Mobility in IPv6

Graduate Thesis

Siv.ing. Degree
in
Information and Communication
Technology

By

Jørn Hunskaar and Trond Almar Lunde

Grimstad, June 2001

Abstract

In the future it is expectable that the number of terminals with wireless access to network resources will be more and more widespread, and it is therefore necessary to integrate mobility support into future generation networks so that users can be online, even while in motion. The increasing use of Internet suggests that the Internet technology can be the best candidate for effective realization of future generation mobile systems. Mobile IP can offer the possibility for use of the mobile Internet in other ways than it is used in the standard wired environment, and may be the solution to increasing mobility demands. Due to this, mobility in IPv6 (MIPv6) is designed to be scalable, stable, efficient and secure, which are the factors considered important for this thesis.

Scalable – The number of users are expected to be so many that MIPv6 is, according to its specification, designed to scale almost as well as Internet without mobility support integrated. This implies the elimination of triangle routing, currently a challenge in MIPv4, and also a reasonable amount of data that must be managed by the nodes involved in mobility. MIPv6 is also designed so that future extensions and modifications are possible by allowing further growth.

Stable – For the adoption rate of this technology to high, the users must be able to depend on the services provided. At present the implementations shows that there are still a few more steps to take until necessary stability is offered, but product quality releases of MIPv6 is expected to be released sometime next year. The implementation tested in this thesis reflects transparent mobility as to simple higher-level applications such as telnet and http, but not real-time applications. The implementation described in this thesis had some initial problems with the procedures for Duplicate Address Detection (DAD), which shall guarantee that all addresses on any given IPv6 network is unique. Some improvements for DAD have therefore been proposed in order to get better solutions as to fault-handling procedures when DAD fails.

Efficient – Base MIPv6 as used in our implementation does not provide the handover efficiency needed for all kinds of applications. Seen from a traditional Internet point of view, the services offered are of best effort quality. A future version of the Internet protocol must, however, be designed to support applications with greater demands to handover latencies, than what a best effort service level can provide. Thus, the handover latency must in these cases be so small that it goes within the boundaries for e.g. demanding real-time applications. Several solutions are proposed for this purpose, but the area of research is still very new and no proposal will be defined for still some time. It seems like the initial mobility deployment phase will be without support for these services, but the technology is very promising and will most likely be integrated as the use of MIPv6 advances.

Secure – In a large mobile environment mobile nodes will not only require Internet access within their own domain. They will also probably visit foreign networks, and as known from GSM infrastructure today, this will not be free of charge. Service providers in foreign domains commonly require authorization to ensure a good business relationship with the client. This leads directly to authentication, and of course accounting (AAA). This AAA infrastructure should be in place before mobile Internet can be deployed worldwide.

Preface

This thesis is a part of our graduate degree (siv.ing.) in Information and Communication Technology at Agder College, and is written for Telenor R&D Agder.

We have elaborated and discussed the concepts of scalability, stability, efficiency and security in MIPv6, and have proposed several solutions that we believe should be taken into consideration in the future development of MIPv6.

We would like to thank our supervisor Alf Martin Sollund and our technical teacher Nils Ulltveit-Moe for help and support while writing this thesis. We would also like to thank our friend Thomas Haslestad for his contributions along the way.

Contents

1. Background

Objectives

- Present the thesis specification.
- Set the boundaries for the thesis.
- Describe the methods used to write this thesis.

Contents

1.1. Thesis Specification

This thesis will focus on several aspects of mobility in IPv6. It will analyze how mobility is implemented, investigate the limitations, and suggest fields for further research. It will especially focus on handoff/handover in Mobile IPv6 (MIPv6) and aims to describe some of its functionality and usability. This will be accomplished by developing scenarios that tests mobility and suggest improvements for the implementations. Scalability and limitations of MIPv6 will also be evaluated.

In order to get hands on experience, and to verify some of the functionality in MIPv6, research implementations will be used for the purpose of testing and evaluation.

Mobility may raise new security issues for the users. For instance, authentication may be required for accounting reasons when a mobile node visits a foreign network. Aspects concerning secure mobile communication in IPv6 will be outlined.

If MIPv6 is going to be utilized, the level of entrance has to be manageable. This implies offering the same basic services as available through traditional IP communication, as well as new and improved services. If these services uncover new limitations or functional requirements to mobility, a short description will be included.

1.2. Thesis Boundaries

The specification states that aspects concerning secure mobile communication in IPv6 will be outlined. This means that our thesis will include the aspects of how AAA services can be integrated into future mobile IPv6 networks and how *mobile nodes* (MN) will be granted access in these. General security issues with IP security (IPsec) are major topics of its own and are not only related to mobility. IPsec will therefore not be elaborated here.

The specification further states that research implementations will be used for the purpose of testing and evaluation of MIPv6 behavior. There are several MIPv6 research implementations available, but interoperability between these is considered out of scope.

We experienced problems with the WLAN equipment we used and it would have been preferable to test other types of equipment in order to try to identify a pattern. This was, however, impossible due to lack of resources and was therefore omitted in this thesis.

The research implementation evaluates mobility according to the base MIPv6 specification, and not according to the proposals described in the chapter about efficient handovers (chapter 5). This is because there are currently few research implementations available capturing procedures for efficient handovers. What currently exists is in an early stage of development and is based on the MIPv6 specification, which is not yet entirely specified.

The current version of the Internet Protocol (IPv4) has some limitations, caused by the Internet's massive growth, motivating the development of a new version (IPv6). Techniques like subnetting and CIDR have helped to contain the rate at which the Internet address space is being consumed. These techniques have also helped controlling the growth in routing table information needed in the Internet routers. At one point, however, these techniques will no longer be adequate and the deployment of IPv6 will be necessary. Since this thesis has the focus of mobility in IPv6, details concerning IPv4 and MIPv4 functionality will not be taken into consideration. For interested readers some of the major differences in functionality between IPv4 and IPv6, and between MIPv4 and MIPv6 are included in Appendix A.

It order to understand the evaluations, conclusions and improvements we have made in this thesis it is recommended that the reader has good knowledge within the areas of internetworking and traditional IP communication. Knowledge about mobility in the Internet protocol is, however, not required because the functionality of MIPv6 will be explained in detail.

With *handover* in this thesis we refer only to the process when a MN moves between network segments and receives a new CoA.

With *roaming* we refer to the process when the MN moves between different administrative domains and must resend its credentials in order to be authenticated at the new network.

1.3. Method

This thesis is based on information taken from various sources, and because MIPv6 is still a new technology, most information is gathered from different IETF draft documents and RFCs. Background material used to set mobility issues into the right context are mainly found in books and white papers.

1.3.1. Information Processing

Mindmaps have been frequently used to be able to manage the large amount of information. This has proven to be an excellent approach, especially during the initial creative phase, to be able to structure the problem domain according to the thesis assignment. The following figure is a snapshot of one of our first superior mindmaps, and as illustrated, our four areas of concentration (scalability, stability, efficiency and security) where already present. It should be noted that mindmaps are organic structures that are being adapted as progression is made. The elements captured during the initial steps gives an indication of the working process, but do not reflect the final result entirely.

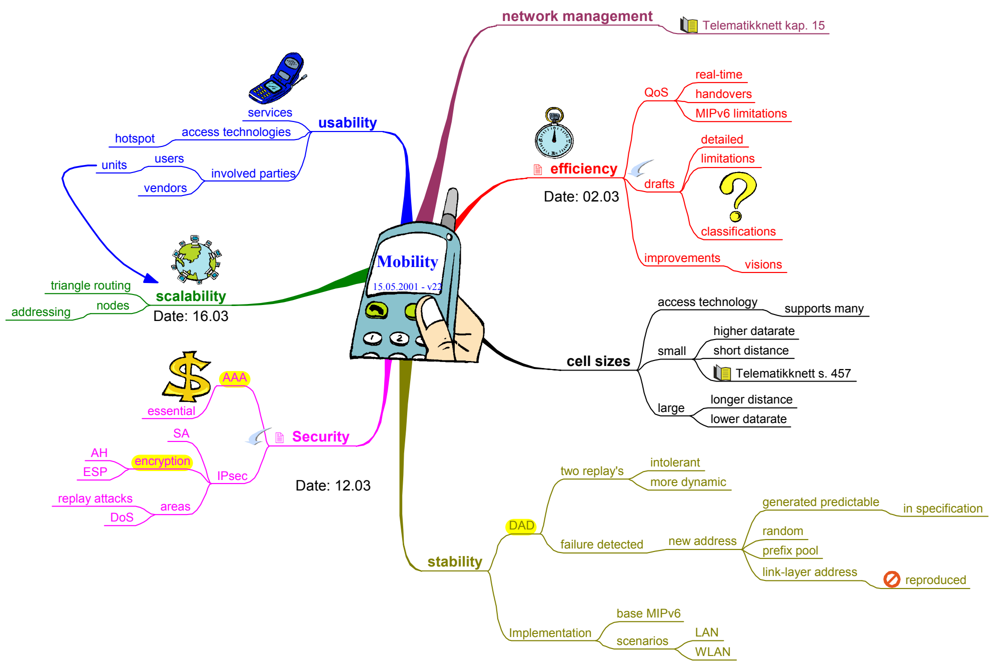


Figure 1.1 – A mindmap developed during the initial creative phase.

1.3.2. Chapter Structure

All chapters in this thesis focusing on technologies are built up using the same basic structure. First of all an *introduction* is included to guide the readers into the actual problem domain. Then a *technical description* explains the technology considered before *evaluating* it with pros and cons. Next, *evaluations* based on the technical description are included followed by a *summary* in which the highlights from the whole chapter are captured. Small modifications to this structure are of course made in each chapter according to the area of consideration, but the basic structure is nevertheless the same.

2. Introduction

Objectives

- Illustrate users need for mobility support on the Internet.
- Explain why mobility should be integrated on the Internet protocol (IP) layer.
- Outline how mobile support can be integrated in the IP-layer.
- Highlight special considerations for a mobile environment.

Contents

2.1. Mobility Characteristics

The last decade has been revolutionary concerning Internet and associated services. This has given us the possibility to access information sources worldwide, but also exchange information in ways that where impossible few years ago. These resources are now accessible from at home, office or school. This perception is however about to change due to the increasing variety of mobile devices and the fact that many people today need access to information while in motion.

In the future it is expected that the number of terminals with wireless access to network resources will increase. The present large number of mobile subscribers today predicts that even a small part of the mobiles demanding wireless IP services, yields a considerable market and important income for telecommunication operators.

Integration of mobility support in future generation networks will therefore enable new and improved services by allowing users to be online, even while in motion. The evolution of mobile networking will differ from that of telephony by the rate of adoption. It took many years for mobile phones to be perceived as convenient, but because wireless mobile computing devices such as PDA’s and pocket organizers have already found user acceptance, mobile computing will likely become popular more quickly [1].

The increasing use of Internet also suggests that the Internet technology can be the best candidate for effective realization of future generation mobile systems. Mobile IP offers the possibility for use of mobile Internet services in other ways than it is used in the wired environment and may be the solution to increasing mobility demands. To provide scalable mobility, these services must use some sort of dynamic addressing structure integrated with the routing mechanisms – e.g. as in

Mobile IP (MIP), where the mobile nodes change their addresses dynamically when moving to a different network segment as described in chapter 3.

2.2. Challenges

MIP allows a mobile node to move from one link to another without changing the mobile node's IP address. A mobile node is always addressable by its *home address*, an IP address assigned to the mobile node within its home subnet prefix on its home network. Packets may be routed to the mobile node using this address regardless of the mobile node's current point of attachment to the Internet, and the mobile node may continue to communicate with other nodes after moving to a new network segment. This means that MIP is designed to provide connectivity with best-effort services and be transparent to higher-level protocols, such as TCP and UDP. Hence, MIP must be scalable, stable, efficient, and secure. To throw light on these aspects this thesis is divided into the following sections:

1. MIPv6 Technology (Ch. 3)
2. Implementation (Ch. 4)
3. Efficient Handover (Ch. 5)
4. AAA (Ch. 6)
5. Duplicate Address Detection – DAD (Ch. 7)

Chapter 3 describes how MIP can provide a scalable solution for future generation mobile systems. This means an explanation of IPv6 and why mobility in IPv6 (MIPv6) may be a better overall protocol than mobility in IPv4 (MIPv4) for the performance in mobile environments, but also an elaboration to the mechanisms that makes integrated mobility in IPv6 possible. Furthermore the general architecture of MIPv6 is explained followed by a closing discussion of the functionality and limitations in MIPv6.

The general architecture of MIPv6 is based on the theoretical draft [2] developed by IETF. By studying this, the functionality of MIPv6 can be studied in theory. In order to get hands on experience and be certain that it function as expected, these theories should also be tested in practice wherever possible. This is done in the implementation chapter and relates directly to the stability of MIPv6. Test scenarios have been developed to examine whether mobility support in IPv6 works according to specifications and these scenarios indicate how mobile nodes should behave and communicate according to the theoretical description. Based on the monitored behavior, new theories, improvements and limitations are elaborated and discussed.

Chapter 5, Efficient Handovers, relates directly to the handover latency when a mobile node moves between different network segments. IPv6 is designed to provide better support for a wide variety of applications with different requirements for data. Some may function even if they do not receive any new data for a long period of time, while others are classified as real-time applications and may fail if they do not receive new data within milliseconds. Even though most applications are quite tolerant to packet loss, it is important to design mobility in such a way that also less tolerant applications when considering packet loss are supported. Elastic applications typically require best-effort services, which is the service level available on the Internet today, and are therefore supported using the base MIPv6 specification. Real-time applications are more sensitive to delays and require more predictable handover qualities. How to improve the support for such applications are elaborated in this chapter through two IETF proposals of mechanisms to achieve more efficient handovers –Fast Handovers for Mobile IPv6 and Hierarchical Handover for MIPv6.

In a mobile environment it is expected that the MNs will visit foreign networks provided by others. With the experiences from today's GSM infrastructure we can guess that this kind of roaming will not be free, and accounting is therefore necessary. Service providers in a foreign domain commonly require *authorization* to be willing to do business with the client. This leads directly to *authentication*, and of course *accounting* – whence AAA. This section is related to the concept of security and explains how these services can be integrated with MIPv6 according to [3]. It will elaborate how a mobile node will be granted access to foreign networks when moving between different network segments by means of the AAA challenge/response mechanism. Integration of AAA services with efficient handovers is discussed, and furthermore, how the GSM SIM authentication mechanism can be used in a MIPv6 enabled environment is elaborated.

The last chapter focuses on the concepts of DAD and how these procedures can guarantee that addresses are unique in an IPv6 network. This is related to the stability of the implementation, and currently, the specification of IPv6 has little or no fault-handling procedures defined for this mechanism. The DAD procedures are explained, and improvements, based on the limitations found, are elaborated.

2.3. Thesis Structure

The thesis is built up around four subjects essential for the design decisions in MIPv6, and this is illustrated in Figure 2.1. These conditions relates to each of the chapters introduced above, thus giving a superior overview of how the thesis is structured throughout this document. It is important to remember that all of the design decisions will have some effect on each of the chapters introduced, but the figure is depicted to show what main areas they relate to.

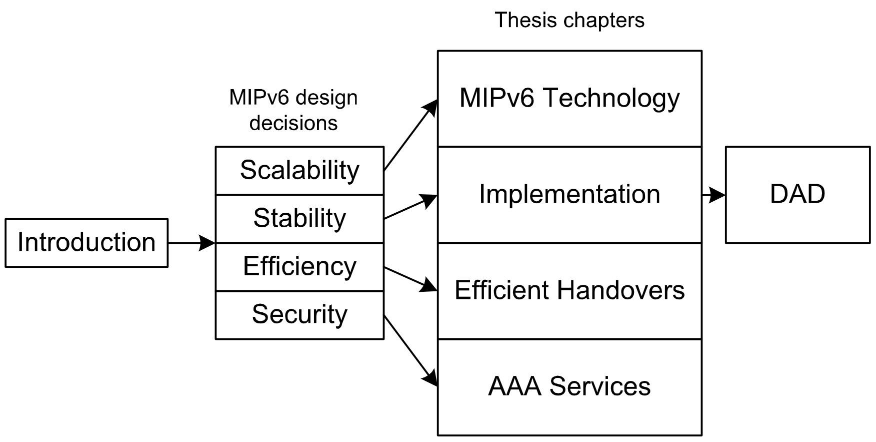


Figure 2.1 – Coherence between design decisions and main topics.

3. MIPv6 Technology

Objectives

- Introduce internetworking with normal IP and describe why this design does not support mobility.
- Explain mechanisms in IPv6 that are essential for mobility integration.
- Describe what extensions that are needed to incorporate mobility into IPv6 (MIPv6).
- Elaborate important design decisions in MIPv6.

Contents

3.1. Introduction

The Internet is all about interconnecting different networks so that these can communicate with each other. The two major problems that must be addressed in doing so are heterogeneity and scale.

Heterogeneity means that users employing one type of network must be able to communicate with users employing other types of network. To further complicate matters, establishing connectivity between networks may require traversing several other networks in between, each of which may be of yet another type. These different networks may be Ethernet, token rings, point-to-point links, or switched networks of various kinds. Each of these will probably have their own addressing scheme, media access protocols and service model. The challenge of heterogeneity is to provide a useful and fairly predictable host-to-host service over this hodge-podge of networks.

To understand the problem of scaling, it is worth considering the growth of Internet, which has roughly doubled in size each year, for the last twenty years. This sort of growth forces us to face a number of challenges such as routing and addressing, but these problems can be solved by use of the IP technology.

The Internet Protocol is the key tool used today to build scalable, heterogeneous internetworks. One way to think of IP is that it runs on all the nodes in a collection of networks and defines the infrastructure that allows these nodes and networks to function as a single logical internetwork. Data from applications and higher-level protocols are encapsulated in IP datagram packets in order to traverse the networks.

The IP datagram is fundamental to the Internet Protocol. A datagram is a type of packet that is sent in a connectionless manner over a network. Every datagram carries enough information to let the network forward the packet to its correct destination, meaning that there is no need for advanced setup mechanisms to tell the network what to do when the packet arrives. When the packet is sent, the network does it best to get it to the desired destination. If the packet gets lost, the network will not attempt to recover from the failure – this must be done on end-to-end basis (e.g. TCP sequence numbers). This is known as an unreliable service and a *best effort* delivery method.

With existing IP technology, there are several elements that makes movement across different network segments difficult. The problem arises when devices are disconnected from the network. Normally when disconnected they cannot continue communication before they have configured a new IP address, the correct netmask and a new default router. Internet and routing protocols deployed today [4] assume that any node will always have the same point of attachment to the Internet. The IP address of a node identifies the link on which the node resides. This means that when a node moves to a new point of attachment without changing its IP address, the address is no longer valid because the address does not reflect where the specific node resides. Communication with this node using existing IP technology will therefore be impossible.

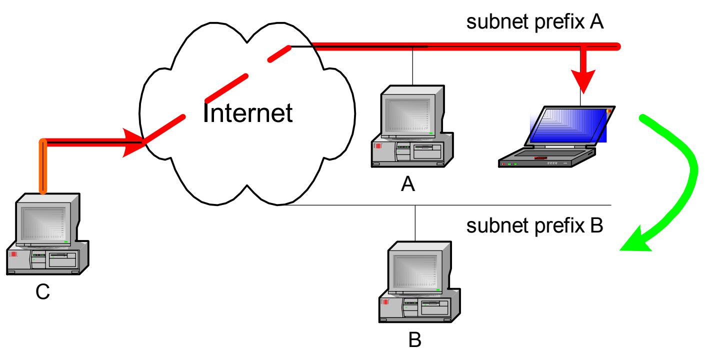


Figure 3.1 – A mobile node moving between network segments.

Figure 3.1 shows a MN moving from one subnet to another. While the MN resides on subnet A, computer C will be able to communicate with it through its regular IP address. This address contains information about the network prefix and packets to the MN will always be routed to link A. When the MN moves to subnet B without changing its IP address, packets will still be delivered to link A and because no node is there to receive them, they will be dropped. In order to communicate with the MN, the MN will have to reconfigure its IP address.

The intention of Mobile IPv6 is to eliminate the problem described above. A node should be able to leave its home link while transparently maintaining all of its connections and still be reachable for the rest of the Internet. This can be done by the use of *home agents* (HA). A MN is registered on its home link and while away from home, the HA delivers packets to the MN on its current point of attachment. The HA must however know where the MN resides in order to deliver the dedicated packets. This is done through update information sent from the MN to the HA whenever the MN changes between different network segments. The current point of attachment for the MN is called *care-of address* (CoA). These mechanisms are according to the draft supposed to be completely transparent to all the layers above IP.

3.2. IPv6 Mechanisms

The basic concepts in IPv6 that render mobility will first be introduced in order to build a foundation for the description of MIPv6, which is elaborated in section 3.3.

The IPv6 mechanisms introduced are *Address Resolution* and *Neighbor Unreachability Detection* (NUD), that both are included in the *Neighbor Discovery* (ND) specification [6]. The procedures for stateless [7] and stateful [8] address configuration are also described.

3.2.1. Address Autoconfiguration

Address configuration specifies the steps a host takes in deciding how to autoconfigure its network interfaces in IPv6. This process includes creating a valid address on the specific link after obtaining the information required, that is whether the address should be obtained through the stateful or stateless approach, or perhaps even both, as described below.

The stateless approach is used when a site is not particularly concerned with the exact addresses hosts are using, as long as they are unique and properly routable. The stateful approach is used when there is a need for improved control of address assignment within a domain. The latter case can be recognized through mechanisms such as DHCP.

IPv6 addresses are leased to an interface for a fixed period of time. When lifetime expires, the binding address becomes invalid and can be reassigned to another interface. To handle the address expiration gracefully, addresses go from preferred to deprecate when lifetime expires. To ensure uniqueness, all nodes run DAD as described in chapter 7.

Stateless

The clear obvious advantage with this method is that it allows hosts to generate their own addresses by combining locally available information and prefixes advertised by routers (see Router Advertisements on page 16). The prefix advertised by routers identifies a subnet associated with a network segment, while hosts have a token that identifies their interface on the specific subnet. By combining the prefix information and the interface identifier, a unique address for each host can be formed. Without prefixes from routers, hosts can only generate *link-local* addresses suited for communication between nodes on the same link, for example on a single link IPv6 network with no routers.

The stateless procedure must start by creating a link-local address for the network interface. This is done whenever a network interface becomes enabled on a new link, by applying the interface token to the well-known link-local prefix. These addresses are known as the IPv6 interface identifier and has the format shown in Figure 3.2.

Traditional interface identifiers for network adapters use a 48-bit address usually known as the IEEE 802 Media Access Control (MAC) address. The IEEE EUI-64 address represents a new standard for network interface addressing in IPv6. The company ID is still 24-bits long, but the extension ID is 40 bits, creating a much larger address space for network adapter manufacturers.

The IPv6 interface identifiers can be derived from the IEEE EUI-64 address or the IEEE 802 addresses. If there is a EUI-64 address assigned to the network card, the 7th bit is complemented to become the interface identifier. If there is no EUI-64 address assigned to the network card, you derive it from the IEEE 802 address by adding the 16 bits of 11111111 11111110 (0xFFFE) into the IEEE 802 address between the company ID and the extension ID. Complementing the 7th bit again creates the interface identifier. To make the nodes addressable, each address includes the prefix fe80, meaning that link-local addresses have the form fe80::/64.

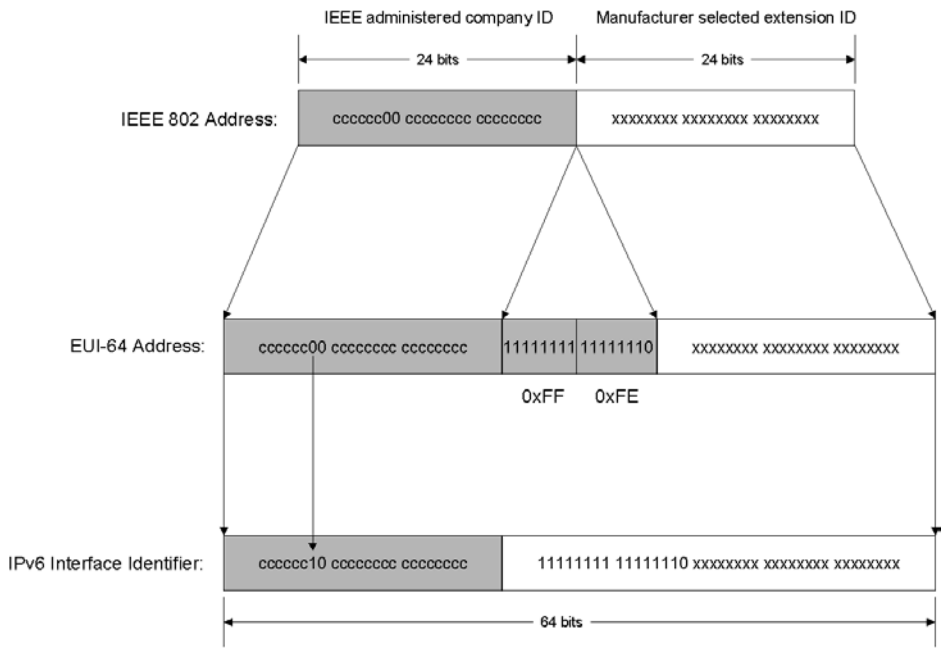

IEEE administered company ID
Manufacturer selected extension ID
24 bits
24 bits
IEEE 802 Address:
cccccc00 cccccccc cccccccc
xxxxxxxx xxxxxxxx xxxxxxxx
EUI-64 Address:
cccccc00 cccccccc cccccccc
11111111
11111110
xxxxxxxx xxxxxxxx xxxxxxxx
0xFF
0xFE
IPv6 Interface Identifier:
cccccc10 cccccccc cccccccc
11111111 11111110 xxxxxxxx xxxxxxxx xxxxxxxx
64 bits

Figure 3.2 – Generate 64-bits EUI-64 address from a 48-bits MAC address.

Before link-local addresses are assigned to their respective network interfaces, they must be tested for uniqueness. This is done with the DAD procedure. It is important to remember that only link-local addresses have to be tested when using stateless address autoconfiguration. If link-local addresses are unique, site-local and global unicast addresses will also be unique, due that they are built from link-local addresses. Appending a prefix to the link-local address forms *site-local* and *global unicast* addresses. The prefix fe80 is in these cases no longer used. The prefixes are obtained from the Prefix Information field in Router Advertisements. The length of the prefix specifies if it should be a site-local or global unicast address.

Site-local addresses are equivalent to the IPv4 private address space. Private intranets that do not have a direct, routed connection to the IPv6 Internet can use site-local addresses without conflicting with global unicast addresses. The addresses are not reachable from other sites and routers must not forward site-local traffic outside the site. They can be used in addition to global unicast addresses and the scope of a site-local address is within the site, e.g. corporate intranet. The format of these addresses is fec0::/48. After the 48 fixed bits is a 16-bit subnet ID field that is provided to create subnets within organizations.

Aggregatable global unicast addresses are equivalent to public IPv4 addresses. They are globally routable and reachable on the IPv6 portion of the Internet known as the 6bone (IPv6 backbone).

As the name implies, aggregatable global unicast addresses are designed to be aggregated or summarized to produce an efficient routing infrastructure. Unlike the current IPv4-based Internet, which is a mixture of both flat and hierarchical routing, the IPv6-based Internet has been designed from its foundation to support efficient, hierarchical addressing and routing. The scope, the region of the IPv6 internetwork over which the address is unique, of an aggregatable global unicast address is the entire IPv6 Internet.

Stateful

In this model hosts obtain interface addresses and configuration information from a server. This is the counterpart to stateless address autoconfiguration and can be recognized as DHCPv6 [8]. The servers specify the addresses that can be used and maintains a database containing information about which hosts that are assigned to the different addresses – similar to the way DHCP function in IPv4. This approach will not be elaborated in this thesis and stateless address autoconfiguration is the mechanism that we used in our implementations (see chapter 4).

Duplicate Address Detection

Neighbor Solicitation and Advertisement messages are used to detect duplicate addresses. This procedure is referred to as DAD. The DAD algorithm is supposed to ensure address uniqueness on any given link and this is elaborated in chapter 7. If a duplicate address is discovered, the address being checked cannot be assigned to the requested interface and the node must be given another address in a stateful manner. During the process of DAD, addresses have the status of tentative, and if they are checked valid, they get the status of preferred and can be used for communication.

3.2.2. Neighbor Discovery

The ND protocol corresponds to the IPv4 protocols ARP [9], ICMP Router Discovery [10], and ICMP Redirect [11]. The purpose of ND is that nodes can use it to decide link-layer addresses for neighbor's known to reside on attached links. Nodes also use ND to discover routers that are willing to forward packets on their behalf and to keep track of which neighbors that are reachable. In the aspect of portability and mobility, ND is used to detect changes in link-layer addresses. Introductions to the essential messages that are a part of Neighbor Discovery are given below. These are Router Solicitation (RS), Router Advertisement (RA), Neighbor Solicitation (NS) and Neighbor Advertisement (NA) and are all ICMPv6 messages encapsulated in IP datagrams. Each of them will therefore contain the IP fields Source- and Destination Address, Hop Limit and Authentication Header.

Router Solicitation

This message is used when interfaces becomes enabled in order to generate router advertisements immediately rather than waiting for them. The message format is shown in the figure below.

Figure 3.3 – Router solicitation message format.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Reserved                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Options ...
+-+-+-+-+-+-+-+-+-+-+-+-
```

The source address is the address assigned to the sending interface or the unspecified address if no address is assigned to it. The destination address is typically the all-routers multicast address.

Router Advertisement

Each router periodically multicasts a router advertisement package on multicast capable links, to announce its availability. The message format is shown in Figure 3.4. The source address must be the link-local address assigned to the interface from which this message is sent. The destination address is typically the source address of the node invoking the router solicitation or the all-nodes multicast address if sent periodically.

Hosts receive these advertisements from all routers, building a list of default routers. Routers generate these advertisements frequently enough that hosts will learn of their existence within a few minutes, but not frequently enough to rely on the absence of these messages to detect router failure. To detect this a separate Neighbor Unreachability Detection Mechanism described later, has been defined.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Cur Hop Limit |M|O|  Reserved |       Router Lifetime         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Reachable Time                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Retrans Timer                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Options ...
+-+-+-+-+-+-+-+-+-+-+-+-
```

Figure 3.4 – Router advertisement message format.

A router advertisement includes information whether the link uses stateless or stateful address configuration, which is determined by the ManagedFlag (the M-bit). If false, the link uses stateless address configuration and the Prefix Information Option should be included in the message. The node should then generate a new address based on its link identifier and the prefix information. If ManagedFlag is true, then the node should invoke the stateful address autoconfiguration protocol to determine its on-link address.

In order to enable centralized administration of critical network parameters, router advertisements also contain Internet parameters such as hop-limit and link MTU. This makes it possible to set these parameters on the routers only, and this information will automatically propagate to all attached hosts.

Neighbor Solicitation

This is the message sent by a link to determine the link-layer address of a neighbor, or to verify that a neighbor is still reachable through its cached address. Neighbor solicitations are also used for DAD. The message format is shown in the figure below.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Reserved                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                       Target Address                          +
|                                                               |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Options ...
+-+-+-+-+-+-+-+-+-+-+-+-
```

Figure 3.5 – Neighbor solicitation message format.

The IPv6 source address is either the address assigned to the interface from which this message is sent, or the unspecified address during DAD check. The destination address is either the solicited-node multicast address corresponding to the target address used when performing DAD, or the target address itself.

Neighbor Advertisement

This is the response to a neighbor solicitation message. This can also be used to send an unsolicited neighbor advertisement message to announce a change in a link-layer address. The message format is illustrated in Figure 3.6. The source address is an address assigned to the interface from which the advertisement is sent. The destination address applies only to solicited advertisements. It can be the source address of the invoking neighbor solicitation, or the all-nodes multicast address if the source address is unspecified (that is if a host has not yet determined its global unicast address).

Figure 3.6 – Neighbor advertisement message format.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|R|S|O|                     Reserved                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                       Target Address                          +
|                                                               |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Options ...
+-+-+-+-+-+-+-+-+-+-+-+-
```

Neighbor Unreachability Detection

Neighbor Unreachability Detection (NUD) is the procedure for determining that neighboring nodes are no longer reachable and is only performed on unicast addresses. There is always a probability for communication to fail, and this can be destination or path failure. If the destination has failed, there are no possibilities for recovery. If however the path has failed, recovery can be possible. NUD is used for all paths between neighboring nodes, and may also be used between routers if no equivalent mechanisms exist.

The recovery procedure initiated by NUD when the path to a neighbor is failing depends upon how the neighbor is being used. If the neighbor is a router, recovery can imply finding a different router. If the neighbor is the final destination, NUD should initiate address resolution again. The recovery required is covered in next hop determination thus NUD signals this need by deleting a neighbor cache entry. NA and NS messages combined with the neighbor cache entries are used for this purpose. The details of NUD are not directly related to mobility, so it will not be elaborated further. RFC 2462 [7] describes the entire functionality for interested readers.

Address resolution

The address resolution procedure in neighbor discovery for IPv6 corresponds to IPv4 ARP resolution mechanisms. The purpose is to determine the link-layer address of a neighbor given only its IP address, using NS and NA messages. The link-layer addresses found are saved in each node's neighbor cache. Address resolution is performed only on addresses decided to be on-link, and never on multicast addresses.

In order to perform address resolution, each node joins the solicited-node multicast address. When nodes want to find the link-layer address of a neighbor, they first check their neighbor cache. If not found, they send a neighbor solicitation with the known IP-address as the target. Each receiving host examines the target of the solicitation to find out if it matches its own. If it matches, it responds with a neighbor advertisement to the soliciting node. The advertisement includes its link-layer address corresponding to the IP-address of the solicitation. The soliciting node updates its neighbor cache with the recently found link-layer address.

Anycast addresses are syntactically the same as a unicast ones. Address resolutions are treated the same way for anycast addresses with two exceptions. The first is that advertisements sent in response to a solicitation are delayed a random time, to reduce the probability of network congestion. The second exception is setting the override flag to zero. This ensures that the first advertisement received is used instead of the last, when multiple advertisements are received on an interface.

Proxy neighbor advertisements are a part of address resolution and indicate that routers can intercept packets on behalf of other nodes. This is for instance used when mobile nodes have moved off-link. This mechanism is identical to the one used with anycast addresses described above. It is also used in MIPv6 when the MN is away from its home network. The HA multicasts a NA on the home link, on behalf of the MN, thus advertising the HAs own link-layer address for the MN's home address. The HA also replies to NS on behalf of the MN. The intercepted packets are tunneled to the MN's CoA.

3.3. General Architecture

In order to explain the functionality of MIPv6 it is essential to know what type of information that is exchanged between the MN and others. All of these messages are IPv6 Destination Options used to carry additional information for mobility support. This section will explain these data structures included in MIPv6 and will conclude with an illustration where a MN moves from one network segment to another.

3.3.1. Destination Options

The four new destination options added in IPv6 is called *Binding Update* (BU), *Binding Acknowledgement* (BACK), *Binding Request* (BR) and *Home Address* (HAddr). BU is used to inform other nodes that it has changed its current point of attachment, thus sending them their new CoA (the care-of address used on the new network segment). If this option requires an acknowledgement, BACK is used for this purpose. BR is used when prompting a node for its current CoA. The last HAddr option is used in packets sent by a MN while away from home to inform the recipient about the MN's home address. MNs generally uses their CoA as source of packets and by including the home address option correspondent nodes can use the CoA directly thus avoiding triangle routing as described in section 3.3.3.

