In this report, we have been concentrating on handoffs in Mobile IP. The main objective of this thesis has been to come up with a solution to improve handoffs by trying to eliminate loss of packets when a handoff happens and to decrease the handoff time (the time it takes for a MN to perform a handoff).

Our solution is to have buffers at the FAs and save incoming packets in those buffers. When a MN moves to a new FA, the new FA notifies the old FA, which then sends all packets in the buffer for the MN to the new FA.

We have discussed and compared different buffer management schemes for our smooth handoff scenario. We have made a buffer size analysis and talked about possible problems with these solutions.

We calculated the inter-arrival time at the MN during a handoff for voice traffic (UDP packets) in the cases when there is an overlap between the cells of adjacent FAs and when there is no overlap. In both cases, the result was satisfactory since it was less than the limit where the quality of a real time application (like voice) becomes unacceptable. This analysis shows that, with our smooth handover scenario under our assumptions, we can satisfy the demands for QoS for voice traffic.

9 Future Work

It would be interesting to do some simulations to see more detailed and accurate results about the handoff time for our suggested solution. It would also be interesting to find out the analysis (or simulation) results in the case when we are sending TCP packets instead of UDP in our scenario. Also, one could do some simulations for our different proposed buffer management schemes and investigate and compare the solutions in more detail.

Finally, it would be very interesting to implement Mobile IP and on top of that our suggested handoff solution to see how it works, but this by itself could be more than enough as a Masters thesis and could be very time consuming.

10 References

[1] W. Stevens, “TCP Slow Start, Congestions Avoidance, fast Retransmit, and Fast Recovery Algorithms”, Network Working Group, RFC 2001, January 1997

[2] C. Perkins, “Mobile IP, Design Principles and Practices”, Addison Wesley, (1998)

[3] C. Perkins, “IP Mobility Support”, RFC 2002, October 1996

[4] S. Deering, Editor, “ICMP Router Discovery Messages”, RFC 1256, September 1991

[5] V. Jacobson, “Congestion Avoidance and Control”. In ACM SIGCOMM ’88. ACM, 1988

[6] C. Perkins and K.Y. Wang, “Optimized Smooth Handoffs in Mobile IP”, Computers and Communications, 4/99. pp. 340 –346

[7] H. Balakrishnan, S. Seshan, and R. H. Katz. “Improving Reliable Transport and Handoff Performance in Cellular Wireless Networks”. ACM Wireless Networks, December 1995

[8] C. Perkins, “Mobile IP Networking Through Mobile IP”, IEEE Internet Computing, January 1998

[9] R. Caceres and V. N. Padmanabhan, “Fast and Scalable Wireless Handoffs in Support of Mobile Internet Audio”, ACM MONET, vol 3, no 4, December 1998

[10] C. Perkins, “Mobile IP”, IEEE Communications Magazine, May 1997

[11] C. Perkins, “IP Encapsulation Within IP”, RFC 2003, May 1996

[12] C. Perkins and D.B Johnson, “Route Optimization in Mobile IP”, Mobile IP Working Group, Internet Draft-work in progress, November 1997

[13] N.A Fikouras, K. El Malki, S.R. Cvetkovic and M. Kraner, “Performance Analysis of Mobile IP Handoffs” Microwave Conference, 1999 Asia Pacific, 1999, pp: 770 -773 vol.3

[14] R.L. Rivest, “The MD5 Message-Digest Algorithm”, RFC 1321, April 1992

[15] B. Schneier, “Applied Cryptography”, New York: John Wiley and sons, 1993

[16] S. McCanne and S. Floyd. ns Network Simulator. http://www.isi.edu/nsnam/ns/

[17] A. Stephane, A. Mihajlovic and H. Aghvami, “Mechanisms and Hierarchical Topology for Fast Handover in Wireless IP Networks”, IEEE Communications Magazine, November 2000, pp: 112-115

[18] S. Pink, C.L. Tan, K.M. Lye, “A Fast Handoff Scheme for Wireless Networks” In the proceedings of the 2nd ACM International Workshop on Wireless Mobile Multimedia (WoWMoM’99), 20 August 1999, Seattle, Washington, pp 83-90.