3.3.2. Data Structures

MIPv6 has defined three data structures. These are *Binding Cache* (BC), *Binding Update List* (BUL) and *Home Agent List* (HAL). The BC list is maintained by each IPv6 node and contains binding caches for each of its addresses. This list is searched when sending packets to a MN. If the address is not found, address resolution is initiated and adds the address to the nodes BC. BU packets will also add bindings to the BC. If the destination for a packet is found, packets are delivered to that CoA. Each IPv6 node also maintains the BUL. This list records information for each BU sent by the mobile node for which the lifetime has not yet expired and contains all BU sent to any nodes.

Each MN and each home agent maintains the last data structure HAL. It records information about each home agent from which the node has received RA with the HA bit set (meaning that the router is acting as a home agent). The Dynamic Home Agent Discovery Mechanism uses this data structure (see page 22).

3.3.3. Scenario Description

The figure below shows a MN moving from its home network to a foreign link. The procedures that will be described in this scenario are Home Agent Registration, Triangle Routing, Route Optimization and Home Agent Discovery. It presumes that all the nodes support MIPv6.

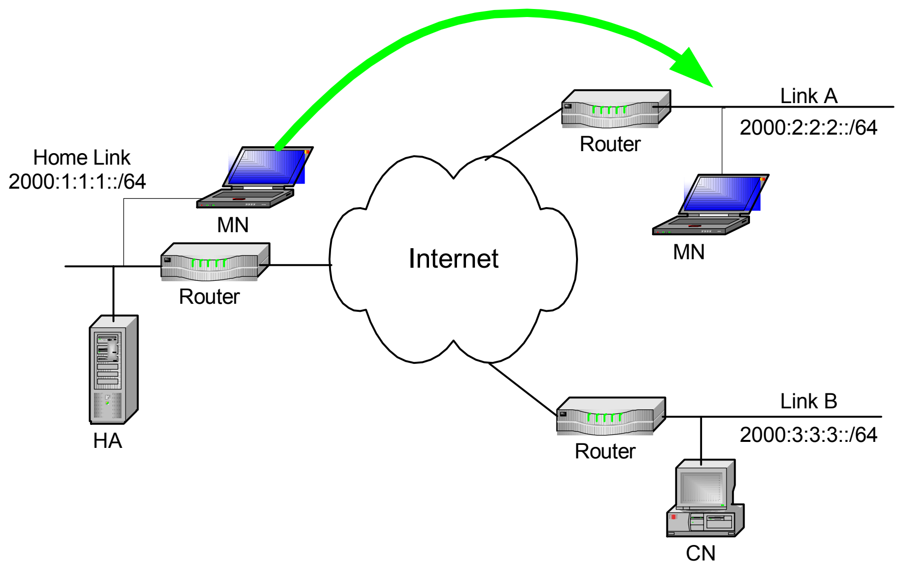


Figure 3.7 – The MN moves out of the home network.

While the MN resides on the home network, the correspondent nodes use its regular home address and no CoA is needed. This address is formed according to stateless or stateful address configuration as described in section 3.2.1. In the stateless approach, the prefix for the home link (i.e. 2000:1:1:1) is combined with the link-level address for the MN (::/64) creating a global unicast address.

Home Agent Registration

Figure 3.8 illustrates when the MN has moved from its home link to link A. Every router in an IPv6 network sends RA to the all-nodes multicast address periodically. The MN needs an RA in order to detect movement and add a new default router. The RA provides the MN with the correct network prefix so that it can start the address configuration procedure. If the delay between each RA is too long, the MN can request it by sending a RS. The RA message on this link specifies the prefix 2000:2:2:2::/64, and the default router will be router A.

Once the RA is received, the MN starts DAD to guarantee the uniqueness of the generated address by sending a NS. If this procedure is successful, the MN has an address ready for use on the new link. This new address is referred to as the MNs CoA.

When the MN has acquired its new CoA, the binding update procedures can begin by sending an IPv6 destination option packet with a BU to the HA. When the HA receives the BU it registers the new CoA and returns a BACK to the MN confirming that the binding is accepted. The MN also send BUs to the nodes listed in its BUL. Authentication Header or Encapsulated Security Payload to avoid misuse by other nodes must protect these binding messages. These aspects are out of scope for this thesis, but interested readers can find it in [12].

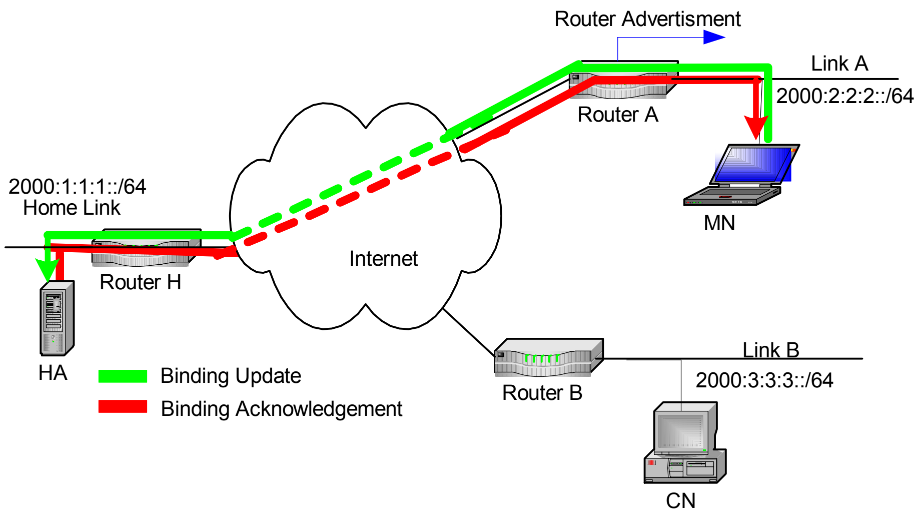


Figure 3.8 – HA is provided with the new CoA using BU and replies with a BACK.

Triangle Routing and Route Optimization

When the MN has received its new CoA, the HA will intercept all packets addressed to the MNs home address. When the MN sends packets, it sets its CoA as the source address and includes a home address destination option. A MN is statically configured with a home address, opposed to the CoA that changes when moving between the network segments. Protocol layers above IP do not, however, see the changing CoAs – only the home address. The MN itself sends packets directly to the nodes it communicates with, known as correspondent nodes (CN). This scenario is known as triangle routing. The problem with this is that packets may need to traverse a longer path than necessary, thus causing longer delays. Consider a CN on the same subnet as the MN far away from the MN's HA. All packets from the CN must traverse through the HA although the MN and the CN resides on the same subnet.

A solution to triangle routing is Route Optimization. This provides means for nodes to cache the binding of a mobile node and tunnel their datagrams directly to its CoA. Route Optimization is illustrated in Figure 3.9. The MN has a CoA that is unknown for the CN. The CN delivers the packet to the MN's home address. The HA intercepts this packet and tunnels it on to the MN.

When the MN receives the tunneled packet, it is able to detect that this packet is originating from the CN, thus sending a BU to that address. The CN will register the new binding in its BC. The next packet addressed to the MN can then be sent directly, thus avoiding triangle routing.

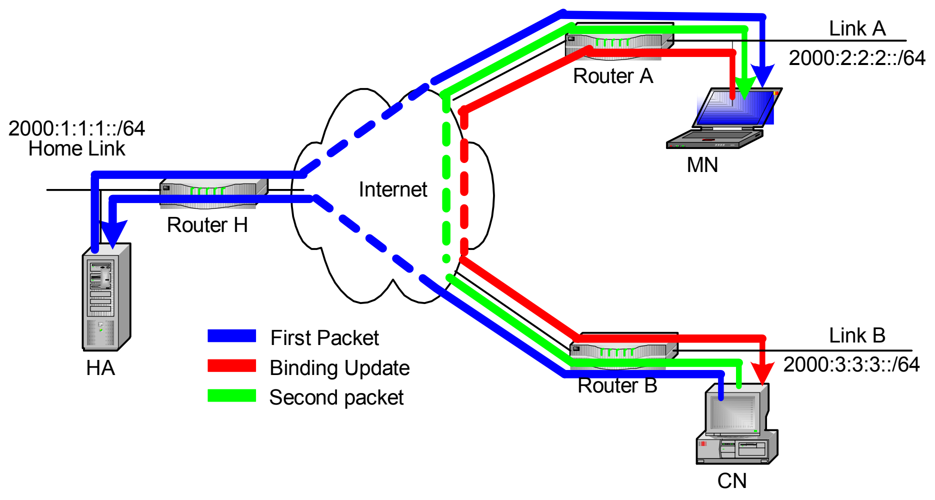


Figure 3.9 – Route optimization eliminates triangle routing.

The BU sent to the MN can decide whether or not the receiver should generate a BACK. If it chooses to disable BACK, it will know if the binding was received by inspecting the next packet from this specific node. If that packet comes from the HA, the binding was not received and the MN should generate a new BU message.

Each binding in the BC is associated with a lifetime. There are two possibilities to renew the binding when the lifetime expires. The MN can send a new BU, or the CN can request a new binding by sending a BR.

Home Agent Discovery

There may occur situations where the MN does not know the IP address of its HA. This can happen if nodes on its home link have been reconfigured while the MN has been away from home, causing the router operating as the MNs home agent to be replaced by another router. MIPv6 has defined the Dynamic Home Agent Discovery mechanism to deal with these situations. This procedure allows the MN to dynamically find routers serving as HAs on the MNs home network.

The MN attempts to find a designated HA by sending an ICMP Home Agent Address Discovery Request message to the home agent anycast address on its home network. It uses its CoA as the source of this message. All home agents serving this subnet receive the message and should reply with one of its global unicast addresses to the MN's CoA, thus letting the MN find its IP-address.

3.4. Evaluation of MIPv6 Technology

The base MIPv6 specification provides mechanisms for transparent mobility support in IPv6, but this service is not necessarily transparent for all kinds of applications. Whether transparent or not depends on the latency in which the MN detects that it has moved and then make the necessary arrangement to continue the communication. Transparency and movement detection will therefore be discussed in the following subsections.

3.4.1. Transparency

MIPv6 is a technology that provides layer three mobility. This implicates that it should be transparent to all higher levels including the maintenance of active TCP connections and UDP port bindings. The discussion on whether this is true is based on what kind of applications and services the MNs are using in order to receive transparent mobility from the network.

Different applications have different requirements as to what kind of delays they can handle. The critical aspect when providing mobility in IPv6 is the delay experienced when a MN performs handover from one network to another and if higher layer applications detect this delay. If time critical applications are used and there is a loss in connection due to handover delays, transparent mobility is not provided. There are however applications that are not so sensitive to delays, such as http, ftp and email, and transparent mobility is easier to provide for these kinds of applications.

The conclusion about transparent mobility is based on what kind of applications that are used. For non time-critical applications, transparent mobility can be provided from the base MIPv6 technology and this is further investigated in section 4.5 when testing mobility for higher-layer protocols. For real time services this is not sufficient. The delays in handover must be reduced in order to use these applications and this can be provided with efficient handovers and is further investigated in chapter 5.

3.4.2. Movement Detection

An important aspect when considering mobility in IPv6 is how the MNs detect that they have moved from one link to another. This is critical in order to provide mobility, since delays in movement detection mechanisms will cause delays in obtaining a new CoA. This will again decrease the performance when moving between different networks. The specification of MIPv6 defines that a MN can use any combination of the mechanisms available to detect that it has moved from one network to another. These are router discovery, neighbor unreachability detection (NUD) and indications from the link-layer.

Router Discovery

When a MN moves between networks it can detect routers by waiting for the periodic router advertisement or it can send a router solicitation. The neighbor discovery protocol has limits for how often a MN can send solicitations, and these limitations can prevent the MN to receive router advertisements quickly, thus delaying the detection of a new router. Therefore, MIPv6 does not impose these strict limitations, meaning that a MN when away from home can send router solicitations more frequently. In addition, the interval between router advertisements can be reduced. This can lead to a reduced overall network performance due to the increased network traffic, but it will also increase handover performance for the MN. It is important to remember that the MNs will reside on wireless links with limited bandwidth, implying that the router solicitation interval will represent a compromise between handover performance and network load.

The main goal with movement detection mechanisms is that MNs should detect their new routers as quickly as possible and this implies that there is a balance between handover performance and the network load created. When considering solicitations, a maximum number that can be sent has been defined to prevent unnecessary network load. The only exception is MNs that are moving to another network. They can exceed this limit if they are currently without a CoA on their new link, thus helping the MNs to detect a new router more quickly.

There is however a problem with this approach. If a MN increases the RS rate without knowing that it has moved to a new link, this will cause extra traffic on the old link, where the MN already is registered with an existing CoA. To prevent this, the MN must know for sure that it has moved to a new link, e.g. by receiving a positive indication from lower protocol layers (see below). It is also important that MNs reduces their RS rate after they have received a CoA, to prevent unnecessary network load.

Neighbor Unreachability Detection

While a MN is away from home and is using some router as default, it is important for the MN to be able to quickly detect when that router becomes unreachable so that it can switch to a new default router and to a new care-of address. Since some links (e.g. wireless) do not necessarily work equally well in both directions, it is likewise important for the MN to detect when it becomes unreachable for packets sent from its default router. The MN can then take necessary precautions to ensure that any CN attempting to communicate with it can still reach it through some other route. This can be done through the NUD procedure, which is a movement detection mechanism that can tell the MN if it has moved to a new link.

The details in the NUD procedure can be found in [2]. It is, however, important to remember that the MN cannot efficiently rely on NUD alone. This is because the network load would be prohibitively high in many cases for a MN to continually probe its default router with NS messages even when it is not otherwise actively sending packets to it. To prevent this from happening it is recommended that the MN can use every received packet from its router as an indication that it is still reachable on its current care-of address. The router should send RA messages periodically and the MN will have frequent opportunity to check if the default router is reachable.

Lower Level Indications

Another possibility for movement detection is obtaining some type of indication about link-layer mobility from lower protocol layers, or device driver software, controlling the network interface on the MN. This is especially important when considering wireless access technologies such as WLAN. A MN can for instance use signal strength or signal quality information for its link with the available routers to decide when to switch to a new primary care-of address that could provide a better connection. It is, however, essential that the MN does not assume that all link-layer mobility indications from lower levels necessarily means a movement to a new network segment. Movement from one WLAN cell to another can be made transparent to the IP layer if all the WLAN access points are operating within the same network segment.

Evaluations of Movement Detection

Movement detection is essential in order to provide mobility in IPv6. Although the default procedure for this is router discovery, it would be preferable to provide additional mechanisms in order to prevent too much network load. The NUD can help the MNs in determining that their default routers are no longer available, thus indicating that they must change to another care-of address. NUD can also help to decide whether a MN can increase their RS rate or not, based on advertisements and indications that the MN really has moved to a new link. The efficiency of movement detection can also be increased if the MN could obtain indications about link layer mobility from lower protocol layers. This means that the MN can detect new routers while still receiving advertisements from its current default router and decide whether or not to switch based on for instance signal strength in a wireless access technology. The conclusion is that a MN should use any information possible in order to perform handover to another network, thus minimizing the delay when changing its primary care-of address.

3.5. Summary

The Internet is all about interconnecting different networks so that these can communicate with each other. The two major problems that must be addressed in doing so are heterogeneity and scale. The existing IP technology (IPv4) has some limitations as to these requirements and IPv6 has been chosen due to its better addressing capability and integrated support for mobility, where neighbor discovery and stateless address autoconfiguration are the central mechanisms that enables this.

In MIPv6 every MN is registered in a home network and while the MN resides on this network, other nodes can communicate with it through its home address. If the MN moves to another network, they receive information from that network through router advertisements and configure a CoA. CoAs are used for communication when the MN is located in other networks. Packets that are delivered to the MNs home address are intercepted by the home agent and delivered to the MN through its current CoA. Through the use of binding update messages, the home agent and other correspondent nodes receive information about the current care-of address when the MN moves to another network.

MIPv6 is a technology that provides layer three mobility, and is supposed to be transparent to all higher-level applications. Whether this is true or not depends on the requirements these applications have concerning delays in a handover scenario. For simple applications such as ftp, http and email, transparent mobility is provided through the use of the base MIPv6 technology as described in chapter 4. For real-time applications, delay requirements are stricter, and transparent mobility cannot be expected through the base MIPv6 technology for these applications. There will be a need for smaller delays, and the need for efficient handovers as elaborated in chapter 5 are needed.

4. Implementation

Objectives

- Explain the motivation for developing mobile test scenarios.
- Illustrate the differences between wired and wireless scenarios using LAN and WLAN.
- Outline implementation specific details for extending IPv6 with mobility.
- Elaborate MIPv6 design decisions that may lead to erroneous behavior, and how to detect this.
- Demonstrate that mobility is transparent to higher layer protocols for best-effort services.

Contents

4.1. Introduction

Different theoretical background material can be used to throw light on how mobility support can be integrated into the future generation of the Internet protocol. These theories should also be tested in practice wherever possible, to validate the expectations. Some test scenarios have therefore been developed to examine whether mobility support in IPv6 works according to the MIPv6 draft. The scenarios indicate how mobile nodes should behave and communicate according to the theoretical description. Based on the monitored behavior in these scenarios, new theories, improvements and limitations are elaborated.

4.2. Technology Decisions

Before configuring a system for mobility support a deliberated decision of how it should be implemented must be made. Factors, such as what kind of platform, operating system and access technology that is appropriate should be determined in advance so that the implementation is useful for the target group. This will be further explained in the following sections.

4.2.1. Platform

To create a mobility scenario it is essential to focus on the different types of entities that should be used. In the scenario described in section 4.4 there are three entities involved that have to support MIPv6. This is the router and the two MNs. The operating systems (OS) considered for these entities are Windows 2000, FreeBSD and Linux due to their reported experimental support for both IPv6 and MIPv6. The decision fell on evaluating only Windows 2000 and FreeBSD. The former because Microsoft is currently developing their IPv6 stack and have just started to integrate MIPv6 functionality, in addition to resources and contact persons in Microsoft Research (MSR). FreeBSD is evaluated because it is a platform where many early experimental stack implementations are developed (because of the OS stability).

During development of experimental software there will always be bugs, either reported or unreported. To minimize the sources of error while testing the implementation another decision was made, and that was to use the same OS and MIPv6 implementation on all network entities.

FreeBSD

The KAME kit [13] implements both IPv6 and MIPv6 support in FreeBSD. This was originally considered to be the ultimate OS for the test scenario, because the implementation has already existed for some time and there are many experiences from previously implementations and test scenarios [14]. We have also in a previous occasion configured FreeBSD with support for basic IPv6 services, so configuring it with mobility support as well should be manageable.

This OS has good support for MIPv6 and the implementation is stable, but the problems are however related to the mobile nodes and different configuration utilities. Wireless equipment, such as PCMCIA WLAN cards, has to be used to test mobility, and FreeBSD has no support for the WLAN adapters available to us. It is, of course, possible to build new drivers for these network adapters, but this would create unnecessary work not directly related to mobility issues in our thesis.

Windows 2000

The most common OS used today on end-user computers is (some sort of) MS Windows, and it is therefore important to implement support for IPv6 on this platform at an early stage. The relevance of providing mobility on the most used platform cannot be ignored, and Microsoft Research released their first (experimental) MIPv6 implementation when we started working on this thesis (January 2001).

When implementing the scenario for the mobile environment it is necessary with a router that can direct packets between the different network segments, and IPv6 hardware routers are expensive and few, with limited functionality. MSR provides, however, a router script [15] that easily can be modified to work in our implementation (described later). Mobility can then be tested with great flexibility, and also be used with a lower level of entrance when it comes to modifications and improvements in the routing functionality. The software that comes with the MIPv6 release from MSR includes a graphical configuration utility where the different parameters can easily be changed [16], such as support for multiple home agents and other parameters that relate to mobility behavior.

When considering future mobile networks, backbone routers will have high demands regarding stability and uptime. In these scenarios FreeBSD would probably be preferred in the backbone networks due to the better stability compared to the Windows 2000 platform [17]. It should however be noted that neither FreeBSD nor Windows 2000 will probably be used as commercial backbone routers. The requirements to this kind of routers can in many cases be so high that only specialized hardware routers, e.g. from Cisco, can deal with the traffic load. Software routers are, in fact, a popular way to build experimental switches when you want to be able to do things like develop new routing protocols because it offers extreme flexibility and a familiar programming environment. They are easier to modify but not very efficient and is therefore appropriate in small, rapidly changing, network environment [18].

Evaluation of OS Choice

When considering the factors mentioned above, Windows 2000 was chosen to test mobility of several reasons:

- The OS is very popular among end-users, so it is important to provide coming technologies as early as possible to support new and increased demands.
- Microsoft is currently implementing IPv6 (and MIPv6 according to the previous version of the MIPv6 specification [19]) support into their operating systems, and they provide the source code to their experimental implementation for development and education purposes. It is therefore possible to be a part of, and contribute, to the ongoing development process.
- MSR was very helpful as to provide support in several challenging test scenarios when testing their stack (see section 4.4.5).

The ideal choice would be to test mobility using different operating systems on the network entities and prove interoperability between different MIPv6 implementations. We are not, however, trying to prove interoperability between different implementations, only mobility, and this is still at an early stage in the specification phase. Versions from different vendors can behave dissimilar since developers that interpret the specifications think differently. Even though it would be very interesting to make dissimilar implementations work together, this would be a very time-consuming task and has been left out to prove for interested readers.

4.2.2. Access Technology

We decided to test mobility in IPv6 on different network access technologies. Local Area Network (LAN) and Wireless Local Area Network (WLAN) were chosen. LAN, and in this case Ethernet, because it is an industry standard supported by most vendors. LAN was initially used to minimize the source of errors and get MIPv6 up running, and is not an access technology that represents true mobility in the test implementation. WLAN was chosen because it represents true mobility in a wireless environment. In a mobile environment covered by communication systems with a small bandwidth, such as GPRS, WLAN cells can be introduced to provide short distance coverage with a high bandwidth – so called hotspots.

An obvious way to make use of some of the opportunities mobility can provide is when traveling. Users may carry out their work almost as when physically situated at their office by using the quality of handovers between different network segments. In this way the home-office can be extended to not only be actually at home, but also while traveling to a meeting in another city or simply preparing today's tasks on way to work.

To give users an improved level of service, WLAN cells may be located in central places to occasionally offer high-speed communication. These hotspots may be used to download large amounts of information, while less bandwidth requiring services, such as e-mail, will function as usual all the time.

If hotspots were the main concern, the ideal solution would actually be to combine different wireless access technologies in a test scenario, but because mobility in IPv6 is still a new technology it would generate too much overhead creating a "bridge" between these access technologies, because there are still no off-the-shelf solutions for this purpose. The support for different access technologies are one of the main reasons that Internet has become so popular and it would of course be a good idea to try this feature, but this is a very time consuming task. But, mobility can much easier be demonstrated in a homogenous environment by moving the mobile node between different WLAN network segments. Since only one wireless and one wired access technology was available to us when testing the implementation, heterogeneity between different equipment and different access technologies is omitted in this thesis.

Wired vs. Wireless

The Internet protocol development has for long been focusing on wired media with small error rates and high bandwidths. There are however great differences between wireless and wired systems concerning the factors of error rates, delays and bandwidths. The rates of errors in wireless systems are greater and the delays are higher and also more variable, and these factors are taken under consideration when testing mobility. The bandwidths in wireless systems are also a natural limited resource due to the frequencies available.

Two service models are defined according to this and the two access technologies chosen, which represents both wired and wireless access technologies. A description is given in the sections below to illustrate the physical differences between LAN and WLAN.

LAN Service Model

The following figure shows the service model for the LAN implementation, and it represents the behavior of communication between a MN and a router.

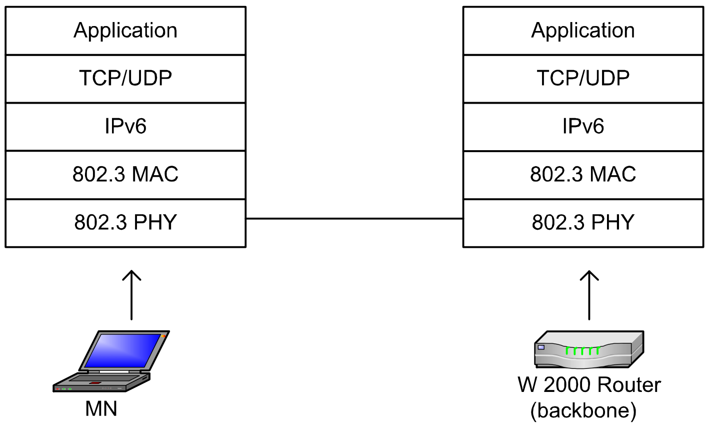


Figure 4.1 – LAN communication between a MN and a router.

The lowest layer is the physical layer of LAN shown as 802.3 PHY and is responsible for moving the bits on the transmission medium. The physical layer defines the mechanical and electrical characteristics of the transmission media. The next layer, 802.3 MAC, is known as the link-layer. This layer uses services from the physical layer to move frames of data from one node to another over a link (802.3 Ethernet link). The main assignment for the link-layer is error protection to provide reliable communication, but it is also responsible for dividing the physical media between its users. The two lowest levels are the ones that are different when using dissimilar access technologies such as LAN and WLAN.

The third layer is the one focused on in this thesis. This is the network layer and represents the use of the Internet protocol. The Internet protocol layer uses transport mechanisms from the link-layer in order to transport packets in the network. The packets are transported between the sender and the receiver independent of the number of links and what kind of physical media that is used to deliver it to the intended recipient. This layer defines the logical addressing scheme and is in our case known as IPv6 addresses.

WLAN Service Model

The service model of our Wireless LAN is quite similar to the one shown for regular LAN, except for the Access Point (AP) that is responsible for the translation between the two different access technologies as illustrated in the following figure. The other levels have the same functionality as the ones mentioned for the LAN service model.

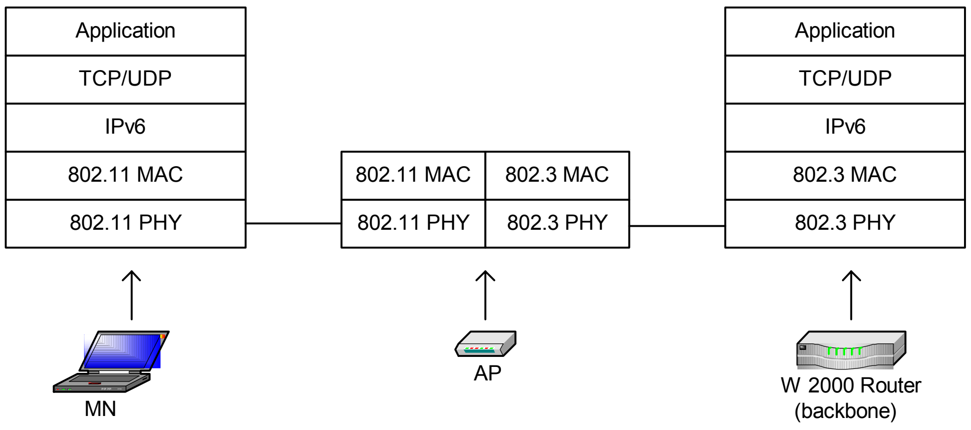


Figure 4.2 – WLAN communication between a MN and a router through an AP.

The two service models discussed follows the Open System Interconnection Reference Model standard [20]. This implies that lower protocol layers are transparent to higher ones. IP can traverse over many different access technologies and the service model implies that the functionality of IPv6 should be the same on WLAN as on LAN. This is elaborated in section 4.4.

4.3. Microsoft IPv6 Evolution

Because we, as concluded above, decided to implement mobility in a homogenous OS environment using Windows 2000, a short description of how this code has evolved is described here.

Early in 1998 Microsoft released their first version of an IPv6 implementation – MSRIPv6 1.0. This implementation has continued to improve and subsequent versions have been released to support new features.

4.3.1. Separate Source Branches

The latest release of the base IPv6 implementation from MSR is version 1.4, which was released in the beginning of 2000. Based on this stack Microsoft has developed a working implementation of Mobile IPv6 on Windows2000 in collaboration with researchers at Lancaster University as part of the LandMARC project [21]. The implemented MIPv6 functionality is described in version 12 [19] of the IETF Mobile IP draft and they are currently reviewing the proposed version 13 [2] for support in the near future, but these versions are not anticipated to interoperate.

Even though the Mobile IPv6 implementation is based on version 1.4 of the MSRIPv6 stack, they do not interoperate. This is because there are modifications to the base implementation as well as additions. As the following figure illustrates, the MIPv6 implementation is developed in a separate branch in the source tree and this implies mixing different versions of some of the modules. The MIPv6 code will be integrated into a future full release of the MSRIPv6 stack.

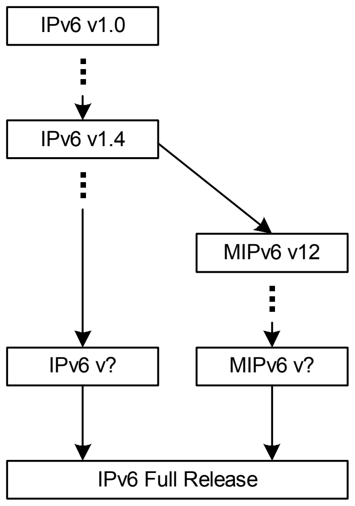


Figure 4.3 – Future integration of the IPv6 and the MIPv6 stack.

Due to external interest Microsoft decided to make a beta version of MSRIPv6 publicly available in both source and binary forms for research, educational, and testing purposes. The software is not commercially supported, so the Windows networking group is in parallel also working on a product-quality IPv6 implementation – IPv6 Technology Preview. According to Microsoft, the IPv6 Technology Preview for Windows 2000 is stable, yet it is not recommended for commercial deployment since 24/7 support is unavailable. Production quality IPv6 is expected to be released in early 2002, but the production level reference is for the Windows XP (a beta version of IPv6 Developer Edition is expected to ship with Windows XP this year).

The MSRIPv6 release comes with source code and supports NT 4, while the IPv6 technology previews require Windows 2000 and are available in binary form only. Thus if NT 4 support is required or alterations of the source code is necessary, MSRIPv6 1.4 must be used. Otherwise it is recommend that the latest technology preview is used since it has a slightly newer version of the stack than what is the case with MSRIPv6 1.4.

IPv6 for Windows XP is supposed to be quite different from MSR and from IPv6 technology preview, with added features and scalability requirements. At this time there are no plans to backport the XP IPv6 to Windows 2000, in part because it would involve a large number of OS internal components. The IPv6 stack itself is a small piece of code compared to changes in related networking code.

The implementation is an experimental prototype for the coming IPv6 version that will be distributed with the Microsoft operating systems, so it shows signs of being an ad hoc implementation. The implementation is poorly structured and the code is not documented adequately. Modifications therefore require thorough studies of the implementation specific details as well as good knowledge of how IPv6 works. The MIPv6 additions are newer and even more unstructured than the base IPv6 implementation. It is mainly developed to test how mobility will function on the

Microsoft platform and will probably undergo major changes before a final version is released. This relates both to the structure of the implementation as well as the functionality specified in the current and future versions of the MIPv6 draft document.

4.3.2. From IPv6 to MIPv6

New features and modifications are necessary to extend the base IPv6 implementation to support MIPv6 functionality. This is only outlined here, but a summary of the modifications can be found in Appendix A.3. A thorough description of all the implementation details can be found in [22].

There were two broad categories of changes needed to implement mobility support in the IPv6 stack – extensions and alterations. Extensions provide the additional functionality or semantic changes needed for mobility support; the second denotes minor corrections to the code necessary to maintain stability. Additionally, there were several minor changes made to the code in order to allow it to compile with the Windows 2000 DDK [23].

To add mobility support, new data structures and sections of code were added to the IPv6 stack. Additionally, several modifications to the route cache structure, packet generation and configuration interface were necessary.

4.3.3. Current Stack Behavior

The implementation has evolved to become a more complete solution as new versions have been released. The implementation supports stateless address autoconfiguration and in the common case, there is no need to perform any manual configuration. By default, a link-local address is always assigned to each Ethernet network interface upon start-up. Non-link-local addresses such as site-local addresses or global addresses are automatically assigned based on the receipt of IPv6 router advertisements. In order to receive additional addresses through IPv6 router advertisements, a properly configured IPv6-capable router must be connected to the network segment.

The stack supports Plug and Play and Power Management for Windows 2000. USB and PCMCIA network interfaces can be added to or removed from the system while on and the stack will reconfigure itself accordingly. Similarly, you can disconnect and reconnect network links or hibernate and resume your system and the MSRIPv6 stack will do the right thing. You can also dynamically unload and reload the stack without rebooting.

4.4. Implementation Scenarios

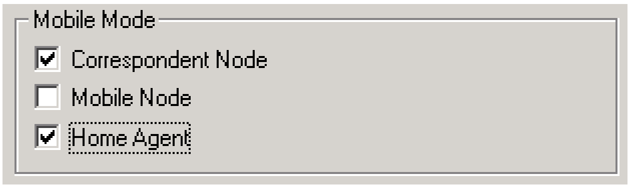
To be able to exploit the functionality specified in the MIPv6 specification of Windows 2000, MSRIPv6 v1.4 must be installed. This section describes important aspects and experiences gained after installing and testing the stack with basic services such as the *ping* operation.

After installing the MSRIPv6 v1.4 implementation the basic IPv6 mobility network operations are supported. This does not, however, unconditionally guarantee support for services on higher protocol layers, and this will explicitly be tested and described in section 4.5.

4.4.1. Network Tools

In order to undertake a more thorough analysis of the MIPv6 implementation, it is necessary to utilize different monitoring and configuration tools.

All the nodes must be configured with mobility support by installing the stack (for installation details, see [24]). The stack can be dynamically configured to run in any combination of modes. These modes include mobile mode, correspondent mode, and home agent mode. When in mobile mode, home addresses can be dynamically added and removed and the security settings for home agents can be configured. This is all achieved via an IOCTL control interface [25]. When any change takes place to the configuration, the new settings are stored in the Windows registry where they are subsequently reloaded during driver initialization.


Mobile Mode

[✓] Correspondent Node
[ ] Mobile Node
[✓] Home Agent

Figure 4.4 – The home agent is configured with the HA and the CN option.

A useful network trace tool for Windows is the MS NetMon (Network Monitor), which is used to capture packets on different network nodes according to the test scenarios described in the following sections. The program is distributed with the SMS 2.0 [26] product available through MSDN [27]. This can parse much of the IPv6 protocol suite, but not (yet) the entirety of MIPv6.

Another useful utility for viewing diagnostic (trace) messages from the MIPv6 stack is the DebugView application available for download from the Sysinternals web site [28]. This application allows the programmer to capture and view KdPrint messages from an application window on the machine running the stack, which are kernel debug printouts used in the MSRIPv6 implementation.

4.4.2. LAN Scenario

The next figure shows the initial configuration of the test implementation when LAN is used as the access technology. The Windows 2000 software router and the two MNs all have MIPv6 support on their network interfaces and the MNs are connected with a network cable to their respective hubs. The figure illustrates two different network segments with one hub on each.