[19] N.A Fikouras, K. El Malki, S.R. Cvetkovic and C. Smythe, “Performance of TCP and UDP during Mobile IP handoffs in Single-Agent Subnetworks”, Wireless Communications and Networking Conference, 1999. WCNC, 1999 IEEE, pp: 1258 -1262 vol.3

[20] L.L. Peterson, B.S. Davie, “Computer Networks, A System Approach”, second Edition, Morgan Kaufmann Publishers, (2000)

[21] J. Conover, “Wireless LANs work their magic”, Network Computing, July 10, 2000, http://www.networkcomputing.com/1113/1113f2.html, available March 2, 2002

[22] Proxim Skyline, “802.11b wireless broadband gateway”, http://computers.cnet.com/hardware/0-7052-407-7345908.html, available March 2, 2002

[23] L. Freed, PC Magazine, “Performance Tests: Wireless LANs”, http://www.zdnet.com/products/stories/reviews/0,4161,2470139,00.html, available March 2, 2002

[24] G. Q. Maguire Jr., J. Ioannidis, “The Design and Implementation of a Mobile Internetworking Architecture”, USENIX Winter, 1993

[25] P. Mehta, S Udani, “Voice over IP, sounding good on the Internet”, January 2001, IEEE, p 36-40

[26] C. N Chen, C. T. Lin, J. M. Zuan, , C. F. Liu, “The Study of Mobile Internet Telephony “, September 2000, IEEE p 179-183

[27] O. Hagsand, K. Hanson, I. Marsh, “Measuring Internet Telephony Quality: Where Are We Today?“, Global Telecommunications Conference – Globecom ’99, May 1999, IEEE

[28] Jon-Olov Vatn, "Improving Mobile IP handover performance", licentiate thesis, Royal Institute of Technology, Stockholm

11 Appendix A

Terminology

Here are some of the terms frequently used in this report:

Mobile Node (MN): A host or router that changes its point of attachment from one network or subnetwork to another without changing its IP address or interrupting existing communications. Except for special cases, the Mobile Node usually uses its home address as the source address of all the IP datagrams it sends

Home Agent (HA): A router on the MN's home network that forwards packets destined for the MN through encapsulation when the MN is away from its home network.

Foreign Agent (FA): A router on the MN's visited network that provides routing services to the MN while the MN is registered with it. The FA decapsulates and delivers packets that were sent by the MN's HA to the MN.

Home Address: A long-term IP address given to the mobile node on its home network. This is the IP address by which the mobile node is known to other hosts on the Internet. The home address remains fixed as the mobile node moves through the Internet.

Care-of address (COA): A temporary IP address given to the MN in the network it is at, at the moment. There are two different kinds of COA: foreign agent COA and co-located COA. A foreign agent COA is the IP address of the foreign agent with which the mobile node is registered; a co-located COA is an IP address temporarily assigned with the mobile node. In this work, we will deal with foreign agent COAs. Unless otherwise specifically mentioned, when we talk about COA in this report we mean foreign agent COA.

Correspondent Node (CN): Any other host on the Internet with which the mobile node is communicating. The Correspondent Node can be either mobile or stationary.

Foreign Network: Any network other than the MN's Home Network.

Home Network: A network that has a network prefix matching that of a MN's home address. Note that standard IP routing mechanisms will deliver datagrams destined to a MN's Home Address to the MN's Home Network.

Mobility Agent: Either a Home Agent or a Foreign Agent.

Mobility Binding: The association of a HA with a COA, including the remaining lifetime of that association.

Binding Update: A message sent by a FA to the MN's previous FA indicating the MN's current location and its new COA.

Agent Advertisement: Messages sent by Mobility Agents (Home Agents and Foreign Agents) to advertise their services on a link. Mobile Nodes use these advertisements to determine their current position. An Agent Advertisement is an ICMP Router Advertisement [4] that has been extended to also carry a Mobility Agent Advertisement Extension.

Agent Solicitation: Messages sent by the MN in order to find a Mobility Agent. The Agent Solicitation is an ICMP Router Solicitation message[4].