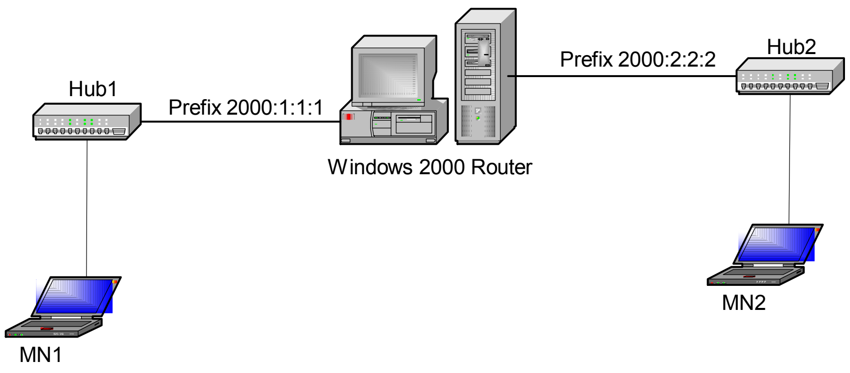


Figure 4.5 – LAN configuration.

The Windows 2000 router was configured with two Ethernet network cards, one 3com and one HP card, each representing different segments. To route traffic between these two segments a software router script was used as shown in Appendix B.3. The first segment was given the prefix 2000:1:1:1 and the second 2000:2:2:2. When this scenario was run, NetMon was used on MN2 to capture the messages that was exchanged. In this capture file the two entities are shown as Hub1 and Hub2 to easily be able to distinguish between the two network segments, and the file can be downloaded from [54].

The routers are responsible for broadcasting the prefixes on their respective links so that the MNs are able to detect movement and retrieve the information necessary for configuration. Explanations of the commands used are included in. The router depicted in the figure is also acting as home agent for these two network segments.

The router script is run as a batch file and can be executed from the command prompt. The prefixes are advertised on their links through router advertisements and received by the mobile nodes. When you accept the new home address, stateless address autoconfiguration is performed and communication can begin as described in chapter 3.

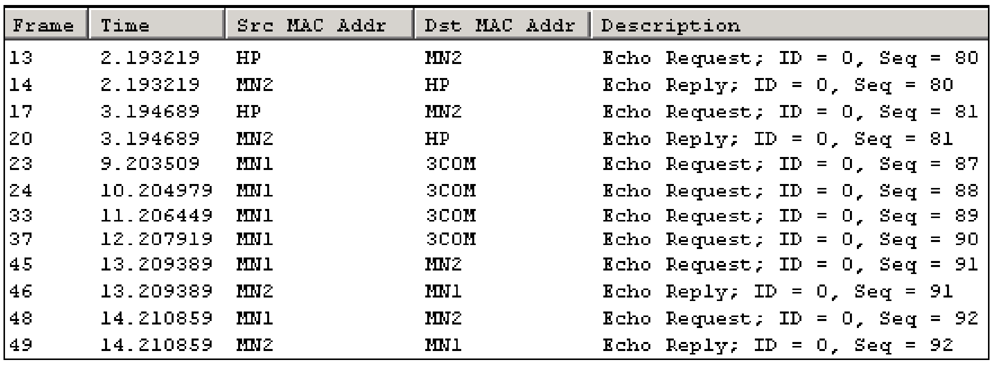
To prove mobility we used the *ping* command to send *ICMP echo request* messages continuously from MN1 to MN2 while the latter where moving between the two network segments. MN2 replies every echo request message it receives with an *ICMP echo reply* and this goes on for as long as the scenario is running. For every echo request MN1 does not receive an echo reply, a "destination unreachable" printout is shown in the command prompt. It is therefore easy to visually detect if MN2 is disconnected and out of reach.

Moving To Foreign Link

Figure 4.6 illustrates MN2 moving from subnet 2000:2:2:2 to subnet 2000:1:1:1. Movement is detected in MIPv6 in two different ways as explained in the introduction – the interval for router advertisements has expired and a router advertisement from a new router is detected. The interval for router advertisements is set default to half a second, but this can be changed.

A complete reference of the message format on all the messages exchanged is included in the capture file [54]. MN2 periodically receives router advertisements while connected to either of the links.

To move MN2 it must physically disconnect from Hub2 and then connect to Hub1. As we can see from line 23 to 46 in the capture file, the ping sequence number increases from 81 to 91.



Frame	Time	Src MAC Addr	Dst MAC Addr	Description
13	2.193219	HP	MN2	Echo Request; ID = 0, Seq = 80
14	2.193219	MN2	HP	Echo Reply; ID = 0, Seq = 80
17	3.194689	HP	MN2	Echo Request; ID = 0, Seq = 81
20	3.194689	MN2	HP	Echo Reply; ID = 0, Seq = 81
23	9.203509	MN1	3COM	Echo Request; ID = 0, Seq = 87
24	10.204979	MN1	3COM	Echo Request; ID = 0, Seq = 88
33	11.206449	MN1	3COM	Echo Request; ID = 0, Seq = 89
37	12.207919	MN1	3COM	Echo Request; ID = 0, Seq = 90
45	13.209389	MN1	MN2	Echo Request; ID = 0, Seq = 91
46	13.209389	MN2	MN1	Echo Reply; ID = 0, Seq = 91
48	14.210859	MN1	MN2	Echo Request; ID = 0, Seq = 92
49	14.210859	MN2	MN1	Echo Reply; ID = 0, Seq = 92

Figure 4.6 – Lost echo request messages in the capture file.

Since the network traffic is monitored from MN2 the capture only displays messages on the link it is currently attached to. This implies that 10 echo requests in between have been lost due to MN2 has been disconnected from the network. Remember that when MN2 moves to the same link as MN1 it will send a BU option as long as the communication is active to avoid triangle routing.

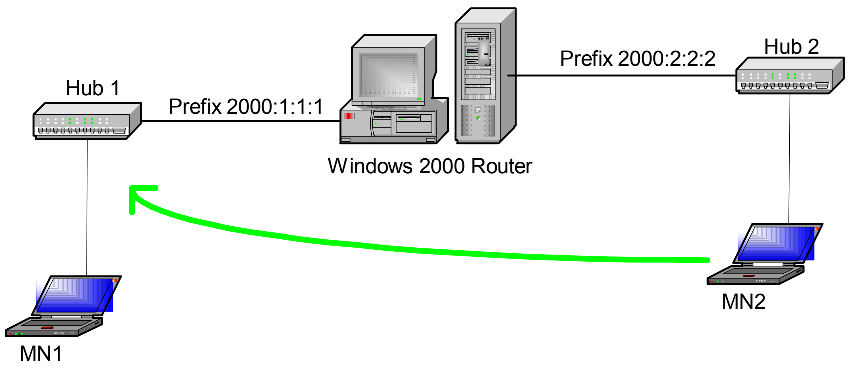
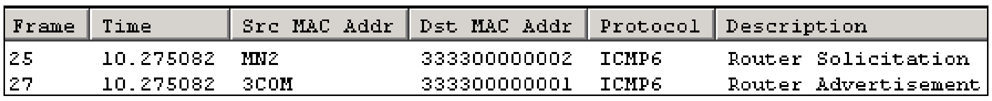


Figure 4.7 – Moving to foreign link.

When MN2 connects to Hub1, it will listen for router advertisements on the 2000:1:1:1 link. As we can see in line 25, MN2 requests an advertisement by sending a solicitation. The correspondent router advertisement is received in line 27 with the included information necessary for configuration.

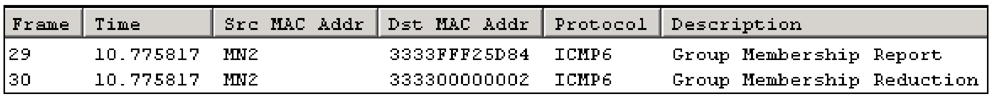
Figure 4.8 – MN2 requests router advertisement by sending a router solicitation message



Frame	Time	Src MAC Addr	Dst MAC Addr	Protocol	Description
25	10.275082	MN2	333300000002	ICMP6	Router Solicitation
27	10.275082	3COM	333300000001	ICMP6	Router Advertisement

The router advertisements are received periodically while connected. The next messages sent are the group membership report and reduction messages (line 29-30). These are used so that MN2 can be a member of the solicited node multicast and the all nodes multicast addresses.

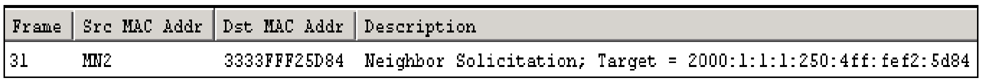
Figure 4.9 – Group Membership Report and Reduction messages.



Frame	Time	Src MAC Addr	Dst MAC Addr	Protocol	Description
29	10.775817	MN2	3333FFF25D84	ICMP6	Group Membership Report
30	10.775817	MN2	333300000002	ICMP6	Group Membership Reduction

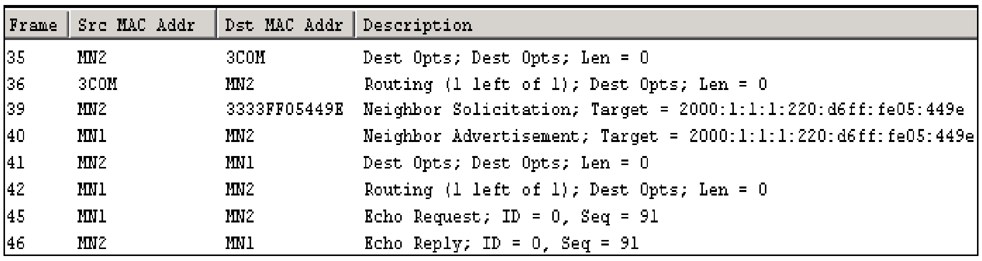
In line 31, a neighbor solicitation is sent from MN2 with the unspecified address as source. This means that the node is performing duplicate address detection. The DAD-check returns success (see section 3.2.1) and the requested target address is assigned to one of the network interfaces of MN2.

Figure 4.10 – DAD check of the target address of MN2.



Frame	Src MAC Addr	Dst MAC Addr	Description
31	MN2	3333FFF25D84	Neighbor Solicitation; Target = 2000:1:1:1:250:4ff:fef2:5d84

The next message of importance appears in line 35. This is an IPv6 destination option message and includes both a HAddr option and a BU option as described in section 3.3.1. This is sent from MN2 to the Windows 2000 router from which it received the router advertisement. The BU option requires a BACK. MN2 also needs to send a BU option to its CNs according to its own local binding update list, and this is sent to MN1 in line 41, which is the only active CN.



Frame	Src MAC Addr	Dst MAC Addr	Description
35	MN2	3COM	Dest Opts; Dest Opts; Len = 0
36	3COM	MN2	Routing (1 left of 1); Dest Opts; Len = 0
39	MN2	3333FF05449E	Neighbor Solicitation; Target = 2000:1:1:1:220:d6ff:fe05:449e
40	MN1	MN2	Neighbor Advertisement; Target = 2000:1:1:1:220:d6ff:fe05:449e
41	MN2	MN1	Dest Opts; Dest Opts; Len = 0
42	MN1	MN2	Routing (1 left of 1); Dest Opts; Len = 0
45	MN1	MN2	Echo Request; ID = 0, Seq = 91
46	MN2	MN1	Echo Reply; ID = 0, Seq = 91

Figure 4.11 – IPv6 destination options and address resolution.

Line 39 and 40 represents the address resolution mechanism in IPv6. The router has detected a new node (MN2) and must send a neighbor solicitation to find its link layer address. As we can see from the solicitation, the target address is the IPv6 address of MN2. MN2 recognizes the target address as its own, and responds with a neighbor advertisement with its link-layer address included.

The IPv6 destination option with an included BACK is received in line 36. This acknowledgement comes from the router and is processed by MN2. When this acknowledgement is received, communication can proceed. As we see in line 45-46, the echo requests reappear with replies, indicating that we also have received a BACK from MN1 (line 42). When these procedures are carried out, regular IPv6 messages traverse the network until MN2 is disconnected from Hub1 in line 78.

Moving To Home Link

Figure 4.12 illustrates that MN2 moves from Hub1 back to its originating home network and reconnects to Hub2. From the capture file we can see that MN2 disconnects from Hub1 between line 78 and 79. Line 78 represents the last echo reply to MN1 when connected to Hub1.

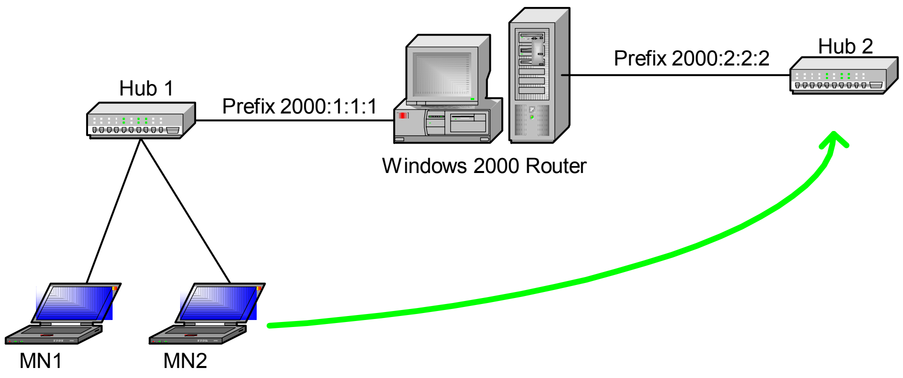
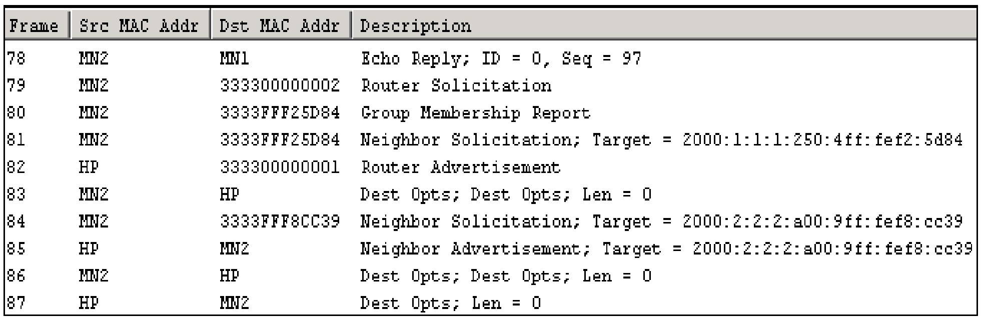


Figure 4.12 – Moving to home link.

When the interval for receiving router advertisements expires, MN2 requests for this by sending a router solicitation (as shown in line 79 in the illustration below). The correspondent router advertisement to this solicitation is received in line 82. As before, MN2 performs DAD by sending a neighbor solicitation with the unspecified source address (line 81). When DAD returns success, the mobile node can send the BU option to the router (line 83). The router performs address resolution with the address of MN2 as the target address (line 84-85).



Frame	Src MAC Addr	Dst MAC Addr	Description
78	MN2	MN1	Echo Reply; ID = 0, Seq = 97
79	MN2	333300000002	Router Solicitation
80	MN2	3333FFF25D84	Group Membership Report
81	MN2	3333FFF25D84	Neighbor Solicitation; Target = 2000:1:1:1:250:4ff:fef2:5d84
82	HP	333300000001	Router Advertisement
83	MN2	HP	Dest Opts; Dest Opts; Len = 0
84	MN2	3333FFF8CC39	Neighbor Solicitation; Target = 2000:2:2:2:a00:9ff:fef8:cc39
85	HP	MN2	Neighbor Advertisement; Target = 2000:2:2:2:a00:9ff:fef8:cc39
86	MN2	HP	Dest Opts; Dest Opts; Len = 0
87	HP	MN2	Dest Opts; Len = 0

Figure 4.13 – MN2 performs DAD when moving back to home link.

The BACK (87) should ideally have been received before the router starts address resolution and, in this case, it has. The timestamps of the messages are the same and are just shown in the wrong order (bug in NetMon, see hole appendix). After the BACK is received, the echo requests from the router on segment 2000:2:2:2 reappear. These echo request messages do actually come from MN1, but are tunneled through MN2's HA. This is also why MN2 does not send a BU to MN1. Its HA performs this task on behalf of MN2 when the node returns to the home network.

4.4.3. WLAN Scenario

The following figure illustrates the test implementation with WLAN as the access technology. According to section 4.2.2 the functionality of MIPv6 should be the same independent of the access technology used, although longer delays and higher error rates are expected with WLAN than with LAN. The figure shows two WLAN cells and each of the Access Points (AP) represents a network segment. AP1 is on the network segment with prefix 2000:1:1:1, while AP2 is on the network segment with prefix 2000:2:2:2. The Windows 2000 router has the same functionality, but the router script is modified to suit the different access points Appendix B.5.

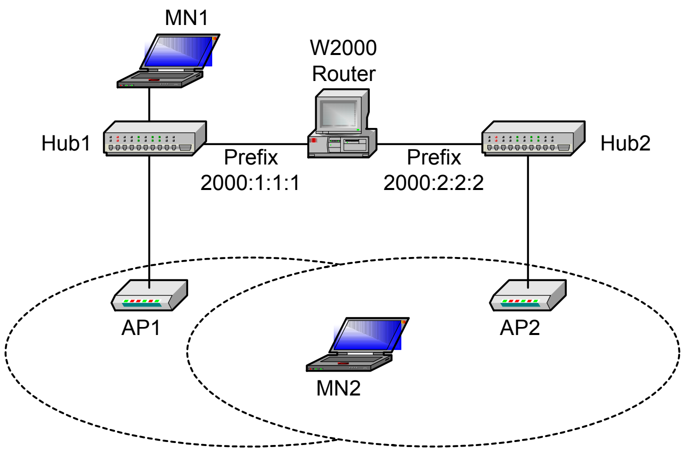


Figure 4.14 – WLAN configuration.

Two different key approaches can be followed in the implementation of a WLAN: an *infrastructure-based* approach, or an *ad-hoc* networking one [29].

An infrastructure-based architecture imposes the existence of a centralized controller for each cell, often referred to as Access Point. The Access Point is normally connected to the wired network thus providing the Internet access to mobile devices. In contrast, an ad-hoc network is a peer-to-peer network formed by a set of stations within the range of each other that dynamically configure themselves to set up a temporary network.

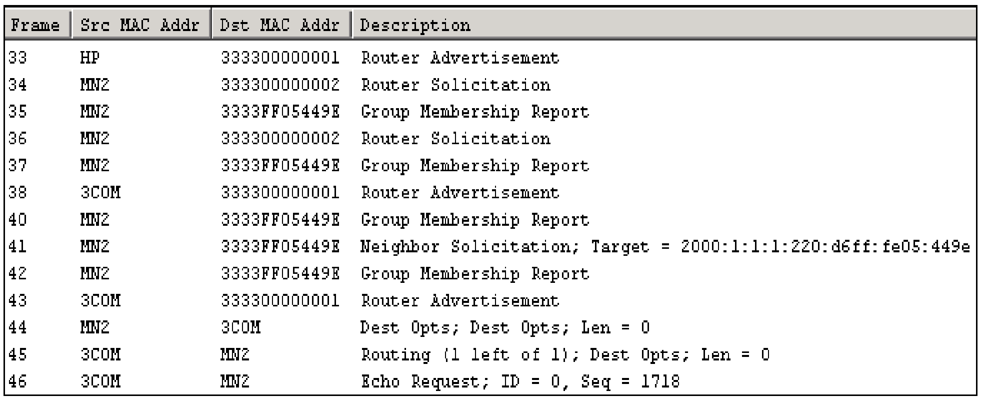
The IEEE 802.11 technology can be utilized to implement both wireless infrastructure networks and wireless ad-hoc networks.

The implementation consists of two infrastructure-based WLAN cells with overlapping radio coverage. The ideal configuration would have been two adjacent cells with minimal overlap in order two demonstrate handover when moving between the cells. This is difficult to implement since both of the access points needs to be connected to the same router. In addition station adapters (such as the mobile nodes) can only be associated with one access point at the time (see IEEE 802.11). When one of the mobile nodes are associated and authenticated with an access point, the specific access point is reset through the use of software in order to demonstrate handover. The mobile node will then associate and authenticate with the other AP and handover between the cells have been carried out.

Moving To Foreign Link

To compare the behavior of MIPv6 when using different access technologies, the same test scenario as used with LAN was configured. This means that MN2 initially was connected to AP2. AP2 was then reset to perform handover to AP1. When connected to AP1, this access point was reset so MN2 returned to its home network (connected to AP2). NetMon was in this scenario also running on MN2 capturing packages when it moved between the two networks. This capture file can be downloaded from [54].

From line 1-33 MN2 is connected to AP2. The first handover (to AP1) is between line 33 and 34 when AP2 is reset. In line 34 and 36 MN2 has detected the absence of router advertisements and sends a solicitation to the all nodes multicast address. The second solicitation is sent because the association of MN2 at AP1 is delayed, but when MN2 is associated (between line 36 and 38) it receive router advertisements from the router sent through AP1. The capture file shows, also in this scenario, that echo request messages are lost when MN2 is disconnected.



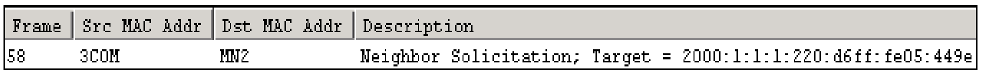
Frame	Src MAC Addr	Dst MAC Addr	Description
33	HP	333300000001	Router Advertisement
34	MN2	333300000002	Router Solicitation
35	MN2	3333FF05449E	Group Membership Report
36	MN2	333300000002	Router Solicitation
37	MN2	3333FF05449E	Group Membership Report
38	3COM	333300000001	Router Advertisement
40	MN2	3333FF05449E	Group Membership Report
41	MN2	3333FF05449E	Neighbor Solicitation; Target = 2000:1:1:1:220:d6ff:fe05:449e
42	MN2	3333FF05449E	Group Membership Report
43	3COM	333300000001	Router Advertisement
44	MN2	3COM	Dest Opts; Dest Opts; Len = 0
45	3COM	MN2	Routing (1 left of 1); Dest Opts; Len = 0
46	3COM	MN2	Echo Request; ID = 0, Seq = 1718

Figure 4.15 – Moving to foreign link with WLAN.

Line 41 shows that the mobile node has received the information broadcasted in the router advertisement and can start the DAD procedure. This is done by sending a neighbor solicitation message with the unspecified address as source because no address have been assigned yet. The target address of the solicitation is the IPv6 address of MN2.

When MN2 has assigned the requested address to one of its interfaces, it can send a BU. This BU is sent to the HA (line 44) according to its binding update list (see section 3.3.2) and requests an acknowledgement. The acknowledgement is received in line 45 and communication with MN2 proceeds as the echo requests reappear in line 46. Further on, router advertisements are received through AP1 periodically and address resolution on MN2 is performed in line 58.

Figure 4.16 – Address resolution with WLAN.

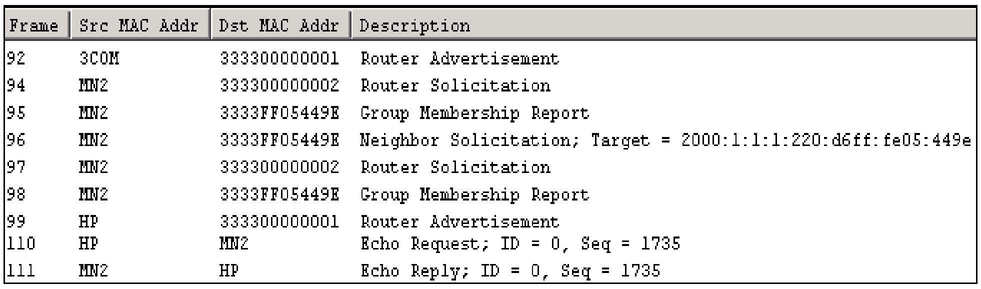


Frame	Src MAC Addr	Dst MAC Addr	Description
58	3COM	MN2	Neighbor Solicitation; Target = 2000:1:1:1:220:d6ff:fe05:449e

Moving To Home Link

AP 1 is reset between line 92 and 94. The last router advertisement received from the router through Hub1 is received in line 92. MN2 detects the absence of advertisements and requests one by sending a router solicitation (line 94 and 97). The advertisement is received in line 99 giving MN2 the information needed for configuration. DAD is then performed with success and MN2 sends binding updates to its CN and HA. Communication is restored as the echo requests reappear in line 110.

Figure 4.17 – Moving to home link on WLAN.



Frame	Src MAC Addr	Dst MAC Addr	Description
92	3COM	333300000001	Router Advertisement
94	MN2	333300000002	Router Solicitation
95	MN2	3333FF05449E	Group Membership Report
96	MN2	3333FF05449E	Neighbor Solicitation; Target = 2000:1:1:1:220:d6ff:fe05:449e
97	MN2	333300000002	Router Solicitation
98	MN2	3333FF05449E	Group Membership Report
99	HP	333300000001	Router Advertisement
110	HP	MN2	Echo Request; ID = 0, Seq = 1735
111	MN2	HP	Echo Reply; ID = 0, Seq = 1735

4.4.4. LAN Behavior

The LAN section (4.4.2) was originally used to minimize the source of errors when testing mobility because LAN is easier to configure and test by means of MIPv6 compared to WLAN, because it is easier to control the movement between network segments. We experienced no errors when testing the stack using this access technology.

By changing the router advertisement interval option and changing between the different modes on router solicitations (were passive and aggressive are the boundaries) [25], it is interesting to see how this affects the handover delays. This is, however, difficult to measure when you must physically disconnect the Ethernet cable from one hub in order to connect it to another. This action represents a variable delay when moving between network segments, with the time granularity so large and variable that the measurements have little value for research purposes. It could have been possible to test time dependent behavior if the MN was connected to a hub that could quickly switch between the different networks segments. This would minimize the variable delay when switching, but there is still impossible to know where in the router advertisements interval the new link are. This influences the delay a MN experiences when changing CoA and testing time-dependent behavior explicitly is therefore omitted.

When using a small router advertisement interval and an aggressive mode router solicitation (were aggressive means that the interval between solicitations are minimized), the communication will reestablish more rapidly when moving the mobile node than when the opposite configuration is used.

4.4.5. WLAN Behavior

For a MN to move from one AP to another when using WLAN, the signal strength from the AP the MN is currently attached to must fall below a certain threshold value. In normal cases this implies that the APs must be at a certain distance so that the signal strength perceived by the MN goes below the threshold value before it tries to connect to another AP. Movement between different network segments based on this method is therefore difficult to control.

By software rebooting the access point in which the mobile node is attached to, the connection is lost for a short period of time and will then reconnect to the new access point. To reset the access points with software, a null-modem cable must be attached to a computer running HyperTerminal. This way you can control the movement the same way as with LAN (see the equipment list in Appendix B.4).

Approach

In our case the scenario did not initially work as intended, and we discovered that the mobile node was provided with a *duplicate address* on the interface to the Breezecom WLAN card immediately after inserting it. Continuous experimenting lead to a *preferred address*, and we were suddenly able to run a successful scenario with a few movements between the network segments in WLAN before the address again was duplicate.

We tried to identify a pattern when, and why, DAD failed, but the failure seemed somewhat arbitrary – the only pattern was that sometimes it worked, but most times it did not. When it worked the number of times the mobile node could move between the network segments were also arbitrary, so we studied the code and the behavior more thoroughly to get a better understanding of how everything functioned. In addition we posted a few questions in a mailing list [31] provided by Microsoft to help identify erroneous behavior in their MIPv6 implementation. Here we made contact with Greg O'Shea in Microsoft Research and discussed how to solve these problems.

We installed NetMon on the mobile node to be able to capture the packets sent and received on the network. We could then study how messages were exchanged between different nodes. Later we started to use DebugView from Sysinternals to catch kernel debug messages from the checked build version of the MIPv6 stack. We also added new debug messages to the code to help us understand how the implementation works and compiled and installed this as described in Appendix C.5.

MIPv6 is supposed to work independent of the underlying access technology, so the WLAN scenario was expected to behave the same way as with LAN. Thus, we suspected that the flaw could be somewhere in the driver to the WLAN card – or it could be a bug in the MIPv6 implementation when loading and unloading this specific adapter (these procedures, *LanBindAdapter* and *LanUnbindAdapter*, can be found in the source file *lan.c* [32]). To detect if the failure was the result of an error in the WLAN driver or a bug in the MIPv6 implementation, another type of WLAN equipment could have been tested, but we did not have these resources available.

Result

Discussing this with Greg O'Shea we decided he should build a new implementation of the MIPv6 stack [32] with a more extensive use of kernel debug messages, which we downloaded from the Microsoft web pages and installed on the MN. We created a new scenario capturing both the packets and the debug printouts using this implementation and carried out the following steps (these files can be downloaded from [54]):

6. Started with the mobile node on home net
7. Stopped the stack: `net stop tcpip6`
8. Cleared down the NetMon and DebugView window
9. Started the stack again: `net start tcpip6`
10. Repeated the following steps until the problem occurred
    a. Allowed things to settle then moved to foreign net
    b. Allowed things to settle then moved to home net
11. Saved the kernel debug log and the capture file

This stack provided many new and informative messages, and we detected several things that implied erroneous behavior in the WLAN driver. We were able to get this scenario to carry out five handovers before it failed as the following figure indicates.

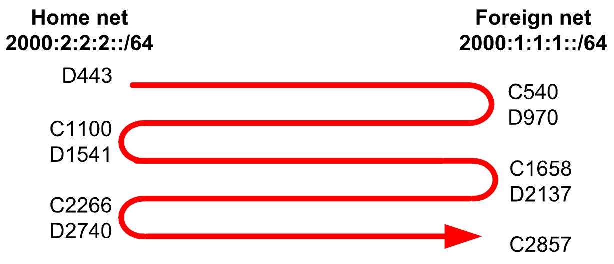


Figure 4.18 – The lines in which the MN connected (C) and disconnected (D) according to the kernel debug message file.

When focusing on driver level issues you must monitor the behavior of other entities as well as using traces of packets going through the networks. The packets exchanged will not show all the concerns because much of the erroneous behavior goes on at lower levels. The following scenario is therefore described using kernel debug messages. By using the implementation downloaded, new and important printouts helped us detecting the problems in the WLAN scenario (some of the lines are shortened to preserve readability).

```
00000000  0.00000000  CreateNTE(IF811a1b88,Addr fe80::(…):449e)->NTE810fa9c8
00000001  0.00036709  LanOpen(81115a88) - media disconnect
00000002  0.00039083  Enter SetIfLinkStatus(IF 811a1b88)->disconnected
00000069  0.00210390  Exit SetInterfaceLinkStatus(IF 811a1b88)
00000070  0.45387975  DADSolicitSend(NTE(810fa9c8), IF(6)  fe80::(…):449e)
00000071  0.45396524  NSReceive(src ::, IF(6) dest fe80::(…):449e)
00000072  1.45516036  Enter IPv6AddressAdded(IF(6), NTE=fe80::(…):449e)
00000093  1.45574005  DADState(4)DADTimer(0) DADCount(0) DADCountLB(0)
00000107  1.58360965  LanStatus(81115a88) - media connect
00000108  1.58364094  Enter SetInterfaceLinkStatus(IF 811a1b88) -> connected
00000162  1.58508442  DADState(2)DADTimer(1) DADCount(1) DADCountLB(0)
00000175  1.58539312  Exit SetInterfaceLinkStatus(IF 811a1b88)
00000176  1.95627601  DADSolicitSend(NTE(810fa9c8), IF(6) fe80::(…):449e)
00000177  1.95636401  NSReceive(src ::, IF(6) dest fe80::(…):449e)
00000178  1.96709358  NSReceive(src ::, IF(6) dest fe80::(…):449e)
00000179  1.96710951  NeighborSolicitReceive: DAD found duplicate!
00000180  1.97092200  enter IPv6 Handoff(IF=811a1b88)
00000201  1.97150755  DADState(1)DADTimer(0) DADCount(0) DADCountLB(0)
00000218  1.97190174  Route RTE 81178d28 fe80::/10 -> IF 811a1b88 released
00000219  1.97194755  Route RTE 81457428 (…):449e/128->NCE 812f8a48 released
00000220  1.97199281  DestroyNTE(NTE 810fa9c8, Addr(…):449e)->invalid(Ref:2)
00000221  1.97205343  NetTableCleanup(NTE 810fa9c8, Addr(…):449e)->destroyed
00000253  1.97282336  exit IPv6 Handoff(IF=811a1b88)
```

Figure 4.19 – The kernel messages show that the interface recovers from DAD failure on link-local scope.

When initiating a stack compiled in checked-build mode, a lot of kernel debug printouts are captured in DebugView. The stack generates an IPv6 address with link-local scope (`fe80::0220:d6ff:fe05:449`) and carries out DAD in line 70 and 71 – even though the media is disconnected.

Figure 4.20 – Values for DAD state is defined in *ip6def.h*

```
#define DAD_STATE_INVALID    0
#define DAD_STATE_DUPLICATE  1
#define DAD_STATE_TENTATIVE  2
#define DAD_STATE_DEPRECATED 3
#define DAD_STATE_PREFERRED  4
```

In line 93 you can see that the DAD state is *preferred* (the value is 4), and after the media is connected the DAD state is reset to *tentative* (see 162).

Next, the DAD sends one neighbor solicitation in line 176, but get two replies from the driver at 177 and 178. So DAD fails according to the specification in IPv6 [7] and the DAD state to this interface address (NTE) is set to *duplicate* (see 201). The user does not, however, notice anything wrong because IPv6Handoff (line 180) clears up the duplicate NTE at 220-221.

Figure 4.21 – The mobile node generates a global unicast address upon reception of router advertisement.

```
00000254  1.97284180  RouterAdvertReceive(IF 811a1b88) - reconnecting
00000255  1.97286303  MIPv6: Adding default router - Lifetime 131070
00000256  1.97291388  AddrConfUpdate: create new addr 2000:2:2:2:(…):449e
00000257  1.97295997  CreateNTE(IF811a1b88,Addr2000:2(…):449e)->NTE81457428
```

After performing DAD, the mobile node receives a router advertisement (line 254) on the home link and generates a new IPv6 global unicast address at line 256 based on the prefix information.

Figure 4.22 – A mobile node detects movement when it receives a new type of router advertisement.

```
00000739  26.50971059  RouterAdvertReceive(IF 811a1b88) - reconnecting
00000740  26.50973210  MIPv6: Adding default router - Lifetime 131070
00000741  26.50978015  AddrConfUpdate: create new addr 2000:1:1:1:(…):fe05:4
00000742  26.50982737  CreateNTE(IF 811a1b88,Adr2000:1(…):449e)->NTE81289a08
00000747  26.99218364  DADSolicSend(NTE(81289a08), IF(6)2000:1:1:1:(…):449e)
00000748  26.99227863  NSReceive(src ::, IF(6) dest 2000:1:1:1:(…):449e)
00000774  27.99392493  DADState(4)DADTimer(0) DADCount(0) DADCountLB(0)
```

When the mobile node receives a router advertisement from another network segment when moving back to 2000:1:1:1::/64, it detects it has moved to a foreign net (line 739). This router it set default and the MN creates a new global unicast address based on the prefix information at line 741. DAD sends a neighbor solicitation at 747 on the foreign net and get one reply from the driver at 748. According to the specification the DAD state is then set to preferred (line 774).

Figure 4.23 – DAD failed on foreign network after the mobile node had connected to the home network.

```
00001219  43.73754150  Exit SetInterfaceLinkStatus(IF 811a1b88)
00001220  44.01676749  DADSolicSend(NTE(81289a08),IF(6) 2000:1:1:1:(…):449e)
00001221  44.01686974  NSReceive(src ::, IF(6) dest 2000:1:1:1:(…):449e)
00001222  44.02920930  NSReceive(src ::, IF(6) dest 2000:1:1:1:(…):449e)
00001223  44.02922690  NeighborSolicitReceive: DAD found duplicate!
00001224  44.03283713  enter IPv6 Handoff(IF=811a1b88)
```

Moving back to the home net DAD is found duplicate the same way as described earlier – it receives two replies at 1221 and 1222 to one neighbor solicitation (line 1220). Note that the prefix is 2000:1:1:1, which is the prefix to the foreign net. In other words it sends one neighbor solicitation and gets back two for the care-of address *after* it has disconnected from the care-of network! This is a clear indication that the problem occurs at the driver level. Once again the user does not notice anything wrong because IPv6Handoff (line 1224) clears up the duplicate NTE.

Figure 4.24 – IPv6Handoff is already executed when DAD occurs and the scenario fails.

```
00002969  105.61910075  enter IPv6 Handoff(IF=811a1b88)
00003032  105.62075655  DestroyNTE(NTE810f89a8,Addr2000:2(…):449e)-> invalid
00003049  105.62121387  NTE(810f89a8) 2000:2:2:2:0220:d6ff:fe05:449e
00003050  105.62124264  IF(6) RefCnt(1) ValidLife(0)
00003051  105.62128148  DADState(0)DADTimer(0) DADCount(0) DADCountLB(0)
00003082  105.62198380  exit IPv6 Handoff(IF=811a1b88)
00003091  106.10594307  DADSolicitSend(NTE(81181768),IF(6) 2000:1(…):449e)
00003092  106.10604951  NSReceive(src ::, IF(6) dest 2000:1(…):449e)
00003093  106.11626701  NSReceive(src ::, IF(6) dest 2000:1(…):449e)
00003094  106.11628377  NeighborSolicitReceive: DAD found duplicate!
```

This behavior reappears a couple of times before it eventually connects to the foreign net and fails at 3094. Recovery is impossible because it has already gone through IPv6Handoff (line 2969 to 3082) that clears up the duplicate NTE after it connected to the link. During the IPv6Handoff procedure it does, however, attempt to delete the NTE that causes the duplication at line 3032, but the *RefCnt(1)* at 3050 prevents this from happening. It cannot delete an NTE while there are references to it. So when DAD is found duplicate this time we are stuck – and this time IPv6Handoff will not come to the rescue.

From the debug messages highlighted above we can see that the order in which the different actions take place is significant for the recovery process when DAD fails. Because there are problems with the driver level when using the Breezecom WLAN cards, our scenario will most likely fail during handover. This particular WLAN driver has problems with its loopback semantics. It cannot distinguish between the packets received on the network. When it sends a neighbor solicitation it can receive two replies from its driver because of this loopback problem.

According to the specification duplicate receptions of such packets will lead to DAD failure, and the handover will therefore also fail.

As described above the DAD can fail even at initiation – that is immediately after inserting the WLAN card or when starting the stack. It may also fail at a later time as described in the scenario above. In fact DAD is failing quite often in the trace. When connecting to a new network segment, it starts with a media connect event which causes DAD to be restarted for all addresses (NTEs). On the reconnecting interface one of two things can happen next. We receive a router advertisement on the interface, or the timer expires and launches DADTimeout (via NetTableTimeout). If DADTimeout runs first then we get an immediate DAD failure, but when the router advertisement arrives we recover from the failure because IPv6Handoff initiates purging of old source addresses. If a router advertisement arrives before DADTimeout then it fails and there is nothing to reset the failed state.

Failure or success therefore seems to depend upon the order in which DADTimeout and IPv6Handoff execute:

- If DADTimeout fails before IPv6Handoff we get away with it
- If DADTimeout fails after IPv6Handoff we are stuck

If we called the function NetTableTimeout at the end of SetInterfaceLinkStatus, which is a function that determines if the mobile node is connected to the link or not, we might get DAD to fail before IPv6Handoff more often and therefore get away with it. This is, however, not a good solution because it would still be a race between the reply (replies) to DAD and the arrival of the first router advertisement, and it will eventually fail. Increasing the router advertisement and reducing the router solicitation interval can also create a more stable scenario, but these solutions do not eliminate the problem – only tries to avoid them.

One obvious solution is of course to fix the problems with the WLAN driver so that it will work under MIPv6. The only problem is that this is something that may occur in other drivers as well, so the specification seems a little fragile concerning DAD. You can find improvements and new solutions to the DAD procedure in chapter 7.

It should be noted that after we detected this problem in cooperation with Greg O'Shea, a description of it is included on Microsoft's web pages [33]. Among other things they say:

"So common and bewildering can such things be that we propose to leave copious tracing in the stack for the meanwhile so that you can more easily determine if your card is badly behaved."

As you can see of the quotation above, they also published a new version of the stack with copious kernel debug printouts – similar to the one Greg compiled and made available to us.

4.5. Application Level Services

To make use of mobile Internet, it is essential to support services on protocols above the IP layer. This includes support for basic communication services as well as high-end user applications. The latter is important in order to provide the existing, and new, functionality to future mobile Internet users. Without new and improved high-end services users will not adopt the technology and there will be little profit involved for the service providers.

The following section describes and demonstrates mobility on the application level and this is important in order to validate that mobility is transparent to network layers above IP.

The IPv6 Technology Preview for Windows 2000 contains a set of services that can be used beyond the standard connection diagnostic tools such as *ping6* and *tracert6*. The technology preview does not support mobility, so this implies combining software from the two branches of the source tree. This can create unstable behavior as it is mixing different versions of some of the modules from the different implementations. A procedure to avoid crashes is described in [34], and involves setting up computers with support for different applications. By using the two configurations found in Appendix C.1 on different computers, it should be possible to run both a successful *telnet* and *http* scenario.

4.5.1. Telnet Scenario

The first application tested was a standard telnet session using a Windows 2000 telnet server and client – configuration details can be found in Appendix C.2. The telnet server running configuration 1 is provided with the technology preview and includes both IPv4 and IPv6 support, while the telnet client must use configuration 2. The tested scenario is illustrated by the following figure.

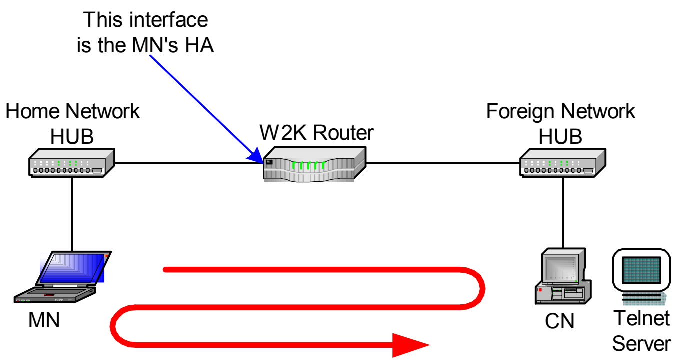


Figure 4.25 – Telnet scenario.

The MN is first connected to its home network and initiates a telnet session with the telnet server (CN) on the foreign network. While the telnet session is still running, the MN moves between the home network and the foreign network. This is a simple command prompt scenario and the MN simply executes commands on the telnet server to prove the mobility. Telnet is run over TCP, and TCP does not allow change of destination address in an active connection. When the MN depicted in the figure moves to another network, the HA is responsible for forwarding the packets received on the MNs old CoA to the MNs new CoA. This keeps the telnet connection active and states that MIPv6 provides mobility that is transparent to higher levels in the protocol stack.

4.5.2. Http Scenario

Telnet was much more common a few years back, but because the http protocol covers some of the same needs, with a user-friendlier interface, telnet has lost some of its market position. It is therefore of current interest to make this service available in a mobile environment. The web server must use configuration 2 and the client must use configuration 1.

To be able to present web pages through IPv6, both the web server and the client browser must support the protocol and the new addressing structure. Today, Unix systems are most frequently used as http servers, while Windows is most used on the client side. Ideally the web server would therefore be implemented using a Unix IPv6 enabled web server and a Windows 2000 MN, but it was instead decided to use Windows with a IPv6 enabled web server to minimize the sources of errors.

In Windows there are currently only a few web servers that support IPv6, and the *Fnord!* web server was chosen when configuring this scenario because it supports IPv6 automatically and is recommended by Microsoft research for the purpose of testing IPv6. To set up Fnord! as an IPv6 enabled web server you need to install the application and run the *FnordCtl* to make a couple of adjustments (set the server up with an IPv6 address and set the path to the web server root where the html documents can be found).

The scenario run to test mobility support in http is similar to the one carried out with telnet as the following figure illustrates.

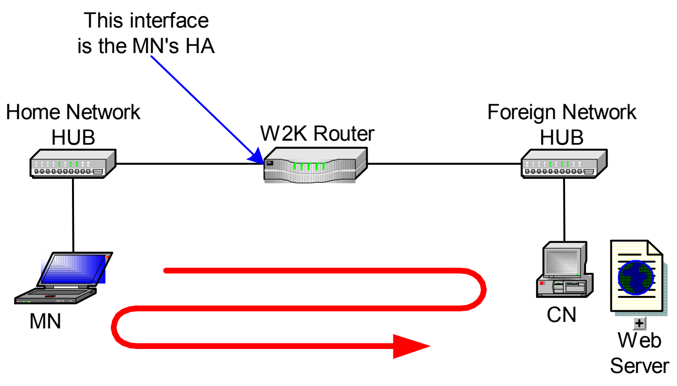


Figure 4.26 – Web scenario.

To show that the MN is connected to the web server continuously, two different web pages are used – see a printout in Appendix D. Both web pages automatically invoke the other every two seconds, so it is possible to watch if the MN stays connected when it moves from one link to another. If it works as intended, the browser will change the web page continuously even when changing network segment. If it does not work as intended, it will stop replacing the web page once it disconnects.

In this scenario it is important to design the web pages so that it is possible to detect whenever the scenario fails. There where some delays in the moment the MN changed link, but it recovered gracefully after a short period of time – similar to what is described in the telnet scenario above.

The IPv6 backbone is called *6bone*. IPv6 provides 340 billionbillionbillionbillion (3.4 x 10^38) internet addresses rather than the current 4.3 billion which are in IPv4. The 6bone is an experimental worldwide network for testing interconnectivity of IPv6 implementations, checking if IPv6 really works or not in actual situations, and so forth. The world 6bone is made up by several regional 6bones. For example, there is a 6bone for Japan region called the WIDE 6bone, which is connected to the world 6bone. Although most of the regional 6bones are made possible by using IPv6-over-v4 tunneling technology, some parts of them are made of IPv6-dedicated leased lines.

A computer set up with configuration 1 can quite easily be connected to the 6bone and access IPv6 enabled web pages by using the procedure described in Appendix C.4.

4.5.3. Behavior

In both the telnet and the HTTP scenario we experienced a couple of seconds delay when moving between different network segments. This was mainly because we physically had to unplug the cable from one hub and plug it into a new one, but the delay was so small that it did not really matter for these two services. When we plugged the cable into the new segment, the stack reconfigured and continued the communication. For real-time services the delay would be too large and other mechanisms are obviously necessary to support such applications. Such mechanisms are described in chapter 5 – Efficient Handovers.

4.6. Evaluation of Implementation

MIPv6 is designed in order to provide transparent and stable mobility for users. The current implementations are, however, still in their initial phases and there are a few more steps that need to be carried out before both these qualities are covered. By that very fact that the specification is not yet set, the deployment speed has also been delayed more than necessary. Product quality implementations are not expected to be released before the beginning of 2002, and the stability of the current implementations are therefore also quite low. This affected our WLAN test scenario and failures due to the driver loopback semantics, in combination with the DAD procedure, where detected. This is the result of how different interpretations of the loopback semantics may affect the stability in MIPv6.

When evaluating the transparency qualities in MIPv6 it seems obvious that this will be provided for higher protocol layers. Currently this will only support best effort services, and for improved service qualities, extensions such as described in the Efficient Handovers chapter are needed.

When developing the handover scenarios, the decision was made to test mobility in a homogeneous environment to avoid interoperability conflicts between different MIPv6 implementations. This decision was taken because mobility can just as well be demonstrated in a homogeneous as in a heterogeneous environment. When MIPv6 is closer to product quality implementations, it will however be very important to test interoperability issues. The same as with the operating system applies to the use of access technologies. Interoperability, by means of handovers for a MN from one type of access technology to another, is very important for the aspect of mobility. It is e.g. not likely that a MN only will utilize mobility between different WLAN cells because of the very limited covering range, but instead visit WLAN hotspots covered by (e.g.) GPRS in between.

Finally, we cannot expect MIPv6 to be deployed worldwide before IPv6 is integrated into the Internet infrastructure. This will, probably sooner than later, be a part of the infrastructure, but the migration costs from IPv4 to IPv6 are huge and this may delay the process. It should be noted that all the large network vendors, including Cisco, supports the deployment of IPv6. Luckily there are also mechanisms that can make IPv4 and IPv6 coexist without reducing the network performance significantly, which can make the transition evolutionary rather than revolutionary.

4.7. Summary

Implementing mobility at an early stage is important to detect challenging issues before the functionality is expected to be commercial available. Microsoft's MIPv6 implementation has most of the functionality described in the MIPv6 draft implemented, even though the stability is somewhat limited. This is due to that it is still a research implementation and we believe that the future commercial releases of this implementation will have the stability needed. Our test scenarios show that MIPv6 can provide transparent mobility on the IP layer

By integrating mobility on the IP layer, new applications exploiting this functionality will probably emerge while other applications will continue to work as before – even when running on MNs because MIPv6 is transparent to higher network layers.

When working with new technology it is important to consider the environment in which it shall run. Installing it in a homogenous environment by means of the platform, operating system and access technology, will ease the interoperability issues but limit the flexibility. A heterogeneous environment would be preferable in a test scenario and this implies testing mobility between different access technologies, such as GPRS and WLAN, to provide so called hotspots. Due to the resources available and the sources of errors that are introduced with this approach we have not demonstrated mobility in a heterogeneous environment. Mobility can, however, just as well be demonstrated in a homogeneous environment as we have shown with our implementation.

Microsoft is currently developing mobility support into their IPv6 implementation, but it is still quite unstable. New versions will subsequently be released and the implementation will continue to improve and the production level reference is for Windows XP early in 2002.

Some basic service must be available to provide mobility support and the implementation embraces two different access technologies – LAN and WLAN. To be able to evaluate if these scenarios are exchanging messages as expected, programs that capture and monitor network traffic can be used. In order to have a more detailed study of the protocol stack, the kernel debug messages can be watched as well. The LAN scenario is quite easy to install, and in our case it worked as is should. It is also easy to control the movement between the different network segments – you only need to plug and unplug the cable between different hubs. In the WLAN scenario is more difficult to control the movement, but resetting the access points can do it. The scenario we tested did not work as intended because of the loopback semantics in the WLAN driver, which resulted in DAD failure. Our LAN equipment did not have these errors and due to this we suspect that other network cards also can experience these problems. We therefore raised the question whether this behavior should be updated in the respective drivers or if DAD is specified too scarce and exposed to failures.

To show that MIPv6 provides transparent mobility for best-effort transport services, we have tested two different scenarios using telnet and http. This, however, involves mixing two different branches of the source tree and makes the stack very unstable. These applications worked as expected and the mobility is therefore transparent to the application levels, but the delays indicate that real-time services will not be supported using basic MIPv6 and other mechanisms that provide lower delays are needed. MIPv6 provides mobility on the IP layer through TCP. We therefore assume that it provides transparent mobility with UDP, since UDP provides IP layer functionality with the exception of support for port numbers, which also is included in TCP.

Modifications in the code can be made and the implementation must then be recompiled and tested. This way you can specialize the stack to different test scenarios, but recompilation of the stack increases the complexity. It also improves the flexibility and more thorough analyzes and improvements can be made. Recompilation is therefore necessary when you need to study the details of the implementation, while the precompiled versions are adequate for testing purposes only.

5. Efficient Handovers

Objectives

- Outline why more efficient handovers than what is possible with base MIPv6 are needed for some applications.
- Evaluate two proposals for more efficient handovers, Fast Handovers for Mobile IPv6 and Hierarchical Mobility Management.
- Outline hierarchical and nonhierarchical approaches from a mathematical point of view.

Contents

5.1. Introduction

MIPv6 is designed to be efficient, but not to a level outside what is expected from best effort services. This is also the service level that we currently are accustomed to on the Internet, and when MIPv6 will be deployed initially, only this service level is anticipated. But, as consumers get used to mobility, the requirements will increase and extensions to MIPv6 are necessary to provide an improved service level, e.g. to support real-time applications. Such extensions are what this chapter will try to illuminate.

Even though mobility support is offered in IPv6, the level of services above the mobility aspect itself is important for the adoption rate. The users will probably not confine themselves to the services that come available to them. They will require the same set of services as when using wired equipment, even when they are in motion.

IPv6 is designed to provide better support for QoS than IPv4, and therefore also a wide variety of applications. These applications are classified according to their requirements for data. Some may function even if they do not receive any new data for a long period of time, while others are classified as real-time applications and may fail if they do not receive new data within milliseconds. Even though most applications are quite tolerant to packet loss, it is important to design mobility in such a way that also other less tolerant applications when considering packet loss are supported.

QoS is, however, a very comprehensive term and a research area of its own. It will therefore not be elaborated in this thesis, but a service model is presented below in order to get an overview of how applications can be classified.

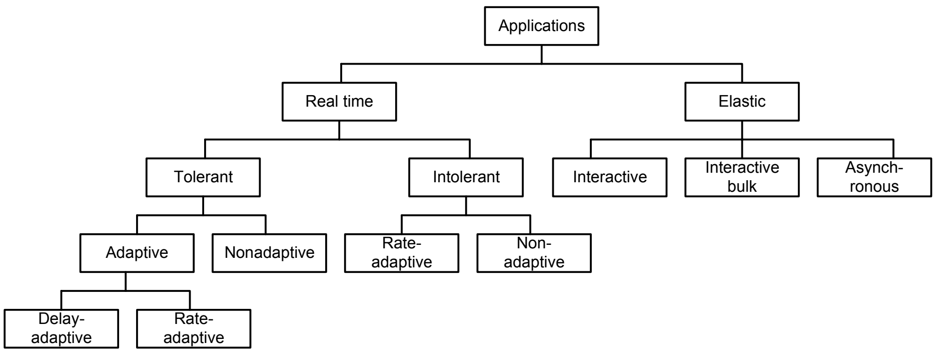


Figure 5.1 – Taxonomy of applications.

These application classifications are described in [36] and should be read for additional details. Elastic applications require typically best-effort services, which is the service level available on the Internet today, and are therefore supported using the base MIPv6 specification [2]. Real-time applications are more sensitive to delays and require more predictable handover qualities, and how to increase the support for such applications are elaborated in this chapter in order to attain efficient handovers and reduce latencies when moving between different network segments.

Several draft documents are proposed in IETF to make handovers between network segments more efficient. These are additions to base MIPv6 and, consequently, these solutions often tend to fall back to the base MIPv6 specification as the worst-case scenario when their preconditions fails. The following two drafts will be elaborated:

- Fast Handovers for Mobile IPv6 [37]
- Hierarchical MIPv6 mobility management [38]

There are a couple reasons for choosing these two drafts. First of all they reflect two different schools of thinking – that is deploying efficient handovers in a hierarchical and in a nonhierarchical manner. They are also currently on the agenda in the IETF, both in the mailing list for mobile IP [39] and at different IETF meetings [40]. A flat deployment is where the MN have to send BUs to the HA and the CNs on every movement between network segments. This is similar structure to what is defined in the base MIPv6 specification. In a hierarchical deployment a MN moving within a local hierarchy needs to send BUs to the same nodes as in the nonhierarchical approach when moving out of a top-level router. When moving within the hierarchy only a BU to the top-level router has to be sent.

5.2. Fast Handovers for Mobile IPv6

Fast handovers for mobile IPv6 is a draft document that we believe will get a high level of acceptance among developers. This is because it addresses several techniques to reduce the handover latency covering a wide range of handover scenarios. So far it has also proven to be adaptive since it has frequently changed to capture the latest progress within this area of research.

The document specifies protocol enhancements to MIPv6 that enable MNs to more quickly become connected at new points of attachment to the Internet, when deploying a flat structure. These protocol enhancements are intended to minimize the time during which the MN is unable to send or receive IPv6 packets due to handover latency.

5.2.1. General Architecture

The following figure illustrates the different network entities that are a part of this proposal for efficient handovers.

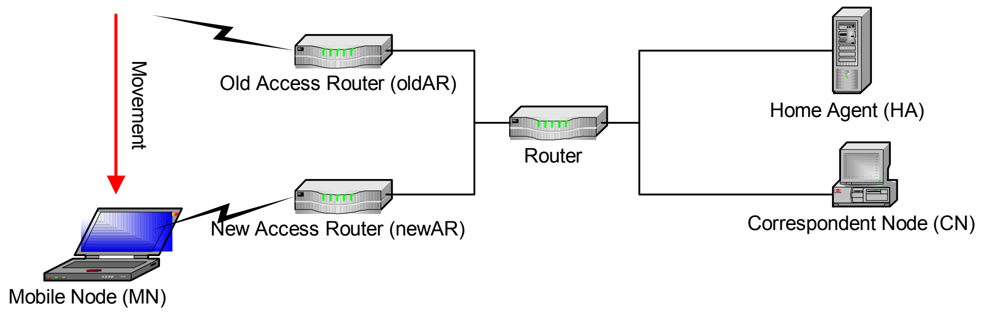


Figure 5.2 – Network elements in fast handovers for MIPv6.

The aim of this specification is to enable the MN to configure a newCoA before it moves to a newAR in a way that it can use this newCoA immediately at connection with newAR. In this respect it will enable fast handover behavior with minimal interruption to packets flowing to a MN as it moves between routers (network segments). The network elements introduced are described in Terminology, page 133.

The design decision has been taken not to consider scenarios in which the handover cannot be initiated until the mobile node has layer-2 connectivity to the newAR. Since the specification deals with layer-3 issues, the handover is considered to require some layer-3 information relevant to the newAR, specifically a newCoA. In parallel, the acquisition of this newCoA should be performed in a way that a duplicate or invalid address cannot be picked (and in the rear case that it does the system is able to recover gracefully). Other scenarios are covered adequately by the base MIPv6 specification and are therefore not included in the specification.

This model applies both to scenarios in which the network and the MN determine that a handover will take place. The model also preserves the mobile IP characteristic of having the MN making the ultimate determination of whether traffic destined to it is diverted to its new point of attachment.

5.2.2. Mobile-determined Handovers

Fast handovers are implemented by adding four new messages and by modifying one of the existing MIPv6 message options. These are implemented between ARs and between the AR and the MN as illustrated in the following figure. This section captures a specific scenario where the MN initiates the fast handover procedure, a mobile-determined handover, but a network-determined approach is also possible (see section 5.2.3).

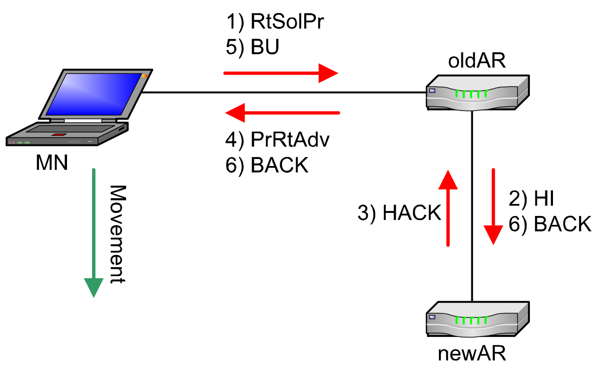


Figure 5.3 – Basic message flow with the fast handovers specification.

If the MN is connected to oldAR and it decides to switch to newAR for a better connection, it can initiate a fast handovers procedure. This can be done to obtain best possible level of service for its applications when moving from one network segment to another, e.g. if one of the applications is a video conference.

To initiate a fast handover the MN sends a Router Solicitation for Proxy (RtSolPr) message (1) to the oldAR containing the link-layer address to the new attachment point. The oldAR will then send a Handover Initiation (HI) message (2) to the newAR with the corresponding link-layer address in order to start the handover process. The purpose of this message is to notify the newAR about the upcoming handover and establish a valid newCoA for the MN before it moves to the new network segment. The response from the newAR is a Handover Ack (HACK) message (3), which can include a valid newCoA for the MN. This information is then passed on from the oldAR to the MN using a Proxy Router Advert (PrRtAdv) message (4). Finally, the MN sends BU (5) to the oldAR so that it can start forwarding packets destined to it on the new network segment. A BACK (6) can then be sent while the MN is still connected to the oldAR or after connecting to the newAR.

The oldAR will therefore receive a HACK message indicating whether the newCoA is valid or not. If valid, it prepares to forward packets to the newCoA, if not, it prepares to forward them to the newAR. The routing will only be changed after the oldAR receives a BU from the MN confirming the handover and the oldAR can then start forwarding packets destined to the oldCoA. These messages will be forwarded either to the newCoA or to the newAR, depending on the value in the HACK message.

The MN must not use the newCoA address as source until it receives a BACK from the oldAR. As mentioned, the BACK may be received while the MN is still connected to the oldAR, in which case the MN will only have to announce itself to the newAR to get full connectivity. In the case were the BACK was sent but not received at the oldAR, there will be a copy of it waiting for the MN at the newAR. If the BACK is not received at all, the MN should assume that the BU was not received by the oldAR and must retransmit the BU to the oldAR.

A detailed description of these messages is included in section 5.2.4.

5.2.3. Network-determined Handovers

A network-determined handover is slightly different than a mobile-determined one, as illustrated in the following figure.

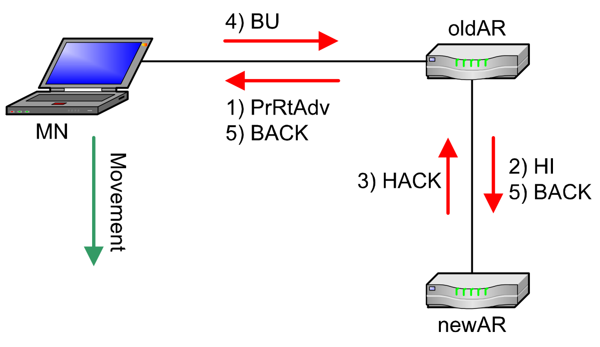


Figure 5.4 – Network determined handover with stateless address autoconfiguration.

If the oldAR realizes that a MN must move to the newAR, which is configured using stateless address autoconfiguration, it compiles a newCoA based on the MN's interface ID and the prefix to the newAR. The newCoA is then sent to the MN together with the newAR's IP address and its link-layer address using the PrRtAdv message (1). At the same time the oldAR sends the HI (2) message to the newAR indicating the MN's oldCoA as well as the recently created newCoA. The rest of the messages are exchanged as described in the mobile-determined handover scenario.

If the network is configured with a stateful approach, an alternate message sequence is utilized as illustrated below.

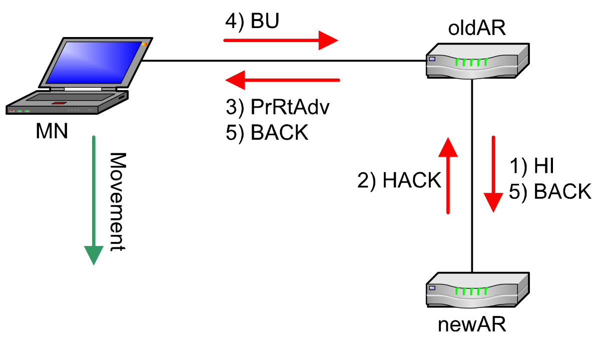


Figure 5.5 – Network determined handover with stateful address autoconfiguration.

The HI/HACK exchange precedes the PrRtAdv message sent from the oldAR to the MN to be able to retrieve the correct contents from the newAR, when this information is not available to the oldAR by other means. The rest of this process is identical to the stateless approach.

5.2.4. Messages

The messages described in the scenarios above are necessary to carry out fast handovers, and their functionality is essential to understand before evaluating the proposal (see 5.3).

- **RtSolPr** – Hosts send this message in order to prompt routers for PrRtAdv. The mobile node must send this message in order to initiate a fast handover and indicates its destination with the new attachment point link-layer address option. This option contains the link-layer address or another identification of the attachment point the MN attempts handover to or requests routing advertisement information for. A PrRtAdv message should be received in response within a short period of time. If not, the MN should resend the request. If the PrRtAdv is not received by the time the

MN disconnects from oldAR, fast handover cannot be performed and the MN should revert back to normal MIPv6 behavior.

- **PrRtAdv** – Routers send out PrRtAdv messages unsolicited if the network is controlling the handover or in response to a RtSolPr message. There are three possible paths from the PrRtAdv message.
    - If this message does not contain a newCoA, it means that the MN should construct a new address out of its interface ID and the prefix information included in the message. If a newCoA is included, then the MN should use that specific address at the newAR. This must be used when stateful CoA configuration is in use, but it can also help all involved parts to calculate the same newCoA when stateless address autoconfiguration is used.
    - If a handover to the new attachment point does not require change of CoA because the access points are within the same network segment, then this is indicated in the message.
    - The oldAR may not be aware of the prefix information requested based on the link-layer information. In this case the MN must give up its attempt to perform fast handover to the new attachment point and may continue using normal MIPv6 instead.
- **HI** – Handover Initiation message is a new message sent by the oldAR to the newAR to initiate the process of MN's handover from one network segment to another. The purpose of this message is to notify the newAR about the upcoming handover and establish a valid newCoA for the MN. Information about the newCoA for the MN will also be included the same way as described in the RtSolPr message. If HACK is not received as a response to this message, then the HI should be resent using a short retransmission timer. If this fails it means that no fast handover can be performed.
- **HACK** – This message must be sent by the newAR to the oldAR in response to HI. The newAR should include the newCoA and information that a new address is assigned, or it can check the validity of the newCoA sent with the HI and reply accordingly. If valid, the newAR should insert the newCoA in its neighbor cache and defend it on behalf of the MN for the period of time expected to use to connect to the newAR. If it is invalid, the newAR should insert a host route for the MN's oldCoA pointing to its mobility interface, which is the interface the MN is expected to connect to when moving to the newAR. The newAR can always refuse the handover, e.g. because of accounting or authorization or even if it has insufficient resources.
- **BU** – This option is defined in MIPv6, but will be modified by adding two flags for supporting bicasting and buffering in the ARs, which they may honor or reject silently. The MN must send the BU so that the oldAR can redirect traffic to it by the way of the newAR. The message should be sent while the MN is still connected to the oldAR and a BACK should be received at the same place. If that is not possible due to missing link layer connectivity, the BU must be sent or resent at the newAR and the BACK must be received before the MN can use the newCoA. The initial retransmission timer for the BU in this particular case should be very short after the MN connects to the newAR. If retransmission does not yield a BACK, the MN must give up the attempt for a fast handover and revert back to normal MIPv6.

Bicasting refers to the process of duplicating packets and sending them to the MN at both its previous and new point of attachment. This is semantically allowed by IP because it does not guarantee packet uniqueness, and higher-level protocols are assumed to eliminate duplicates whenever that is important for the application.

Buffering can also be used to improve performance for a MN, one may use buffering at either the newAR or the oldAR (or both) to avoid dropping packets during the time, which a MN may be disconnected from both network segments [41].

5.2.5. Performance

Some mechanisms may affect the performance described in the specification and may add delays to the MIPv6 handover. One of them is DAD, described in chapter 7.

The chance that a duplicate address will be generated can be considered very low because the address (in the stateless approach) is generated through the use of a prefix and an IEEE EUI-64 address, but it cannot be ignored. There are, however, possible for an AR to keep a list of all the nodes connected to a network. The addresses can then be checked for validity and the result can be attached in the HACK message to the oldAR. If no such information were available, the MN would have to perform DAD on the new link, preferably before connecting to it to avoid such delays.

5.3. Evaluation of Fast Handovers

Fast Handovers for Mobile IPv6 are currently an important subject of discussion in the IETF mailing list for mobile IP and is changing rapidly to reflect the latest progress within this area of research. Thus, the technology is new and there are several aspects that need to be investigated before the proposal can move from draft to RFC status [42].

Some of these issues will be elaborated here and it is essential to have a thorough understanding of how the fast handovers procedure works before reading this section.

5.3.1. Acquiring newAR Identification

If the MN wants to initiate a fast handover it sends an RtSolPr message to the oldAR indicating that it desires to carry out the fast handover procedure to a new attachment point. The message will contain some form of identifier to specify the new attachment point it requests information for, either the link-layer address or some other identification.

In this draft the design decision has been taken only to consider the scenarios in which layer-2 connectivity to the newAR is present before moving from the oldAR. This information will be used to identify the newAR when a handover is initiated. For the MN to be able to retrieve link-layer information to the new point of attachment while it is connected to the old, wireless access technologies are most relevant for the fast handovers scenarios – such as WLAN used in our implementation, section 4.4.3.

The MN can in a wireless environment detect other access routers by investigating accessible layer-2 radio interfaces in the air – that is available radio signals the MN can perceive from its current position. These signals are most likely overlapping so the link-layer address to the newAR can be obtained while the MN is connected to the oldAR, which is consistent with the design decision taken.

The MN is also considered to require some layer-3 information relevant to the newAR in the handover procedure, specifically a newCoA and the new network prefix. How the MN can obtain this layer-3 information from the newAR before connecting to the new network segment is not discussed in the specification, but several solutions are possible:

- Insert layer-3 information on the link-layer so that the MN can address the newAR by its global unicast address.
- Use a neighbor protocol so that all access routers knows about their immediate neighbor ARs – both by link-layer and global unicast address.
- Let the ARs maintain a list of link-layer addresses with corresponding global unicast addresses based on previously successful handovers.

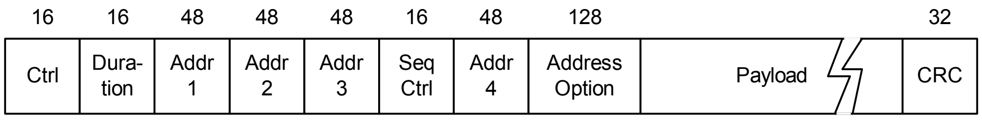
These solutions will be explained in the following sections.

Layer-3 Information on the Link-layer

According to the specification one of the presumptions is that the MN is allowed to access link-layer information from the new attachment point. Assumed that the MN needs to be authenticated in order to access layer-3 information, the link-layer can include the global unicast address instead of retrieving this information through RAs. This address can then be used as the identification to the newAR in the RtSolPr message.

Using the 802.11 frame format as an example, it can be modified to include the 128 bits address option in the start of the payload as illustrated below.

Figure 5.6 – Modified 802.11 header format.



Note that IPv6 should be independent of the underlying protocols and that information from lower layered protocols therefore not should be necessary to make it work. In this approach, however, the MN can utilize link-layer information to improve the efficiency, but it does not depend on it to work properly. Note that MNs can still move between ARs without this information on the link-layer by falling back to base MIPv6 functionality, but then it cannot perform a fast handover.

If the oldAR is unaware of the address to the newAR then the MN can provide the necessary information for them to establish a connection. The MN knowing the global unicast address to the newAR instead of only the link-layer information will, for the oldAR, eliminate the problem of knowing the link-layer addresses to its neighbor access points. The handover latency can therefore become quite predictable, even if the involved ARs are unaware of each other.

Neighbor Protocol

A neighbor protocol can also be used to make the oldAR aware of the newAR. The protocol must handle the exchange of link-layer addresses between ARs that are considered to be neighbors. They are neighbors if a MN can move directly between them without having to go through a transient AR. In this approach the MN will use the link-layer option as an identifier to the newAR as defined in the RtSolPr message according to the fast handovers specification.

If these ARs were directly connected to each other, this protocol would have to make sure that they know the link-layer address of their immediate neighbors (that is requesting all available links and set the HopCount in the IPv6 header to 1). The following figure illustrates directly connected ARs where AR1 will have to know the link-layer address to AR2 and vice versa.

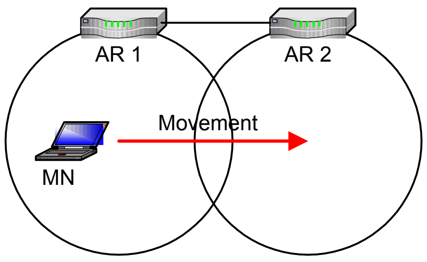


Figure 5.7 – Movement between directly connected ARs.

The problem is that all ARs are not necessarily directly connected to each other. There could actually be a substantial topological network structure between the different ARs as shown in the following figure:

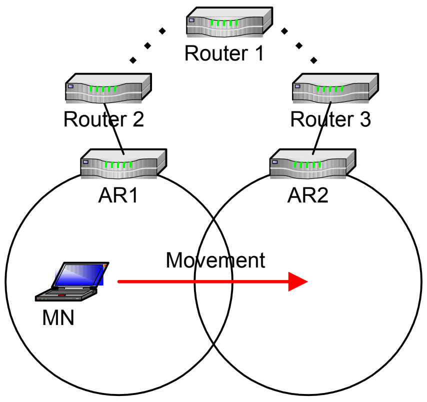


Figure 5.8 – Movement between ARs connected in a hierarchy.

This will increase the complexity of the protocol because the ARs must use some form of detection algorithm to identify which ARs that are neighbors. How the physical topology of the network looks like, and the coverage area to the different ARs, are of course difficult to determine. The protocol will be simpler by letting the ARs know all link-layer addresses to the other ARs below a certain level in a hierarchy – in the figure above a natural choice for the top level of the hierarchy are below Router 1. The number of ARs within a limited hierarchy is in most cases so few that it will not be too much information to keep track of for each AR.

Successful Handovers History

It will be easier for ARs to implement a similar functionality, that is getting to know its neighbor access routers, by registering a list of successfully accomplished handovers. When the first MN moves from one specific location to another, it may not be possible to carry out a fast handover procedure because the oldAR does not know the link-layer address to the newAR. After the first handover, the link-layer addresses to each other are cached in both involved ARs, providing the information necessary for a fast handover the next time a MN is about to move between these links. The cache entry can be set valid for a predefined period of time, which is updated in every handover, preventing the ARs from saving unnecessary information if the entry is out of date (due to restructuring of the network topology etc.).

Evaluation of newAR Identification

Integrating layer-3 information on the link-layer provides the MN with the global unicast address to the newAR. The oldAR does therefore not have to be aware of the newAR in advance and this reduces the complexity in the AR architecture. If the MNs are not aware of this integration, they are still able to use the network but not capable of performing fast handovers. Anyway, this will still violate the separation between the access technology and the IP layer, which is one of the reasons Internet has been so widely deployed.

Defining a neighbor protocol seems at first glance like a good solution in order to exchange information between neighbor ARs so that fast handovers are possible. Such a protocol will be simple if all ARs are connected directly to its immediate neighbors, but this is not possible in parallel when preserving a network topology for the purpose of efficient routing. Complicated protocols and several messages will have to be exchanged to be able to achieve this functionality, and this will add more network load as well as increase the complexity.

A better solution is probably the last approach were a cache is used at the ARs to keep a history of previously successful handovers. The limitation of having to perform one normal handover before fast handovers can be carried out should be acceptable if there are many MNs using the network. In smaller networks with less traffic load, this may lead to that a majority of the handovers will be normal instead of fast – and this is clearly not the intention of the fast handover procedure. To deal with this in smaller networks, the administrators can preconfigure the ARs with some neighbor link-layer addresses that are permanent, or the timeout value for the dynamic cache entry could be set very high. This solution is, however, not very scalable because the administrator has to keep track of this manually.

5.3.2. Message Improvements

The messages used to carry out fast handovers may be improved due to reducing the number of messages exchanged, providing more stable handover latencies and modifications allowing future growth. Other changes that can be done, especially in order to simplify the specification and reduce the overhead, are not discussed in this section, but Appendix E elaborates several of these aspects.

Flag Utilization in BUs

Two flag bits are added in the binding update option. The B flag is set if the MN would like the AR to do bicasting, and the U flag is set if it would like it to do buffering. BUs are sent to the oldAR so that it can register the newCoA when the MN is about to move to the new network. The draft also specifies that the ARs may honor these requests, or reject them silently.

When a BU is sent to the oldAR, the traffic will be redirected to the MN by the way of the newAR. Since a fast handover procedure has been initiated, the MN would like the handover to be fast with minimal packet loss. Consequently, both the B and U bit will probably be set by the MN to improve the handover quality.

Both bicasting and buffering are important to improve the quality of the fast handovers. The reason for these efforts to be voluntarily implemented in the ARs, or else be silently ignored, is therefore vague. The ARs supporting fast handovers can implement this functionality to help speeding the process, but as long as this is not mandatory, no time guarantees can be given. By specifying that these services must be supported in the ARs, the handover qualities will improve and the MNs can expect a better overall performance.

If the MN asks for these services and the AR cannot do as requested, either because the services are not supported or due to lack of resources, the AR should not silently ignore this. If the ARs do not support these services it should indicate so in the BACK so that the MN can take the necessary precautions in case of packet loss. This was also the general opinion after discussing this in the IETF Mobile IP Working Group mailing list [39].

If all ARs supporting fast handovers implements this functionality, then there are few reasons for the MN to explicitly set these flags in the BU option. The MN must however, one way or the other, request for these services because the timer in the BUL must be aligned with the lifetime in the BC entry in the AR according to the lifetime of the BU. This is because the newAR must defend the new address on behalf of the MN for a certain period of time. By setting these flags, the services are asked for explicitly, but the lifetimes could probably also be aligned implicitly, e.g. at the time the BU option is received.

If the support stays optional, then most MNs will probably set the flags anyway to try to achieve the best possible performance. The need for these flags will therefore probably diminish and the usage should be evaluated in future releases of the fast handovers specification.

Single BU Transmission

A MN moving from one network segment to another using the fast handovers procedure must send BUs to the oldAR, the HA and the CNs with information about its newCoA to exploit the route optimization capabilities. The oldAR and the HA must be updated explicitly as the following figure illustrates.

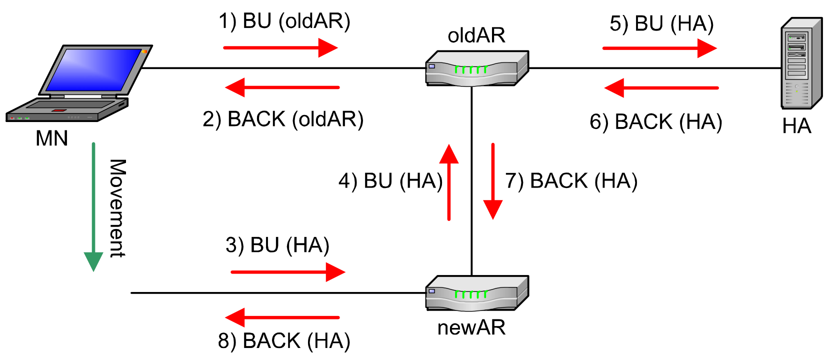


Figure 5.9 – Messages exchanged with basic BU structure.

This approach is currently debated in the IETF mailing list for fast handovers in mobile IP, and one suggestion is to use only one BU to update both the oldAR and the HA about the newCoA. A single BU message will in this case be sent to the oldAR, which registers the newCoA and forwards the packet to the HA as shown in the following figure.

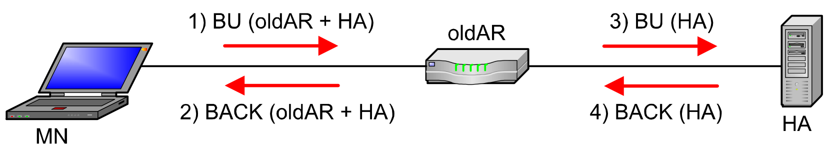


Figure 5.10 – Messages exchanged with a single BU

Some consequences of utilizing such a mechanism will be outlined in the following paragraphs, but a MN cannot receive a BACK (2) from the HA before the oldAR has received a BU from the HA itself. The MN should ideally receive the BACK before disconnecting from the oldAR, but the oldAR cannot wait for the BACK from the HA (4) if such a message is delayed. This is because once the oldAR receives a BU from the MN, it will know that the MN is about to change to another network segment. A solution can therefore be for the oldAR to answer with a BACK on behalf of the HA and itself. If the oldAR does not receive the BACK from the HA, it must notify the MN via the newAR, and the MN should then be responsible of sending the BU.

This approach will reduce the network load because the number of messages transferred will be reduced, and this is important when looking at wireless links with limited bandwidths. The MN will in this case only need to send the BU to the oldAR, and not to the HA. The oldAR can then decide whether or not to send the BU to the HA depending on how long the MN have stayed on the link, which can reduce the number of BUs sent to the HA.

The oldAR should take into consideration how long the MN have stayed on the link before forwarding the BU message to the HA. If it has only stayed for a short period of time, this may generate more network traffic than necessary. In several situations, the MN would probably be best suited to make this decision itself. If there are many MNs in a network, the ARs may be exposed to a lot of extra workload if it shall control this for every MN, thus reducing the scalability. By increasing the workload on the AR the stability is also reduced because it introduces a single-point-of-failure. The scenario may also add further complexity due to the new scenarios that can fail and must be treated, e.g. the AR may go down and the MN can therefore loose packets destined to it from the HA or CNs. This solution will therefore reduce the number of messages exchanged (as long as successful behavior is assumed), but also add further complexity to the scenario.

Specialized BU for Fast Handovers

The draft document indicates that a future release of it may contain new types of message options, namely Fast Handover BU (FBU) and Fast Handover ACK (FBACK). These are not included in the current specification, but will probably define that the oldAR only can change the route to the MN if *both* FBU from the MN and HACK from the newAR are received.

In the MIPv6 specification as well as in the current fast handovers draft, only one single BU message is necessary to change the routing information in the oldAR to the MN. A new message option type for the BU is therefore necessary for the routing change to occur after the oldAR receives both messages.

An obvious advantage of using this approach is that the routing information in the oldAR cannot change before both the MN and the newAR agree to the movement. When both parts agree, they are able to make the necessary preparations to accomplish the handover as fast as possible.

Introducing a new message type option instead of extending the existing renders several new features. The current draft document specifies an extension in the BU message by adding two more flags. This leaves only two free flag bits for future extensions to the BU message to use in all future extensions of MIPv6, and related technologies using these features, so the introduction of new message options may therefore be necessary. This will not only make modifications to the standard BU message option easier, but also help future additions to the fast handovers procedure by only having to change the FBU/FBACK messages.

This extension will, however, add extra complexity to the whole handover scenario because the ARs supporting fast handovers must keep track of several message types, depending on whether the MN uses standard MIPv6 or the fast handovers algorithm. Extending the standard BU message option may reduce the complexity because it is integrated into the existing message infrastructure. Small modifications to support some new services in the AR are necessary, and it will also quite easily coexist with the base MIPv6 implementation. ARs supporting only these new services can at least perform a standard MIPv6 handover when MN requests a fast handover. This kind of “backward compatibility” is not possible when utilizing a new BU option (FBU) because the base MIPv6 implementation does not know how to process the new option types. The MN must then revert to the base MIPv6 procedure to carry out a handover, which will increase handover latency.

ARs supporting the new option types can also implement some kind of queuing technique to give MNs performing a fast handover higher priority than the MNs using the base implementation. This will of course increase the performance when fast handovers are carried out, but the other nodes may suffer a lot if an unfair algorithm is chosen. “Fair” algorithms will not, however, be elaborated in this thesis.

5.3.3. Movement Detection

The fast handovers procedure can improve its performance by implementing new elements for movement detection. A MN can use any combination of mechanisms available to detect that it has moved from one link to another, and this can affect the total performance. At the extreme, the handovers can be totally determined by the network or by the mobile node. Solutions somewhere in the middle can of course also be used.

One possibility is to exploit the movement detection from the base MIPv6 implementation. If the MN does not receive the router advertisement, it assumes that this default router is no longer reachable and decides to switch to another router from which it may currently receive router advertisements. This approach is very simple and will generate few error conditions, but is not very flexible when it comes to support fast handovers. For this purpose more aggressive approaches are needed.

Layer-2 Triggers

Most of the documents discussing procedures for efficient handovers suggest using layer-2 information where available to initiate the handovers, and this also includes the fast handovers procedure. If this kind of information is not accessible, the MN should fall back to the base MIPv6 implementation. Under the circumstances where the link-layer does not provide this information, the handover latency will increase and other kinds of triggers are necessary to achieve a fast handover.

The layer-2 triggers indicating that a MN should move from one link to another in a wireless environment are usually based on the signal strength received. If the level reaches a certain threshold level, the MN will search for new alternative APs with better signal strength. The threshold value does not usually alone determine if a MN should move from one AP to another – also the current load on the different APs should be taken into consideration.

Layer-3 Triggers

A number of fast handover schemes currently proposed in the IETF Mobile IP Working Group require a handover trigger specified by the link-layer events. These layer-2 triggering events are interpreted to initiate the mobile IP handover at the network layer (layer-3). Although a handover solution should not depend on the wireless access technology, it is an obscure task to universally specify the layer-2 triggering events from different access technologies. Furthermore, one cannot expect layer-2 triggers from every wireless device. To avoid these air-interface and layer-2 diversity issues, a generalized layer-3 methodology or layer-3 triggering mechanism that is independent of the access technologies can be developed [43]. These layer- 3 mechanisms are beyond the scope of this document and will not be elaborated here, but the concept of using triggers on this layer contra on the link-layer will be discussed in general terms.

Evaluation of Triggers

One obvious advantage by using layer-2 triggers is that it can exploit functionality present on a lower protocol layer. This is efficient because the triggering mechanisms will only be executed once, not both in layer-2 and layer-3, and implementations are in most cases more efficient when running in lower protocol layers. Lower layers are executed before higher ones in the protocol stack and these issues can therefore be treated immediately when the layer-3 implementation starts executing. No new algorithms will make layer-3 more complicated and no new messages will have to be exchanged – this can lead to very efficient layer-2 trigger implementations.

In a homogenous wireless environment where the link-layer access technologies have become more or less a *de facto* standard, the layer-2 triggers can prove to be more efficient than layer-3 triggers. It is likely that some access technologies will become more common than others, such as Ethernet today in LANs, to enable the MNs to move freely around without having to support several of these technologies. This is also more cost effective for the users because network adapters can be produced in large quantities, thus reducing the total cost. The Internet providers may also reduce their costs because the number of base stations that have to be installed will be reduced, and this may again, lead to lower expenses for the end-user. Examples of access technologies that may become de facto standards are UMTS and WLAN (or HIPERLAN). New PC network cards with support for these technologies will probably be developed, and a seamless handover between these access technologies are probably a functionality that the users want. Considering this scenario, it will probably be more efficient using layer-2 than layer-3 triggers.

If changes in the access technology should not affect layer-3, a faster deployment of new access technologies are possible, but not at the level that can be expected for a layer-2 implementation. In general, layer-2 triggers will probably be faster than similar implementations in layer-3 because they are specialized to work on a specific access technology. The handover latency is however not necessarily the only consideration when making this design-decision. The layer-3 triggers may in fact be a bit slower in general, but the delay will be more predictable because it does not depend on the nature of the access technology.

Layer-3 triggers will therefore be more flexible in a heterogeneous environment than layer-2 triggers. A broader handover concept will make inter-access technologies easier, and also changes to newer and improved access technologies will be better supported. In other words, a layer-3 trigger solution will be more flexible than a layer-2 one when it comes to support for different access technologies because it is independent of it. It would also be difficult to specify all possible layer-2 triggers from every type of wireless device.

As a final observation it is important to know that the layer-3 triggers will increase the IP layer complexity for fast handovers. It will also differ remarkably from the current MIPv6 draft and this can lead to difficulties and erroneous behavior when implementing it. The approach will however be "cleaner" since everything deals with layer-3 issues, thus supporting the most important design goal of IP: To be scalable in a heterogeneous environment.

5.4. Hierarchical MIPv6 Mobility Management

This approach aims to enhance the network performance while minimizing the impact on MIPv6 and other IPv6 protocols. Because of these design goals the approach may successfully be combined with other handover frameworks as well as being an independent solution.

To improve the network performance, HMIPv6 suggests extensions to MIPv6 that will reduce the amount of signaling and improve the performance in terms of handoff speed. Introducing a new node called the Mobility Anchor Point (MAP) in a hierarchical network structure will help making these enhancements. The MAP operation is in many ways similar to the ones used by the HA, but moves these operations physically closer to the MN. Moreover, HMIPv6 is suited to implement access control and handoffs between different access technologies.

5.4.1. General architecture

Mobile IPv6 can benefit from reduced mobility signaling with external networks by employing a local hierarchical structure as shown in Figure 5.11. The new node, MAP, can in this scenario be located at any level in a hierarchical mobile IPv6 network including the access router, but it is not required on each subnet. Two different MAP modes are proposed, named extended and basic, and these will be elaborated below. Extended mode is used when the MN use a MAPs address as an alternate CoA and basic mode whenever a Regional CoA (RCoA) is formed on the MAPs network segment while roaming within a hierarchical (MAP) domain, where such a domain involves all ARs advertising that MAP.

In HMIPv6 there are minor extensions to the MN and HA operations – the CN operation is left unchanged. HMIPv6 is based on the standard MIPv6 specification, but the modifications are all additions, so they may operate together in the network. Consequently the new facilities do not have to be exploited if not decided by the MN.

The MAP will limit the amount of MIPv6 signaling outside the domain because it will operate as a regional HA. The MN can therefore reduce the amount of signaling to its HA and most of the time use a closer MAP, which will help supporting efficient handovers. By adding bicasting to a MAP the packet loss will be reduced. This will improve the throughput of best-effort services but also performance of real-time data services over a radio interface.

The introduction of a MAP concept will further diminish signaling generated by MIPv6 over a radio interface. The MN only has to perform one local BU when changing its layer-3 AP within the MAP domain, opposed to standard MIPv6 where at least two BUs will be sent (CN and HA). AAA services may also interact with the MAP to perform key distribution during handoffs and eliminate the need for re-authentication, which will is described in section 6.3.2.

When a MAP receives packets addressed to the MN, they are encapsulated and forwarded to the MNs current address. If the MN changes its current address within a local MAP domain, it only needs to register the new address with the MAP, since the global CoA does not change.

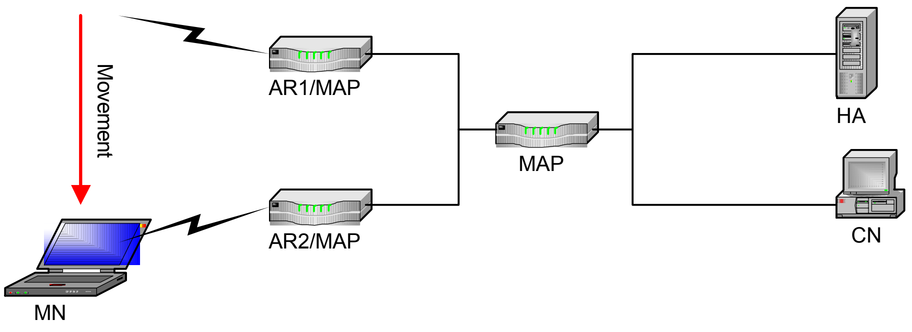


Figure 5.11 – Network elements in hierarchical mobility management.

The figure above illustrates the use of MAP and can help provide more efficient handovers for the MN as it moves from AR1 to AR2 while communicating with the CN. It is possible to use multi-level hierarchies of routers and implement the MAP functionality in AR1 and AR2.

Upon arrival in a new link, the MN will automatically discover the global address of the MAP when receiving a RA, stored in all ARs within the administrative domain. If the address to the MAP is the same as on the old link, it moves within the same administrative unit and a BU only have to be sent to the MAP. If the MN moves to a new administrative unit, the RAs will not contain the same MAP address and BUs has to be sent to the HA and all CNs.

A MN is not limited to register in only one MAP. Multiple registrations can be utilized simultaneously, e.g. using different MAPs depending on which CN the MN communicates with, to be able to use the network bandwidth in a more efficient manner. This will avoid sending all packets via the "highest" MAP in the hierarchy and hence improve performance and scalability.

5.4.2. Basic Mode

When a MN is connected to an AR within a hierarchical network in basic mode, messages are exchanged between nodes as illustrated in the figure below.

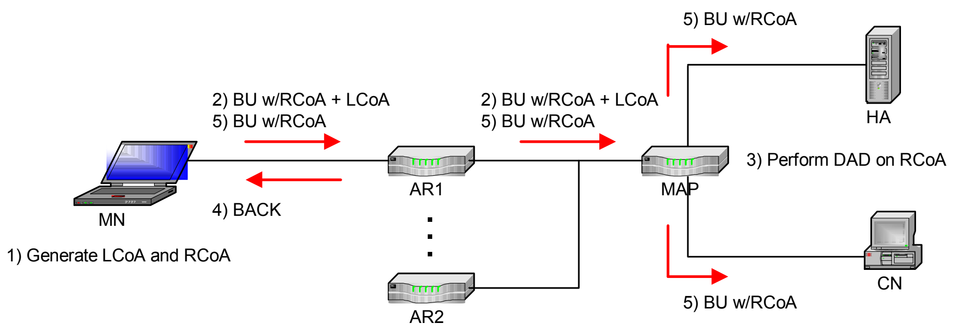


Figure 5.12 – Messages exchanged with MAP in basic mode.

In this scenario the MN needs two addresses (1), RCoA on the MAPs subnet and an on-link CoA (LCoA) on the subnet to AR1. These addresses may be formed in a stateless or stateful manner and are generated using the MAP information included in the RA messages. After forming the RCoA, the MN sends a regular MIPv6 BU to the MAP (2). This BU will bind the MNs RCoA (similar to a home address) to its LCoA. The MAP (acting as a HA) will then perform DAD for the MNs RCoA on its subnet (3) and return a BACK according to the MIPv6 specification (4).

The MN must also register its new RCoA with its HA by sending a BU that specifies the binding as in MIPv6 (the home address option of the BU is set to the home address, and an alternate CoA suboption is used and set to the RCoA). It can also send a similar BU to its current CNs (5).

The MN should wait for the BACK from the MAP, before registering with its HA. To speed up handover between MAPs, a MN may send a BU to its previous MAP, specifying its new LCoA. Packets in transit that reach the previous MAP are forwarded to the new LCoA. When the MN moves locally, i.e. if the MN moves from AR1 to AR2, its MAP does not change and it must only register its new LCoA with its MAP. In this case, the RCoA stays unchanged and no BUs has to be sent to the HA and CNs. Note that a MN can send a BU containing its LCoA (instead of its RCoA) to CNs who share the same link. Packets will then be routed directly without going through the MAP.

The MAP will receive packets addressed to the MNs RCoA (from the HA or CNs) and tunnel them to the MNs LCoA. The MN will decapsulate these packets and then process them in the normal manner.

5.4.3. Extended Mode

In some mobile scenarios it may not be possible for a MN to get an RCoA on the MAPs subnet, e.g. if the MN is also a router to which several MNs may be connected. Network operators may however also have other reasons for choosing this HMIPv6 mode of operation.

If the MN connects to an AR using extended mode, the messages exchanged are quite similar to the scenario described in basic mode. One major difference, though, is that when the MN sends a BU to the MAP, this BU contains the LCoA (as source address) and the HAddr – not the LCoA and the RCoA as in basic mode.

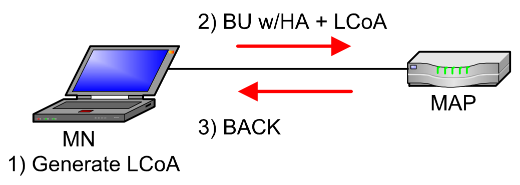


Figure 5.13 – The HA option is included in the BU message in extended mode.

After MAP discovery has taken place, a MN can register with one or more MAPs. The MN performs this local registration by sending a BU to the MAP with the appropriate flags set. When registering with a MAP the MN should wait for a BACK from the MAP, before using the MAP address as an alternate CoA in its BUs.

After successfully performing registration with a MAP, a MN can start sending BUs with its alternate CoA to the HA and the CNs. The MAPs IP address must be included in the alternate CoA suboption.

When the MN is in a foreign network, it needs to know which path a packet has taken from the CN to the MN. That is, whether triangular routing was used via the

HA or if route optimization was used. All packets received by the MAP from CNs will be tunneled to the MN. When using the MAPs address as a CoA, packets tunneled by the HA to the MAP will be decapsulated and then encapsulated again with the MAPs address as the source address of the outer header. Therefore, the MN cannot decide whether route optimization is required by checking if the packet was tunneled from the HA. It would have to check for the existence of a routing header in the encapsulated header where the MNs home address is the final address. If the routing header does not exist, the MN should send a BU with the appropriate information to initiate route optimization.

The only impact to extended mode on the HA implementation is when tunneling packets to the MN with destination addresses other than the addresses registered by the MN in its home registration (such as site-local addresses). Here the HA should include a routing header in the outer header with the MNs registered HAddr as the final destination. This is done to enable the MAP to find the right routing entry for the MN, since it has no knowledge of a non-unicast global home address for the MN.

5.4.4. MAP discovery

When a MN moves between different network segments it must be able to obtain the MAP address and the subnet prefix. In HMIPv6 two different ways are defined: *Dynamic MAP Discovery* and *Router Renumbering*.

Dynamic MAP Discovery

This method is based on propagating the MAP option from the MAP to the MN through certain (configured) router interfaces within the hierarchy. This would require manual configuration of the MAP and the routers receiving the MAP option, to allow them to propagate the option on certain network interfaces.

A MAP will be configured to send its option or relay other MAPs' options on certain network interfaces. The choice of network interfaces is done by the network operator and depends upon the network architecture.

The information is exchanged between MAPs, ARs and MNs in router advertisements by introducing a new message option. The ARs are required to send the MAP option in all RAs containing:

- Distance (number of hops) from the MAP to the MN
- MAP preferences
- MAP’s global unicast IP address and subnet prefix.

The ARs can be configured manually or automatically with this information. In the automatic approach the routers must copy the information in the MAP option and resend it on certain network interfaces, after incrementing the distance field by one. If the router is also a MAP, and this is allowed because a hierarchical structure is both possible and recommended, it should also send its own option in the same RA. If a router receives more than one MAP option for the same MAP, from two different network interfaces, the option with the smallest distance field should be forwarded.

In this manner, information about a MAP at a certain level in a hierarchy can be dynamically passed to a MN. Furthermore, different MAP nodes are able to change their preferences dynamically based on the local policies or the load on different nodes.

Router Renumbering

The other method, Router Renumbering, requires no manual configuration for routers within the hierarchy. The MAP option is sent directly from a central node to all ARs within a MAP domain. It uses a mechanism to define a set of messages that can be used to renumber certain network interfaces without manual configuration of such routers.

Upon reception of a Propagate command, which is a command defined in this approach for propagating information between routers within a hierarchy, a router will propagate the MAP option on the designated network interface after incrementing the distance field by one, but this is beyond the scope of this document (see [38] for details).

5.5. Evaluation of HMIPv6

The Hierarchical MIPv6 mobility management is currently very interesting because it addresses several techniques that are similar to well known telecommunication systems, such as GSM. These techniques are thoroughly tested and can be applied with less effort and risk than whole new procedures – even though the draft proposes elements that are new, especially within computer communication.

5.5.1. Comparisons Between Basic and Extended Mode

Two different modes of operation are defined for HMIPv6 depending on the MNs choice of CoA when located in a foreign network. The selection of the operation mode is done based on the information sent in the MAP option, and the network administrator must configure this information manually. Some administrators may allow a MN to only obtain an RCoA or use the MAPs address as an alternate-COA. To simplify MN and MAP implementations, the MAP uses only one mode at a time.

The administrator configures the MAP operation mode in the network depending on the properties required. A brief comparison shows that the basic approach is very simple because it is completely independent of the HAs implementation. The extended approach requires that the HA includes a routing header for certain non-global unicast addresses as described earlier. This mode will therefore affect the base MIPv6 implementation, and probably make it easier to get acceptance for basic mode, because it is fully backward compatible and only additions are needed.

Both the basic and the extended mode implements functionality that will lead to increased processing demands on the MAP routers compared to normal routing functionality. The basic mode MAP routers will only encapsulate and forward intercepted packets to the MN. The extended mode MAP routers will decapsulate the packets, because the destination is its own address, and encapsulate them again using the LCoA as the MN's destination address. The extended mode will therefore require more processing of each packet than what is necessary in the basic mode. Basic mode may therefore prove to be more scalable because of the reduced MAP load, but if the domain is configured with a multi-level hierarchy of MAPs this may solve these scalability issues in extended mode as well.

Two additional IPv6 headers are needed in the basic approach, while only a single additional header is needed in the extended one. If an appropriate header compression mechanism is used, this may not be an issue of concern. In networks with limited bandwidths, such as wireless networks, compression of IP and transport headers may be employed to obtain better utilization of the available spectrum capacity. When header compression is used along with handovers in such networks, the header compression context needs to be relocated from one IP AP (i.e. a router) to another, in order to achieve an efficient handover operation ([44]).

Header compression will increase the complexity, in a mobile environment especially, since the compression contexts have to be transferred from one router to another every time a MN changes its point of attachment, to seamlessly continue using existing compression contexts. The deployment speed of standard MIPv6, and of course also procedures based on this, will then be reduced due to the increased complexity.

It may be more efficient for the MN to use the MAPs CoA as an alternate-CoA in extended mode if no header compression is used. This implies reducing the overhead in each packet compared to the basic approach, which is a valuable contribution to a wireless link with limited bandwidth. The complexity will also be reduced, something that may have a positive effect on the deployment speed.

Basic mode requires that DAD has to be performed for the (RCoA), but this is not the case in extended mode. Avoiding the DAD check can lead to a faster registration when using the extended mode if inter-MAP handovers are expected. Basic mode may also reduce this latency by registering its new LCoA with its previous MAP. The packets are then forwarded while DAD is performed, but this increases the complexity.

In respect to the DAD check necessary for the RCoA in basic mode, the efficiency depends on the network topology were the MNs move. If the MAPs are configured in a topological efficient manner, the movement between MAPs will be reduced to a minimum. This means that DAD does not have to be carried out very often, so the DAD overhead will decrease. If the MNs move between different MAPs too often, this will increase the overhead and there will probably also be difficult to achieve fast handovers due to the increased delays.

According to what is described above, extended mode will increase the load on the MAP because the packets are decapsulated there, and then encapsulated again with the MN as the destination. In basic mode, the packets are only encapsulated in the MAP before forwarding them to the MN. This reduces the MAP load, but leads to increased overhead in each packet. Header compression techniques can solve this, but adds further complexity to the mobile environment.

It is therefore important to find the balance between increased load on the MAPs and increased overhead in each packet. It is difficult to say what solution that will be the best since this depends on several issues. If there are many MNs attached to the same MAP, the processing capability in the MAP may be important to avoid unnecessary delays, and basic mode would be the best solution. Note that overlapping MAPs may reduce the load on a single top-level MAP and therefore increase the scalability in extended mode as well. If the radio interface is of very limited bandwidth, the increased overhead may be more important than the increased MAP workload. Extended mode only use one additional header compared to two additional headers in basic mode, meaning that extended mode will increase the performance. DAD does not have to be performed in extended mode, so this will reduce handover delay when the MN moves between MAPs.

The choice of whether to use basic or extended mode must be taken by the administrators based on what they expect their administrative domain will look like, considering both the available bandwidth, and the expected number of MNs.

5.5.2. MAP Discovery

The two different ways of obtaining the MAP address, dynamic MAP discovery and router renumbering, both have advantages and disadvantages depending on the network topology.

Because the automatic router renumbering approach exchange messages between routers to notify the ARs about available MAPs, this will probably lead to more signaling messages than when using a manual approach such as dynamic MAP discovery. The increased signaling load may of course reduce the scalability, but the messages are most likely exchanged between routers using link technologies with a high network capacity, so this may not be an important issue.

Router renumbering could therefore be more flexible and scalable, except for the increased signaling load. This does not, however, mean that router renumbering is the best solution in all situations. The dynamic MAP discovery can probably successfully be used in smaller networks when the topology is not changing too frequently. This approach is also simpler and the signaling between routers will be reduced.

Another important issue is to make the system manageable, even if a different part of the network is under another administrative domain. Manual configuration between these administrative domains is almost impossible when considering large networks. Internet has become very popular precisely due to its scalability in a heterogeneous environment, so supporting third party network vendors is very important for the sake of diversity – and manual configuration is therefore not an option.

Even though router renumbering is a more complicated approach, it offers increased flexibility and scalability compared to dynamic MAP discovery. By allowing both approaches, the complexity will increase even further. For smaller networks within the same administrative domain, dynamic MAP discovery may provide the functionality needed, but also router renumbering will function well. The total complexity by allowing both approaches can therefore be reduced, while preserving the best functionality, by only offering the router renumbering approach.

5.5.3. Signaling Load

The main reason for introducing a hierarchical MIPv6 scheme is to diminish the signaling generated over a radio interface. This is due to the fact that a MN only needs to perform one local BU when changing its layer-3 access point within the MAP domain – compared to at least two BUs needed in the base MIPv6 specification (CN and HA).

The MAP will therefore undertake some of the same functions as the HA, only physically closer to the MN. The handover performance for the MN will in general improve because the number of messages that are transferred is reduced, which is important when communicating on a wireless link with limited bandwidth. The link quality is also usually better when communicating within a limited region because it has a more predictable latency pattern (shorter distance and higher throughput leads to lower latency).

The hierarchical scheme will not only increase the performance, but also the flexibility. HMIPv6 is designed to coexist with several protocols, and among these the base MIPv6 specification and AAA services. Compared to MIPv6 it only adds new functionality (except for the modification of the HA in extended mode), and the MN is therefore able to choose whether to use the HMIPv6 implementation (where available) or only the base MIPv6 implementation. HMIPv6 will in this case decrease the number of messages exchanged compared to standard MIPv6, but increase the complexity. HMIPv6 can also be designed to operate together with AAA services (as suggested in chapter 6), and this will reduce the number of messages that needs to be exchanged with the AAAH by establishing a local AAA authority.

Within a multilevel hierarchy of MAPs, the number of MAP options included in the router advertisements may be quite large. This will of course require more bandwidth than if only a single MAP is advertised, but if several are advertised the MN can choose the most appropriate MAP itself. This is a question that balances the increased network load versus the flexibility and must be decided according to the total network bandwidth and the workload on each MAP. If the network bandwidth is so limited that all MAP options cannot be included using the current access technology, maybe a new access technology can be considered to increase the flexibility. Another solution can be to reduce the hierarchy of MAPs announcing their presence to the ARs and the MNs, thus reducing the number of MAP options included in RAs.

5.6. Fast Handovers Combined with HMIPv6

It is difficult to say whether fast handovers or HMIPv6 will provide the most efficient handovers, because they are very different and rather supplementary than complementary. It is more advantageous to see if these can be combined on the same network. This combination employs the traditional telecommunication way of thinking through HMIPv6 and newer computer communication way of thinking through fast handovers.

This section will briefly describe our proposal to how these two technologies can be merged. It will not, however, elaborate this in detail because the behavior is similar to what is described in the respective drafts. The intention of including this section in this thesis is just to indicate that a single solution does not need to embrace the whole aspect of implementing efficient handovers – different solutions can coexist.

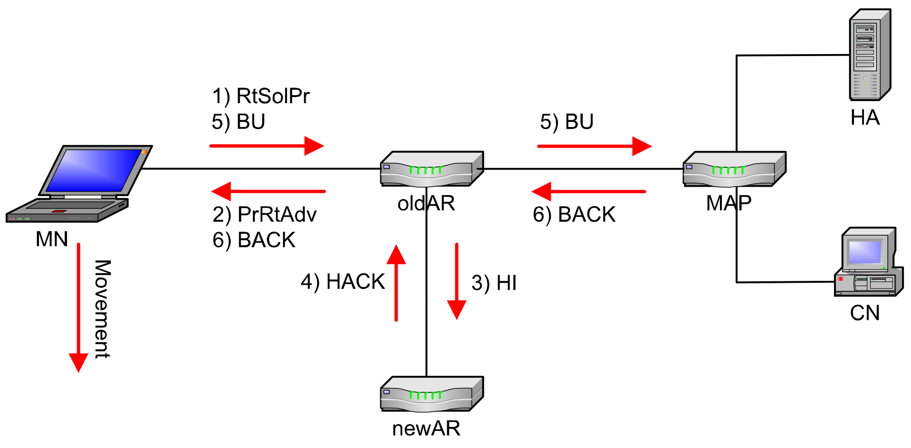


Figure 5.14 – The messages exchanged when combining fast handovers and HMIPv6.

The figure above illustrates a mobile-determined handover combining the infrastructure from both fast handovers and HMIPv6. This illustration reminds most of the scenario in fast handovers, but one major difference is that no BU needs to be sent to the HA and the CNs when moving within the MAP hierarchy. BUs are only sent when moving between different MAPs, which can be in either basic or extended mode.

The illustration is simplified, and in the case of a basic mode MAP the MN will initiate the scenario by generating LCoA and RCoA (see Figure 5.12) based on the reception of a MAP option in the router advertisement. Further it will send a BU containing these addresses to the MAP that will perform DAD on the RCoA and reply with a BACK to the MN. When the MN moves into a new MAP domain, a

BU containing the RCoA will only be sent to the HA and the CNs in the same way as specified in basic mode.

This approach will have many of the same advantages and disadvantages as described in fast handovers and in basic mode HMIPv6. Using the functionality provided by fast handovers may carry out efficient handovers and only a single BU is sent to the MAP – the HA and the CNs are not affected as long as moving within the hierarchy. The signaling will therefore diminish inside the MAP hierarchy by only having to send a single BU to the MAP when moving within the hierarchy. It will also diminish outside because the BUs are sent less frequently than when using only the fast handovers procedure.

By combining these two technologies the flexibility will increase because it is possible to take advantage of the functionality provided in both of them. The complexity will increase of the same reason, but it seems like both of these features are good solutions when designing service models for efficient handovers.

5.7. Mathematical Analysis

After evaluating Fast Handovers and HMIPv6, and proposing a solution for combining both to improve the handover efficiency, this section will use a mathematical evaluation to put things into perspective. The following analysis is based on an ongoing discussion in the IETF mailing list [39], which compares the signaling overhead generated on hierarchical vs. nonhierarchical designs. The details of the analysis are included in Appendix F, but the results will be presented here.

5.7.1. General Proposal

The general proposal takes the number of signaling messages, and their delay, transferred between nodes involved in a handover into consideration – both in hierarchical and nonhierarchical designs.

In the nonhierarchical design the MN must send BUs to the HA and all CNs every time the MN moves from one network segment to another. The hierarchical approach, on the other hand, assumes that the MN only must send BUs to the HA and the CNs when it moves between different hierarchical domains. In addition BUs must be sent to the localized mobility manager (LMM) when the MN moves within the same hierarchical domain.

The analysis concludes, assuming the presumptions taken are correct, that the signaling latency in the hierarchical case will be the best solution when considering the signaling load.

5.7.2. Binary Tree Proposal

The MN's mobility has to be taken into account when talking about performance of (non)hierarchical mobility support. Assume you have a network with a tree-like topology, like a cellular network, where MNs are attached to the leaves (base stations). Hierarchical mobility support means that each node is a mobility agent (HA, Foreign Agent, or MAP). When a MN moves from leaf A to leaf B, exactly the mobility agents on the path from A to B are affected by the move, i.e. they have to change the bindings for the MN. Let us call the length of that path the "distance" between A and B (it is just the distance in the graph-theoretic sense) and consider it as a measure for BU performance.

In order to arrive at some numbers, consider as topology a complete binary tree, as illustrated in the figure below, of height n and, for a better understanding of the following, draw it as a wheel of radius n where:

- The root becomes the nave
- The branches become the spokes
- The leaves are arranged on the felly

The MN moves on the felly, and by accident such a move may be very short (of distance 2 in case of two base stations served by the same controller – A and B) or very long (of distance 2n if the root/nave is traversed – A and C).

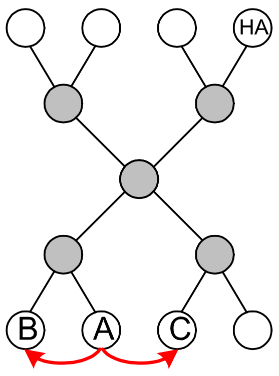


Figure 5.15 – MN movement of different distance.

In order to have some better average cost analysis, let us consider complete travels around the wheel, i.e. permutations of the leaves/felly. We can see that the BU traverses the very short distance of 2 in 50% of all cases. On the other hand, it is easy to imagine travels where each move has distance 2n because it crosses the nave.

In the nonhierarchical mobility support, i.e. base MIPv6, you fix a leaf (HA) on the felly and all binding updates have to go to it. This means that at least 50% of them have to traverse the root/nave.

5.7.3. Evaluation of Mathematical Analysis

Even though the first analysis indicates that the hierarchical approach will reduce the signaling load, there are some issues that can be discussed according to the presumptions taken, and how much the handover latency actually will be reduced.

When analyzing the nonhierarchical case it is assumed that all messages are exchanged in sequence and that the total latency therefore is the sum of each of the message delays. One can argue that the BUs will probably be sent in parallel to both the HA and the CNs and the latency therefore will be a maximum of a set of numbers – not a sum. The same can also be said about the hierarchical case where the LMM AR sends the BU to the top level LMM independently of replying to the MN, so that the MN does not have to wait.

The MN should probably also wait for the BACK from the HA before proceeding with sending BUs to the CNs. These messages would in this case have to be sequential. In any case this may be seen as the worst-case analysis and there is of course potential for optimization. It might be a better solution to simulate this instead of analyzing it using only theoretical background material, but the models should despite this give and indication of what to expect.

It is obvious that updating a node that is two hops away is faster than updating one that is three hops away. The point is actually whether a hierarchical solution will make any significant improvement compared to a nonhierarchical approach. The numbers of delays over the air interface are much higher than on wire, which makes the forwarding delays on wire within a local domain almost insignificant. In order to make this approach help the overall performance, we need to see an order of magnitude reduction in delays for any of this to be significant.

Exemplifying this argument with numbers can illuminate the actual problem. The typical figures for forwarding delays in some major routers are roughly 1-2 ms (some routers can do much better). Delays over the air in CDMA (e.g. WCDMA) are 60 - 80 ms (80 being the worst-case scenario). If we can save 3 or 4 ms in a 70 ms delay, e.g. reducing the delay from 76 to 73 ms, there may not be any significant improvements by deploying a more complicated hierarchical structure event though it in theory is proven to be a more efficient solution.

Looking into the second binary tree proposal, the hierarchical mobility support is in some cases provably better than the nonhierarchical counterpart also here. But these cases do not occur very often. Hence it is questionable to attempt an optimization because it will have an effect only in rare cases. Maybe, one should distinguish mobility by feet, by car/train, and by plane because they show different travel characteristics in the binary tree model described above.

5.8. Summary

To support real-time applications and QoS it is important to minimize delays when performing handover from one network segment to another. The base MIPv6 implementation does not provide these properties alone, but it supports extensions that can reduce the handover latency. Several draft documents are proposed for this purpose and two of them are elaborated – Fast Handovers for Mobile IPv6 and Hierarchical MIPv6 mobility management.

Fast Handovers for Mobile IPv6 aims to enable the MN to configure a newCoA in a way that it can use this address immediately after connecting to the newAR. If the MN initiates a handover it sends an RtSolPr message to the oldAR including the link-layer address to the newAR in which it wants to connect. Based on this message, or if the movement is initiated by the network, the oldAR sends a PrRtAdv to the MN with either the new address or with the information necessary so that it can generate one. The oldAR also sends a HI to the newAR to notify it about the upcoming handover and establish a valid newCoA for the MN. The newAR replies with a HACK in response. This contains information about the address the MN can use, and the newAR will then defend this address for the period of time the MN is expected to use when connecting to the newAR. The routing will only be changed after the oldAR receives a BU from the MN confirming the handover, and the oldAR replies with a BACK either through the old or the new network segment.

This technology is new and has still a few shortcomings. How the MN can retrieve the identification to the newAR is one challenge, while others are associated with the utilization of different option fields, flag bits and movement detection.

Hierarchical MIPv6 mobility management aims to reduce the mobility signaling with external networks by employing a local hierarchical structure using a new node called MAP. The MAP will limit the amount of mobile IPv6 signaling outside the domain because it will operate as a regional HA. The MN can therefore reduce the amount of signaling to its HA and most of the time use a closer MAP, which will help supporting fast handovers. Upon arrival in a new link, the MN will automatically discover the global address of the MAP when receiving a RA stored in all access routers within the administrative domain. If the address to the MAP is the same as it where on the old link, it moves within the same administrative unit and a BU only have to be sent to the MAP. If the MN moves to a new administrative unit, the RAs will not contain the same MAP address and BUs has to be sent to the HA and all CNs.

A MAP can operate in one of two different modes, extended and basic, which has different strengths and weaknesses with respect to network overhead and load. Other challenges, such as the propagation of the MAP information, are also discussed according to the network topology.

Different solutions for carrying out efficient handovers may coexist – they do not necessarily need to embrace the whole problem domain alone. It is described how fast handovers can be combined with HMIPv6 to generate a solution that is more complicated, but also more flexible because it embraces both technologies.

Finally the chapter introduces a mathematical evaluation of a hierarchic approach compared to a nonhierarchical one. In theory it seems like the signaling load, which is an important aspect of enabling efficient handovers, is reduced when using hierarchic. Theory is however not the same as practice, and the deployment of a hierarchical structure may increase the complexity of an implementation. More thorough studies are, however, necessary to determine whether such deployment will reduce the handover latency when considering the order of magnitude needed for effectuation.

6. AAA

Objectives

- Describe why AAA services are necessary in a mobile environment.
- Outline how the necessary security mechanisms function.
- Explain how AAA mechanisms can be integrated with MIPv6, also in consideration with efficient handovers issues.
- Elaborate deployment issues of an AAA infrastructure, such as the propagation of AAA servers, client identifiers and integration with GSM.

Contents

6.1. Introduction

In a mobile environment, MNs will not only require Internet access within their own domain. They will also visit foreign networks provided by others and with the experiences from today's GSM infrastructure we can guess that this kind of roaming will not be free, and accounting is therefore necessary. Service providers in a foreign domain commonly require *authorization* to ensure a good business relationship with the client. This leads directly to *authentication*, and of course *accounting* – whence AAA.

This section explains how these services can be implemented in MIPv6 according to[3]. It will first of all elaborate how a MN will be granted access to foreign networks when moving between different network segments by means of the AAA challenge/response mechanism. The challenge/response procedure is a mechanism that enables one entity (the foreign network) to challenge another (the MN) with given credentials so that the other entity replies in order to be authenticated and therefore granted access to the foreign network.

According to the discussion on stateless and stateful address autoconfiguration, stateless mobile scenarios will be focused on in this thesis. There will first be an explanation to the general architecture, followed by a discussion on how to integrate AAA services with Fast Handovers.

The mobile nodes will need a way to offer credentials to an AAA server located in the roaming area network. These credentials will be forwarded to an AAA server located in the MNs home network. This AAA server will then determine if the credentials offered are valid for the specific MN and if it may be granted access to the local network. The intention is to integrate these mechanisms with the existing

IPv6 routers so that AAA services are propagated throughout the Internet. These AAA services aims to protect the MN from replay attacks and other DoS attacks.

A *denial of service* (DOS) attack is one that is intended to compromise the availability of a computing resource. Common DOS attacks include ping floods and mail bombs – both intended to consume disproportionate amounts of resources, starving legitimate processes. Other attacks are targeted at bugs in software, and are intended to crash the system. The infamous *ping of death* and *teardrop* attacks are examples of these [46].

6.2. General Architecture

This section will describe the general AAA architecture. First, the SAs needed between different network entities will be outlined, followed by the network topology and then the basic operations.

6.2.1. Security Associations

This background section is based on [45] to help explaining the AAA mechanisms needed in a mobile environment, which will be discussed below. This is included to give an indication of how AAA services only are a part of a bigger security system, IPsec, which is extensively used in IPv6 but that not will be elaborated here.

Within the Internet, a client belonging to one administrative domain (the home domain) often needs to use resources provided by a foreign domain. An agent in the foreign domain that attends to the MN's request (called the *attendant*) is likely to require that the MN provide some credentials that can be authenticated before access to the resources are permitted.

The attendant may not have direct access to the data that is needed to complete the transaction. Instead, the attendant is expected to consult a local authority in the same foreign domain in order to obtain proof that the MN has acceptable credentials. Since the attendant and the local authority are part of the same administrative domain, they are expected to have security relationships that enable them to securely transact information locally.

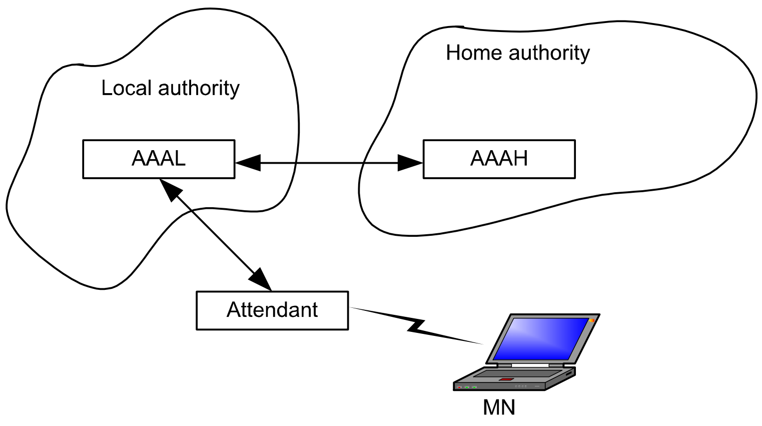


Figure 6.1 – MN contacts the attendant to obtain necessary credentials.

The local authority in the foreign domain (AAAL) itself may not have enough information to verify the credentials of the client. Nevertheless, in contrast to the agent attending the MN, AAAL is configured in a manner that enables it to negotiate the verification of MN credentials with an external authority. The local and external authorities are configured with sufficient security relationships and access control, so that they can negotiate the authorization, which enables the MN to have access to the requested resources. This authorization commonly depends on secure authentication of the MN's credentials.

Once the local authority has obtained the authorization, and the authority has notified the attendant of the successful negotiation, the attendant can provide the requested resources to the MN.

There is a security model implicit in Figure 6.1 and it is crucial to identify the specific security associations (SA) assumed in that security model, as illustrated in Figure 6.2.

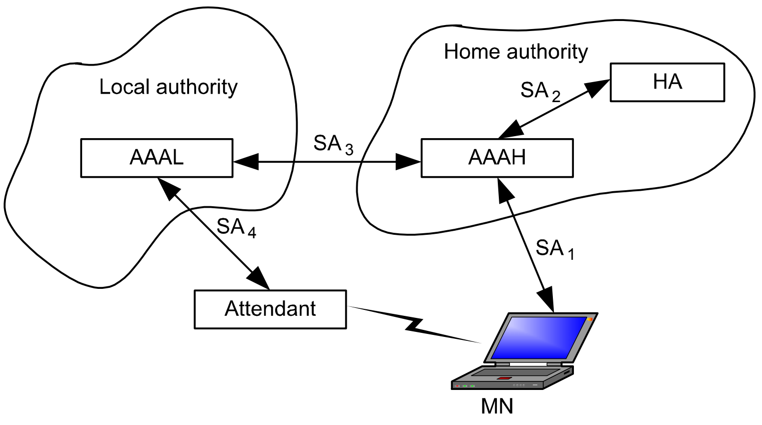


Figure 6.2 – Security associations in a mobile AAA scenario.

These are the security associations needed:

- **SA_1** – The MN has a SA with the AAAH because it is belonging to the home domain.
- **SA_2** – A SA between AAAH and HA will allow the MN to maintain only a single SA.
- **SA_3** – AAAL and AAAH have to share a SA because otherwise they could not rely on the authentication results, authorization, or even the accounting data that might be transacted between them.
- **SA_4** – The attendant must share a SA with AAAL so that it know that it is permissible to allocate the local resources to the MN.

How these SAs are established is left out because it is not directly related to mobility issues. Details about this can be found in [12].

6.2.2. Network Topology

The entities that exist in a proposed AAA mobile network is depicted in Figure 6.3 from a functional point of view. From a system point of view we refer to Figure 6.4.

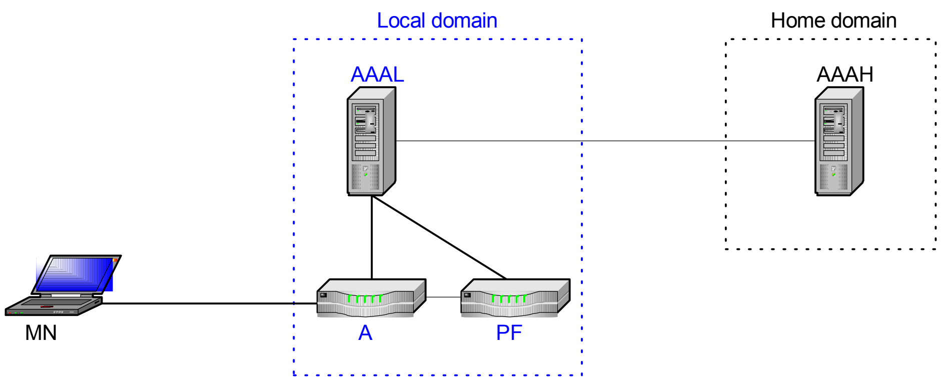


Figure 6.3 – Network entities in the proposed mobile AAA environment.

The MN is the client asking for access to the local domain. The attendant (A) is responsible for extracting identification data from the MN in order to forward them to the local AAA server in the local domain. This server is called AAAL and is responsible for mediating local traffic to the AAA infrastructure. The packet filter (PF) can be considered as a security gateway, responsible for disallowing unauthorized datagram traffic. When clients are authorized, access control lists are updated with the MNs IP address. The uncontrolled part of the router system is subject to clients granting access. The authorized clients are dealt with in the controlled part. The AAAH server is in the MNs home network and has the possibility of authorizing each of its clients.

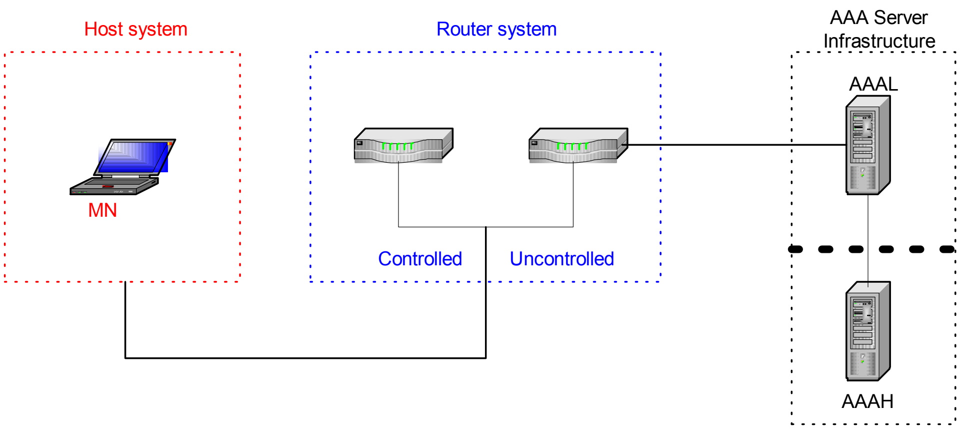


Figure 6.4 – The router system is divided into a controlled and an uncontrolled part.

The MN needs to provide some kind of credentials in its authorization request to the router system. One solution of how to obtain these credentials can be a message authentication code constructed using a secret key between the MN and AAAH. These credentials will be created the first time the MN registers in its home network. The suggested solution for these credentials consists of the following elements.

- A client identifier.
- The local AAA challenge.
- A mechanism for preventing replay attacks.

AAAL needs a client identifier to recognize a suitable AAAH for which it will perform the necessary authorization steps for the MN. This identifier can be either the network access identifier (NAI), or the IPv6 global unicast address of the MN. Which identifier to use is discussed in section 6.6.

Every participant in this scenario should verify the validity and freshness of the different messages in order to protect themselves from replay attacks. The messages exchanged between the attendant and AAAH and between AAAH and AAAL are explained in the next section. Replay attacks between the MN and the other entities can be prevented in numerous ways. The suggested solution is to include the local AAA challenge and a timestamp in the request.

6.2.3. Basic Operation

The basic operation of the protocol is shown in Figure 6.5. It is assumed that the MN has recently moved to a foreign network and receives a RA. This advertisement includes an AAA challenge option as shown in the figure (1). When the MN receives this message it constructs a tentative IP address according to the specification of stateless address autoconfiguration, and replies with the options shown in (2) – AAA Request. The local challenge option (LC) is copied into the message so that the uncontrolled part of the router system can verify that the reply is to the recent challenge. The next option is the reply protection indicator (RPI) used to prevent replay attacks between the MN and AAAH. The tentative IPv6 address (ID) in addition to the long-term IPv6 address or NAI [47] is also included to make sure the router system can identify the MN. The last option is the MN's credential. The MN can perform DAD before sending the AAA reply as discussed in 6.3.1.

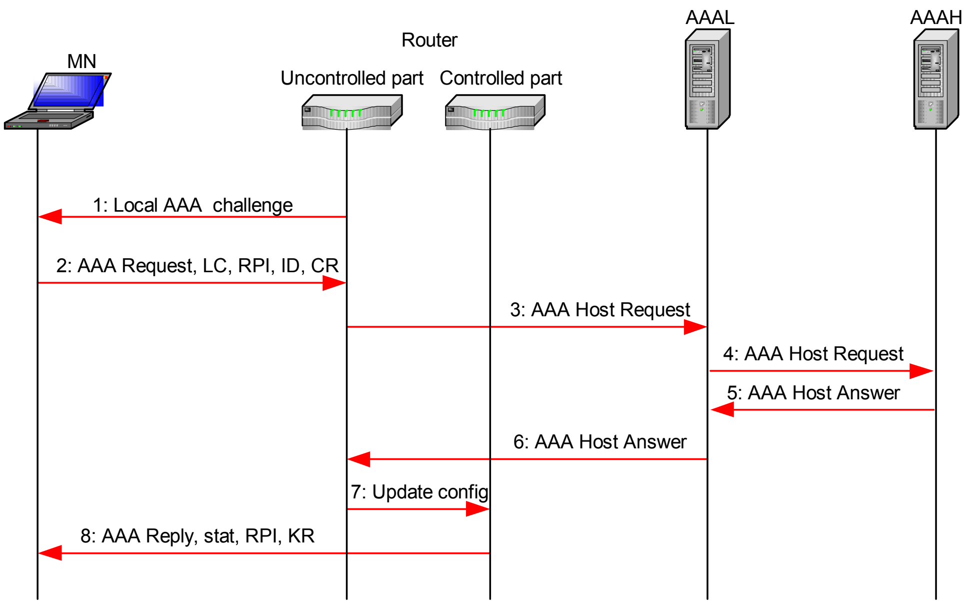


Figure 6.5 – Mobile AAA message structure.

The attendant receives the reply (2) on one of its network interfaces and must check if the chosen tentative address is valid and reply with an address in use if not. If the address is determined valid, the attendant will extract the AAA data provided.

The AAA data will be forwarded to AAAL (3) using the AAA protocol. These data are then delivered to AAAH (4). AAAH can evaluate the credentials provided by the MN. AAAH is then responsible for representing these data so that the attendant can extract them. These data are sent back to AAAL (5) with the following options:

- Timestamp, or AAA Challenge and Host Challenge
- Key Reply
- Lifetime
- AAAH Authenticator

The timestamp or challenges are still included to prevent replay attacks. If AAAH chooses a key to be used between the attendant and the MN, this key should be encoded in this reply. The lifetime option specifies the validity period for this authentication. Finally, an AAAH authenticator is computed by applying the algorithm specified in the security association that exists between the MN and AAAH.

If the credentials provided are determined to be correct, AAAH replies with this AAA Host Answer (5) to AAAL, who then will forward this to the attendant (6). The attendant is then responsible for adding an entry for the mobile node in its neighbor cache in addition to updating its packet filter (7) with an entry for the MN's IP address. The attendant then sends the reply back to the MN (8) that must verify the AAAH authenticator and then, if success, create a SA for the attendant.

6.3. AAA in Efficient Handovers

Section 5.6 in the efficient handovers proposes integration between the fast handovers draft and HMIPv6. It emphasizes that this kind of integration must be possible to allow future extensions and modifications. As described in the introduction, MNs will visit foreign networks provided by others, and this will probably not be free of charge. The users will still, however, require the same services, thus AAA services must be incorporated with mechanisms that makes efficient handovers possible.

The MN initiates DAD while it is still connected to the oldAR to reduce handover latency because it prepares the new network segment for the new node. Of the same reason AAA services must also be carried out at this stage. The MN will then know in advance if it is granted access to the new link – and to what terms.

At first sight it seems natural to include AAA services with DAD because this is already a part of the messaging infrastructure and have to be carried out when connecting to a new link. When the MN moves between different administrative domains it must account to the right network vendor. If the MN moves to a new network segment within the same administrative domain it is not necessary to renegotiate the AAA credentials, but the MN cannot know this before initiating the handover procedure. When moving within the same administrative domain, each local packet filter will let the MN through.

The following proposals we have made for integrating AAA services and efficient handovers require thorough knowledge in both technologies. The basic AAA functionality is described above, while efficient handovers are described in chapter 5.

6.3.1. AAA and Fast Handovers

Integrating AAA services into the fast handovers proposal can be done quite simple by including different AAA information into the messages that are already defined.

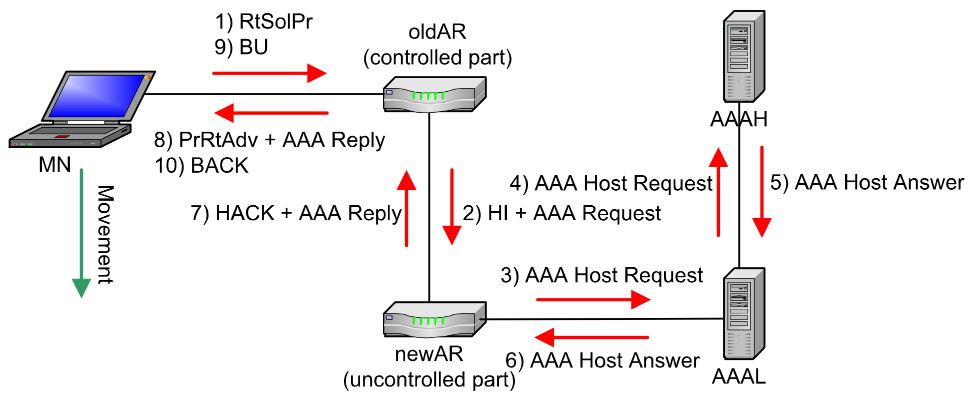


Figure 6.6 – Fast handovers (mobile determined) for mobile IPv6 integrated with AAA services.

When a MN detects a newAR it decides to make a handover to, it sends a RtSolPr message (1) to the oldAR. The oldAR is the AR in which it is currently attached and it can according to AAA therefore be looked at as the controlled part. If we assume that the MN is already authenticated on the old network segment, the oldAR have the AAA credentials to the MN. It can therefore include the MN-identifier and an AAA request when sending the HI message (2) to the newAR, which in AAA terminology is the uncontrolled part.

If the newAR and the oldAR are under the same administrative domain, no new AAA credentials are needed for the MN. The newAR then sends a HACK (7) with an AAA reply included, while the oldAR includes this in the PrRtAdv message (8) to provide the MN with updated information. Including this is not strictly necessary because the AAA negotiation process can work transparently to the MN, but in the case where new AAA credentials are needed the MN should know the AAA Reply.

If both ARs are under different administrative domains, new AAA credentials are needed for the MN. The newAR must then on the reception of the HI message (2), negotiate new credentials using normal AAA procedures (message 3, 4, 5 and 6) before continuing as described above.

6.3.2. AAA and HMIPv6

AAA services can also be integrated into the HMIPv6 messaging infrastructure proposal as shown in the figure below.

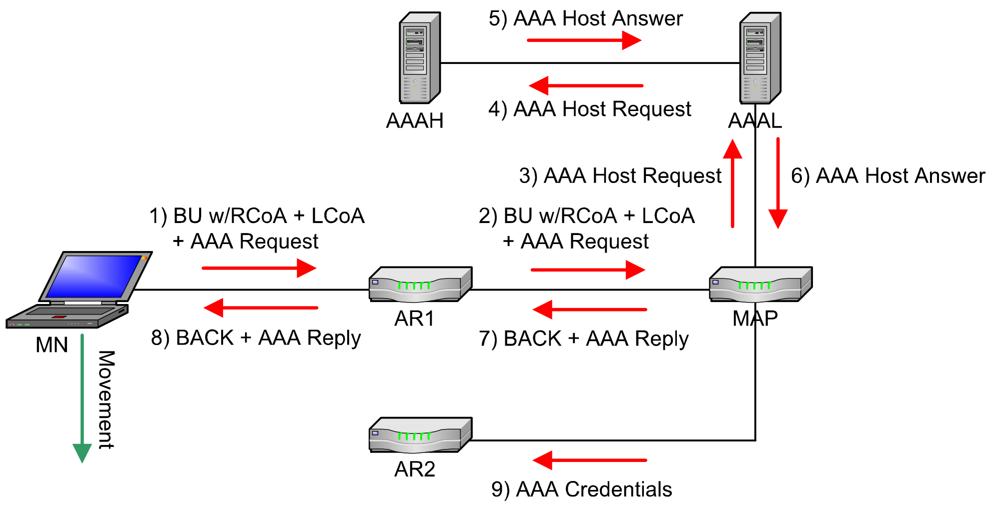


Figure 6.7 – HMIPv6 integrated with AAA services.

Consider a scenario where the MN has just connected to AR1 and received a RA containing information about the MAP. It will then, assuming the MAP is configured in basic mode, generate the LCoA and RCoA before including them in a BU option that sent to the MAP (message 1 and 2). It will from the information received also know that certain AAA credentials are needed to be able to continue the communication. It therefore includes an AAA request together with the BU.

When the MAP receives this message it carries out the normal AAA procedures (step 3, 4, 5 and 6). When it receives the AAA host answer (6) it returns a BACK including an AAA reply (7), which in this scenario passes through AR1 before reaching the MN (8). The MN is then equipped with the AAA credentials needed for further communication on the network segment.

If the MN decides to move to AR2, which is under another administrative domain, a similar scenario to the one described above will be executed. If AR2, on the other hand, is under the same administrative domain, no new AAA credentials are needed for the MN. When it sends a BU to the MAP (step 1 and 2), the latter does not need to make new arrangements with AAAL and AAAH. It only needs to pass on the MN's AAA credentials to AR2 (9).

6.3.3. Evaluation of AAA in Efficient Handovers

It fell natural for us to use the existing messaging infrastructure, described in the efficient handovers procedures, when creating the proposals for integrating it with the AAA services. By utilizing the existing infrastructure less complexity is added to the scenarios than when adding new messages. As we can see in both the scenarios described above, the messages are exchanged in very much the same way as without the AAA services integrated.

A common denominator in these scenarios is that they are simplified a great deal if the MN moves within the same administrative domain. The AAA credentials are in this case moved from the old AR to the new, without having to go all the way through AAAH to determine if the MN shall be granted access. This approach is simpler in HMIPv6 because the infrastructure already provides mechanisms that can help the MN determine if it moves into a new administrative domain through the propagation of MAP information in the RAs.

The fast handovers proposal is more complex, but still possible. The oldAR and the newAR must have a SA between them to be able to carry out efficient handovers. They can use an identifier when establishing this, e.g. the network prefix, to determine if they are under the same administrative domain. When moving to the newAR the oldAR already knows if the new AAA credentials are needed for the MN to be granted access to the new network segment. The necessary precautions can then be taken to avoid, or reduce, the handover latency.

When these precautions are taken before the MN moves from one network segment to another, it should be possible to carry out efficient handovers in both scenarios – even when new AAA credentials are needed to access the network segment. If the MN only moves within the same administrative domain the handovers will be even more efficient. This may allow the QoS aspect to live, yet with the AAA services integrated.

6.4. HA and AAAH Infrastructure

Different architectures can be used when deploying AAA services in a mobile networking environment. The architectural difference is whether or not the HA and AAAH belong to the same domain. If they belong to the same domain it may be more efficient to bundle the BU to the HA in the AAA Request message, so that the delay is minimized. If they belong to different domains, the BU and the AAA Request message must be sent in two different messages. The following will explain the architecture for the proposed solutions and discuss their advantages and disadvantages according to the following factors:

- General security
- Network load and overhead
- Delays
- Complexity

The section is divided in three sections. The first section explains the architecture when AAAH and HA belong to the same administrative domain, whereas the next explains the architecture when they belong to different administrative domains. The last section compares the different solutions in order to propose what might be the best solution for AAA services in a mobile networking environment.

6.4.1. HA and AAAH in Same Domain

Figure 6.8 shows the architecture (see [45]) when the HA resides in the same administrative domain as the AAAH server. This means that both the BU designated to the HA and the AAA Request should be sent to the same subnet. It is suggested to define a new option called Embedded Data Option (EDO), which can bundle the BU in the AAA Request as payload. The designated packet will then traverse through the AAA entities, and the AAAH server is responsible for unpacking the option and delivers the binding update to the HA as shown in the following figure. Each BU option should be responded with a BACK and this is sent back to the AAAH server. The AAAH server will further bundle the acknowledgement into the AAA reply.

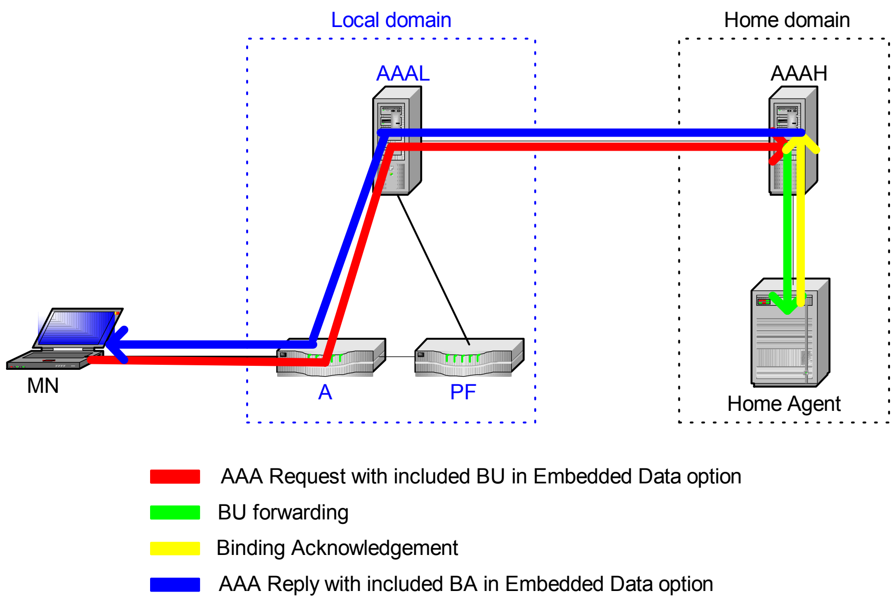

Local domain
Home domain
AAAL
AAAH
MN
A
PF
Home Agent
🟥 AAA Request with included BU in Embedded Data option
🟩 BU forwarding
🟨 Binding Acknowledgement
🟦 AAA Reply with included BA in Embedded Data option

Figure 6.8 – HA and AAAH in the same domain.

The basic problem when considering the general security factor is the well-known problem of distributing keys between the network participants. The basic solution to this problem is based on the use of digital signatures, where the general idea allows the building of chains-of-trust.

The *chain-of-trust* can be that entity X certifies that a certain public key belongs to Y, and then Y goes on to certify that another public key belongs to Z, and then there exists a chain of certificates from X to Z, even though X and Z have never met. If Z wants to provide his public key to A, he can provide the complete chain of certificates – The certificate for Y's public key issued by X and the certificate for Z's key issued by Y. If A has the public key for X, he can use the chain to verify that the public key of Z is legitimate.

The stated principle above also applies to the MNs assuming that the AAAH and the HA reside on the same domain. When a MN visits a foreign network, the basic authentication procedure shown in Figure 6.5 must be followed. When the MN has provided the necessary information in the AAA Request, the uncontrolled part of the router sends this information in an AAA Host Request to the AAAL. The AAAL then forwards this information to the AAAH resided on the MNs home network. The AAAH server evaluates the credential provided by the MN, in order to reply with an AAA Host Answer whether or not the MN should be granted access. This is when the chain-of-trust problem arises. When AAAH tells AAAL that the MN should be granted access, AAAL must trust AAAH that the MN has given valid credentials. However, the AAAL has no guarantee that the AAAH granting access is doing its job properly. There is a possibility that the AAAH server has granted access even without checking the credentials provided by the MN.

Another problem related to the chain-of-trust and that the AAAH servers are distributed throughout the network is that possible intruders can build their own AAAH system network, just to provide their MNs with access in foreign networks. These aspects compromise the general security when AAAH and the HA resides in the same domain.

In order for the proposed solution to be considered used in MIPv6 enabled scenarios, there must be some sort of generalized control between the different

AAAH entities. This complicates the solution, and if generalized control is introduced, it is similar to the solution discussed below, where AAAH and HA resides in different networks, with generalized control.

There are many mechanisms that must be performed when a MN registers in a foreign network. It would be desirable to perform some of these mechanisms together, in order to minimize network load. This can be done by the use of the EDO as suggested in Figure 6.8. The number of initial packets sent in order to enable communication for the MN is reduced, thus decreasing the signaling load. This may increase the efficiency of handover and may be important when many MNs are requesting access to the foreign network. The complexity is however increased when the BU is embedded in the AAA Request. The AAAH can fail to unpack the EDO, when forwarding the request to the HA and it can fail to bundle the acknowledgement when sending the reply to the MN. This increases the source of errors when a MN is trying to change its CoA, thus making the authentication procedures more complicated than if the BU and AAA Request were sent in two different messages. The bundling of these two options in on message also increases the processing demand one each AAAH server, due to that it must unpack the BU and forward it to the HA, and pack the BA and forward it to the MN.

When the HA and AAAH are residing in the same administrative domain, each domain will need to have their own AAAH server. When the AAAH servers are used in a distributed manner as depicted in Figure 6.8, each system administrator will be responsible for management and maintenance of the AAAH server. The network administrators will have more responsibility, and this might increase the complexity of each domain.

6.4.2. HA and AAAH in Different Domains

Figure 6.9 shows the intended scenario when the HA and the AAAH server are in different domains. This implicates that the BU option and the AAA request must be separated and sent to different domains as shown in the figure below. This proposal presumes that the MN first obtains a CoA and performs DAD with success. When the MN is authenticated and has received the AAA Reply, it can send its BUs to the HA and possible previous routers as described in chapter 3. How DAD and stateless address autoconfiguration is performed according to this scenario is described in 6.3.1. Remember that the MN’s home IP address must be included and sent to the attendant.

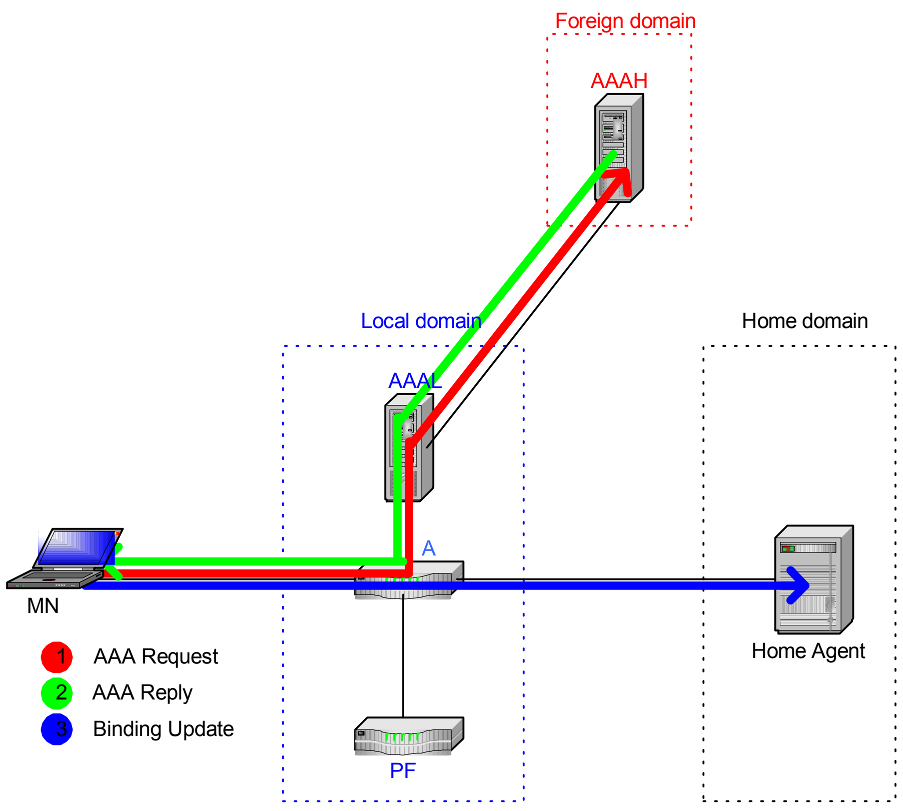


Figure 6.9 – HA and AAAH in different domains.

The obvious advantage with the solution provided in Figure 6.9 when considering general network security is that one AAAH server can be responsible for the authentication procedures for several domains. There could for instance be one AAA server for a country, or one AAA server for each region in a country. This is similar structure as provided today when using dialup connections to access the Internet. The users dial a telephone number, and provide their username and password in order to authenticate against the dialed Internet Service Provider (ISP). This is a form of centralized administration that includes all the principles of authentication, authorization and accounting. When deploying such centralized control in a mobile AAA network, system administrators in the different domains have no control over the authentication procedures. This implicates that each domain only needs to trust the centralized security authorization, and could improve the overall access security due to the chain-of-trust principle explained in 6.7.1.

As we see from Figure 6.9 the BU and the AAA Request are sent in two different messages. When the MN has received an AAA Reply, it can send the BU to the HA. This causes more overhead and load compared to when the entities are in the same domain, since two messages has to be sent in order to authenticate and perform handover to a foreign network. This would also increase the delay, since the MN cannot send its BU and be granted access to the new network until the AAA Reply is received.

When HA and AAAH are in different domains, the complexity of the solution decreases compared to when they are in the same domain. This is because the AAA Request and the BU are sent in two different messages and there is no need for packing and unpacking them. This reduces the source of errors when a MN is moving between networks. This also reduces the processing requirements on the

AAAH server. It does not have to unpack the EDO option, and neither does it need to bundle the BA when sending the AAA Reply back to the MN.

6.4.3. Evaluation of HA and AAAH Infrastructure

The factor of centralized control is very important when considering which of the two proposed solutions that would be best when deploying AAA services for MNs. When having HA and AAAH in the same domain, the complexity increases due to that each domain must administer their own AAAH. In addition, the general security would be better when having centralized AAAH servers because of the chain-of-trust. Centralized control with authorization entities responsible for the AAAH servers would improve general network security. The complexity also increases when the EDO is used. These arguments are in favor of the separate domains solution.

The existing network infrastructure is supported when AAAH and the HA are used in different domains. The only thing needed is a centralized AAAH server and an interface used for communication between the different domains and the AAAH server. When they are used in the same domain, each domain will have to upgrade their infrastructure by installing an AAAH server. This would increase the costs for each domain when migrating to IPv6 and MIPv6 in future networks. The different domains solution is also more flexible when considering different gateway-solutions, such as the GSM-AAA when no AAAH entities are available to AAAL, as described later.

These arguments suggests that deploying the AAAH and HA in different domains would be the better overall solution for AAA services in an mobile environment such as MIPv6. The general security is better, the complexity is smaller and the fact that larger delays is introduced when sending two messages instead of one can be somewhat negligible, since the authentication in fact should be done before the actual handover takes place.

6.5. Roaming History Database

It would be preferable if the authentication of MNs when moving to a foreign network can be done while the MNs are connected the old link. This would minimize delays when performing handover. This section explains how we propose that these delays can be minimized if a database, containing the history of MNs that have been granted access to a certain domain, is implemented in each AAAL. Furthermore, it discusses if this could be an appropriate solution for AAA services in an IPv6 mobile environment.

6.5.1. General Architecture

If a MN visits a foreign network it must authenticate itself according to section 6.2. This means that a MN must send an AAA Request with the appropriate options to AAAL. AAAL further forwards this request to AAAH, who sends an AAA Reply to AAAL stating whether or not MN should be granted access in the foreign network. Even though it is assumed that the authentication takes place while connected to the old network segment, it is desirable to minimize the delay of the authentication. This is because MN sooner or later has to move from the old network segment. The radio coverage of for instance WLAN cells is limited, and if MN is moving with high speed into a foreign network, the authentication must be performed and access granted to the new network before the radio connection is lost.

In order to speed up the process of authentication, each AAAL could implement a history database that saves entries for each MN that connects to the network that

AAAL is responsible for. This database could store the NAI value of a MN associated with a lifetime. The lifetime for the specific entry can be set by AAAH for MN in the lifetime option. When the lifetime expires, the database entry is deleted, and AAA Requests must be performed again. The first time a MN connects to a foreign network, the authentication procedure is performed. If the authentication is successful, the NAI value of MN is stored, and the MN can be granted access immediately when visiting the specific foreign network again. There are however two possibilities when immediate access is granted to MN. These are:

- No further AAA actions
- AAA Request sent to AAAH.

The first approach performs an inquiry to the database to check whether or not the MN has an entry. If the entry exists with a valid lifetime, the MN is granted access. The MN must however send a new AAA Request to AAAL when the lifetime for its database entry expires. The second possibility is that the MN is granted access immediately, and the AAA Request is sent while connected to the new link. If AAAH responds that the MN should not be granted access, AAAL will refuse the MN to communicate on the new link. Whether or not the AAA Request is sent after the MN has authenticated through its NAI database entry, is not essential in this proposal. This is a security concern for each of AAAL and we suggest that the network that controls each AAAL can decide whether or not the AAA Request should be sent to AAAH in order to check the database entry for the specific MN.

6.5.2. Evaluation of Roaming History Database

The obvious advantage of this approach is that the delay for the authentication procedures is minimized. The MN is granted access immediately if it has visited the specific network before. Another advantage is that it could minimize the processing required on AAA entities, since the AAA Request an Reply messages does not have to be sent every time a MN visits a foreign network, since the NAI value is stored with an associated lifetime. The disadvantage is however that each AAAL must implement a database to store the different MNs that visits. This is however quite simple and the advantages will probably outweigh the disadvantages. The history database solution can be a good solution to minimize the authentication delay in a mobile networking environment.

6.6. Client Identifier

The intention of the client identifier is that AAAL can identify the designated AAAH for the MN sending an AAA request to be able to carry out all necessary authorization steps. Two different identifiers are suggested for this purpose – the Network Access Identifier (NAI) and the global unicast IPv6 address of the MN [45]. Combinations of these two are also possible. Note that both identifiers may identify the user to AAAL, although this is not necessary. The user part of the identifiers may be a pseudonym intelligible only to AAAH.

The following sections present a short description of both identifiers, and then a discussion of what that could be the best-suited identifier under different circumstances is included.

6.6.1. NAI

The NAI identifier [47] was originally designed for AAA servers to identify dial-up computers in the Internet, meaning the user identification submitted during PPP authentication. This has later extended to support roaming within ISP's, but can also be used in future mobile networks supporting MIPv6 for identification purposes.

The formal definition of this identifier is *username@realm*, where realm is the users home domain name (e.g. *online.no*). The realm enables the AAAL servers in the roaming network to identify the MN AAAH server by using DNS lookups.

6.6.2. IPv6 Global Unicast

This approach is similar to NAI, but in a way it is at a lower level. The identifier is in this case structured with *userid@ip-address*, which actually is the same structure as NAI after the DNS request is carried out. There are currently not common to use this approach, but mechanisms in IPv6 renders this possible and it is therefore included.

6.6.3. Evaluation of Client Identifier

Because of the similarities between NAI and an IPv6 global unicast address identifier, the following discussion is actually based on whether identifiers needs a name service, thus readable by humans, or if it should use a routable address directly.

One obvious advantage using NAI is that the technology has been used in different systems for several years. Not all of these systems use NAI exactly as defined in the specification, but in many cases a solution that is very similar – an example of this is IMSI for GSM (see section 6.7). The format is therefore well known and the implementation is quite likely straightforward.

Naming is an issue that easily can be overlooked but is nonetheless fundamental in distributed system designs [48]. Names facilitate communication and resource sharing and are necessary because of the possibility of relocation of services – such as the AAAH server. AAAL will have to contact a DNS server to obtain the corresponding IP address to the AAAH server, thus the naming service will function as a layer above IP, supporting migration of services between different network segments. The only thing that has to be updated on migration is the DNS server's binding between the name and the object (the AAAH server).

In naming services there is a distinction between pure and non-pure names. Pure names are simply uninterpreted bit patterns, while non-pure contain information about the object that they name – in particular the location of the object. NAI is a non-pure DNS naming pattern. Despite its geographic-sounding suffix *no*, a domain such as *telenor.no* could have data located in the Irish office of Telenor, a Norwegian company. In other words, even geographic sounding domain names are conventional and are completely independent of their physical location.

The DNS infrastructure is also propagated throughout the Internet, so using NAI does not introduce new network components, thus keeps the level of complexity low, and it is therefore easy for AAAL to detect the address of AAAH. There is however a couple of considerations using naming services:

- AAAL must always be able to access a DNS to carry out the necessary AAA operations because NAI is not addressable by itself. The same discussion applies actually to the use of name services in general, and the requirements for uptime for the DNS infrastructure on the Internet are very strict (duplication of servers, etc.).
- The delay will increase some by having to make requests to the DNS looking up the corresponding address. The realm part of the NAI will, however, probably be buffered for some time at the AAAL so that the request to the DNS only have to be carried out for that specific realm once in a while (with a timeout value for each binding).

Note that all of these names must be readable by human to preserve the advantage of using NAI to ease migration of servers, since users and system administrators need to refer to the major components and configuration of distributed systems. If this is not the case, the implementation of an extra “layer” will only lead to extra overhead in the AAA operations. It should then probably use the IPv6 global unicast approach instead and be able to utilize the capability of routing directly to AAAH without using DNS – thus reducing the overhead and probably also the handover latency.

One interesting aspect with both of these identifiers is that AAAH can hide the user identity. The username/userid part does not necessarily say anything about who the user is to AAAL. It is probably not essential to know this either as long as AAAH provides the credentials needed for the MN so that it can be granted access to the network. The respective AAAH servers can therefore be the only one, except for the MN itself, which knows the client identity, so it is possible to roam between different networks in privacy. It is of course possible to let the username part of the identifier be the users real name as well, but this will reveal the user id to AAAL, that in some situations may compromise the protection of privacy for the users.

When this identifier is in transit, both from the MN to AAAL or from AAAL to AAAH, eavesdropping may occur and others can therefore get to know the IP address to AAAH. This can either happen directly if a IPv6 global unicast address identifier is used, or indirectly by looking up the address using DNS if NAI is being used. The result can be DoS attacks against AAAH so that the MN is not granted access to a foreign network. To protect from this it is essential that SAs are established between the respective parts so that the identifier is transferred encrypted.

Summing up these elements most things goes in favor of using NAI, and therefore the use of a naming service, instead of using the IP address directly in the identifier. A naming service introduces extra overhead and complexity compared to the other approach that is routable, but the flexibility due to simple migration and administration of services may outweigh these disadvantages.

6.7. GSM SIM Authentication

With reference to section 6.2.1, many security associations are needed when deploying this kind of technology. These security associations could be established with use of the GSM Subscriber Identity Module (SIM) [49] authentication procedure to authenticate the mobile users and generate keys for these. When using the SIM key exchange there will be no need for other preconfigured security associations. The GSM SIM authentication procedure is shown to demonstrate how this could be done or integrated with MIPv6 both with and without access to a global AAA infrastructure.

6.7.1. Introduction

The idea of this authentication procedure is not to use the TDMA/FDD access technology in GSM, but to use the authorization with MIPv6 over any type of link layer such as WLAN described in section 4.2.2. The Internet draft [49] describes the basic operation to this, but related to MIPv4. We have adapted this solution with modifications, to suit the specifications of MIPv6. GSM authentication is based on a challenge-response mechanism where the network has the possibility of authenticating the mobile phone. In MIPv6 there is in addition a need for enabling the MN to authenticate the network. This is discussed in section 6.8. The entire functionality of the authentication procedures in GSM is specified in GSM AKA

[50] but to understand the concepts and how this relates to MIPv6 there will be a short introduction.

An important issue is to include AAA into MIPv6 because of the roaming possibilities. The ideal solution is of course to build an AAA infrastructure into the existing IP network (include AAA functionality into the routers), but for early adoption this functionality has to be available. By exploiting the already existing accounting possibilities in GSM the AAA services does not necessarily have to be deployed everywhere before users can be authenticated because the GSM infrastructure can be used instead.

6.7.2. Authentication Procedure

The GSM SIM card is a smart card that is distributed by GSM operators and is located inside the cellular GSM telephones. The authentication algorithm that runs on this SIM can be given a 128-bit random number (RAND) as a challenge. SIM than runs an operator specific confidential algorithm with the RAND and a secret key Ki (stored in SIM) as input and produces a 32-bit response (SRES) and a 64-bit long Kc (to be used for encryption in the air interface) as output. This is an example on how the network authenticates the cellular phone.

When this method is used in MIPv6, several RAND challenges are used to generate more 64-bit Kc keys, which are combined to constitute a longer MIPv6 registration key to be used within the BU destination option. The session keys are generated and can be used when registering the MN in a foreign network. Some questions has been raised if the GSM authentication procedure is good enough, but despite the quality, we believe that it will represent a better security than the existing access security solutions that exist on the Internet today.

6.7.3. Entity Explanation

As in previous figures we use the term AAAH to denote the AAA server that can authorize each of its users, shown in Figure 6.10. Related to GSM this can be compared with the Authentication Center (AuC) that often is located together with GSM Home Location Register (HLR).

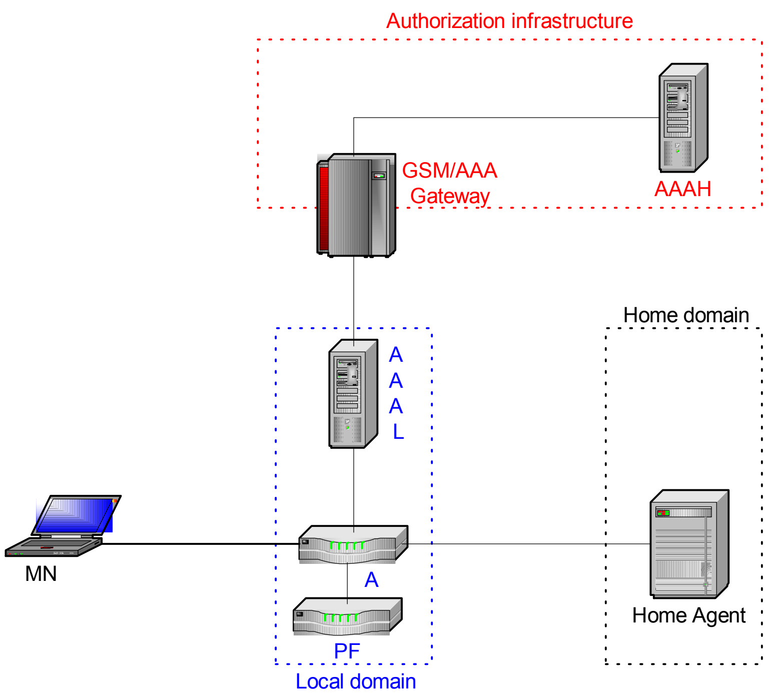


Figure 6.10 – Using the GSM infrastructure for authentication.

The AuC in GSM can authenticate their users based on the unique International Mobile Subscriber Identity (IMSI), a 15-digit number consisting of the Mobile Country Code (MCC, 3), Mobile Network Code (MNC, 2) and Mobile Subscriber Identification Number (MSIN, 10). MCC and MNC identify the specific GSM operator for the user and MSIN identifies the user. This relates to NAI in MIPv6 where the username portion identifies the user and the realm identifies the home network or the AAAH server. The elements are the same as before with the addition of the GSM/AAA gateway in the border of the Internet AAA network that can route the requests to the AAAH server.

As we can see from the figure, AAAH and the HA are in separate domains as discussed in 6.4.2. The only requirement to the local AAA infrastructure is that it can reach the appropriate GSM/AAA gateway. In the following sections it is assumed connectivity between the local AAA infrastructure and the GSM/AAA gateway. An important point is that the AAAH and the HA are in separate domains and we do not need an AAA infrastructure that spans from the local domain to the home domain of the MN. The connectivity to the GSM/AAA gateway can be used instead.

The original GSM SIM authentication procedure suggests that the MN transmit the user's IMSI as a MN-NAI extension [51] in the registration request in MIPv4. These registration messages are not present in MIPv6 so the initial MN-NAI extension will have to be transferred in the ID field of the initial AAA request. When IMSI is encoded as NAI, the username portion of it contains the IMSI as a string of digits and the realm identifies the AAAH server. By including the realm the MN can be able to select a specific AAAH server to process the AAA request instead of the AAAH server located in the GSM/AAA authorization structure. The realm portion of NAI is also used if the AAA request contains a BU as an EDO.

The figure also shows that the AAAH server and the HA can reside in different domains. GSM roaming allows this, thus TMSI insures that new VLR always finds old VLR and HLR (AuC). By sending the new NAI we can ensure that the correct

GSM/AAA gateway is found and that the appropriate AAAH server checks the AAA request.

In the draft [49] it is stated that IMSI contains enough information to identify the specific GSM operator and route the AAA request to the subscriber's home GSM operator. It is further stated that the mobile node necessarily does not need to specify the exact AAAH server in the realm part of NAI, but instead use the special NAI realm GSMSIM_NAI_REALM. This parameter is supposed to allow any GSM SIM aware AAA server to route the request to the nearest GSM/AAA gateway. This is to be performed whenever the local AAA infrastructure cannot reach the designated AAAH server for the mobile node. Local AAA servers that not are GSM SIM aware process this parameter as an ordinary AAA request.

6.7.4. Key Exchange

The structure of the messages sent in order to authenticate a MN will be the same as the ones described in 6.2.3. How the authentication and possible accounting issues is to be solved will be related to whether or not the SIM key exchange procedures has connectivity with a global AAA infrastructure or not. The exchange of messages is beyond the scope of this document. It is however important to remember that three new extensions to registration messages between the MN and the foreign agent are needed in order to authorize the mobile node and agree upon the session keys, SIM Key Request extension, SIM Key Reply extension and SRES extension. In MIPv6 we propose that this is included as extensions to the BU destination option instead. The foreign agent is not present in a MIPv6 network and it is assumed that the attendant in the visited network has the capability of generating RA and also forward all AAA requests sent from any MN.

6.7.5. Evaluation of AAA and GSM Integration

One obvious advantage when integrating a GSM/AAA gateway in the mobile AAA network environment is that we can have a centralized authorization unit for all mobile communication devices. Cellular phones, portable computers and PDAs can authenticate against the same interface, assuming that they all have a SIM card included. This can simplify the AAA network topology and enables that all AAA services for different communication entities can be related to one account through the SIM cards.

Another aspect that could be of great importance is that the GSM/AAA solution can be used in combination with the already existing infrastructure. This could imply that the GSM/AAA gateway can act as a backup solution if communication between AAAL and the designated AAAH for a MN fail to respond. The authentication can than be routed through the GSM SIM aware gateway that can authenticate the MN.

When deploying AAA services in a MIPv6 enabled networking environment, it will require a lot of work to build the proposed infrastructure shown in Figure 6.5. In order to get AAA services up and running it would be preferable if one could use already existing infrastructure in order to perform the AAA services required, e.g. using the GSM/AAA solution described above.

There are however drawbacks with the proposed solutions. One of them is that it could be difficult to integrate the authentication procedures for MNs into the existing GSM infrastructure. This means that one would have to enable all MNs to have a SIM card included, in order to send the authentication data required. In addition one would have to develop a GSM-SIM aware GSM/AAA gateway. This solution will also create more signaling load in the GSM system as the number of MNs increases, thus maybe decreasing performance for users in both systems.

The idea that one can have one centralized system that can perform AAA services for different communication equipment is a tempting fact that also can be of great value in future generation mobile systems such as UMTS. This could make AAA services easier for the different vendors, and at the same time provide a better guarantee for that the authentication of a mobile device always can be performed. Even though this would create more signaling load and could be difficult to implement, the advantages may outweigh the disadvantages since the authentication procedures in GSM already exists and is known to be working. This may lead to an easier deployment of AAA services, and we suggest that the general idea should be adopted in the further development of AAA services in future generation mobile networks. Integration against mobile cellular networks will in any case be beneficial for the future mobile Internet.

6.8. Two-way Challenge-response

A MN may share a SA with its home AAA server to allow the MN to be authenticated when roaming to different visited domains. The MIPv6 framework has defined some extensions enabling challenge response based authentication mechanisms [52]. Currently; the challenge used for the authentication is generated by the visited domain and broadcasted in Router Advertisements messages. The mobile node uses this challenge to compute authentication data when it wants to register to the network. In order to allow for an easy deployment of Mobile IP for cellular networks, the security of MIPv6 should be enhanced at least to match the level of security available nowadays in cellular networks.

For this reason the home or foreign domain must have the ability to ask a MN to provide authentication data at any time during a session. The result of this authentication determines if the session can continue or if communication must be aborted. The MIPv6 framework has defined some extensions that enable challenge-response based authentication mechanisms. This section explains the general architecture according to [52] and includes a discussion of these solutions. There is a general section that discusses the benefits and limitations of using the challenge-response mechanism with MIPv6. Furthermore a challenge specific part is included. It is important to remember that these challenges can be categorized in the following:

- The home network challenges the MN
- The foreign network challenges the MN
- The MN challenges home or foreign network

There are no specific functional differences between the messages that are sent even though different entities are authenticating each other. The only difference is which entity that initiates the other, thus which entity that demands the other to authenticate.

The reason why MIPv6 should support this mechanism is to minimize fraud. This can be that non-authorized users “steal” the identity of MNs in order to access information and resources in home and foreign networks. In addition, the MNs should be able to authenticate the network in order to prevent someone to “simulate” a router and steal identity messages and other types of user specific information provided by the MN to the router.

6.8.1. General Architecture

The general architecture is based upon Figure 6.5 and has almost the same type of messages concerning the authentication requests and replies. The figure shows that the visited domain generates the local AAA challenge used for authentication, thus broadcasting this challenge through an RA message. The mechanism provided here is instead user specific. The challenge generated by the network is directed to a particular MN. Since the random number for the challenge is changed for each operation, the proposed authentication mechanism provides a much more secure user authentication. Whether the first authentication procedure succeeded or failed, the user specific challenge authentication can serve as a double check on the authenticity of the MN. The procedure for the MN to authenticate the network is similar.

The two messages used for the purpose of challenge-response authentication in mobile IPv6 are defined as IPv6 destination options. This means that they can be sent as own packets or included in any existing packet sent. The two messages are called Mobile IP Authentication Request Extension and Mobile IP Authentication Response Extension. These are similar whether the network is challenging the MN, or vice versa.

The Mobile IP Authentication Request Extension is a request sent to the network or to the MN stating that it should authenticate. This message has a challenge that is used to compute the authentication data that is to be provided. When entities (read MN or network) receive this message, they should reply with a Mobile IP Authentication Response Extension. The message formats can be found in [52].

6.8.2. Evaluation of Two-way Challenge-response

The two-way challenge response mechanism will be discussed according to the following factors:

1. When and how often challenges should be sent
2. Why we need a two-way challenge-response
3. Quality of the authentication mechanism

The first point relates to a compromise between how often one should authenticate the different entities, and the network load. The ideal situation would be to find a solution that provides secure enough authentication without decreasing the network bandwidth for the MNs. The MNs are as described earlier (see page 29) expected to use some kind of wireless access technology that represents lower bandwidth than traditional wired media. It is assumed that every MN has a secret key with its home network.

It is essential for the MNs to authenticate, whenever they move from one administrative domain to another, so that the new network can check that the MN should be granted access. This is according to the general architecture. The MN comes within range of a new administrative domain, and the network challenges the MN. This can be done with the two-way challenge-response mechanism. It is however not clear when the MN should authenticate the network, in order to prevent that they send information snooped by fake routers. The latter can be known as message tampering.

Tampering is defined as unauthorized alteration of information meaning the interception of messages and alteration of their contents before passing them on to the intended recipient. The *man-in-the-middle* attack is a form of message tampering in which an attacker intercepts the very first message in an exchange of encryption keys to establish a secure channel. The attacker substitutes compromised keys that enable him to decrypt subsequent messages before re-encrypting them in the correct keys and passing them on.

When moving to a foreign network, the challenge from the MN to authenticate the network must be prior to, or after the network has authenticated the MN. Which challenge-response request that is executed first will be situational, but as long as both entities authenticate each other prior to granting access to each others resources, the order of which of them that authenticates the other first, is immaterial.

The other aspect that needs discussion is whether or not challenges should be used when the MNs are within range of one administrative domain. The probability that someone tampers information when the entities already are authenticated is low, but to prevent this occurrence, one could initialize a time interval for the authentication. When the lifetime of the authentication expires, the entities must again send their challenges.

In GSM they use a challenge-response mechanism. These procedures are, however, only defined so that the GSM backbone network (from AuC) can authenticate the cellular phone. The cellular phone has no possibility for authenticating the network. The motivation for this type of authentication can be that the designers have thought that there is little or no probability that someone can fake a GSM network entity, in order to steal information sent by the mobile node, which is built upon SS#7.

When considering the challenge-response authentication procedure in MIPv6, it is important to remember that the backbone network is IP. We believe that potential intruders have more knowledge about IP than they have about SS#7, thus increasing the probability for intruders to steal information in an IP based network. The MNs are also expected to have access to the Internet, and this means that it would be easier for an intruder to access a MN, than it is for an intruder to access a particular cellular phone in a GSM system. This results in higher security demands on the MNs in a MIPv6 based network. The result of these factors is that it is easier to fake a network entity in an IP based network than it is in GSM. This leads to the fact that it is also important for the MNs to be able to authenticate the network. The two-way challenge-response mechanism provides integrity for both the user and the network.

The quality of the authentication mechanism used in a challenge-response scenario will be dependent upon the computational challenge used. This is beyond the scope of this thesis and it is assumed that the computational challenge used will meet the security requirements of a future MIPv6 enabled network. One obvious advantage by using the challenge-response based authentication scheme is that it is a known security mechanism deployed and tested in many computer and telecommunication systems. The other obvious reason is that both entities in the network can authenticate the other at any given time, improving general security in the network. A disadvantage can be that the network load will increase whenever an authentication is initialized within an administrative domain. Again, there will be a compromise between the bandwidth provided to the MNs, and the security level deployed in the network.

6.9. Summary

MNs are expected to move between different networks. Different service providers for these networks will require authorization in order to be willing to do business with the client. Furthermore they will require authentication from the MNs in order to deny access from unauthorized users. The last element is accounting, since it is expected that the different network providers will not offer the requested services free of charge. These are the aspects of AAA services in a mobile networking environment such as MIPv6.

The general AAA architecture that is proposed to support these AAA services is elaborated. This method is based on the challenge-response security mechanism and provides means for the local AAA server (AAAL) in a foreign network to challenge a visiting MN in order to authenticate itself. The authentication is checked through the credentials offered by the MN. These are the client identifier, the local AAA challenge and a mechanism for preventing replay attacks. The client identifier can be the NAI value or the IPv6 global unicast address, preferably the NAI value due to its simple migration and administration of service. AAAL further forwards these credentials to the designated AAA server (AAAH) in the MNs home network, that verifies if the MN should be granted access or not. The reply is sent to AAAL, and the MN is granted access or not, based on this reply.

Different proposals can be integrated into a single, complete solution. AAA is necessary in future mobile networks, and it should therefore be integrated with different solutions for efficient handovers to provide the services demanded by users - such as real-time applications. Two solutions therefore describe how to integrate AAA services into two proposals for efficient handovers, namely Fast Handovers and HMIPv6 as elaborated in chapter 5. It is clear that such integrations will increase the delays during handovers, but some simple techniques can improve the performance – i.e. by passing on AAA credentials from the old to the new access router as long as they are under the same administrative domain.

When such an AAA network is used, the AAA infrastructure has two possibilities. The AAAH and the HA can be in the same or in different domains. When they are in different domains, there is a possibility for bundling the BU and BACK in the AAA messages in order to save time when performing handover to a foreign network. This is based on the idea that each domain must have an AAAH server, and would not be a good security solution due to a large chain-of-trust. The solution suggested here is that they should reside in different domains. This provides an AAA architecture that enables centralized security control by the means of an authorization organization. This is in general a better solution that will scale well and has a smaller chain-of-trust. An aspect that also should be taken upon consideration is how often the MNs are expected to move between different administrative domains. This is however difficult to know, and depends upon the number of MNs and the size of each administrative domain.

Furthermore the chapter explains how these AAA services could be integrated with the ones that exist for GSM in order to provide one central authentication system for all types of communication equipment. The GSM SIM authentication procedure can provide this and would make it easier to deploy AAA services in future mobile networks

The challenge-response mechanism used in GSM only enables the network to authenticate the MN. In a MIPv6 enabled network it is also important that the MN can authenticate the network, because it could be possible to act as a fake network identity and steal information provided by the MN. The order of which entity that challenges the other first for authentication is unessential as long as both entities are authenticated against each other.

7. Duplicate Address Detection

Objectives

- Describe the motivation for DAD and how it works.
- Describe how DAD may fail due to diversity in loopback semantics.
- Outline possible improvements of how a MN can obtain a new valid address if DAD fails.

Contents

7.1. Introduction

This chapter aims to introduce the concept of DAD when using stateful or stateless address autoconfiguration. DAD is the procedure that shall guarantee that all given IPv6 addresses are unique. A description of the messages (NS and NA) used for this purpose will also be explained. This section will mainly focus on DAD for the stateless address autoconfiguration approach, since this is the one we have decided to focus on in our thesis. Furthermore, there will be a discussion on the limitations in DAD followed by general suggestions for improvement.

7.2. General Architecture

The implementation chapter (4) outlines some problems with DAD based on experiences gained when testing the WLAN scenario. We believe that these errors occurred because the specific WLAN driver had some problems with the loopback semantics, but it gave an indication of some weaknesses that can lead to faulty behavior of DAD.

Neighbor solicitation and advertisement messages are used to detect duplicate addresses. If a duplicate address is discovered, the address being checked cannot be assigned to the requested interface. During the process of DAD, addresses have the status of tentative. These addresses are not considered assigned to an interface and cannot be used for regular communication. The only messages these interfaces can accept are NS and NA containing the tentative address in the target address field, used for checking the uniqueness of an address.

A node must ensure that it receives neighbor advertisements from other nodes that are already using the tentative address requested. To guarantee this, a network interface joins the all-nodes multicast address before sending a NS for the address being checked. A node must also join the solicited-node multicast address to ensure that any two nodes trying to use the same address detect each other's presence.

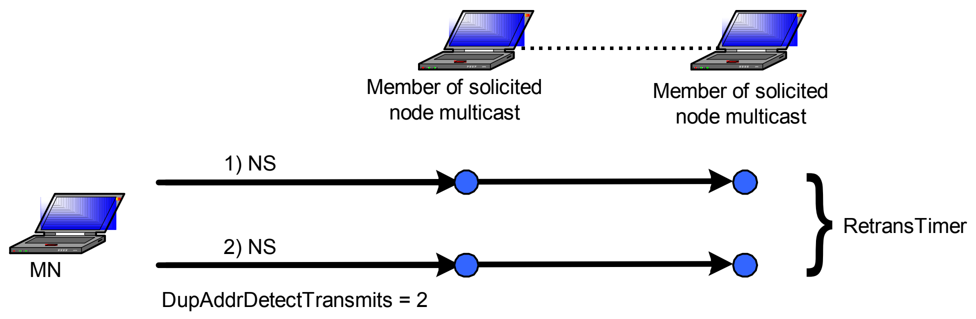
One might think that the all-nodes multicast address could be used for both the purposes mentioned above. This would however create unnecessary traffic on the network, and to eliminate this traffic, the solicited-node multicast address is created. When MNs register in a foreign network, they become a part of the solicited-nodes multicast address. This address is then used to perform the DAD check. The only nodes receiving the solicitations sent are the MNs that still not have received a preferred address. When the MNs has been assigned with a preferred address, they are no longer a part of the solicited-nodes multicast address.

In an IPv6 network, DAD is performed on all types of addresses whether they are obtained through the stateful or stateless approach. The following section aims to describe how DAD is performed and what scenarios that causes DAD to fail.

7.2.1. DAD Success

Figure 7.1 illustrates a MN performing DAD with success. DupAddrDetectTransmits, which is a variable set in the MN, specifies the number of consecutive NS messages sent during this process. A value of zero implicates that DAD is not performed on tentative addresses. Default value is one and indicates a single transmission of the NS message with no follow up retransmissions. The illustration indicates a value of two, meaning two consecutive messages.

There is also a variable named RetransTimer. This specifies the delay between consecutive NS messages (if DupAddrDetectTransmits is greater than one), as well as the time a node waits after sending the last NS before ending the DAD process.



Member of solicited node multicast
Member of solicited node multicast
1) NS
2) NS
MN
RetransTimer
DupAddrDetectTransmits = 2

Figure 7.1 – DAD success.

When the MN performs DAD, the source address of the solicitation is set to the unspecified address as shown in the figure below. In addition the target address has the status of tentative.

```
IP6: Proto = ICMP6; Len = 24
    IP6: Version = 6 (0x6)
    IP6: Traffic Class = 0 (0x0)
    IP6: Flow Label = 0 (0x0)
    IP6: Payload Length = 24 (0x18)
    IP6: Next Header = 58 (ICMP6)
    IP6: Hop Limit = 255 (0xFF)
    IP6: Source Address = ::
    IP6: Destination Address = ff02::1:ff05:449e
    IP6: Payload: Number of data bytes remaining = 24 (0x0018)
ICMP6: Neighbor Solicitation; Target = 2000:1:1:1:220:d6ff:fe05:449e
    ICMP6: Checksum = 0xFA3D
    ICMP6: Type = 135 (Neighbor Solicitation)
    ICMP6: Code = 0 (0x0)
    ICMP6: Reserved
    ICMP6: Target Address = 2000:1:1:1:220:d6ff:fe05:449e
```

Figure 7.2 – IPv6 source address is unspecified and the target address being checked.

If the source address of the solicitation is from another node, the target address is a duplicate and should not be used by either node. Since nodes have different loopback semantics, the solicitations can come from the node itself without indicating the presence of a duplicate address. NS messages used for DAD is always sent to the solicited-node multicast address. The illustration above receives no NA for the tentative address, indicating that it can assign the tentative address to one of its own network interfaces.

7.2.2. DAD Failure

There are three different scenarios where DAD will fail. Figure 7.3 shows a solicitation for a tentative address. One of the nodes in the all-nodes multicast group is already using this address, sending an advertisement back to the MN. All advertisements received with the target as a tentative address indicates a duplicate address, and the address cannot be assigned to the requested interface. The MN must than use stateful or manual configuration in order to receive a preferred address. How this configuration is done is beyond the scope of the specification [7], but we will illuminate this later in this section by proposing two different approaches that can be used.

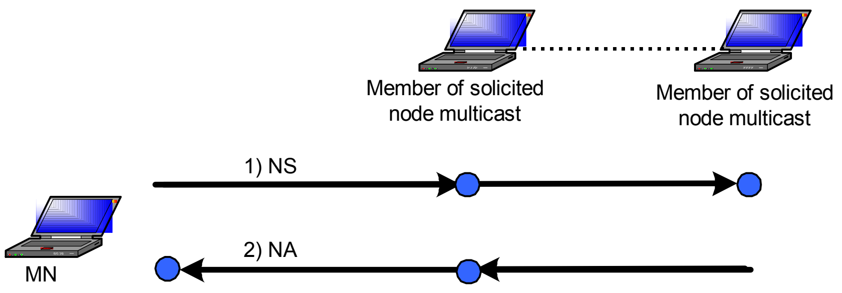

Member of solicited node multicast
Member of solicited node multicast
1) NS
2) NA
MN

Figure 7.3 – DAD failure 1.

Figure 7.4 shows another scenario where DAD fails. When a NS message is received on a network interface prior to having sent one, the tentative address checked is a duplicate and should not be used. When network interfaces become enabled, and NS is the first message being sent, they select a random delay value before they send this message. This is specified between 0 and the parameter MAX_RTR_SOLICITATION_DELAY. The reason for selecting this random value is to alleviate congestion when many nodes start up at a link on the same time, due to e.g. power failures. This may help avoiding race conditions between nodes trying to solicit for the same address at the same time.

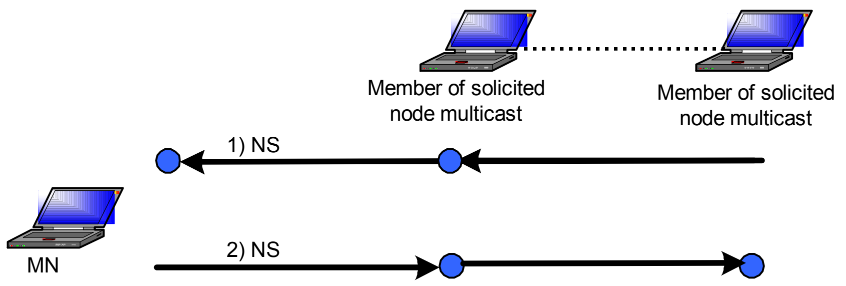

Member of solicited node multicast
Member of solicited node multicast
1) NS
MN
2) NS

Figure 7.4 – DAD failure 2.

The reason why the figure illustrates a duplicate address is that the MN and another node in the solicited-node multicast group runs DAD simultaneously, but transfer their initial solicitations at different times. The MN selects a greater random delay value and receives a solicitation for the tentative address it has planned to use. The result is that the tentative address cannot be assigned to the requested interface.

The last occurrence where DAD can fail is when the number of NS messages received is greater than the number expected, based on loopback semantics. This is the DAD failure we experienced in our WLAN scenario and is shown in Figure 7.5. The MN has in this case disabled loopback of multicast packets at the hardware level on the network interface, yet still receives a NS message for the tentative address it has determined to send a solicitation for. This means that the address is a duplicate since another node has already accessed it and the address cannot be assigned to the requested network interface. This situation can occur when to nodes are performing DAD on the same address simultaneously and transfer their initial solicitations almost at the same time. If the network interface had enabled loopback of multicast packets it would have expected duplicate receptions of NSs, but not the solicitation from another node. In this case DAD would therefore have failed anyway.

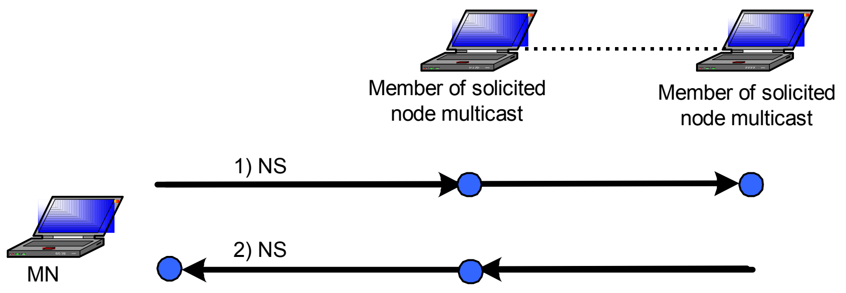

Member of solicited node multicast
Member of solicited node multicast
1) NS
MN
2) NS

Figure 7.5 – DAD failure 3.

7.3. Implementation Problems

To recap what we found out when implementing the WLAN scenario, one of two things could happen when the MN moved to a new network segment:

1. A router advertisement is received on the interface
2. A half-second timer expires

If the half-second timer expired first we got an immediate DAD failure. It also failed when the RA arrived first, but the MN recovered because the implementation initiated purging of old source addresses.

7.4. Limitations

There are a few limitations that can complicate implementations in the way DAD is specified in [7]. These issues concern various interpretations of the loopback semantics in different, but also within the same, access technologies. The specification says the following about how a node shall behave when receiving a NS message, which is equivalent to the third scenario describing DAD failure above:

"If the solicitation is from the node itself (because the node loops back multicast packets), the solicitation does not indicate the presence of a duplicate address."

Many interfaces provide a way for upper protocol layers to selectively enable and disable the looping back of multicast packets. The details of how such a facility is implemented may prevent DAD from working correctly, which is the reason that our WLAN scenario failed.

Determining whether a received multicast solicitation was looped back to the sender or actually came from another node is implementation-dependent. A problematic case occurs when two network interfaces attached to the same link happen to have the same identifier and link-layer address, and they both send out packets with identical contents at roughly the same time (e.g. a NS for a tentative address as part of DAD messages). Although a receiver will receive both packets, it cannot determine which packet was looped back and which packet came from the other node by simply comparing packet contents (i.e., the contents are identical). In this particular case, it is not necessary to know precisely which packet was looped back and which was sent by another node; if one receives more solicitations than what was sent then the tentative address is a duplicate. However, the situation may not always be this straightforward.

Service interfaces can be designed to provide a way for an upper-layer protocol to inhibit local delivery of packets sent to a multicast group that the sending host is a member of. Some applications know that there will be no other group members on the same host, and suppressing loopback prevents them from having to receive (and discard) the packets they send out. If the access technology implements and allows a facility to disable loopback at the hardware level, a node running DAD simply counts the number of NSs received for a tentative address and compares them with the number expected. If there is a mismatch, the tentative address is a duplicate.

In those cases where the hardware cannot suppress loopbacks, software heuristic to filter out unwanted loopbacks is to discard any received packet whose link-layer source address is the same as the receiving network interfaces. Unfortunately, use of that criterion also results in the discarding of all packets sent by another node using the same link-layer address. DAD will fail on interfaces that filter received packets in this manner:

- If a node performing DAD discards received packets having the same source link-layer address as the receiving network interface, it will also discard packets from other nodes using the same link-layer address – including NA and NS messages required to make DAD work correctly. This particular problem can be avoided by temporarily disabling the software suppression of loopbacks while a node performs DAD.
- If a node that is already using a particular IP address discards received packets having the same link-layer source address as the interface, it will also discard DAD-related NS messages sent by another node using the same link-layer address. Consequently, DAD will fail, and the other node will configure a non-unique address. Since it is generally impossible to

know when another node is performing DAD, this scenario can be avoided only if software suppression of loopback is permanently disabled.

Thus, to perform DAD correctly in the case where two network interfaces are using the same link-layer address, an implementation must have a good understanding of the network interfaces multicast loopback semantics, and the network interface cannot discard received packets simply because the source link-layer address is the same as the network interfaces.

7.5. DAD in a Mobile Environment

The specification of stateless address autoconfiguration establishes that DAD should be performed on all unicast addresses, whether obtained through the stateful or stateless approach. If every node in an IPv6 network had used successful stateless address autoconfiguration to configure their addresses, there would have been no need for DAD. This is because link-local addresses are generated with the help of the MNs MAC address, and the MAC address is always unique. Stateless address autoconfiguration is, however, not used in all situations, and it is assumed that some nodes will use the stateful approach of some reasons, thus supported in IPv6. To prevent conflicts between the different manually configured addresses and those created with the stateless approach, DAD is needed.

The specification of DAD states that if a node determines that its tentative link-local address is not unique, autoconfiguration stops and manual configuration of the interface is required to be able to start/continue the communication. To simplify recovery when DAD fails, it should be possible to supply the MN with an alternate interface identifier that overrides the default identifier.

The idea to supply the MN with a new interface identifier is categorized under fault-handling procedures if DAD fails for a specific address. In the following sections we suggest solutions covering this feature.

It should be noted that we believe that these proposals actually should be a part of the specification dealing with address autoconfiguration, not just a de facto solution that is optional for the administrators of the different network segments. It is important to work out a solution that will be carried out exactly the same way every time DAD fails so that a predictable address for the MN can be obtained. It is more likely, however, that mechanisms providing MNs with new addresses on DAD failure therefore only will become a de facto standard because the address autoconfiguration specifications have already reached RFC (Request For Comments), and when specifications reach this level, they are hard to change in the IETF.

7.6. Improvements

According to the previous sections, there are some initial problems related to DAD and to how it works in a mobile environment. This section will elaborate some approaches that can be used when DAD fails. The first proposal outlines a router-controlled solution delegating a new address from a predefined pool of addresses to the MN without having to carry out DAD again. The second proposal outlines a MN-controlled approach using random generation of addresses where DAD must be reprocessed.

These proposals we suggest are all based on the scenario where a MN moves into a foreign network. In order to check the uniqueness of its stateless configured address, the node must perform DAD. If DAD detects a duplicate address, the requested address cannot be assigned to the MNs interface and the connection link is broken. If the MN wants to communicate again, it must according to the specification manually configure its address with the help of the administrator of the specific domain.

7.6.1. Pool of Addresses

Performing DAD on the new address is time-consuming and causes loss in connection. This situation can however be prevented if each network segment has a pool of addresses that can be given to the MNs when DAD fails. The routers can be used to control the distribution of such addresses, which in this case must be unique because no new DAD will be carried out.

For the routers to be able to control this they can administer a pool of addresses on each network segment. If there is only a single router the administrator can configure it with a pool of addresses that it will maintain itself. If there are two or more routers on a network segment, however, they must be configured in a way so that they maintain the same pool of addresses to avoid delegation of a new duplicate address. Two different solutions supporting these scenarios will now be described.

Manual Distribution

The administrator can manually configure the routers on a network segment with an address space that they maintain themselves. E.g. if there are two routers, each of them can maintain half of the address space. In smaller networks this is a very simple approach, but in more extensive networks this may create some administrative impediments.

To illustrate how this proposal can integrated with the existing IPv6 message infrastructure, the following figure assumes that DAD has failed and the MN must obtain a new CoA according to this solution. If there is more than one router on the network segment, the MN may receive RAs from each of them, but this approach can still be used because the routers maintain different parts of the address pool. Both the RS and RA messages can utilize a new destination option that each can carry the information required, which is defined as:

1. Request Pool Address (RPA) – RS
2. Provide Pool Address (PPA) – RA

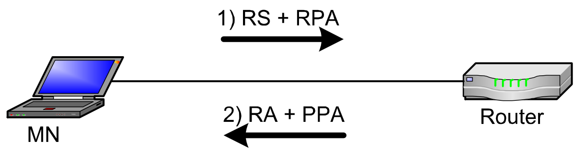


Figure 7.6 – A MN can request the router for a new address using RPA destination option.

The first option (RPA) tells the router that DAD has failed for the MN, and the router must send a RA message with an available pool address. This is done through the second destination option (PPA) that is given to the MN. The MN can then continue communication on its new link with the designated pool address as its new CoA.

Multicast Distribution

Instead of including the distribution of a new address as a part of the existing IPv6 message infrastructure, a multicast address can be used. All routers on a network segment can join the well-known multicast address when they are initiated, and if DAD fails then the MN can send request for a new pool address to the multicast address containing a similar structure as the RPA destination option. It is also possible to utilize the destination options described above, but they must be wrapped in new ICMP messages.

When the MN sends a message to the multicast address, all routers will receive the request for a new pool address. By configuring the routers with the same address space and delegate addresses incrementally, it should be possible for all routers to maintain a consistent pool of addresses even though it is distributed across several routers.

One consideration that may complicate this approach is how the routers can decide who will send a reply to the MN providing a new address (using PPA destination option). The number of routers on a single network segment is however limited, so all can probably reply without causing an “ACK-implosion”. If more than one MN uses this feature at almost the same time, the different routers can in theory pick different addresses from the pool. This can, however, quite easily be discovered if the MN compares the addresses provided from the routers. If all of them are equal, then the MN can assign the address as a new CoA. If they differ, the MN has to request for a new address once more – maybe after a random delay to avoid this situation to happen again.

Evaluation of Pool of Addresses

If it is decided to use this approach of maintaining a pool of addresses, DAD does not have to be performed because the addresses are already guaranteed to be unique by the routers. This method requires each router to maintain a list of the pool addresses that are in use at any given time. This is to prevent the router from offering the same pool address to multiple nodes. When a MN that has a pool address moves to another network, the pool address is made available again in the pool address list and the MN then reconfigures a new address by using normal stateless address autoconfiguration. Another factor that must be decided when using this approach is the size of the pool address list. This is difficult to determine, but factors such as size of the network segment, expected roaming load and the possibility for DAD failure, must be taken upon consideration when deciding this. This thesis will not elaborate these factors.

The major advantages of using the manual approach are that it is integrated into the existing message infrastructure and it provides a new address for the MN without any possibility for duplicate addresses in a simple way. The address pool must however be distributed and configured manually by the administrator, which can be a time-consuming task.

Multicast messages can also be used, but new messages must be defined for this purpose. Of this reason, and because MNs must implement semantics for how to compare and detect different pool addresses, the proposal is more complicated than manual distribution. The routers will in this case only have to be configured with the same pool of addresses and distribute these addresses to the MNs upon request. The administrators will then have an easier task when configuring the network.

7.6.2. Random Address Generation

The second approach involves random generation of a new CoA for the MN. This may be integrated with the previous approach by letting the routers build up a pool of addresses generated using random values. DAD must be performed on each of these addresses for the routers to be able to guarantee their uniqueness.

It is easier to implement this if the MN controls the random generation of a new address itself. When DAD fails it simply generates a new address using a random value, but this procedure cannot guarantee the address uniqueness, implying that

DAD must be performed again. This goes on until a unique address is found on the given network segment, but this is of course likely to happen at first try.

This method does not require a pool address list on the router and the complexity is low because no messages or destination options needs to be defined. The latency before the MN is able to communicate again will however be higher because DAD has to be performed again.

7.6.3. Evaluation of Improvements

The complexity will increase using a pool of addresses instead of generating a new address on the MN, but the solution can improve the performance when roaming between networks because DAD does not have to be carried out again – even though it has failed once. Minimizing the delay is very important and of course preferable in many situations, especially by means of performing efficient handovers where DAD has to be complete before the MN moves to a new network segment. But the deployment of such method is difficult and this can favor the MN-controlled random generated address.

As a curiosity it can be mentioned that if all NSs included a digital signature, the nodes would easily be able to detect whether the messages originated from themselves or not. None of the DAD problems described above, such as implementing different loopback semantics, would affect the execution of DAD. Addresses could in this case also be duplicate and the mechanisms described here for providing a new one is therefore necessary, but DAD would never fail unless two different nodes actually tried to use the same address.

7.7. Summary

When a MN connects to a new network segment it must obtain a new address, and DAD is used to ensure the uniqueness of this. If uniqueness is ensured then the procedure is successful. It can, however, fail in a number of scenarios, even though the probability is very small. When it fails the MN is, according to the specifications, without a valid address and it must manually configure a new address before it can continue communication.

DAD may fail even though no other nodes uses the same address because different vendors have dissimilar implementation practice of the loopback semantics on the network interface. This is consistent with what was experienced when testing mobility using WLAN in the implementation chapter, which lead to the proposed solutions of how to obtain a new address even when DAD fails.

One of these solutions is to let the routers on a network segment maintain a small pool of addresses that can be assigned to the MNs if DAD fails. These addresses have already been tested for uniqueness and can therefore be used immediately by a MN without having to perform DAD again.

Another solution is to let the MN generate a new address itself, either based on the interface identifier or through the generation of a random number. This solution is simpler than when controlled by the routers, but the MN must carry out DAD again and this will increase handover latencies

8. Conclusion

The MIPv6 specification is built in a way that makes it scale well. This is because little extra overhead is created due to the elimination of triangle routing. Some extra processing to keep track of where the MNs are currently situated are however needed, but this is within the extent that are acceptable for each node. MIPv6 is a technology that provides mobility on the IP layer, which as shown in this thesis also will be scalable.

Integrating mobility on layer-3 implicates transparency to all higher levels. This also includes the maintenance of active TCP connections and UDP port bindings. Different applications have different requirements as to what kind of handover delays they can handle. For some applications, such as http, ftp and email, MIPv6 provides the necessary functionality and service quality needed for transparent mobility as tested and documented.

For time stringent applications, a short loss in connection due to handover delays will lead to failure. For this case we believe that basic MIPv6 does not provide the service qualities needed, especially in a heterogeneous environment. Extensions to MIPv6 can be used for providing stable and efficient handovers, but mobility will in this case not be transparent for higher protocol layers unless all nodes support the same efficient handover procedure. Fast Handovers and HMIPv6 were evaluated for this purpose, and as the situation is today the prior are closer to a worldwide deployment. Fast Handovers are, however, still quite new and it will probably continue to evolve for still a couple of years.

MNs will move between different network segments, also foreign networks managed by other service providers. This means that security, by means of AAA control, must be incorporated to the mobility infrastructure. It is therefore essential that AAA services are available wherever public networks are deployed. To be able to deploy an AAA infrastructure worldwide, the number of AAA servers must be within a manageable level to minimize the chain-of-trust between AAAL and AAAH servers. A central Internet authority should delegate such responsibility to e.g. ISPs.

Another important issue for a flexible infrastructure of AAA services is to allow the HA and AAAH to reside in different domains with centralized AAA control. This reduces the chain-of-trust because such an infrastructure reduces the number of AAA servers, but it also offers flexibility due to the network topology. AAA services can also be integrated with procedures for providing efficient handovers, such as Fast Handovers and HMIPv6. The clue is here to obtain the necessary AAA credentials before connecting to the new link, utilizing the same principle as when obtaining a new valid CoA before changing network segment in the efficient handovers procedures.

The current implementations of MIPv6 revealed some problems. Many of these are probably specific for the version of MSRIPv6 stack that were used in the LAN and WLAN scenarios, so these will probably be corrected soon and provide stable mobility. On the other hand it was, during tests, discovered indications of a possible weakness in the DAD algorithm specified for IPv6. DAD may fail due to different interpretations and implementations of the loopback semantics in the network interface drivers. Some drivers are implemented in a way that a connected node will not “see” its own packets, while others do. A MN must know in advance how each driver works in order to deal with this. If DAD fails then the MN, according to the specification, must manually configure a new address to use on a network segment – but how this should be done is not standardized.

It is within this report proposed two different solutions that can be used in order to obtain a new address. The simplest approach is for the MN to generate a new address based on a random value and run DAD again, which will increase the handover latency. This should be used when the MN is only running applications requiring best effort services. Another, more complex approach, is to let the routers on a network segment maintain a pool of addresses that can be delegated to a MN upon DAD failure. This address is guaranteed to be unique so DAD does not have to be carried out again, which is an important aspect for time stringent applications. This approach should probably be used when such applications run on the MN.

Appendix A

This appendix outlines some of the major differences between IPv4 and IPv6, MIPv4 and MIPv6 according to the specifications. It also describes structural changes of how Microsoft has extended their standard IPv6 implementation to include mobility support.

A.1. IPv4 vs. IPv6

Some of the major differences between IPv4 and IPv6 are outlined in the following bullets.

- Expanded Addressing Capabilities – IPv6 increases the IP address size from 32 to 128 bits, to support more levels of addressing hierarchy, a much greater number of addressable nodes and simpler auto-configuration of addresses. A new type of address called anycast is defined, used to send a packet to any one of a group of nodes.
- Header Format Simplification – Some IPv4 header fields have been dropped or made optional, to reduce the common-case processing cost of packet handling and to limit the bandwidth cost of the IPv6 header.
- Improved Support for Extensions and Options – Changes in the way IP header options are encoded allows for more efficient forwarding, less stringent limits of the length of options, and greater flexibility for introducing new options in the future.
- Flow labeling capability – A new capability is added to enable the labeling of packets belonging to particular traffic “flows” for which the sender requests special handling, such as non-default quality of service or “real-time” service.
- Authentication and Privacy Capabilities – Extensions to support authentication, data integrity and data confidentiality (optional) are specified for IPv6.

A.2. MIPv4 vs. MIPv6

This section provides an overview of the features that are different between Mobile IPv4 and Mobile IPv6. The design of Mobile IP in IPv6 represents a natural combination of the experiences gained from the development of Mobile IP in IPv4, together with the opportunities provided by the design and deployment of the IPv6 itself and the new protocol features offered. Mobile IPv6 thus shares many features with Mobile IPv4, but the protocol is now fully integrated into IP and provides many improvements over Mobile IPv4. This section summarizes the major differences between Mobile IPv4 and Mobile IPv6:

- Support for what is known in Mobile IPv4 as "Route Optimization" is now built in as a fundamental part of the protocol, rather than being added on as an optional set of extensions that may not be supported by all nodes as in Mobile IPv4. This integration of Route Optimization functionality allows direct routing from any correspondent node to any mobile node, without needing to pass through the mobile node's home network and be forwarded by its home agent, and thus eliminates the problem of "triangle routing" present in the base Mobile IPv4 protocol. The Mobile IPv4 "registration" functionality and the Mobile IPv4 Route Optimization functionality are performed by a single protocol rather than two separate (and different) protocols.

- Support is also integrated into Mobile IPv6 and into IPv6 itself, for allowing mobile nodes and Mobile IP to coexist efficiently with routers that perform "ingress filtering". A mobile node now uses its care-of address as the Source Address in the IP header of packets it sends, allowing the packets to pass normally through ingress filtering routers. The home address of the mobile node is carried in the packet in a Home Address destination option, allowing the use of the care-of address in the packet to be transparent above the IP layer. The ability to correctly process a Home Address option in a received packet is required in all IPv6 nodes, whether mobile nor stationary, whether host or router.
- The use of the care-of address as the Source Address in each packet's IP header also simplifies routing of multicast packets sent by a mobile node. With Mobile IPv4, the mobile node had to tunnel multicast packets to its home agent in order to transparently use its home address as the source of the multicast packets. With Mobile IPv6, the use of the Home Address option allows the home address to be used but still be compatible with multicast routing that is based in part on the packet's Source Address.
- There is no longer any need to deploy special routers as "foreign agents" as are used in Mobile IPv4. In Mobile IPv6, mobile nodes make use of IPv6 features, such as Neighbor Discovery and Address Autoconfiguration, to operate in any location away from home without any special support required from its local router.
- Unlike Mobile IPv4, Mobile IPv6 utilizes IP Security for all security requirements (sender authentication, data integrity protection, and replay protection) for Binding Updates (which serve the role of both registration and Route Optimization in Mobile IPv4). Mobile IPv4 relies on its own security mechanisms for these functions, based on statically configured "mobility security associations".
- The movement detection mechanism in Mobile IPv6 provides bi-directional confirmation of a mobile node's ability to communicate with its default router in its current location (packets that the router sends are reaching the mobile node, and packets that the mobile node sends are reaching the router). This confirmation provides a detection of the "black hole" situation that may exist in some wireless environments where the link to the router does not work equally well in both directions, such as when the mobile node has moved out of good wireless transmission range from the router. The mobile node may then attempt to find a new router and begin using a new care-of address if its link to its current router is not working well. In contrast, in Mobile IPv4, only the forward direction (packets from the router are reaching the mobile node) is confirmed, allowing the black hole condition to persist.
- Most packets sent to a mobile node while away from home in Mobile IPv6 are sent using an IPv6 Routing header rather than IP encapsulation, whereas Mobile IPv4 must use encapsulation for all packets. The use of a Routing header requires less additional header bytes to be added to the packet, reducing the overhead of Mobile IP packet delivery. To avoid modifying the packet in flight, however, packets intercepted and tunneled

by a mobile node's home agent in Mobile IPv6 must still use encapsulation for delivery to the mobile node.

- While a mobile node is away from home, its home agent intercepts any packets for the mobile node that arrive at the home network, using IPv6 Neighbor Discovery rather than ARP as is used in Mobile IPv4. The use of Neighbor Discovery improves the robustness of the protocol (e.g. due to the Neighbor Advertisement "override" bit) and simplifies implementation of Mobile IP due to the ability to not be concerned with any particular link layer as is required in ARP.
- The use of IPv6 encapsulation (and the Routing header) removes the need in Mobile IPv6 to manage "tunnel soft state", which was required in Mobile IPv4 due to limitations in ICMP for IPv4. Due to the definition of ICMP for IPv6, the use of tunnel soft state is no longer required in IPv6 for correctly relaying ICMP error messages from within the tunnel back to the original sender of the packet.
- The dynamic home agent address discovery mechanism in Mobile IPv6 uses IPv6 anycast and returns a single reply to the mobile node, rather than the corresponding Mobile IPv4 mechanism that used IPv4 directed broadcast and returned a separate reply from each home agent on the mobile node's home link. The Mobile IPv6 mechanism is more efficient and more reliable, since only one packet need be sent back to the mobile node. The mobile node is less likely to lose one of the replies because no "implosion" of replies is required by the protocol.
- Mobile IPv6 defines an Advertisement Interval option on Router Advertisements (equivalent to Agent Advertisements in Mobile IPv4), allowing a mobile node to decide for itself how many Router Advertisements (Agent Advertisements) it is willing to miss before declaring its current router unreachable.
- The use of IPv6 destination options allows all Mobile IPv6 control traffic to be piggybacked on any existing IPv6 packets, whereas in Mobile IPv4 and its Route Optimization extensions, separate UDP packets were required for each control message.

A.3. From IPv6 to MIPv6

In order to allow for greater flexibility, the Mobile IPv6 implementation was designed to allow for the simultaneous use of multiple home addresses. The implementation maintains a list of currently active home addresses, known as the Home Address List, which consists of zero or more Home Address Entries (HAEs). Each HAE maintains the appropriate IPv6 home address and prefix length, together with a reference to the NTE (read as the source address) matching that home address (and the physical interface if the home address is currently bound to one). Each HAE also maintains a set of flags and an individual binding update list. The NTE field is used primarily by the RouteToDestination code during Route Cache Entry (RCE) construction.

The most fundamental extension to the existing IPv6 code is the Route Cache. The purpose of the cache is to maintain state on a particular route from an IPv6 node *to* a specific destination, in order to speed up the process of packet transmission and forwarding. The cache maintains a RCE for each active route. In order to efficiently support the transmission of packets *from* a MN, further information needs to be encoded within the route cache – access to both the home address and care-of address to be used for packet sourcing is required. This additional information is added through the use of a second RCE reference within the RCE structure.

Smaller additions where also added to be able to support MIPv6. Functions were e.g. added to support the reception of the Mobile IPv6 binding acknowledgement and binding request destination options. Movement detection is achieved by listening to IPv6 router advertisement messages. If a RA is received which does not match the previous one, then a handover is deemed to have occurred. Once a new address has been acquired and any necessary DAD has taken place, then the binding update lists are traversed and binding update messages are sent to all correspondent nodes found there.

When home agents receive home registrations, the binding update is validated to guarantee that this node is capable of acting as a home agent for the mobile node originating the binding update. This is achieved by ensuring that the node has at least one unicast IPv6 address bound to an interface which corresponds to the same IPv6 network as the home address supplied in the binding update message.

In order to provide the IPv6 address proxy functionality required by home agents, a new address type, the Proxy Address Entry (or PAE) has been defined. When a MN requests a home registration for a specified home address, a PAE corresponding to that home address is added to the relevant network interface on the HA, and the relevant IPv6 neighbor discovery multicast groups are joined. PAEs are removed from interfaces upon the removal of the corresponding entry from the binding cache (either explicitly via a registration, or indirectly via a cache expiry). Upon reception of a packet destined for a PAE, the home agent tunnels this packet to the mobile node’s care-of address, as specified in the binding cache.

Appendix B

The following sections explain the IPv6 router commands used in our implementations. Furthermore it describes the equipment in LAN and WLAN test scenarios in addition to the router configuration scripts.

B.1. Router Commands Explanations

```
ipv6 if [if#]
```

Displays information about interfaces. If an interface number is specified, information about only about that interface is displayed. Otherwise, information about all interfaces is displayed. The output includes the interface's link-layer address and the list of IPv6 addresses assigned to the interface. It includes the interface's current MTU and the maximum (true) MTU that the interface can support.

```
ipv6 ifc [forwards] [advertises] [-forwards] [-advertises] [mtu #bytes]
[site site-identifier]
```

Controls interface attributes. Interfaces can be forwarding, in which case they forward packets whose destination address is not assigned to the interface. Interfaces can be advertising, in which case they send router advertisements. These attributes can be independently controlled. An interface either sends router solicitations and receives router advertisements, or receives router solicitations and sends router advertisements.

```
ipv6 adu if#/address [lifetime VL[/PL]] [anycast] [unicast]
```

Adds or removes a unicast or anycast address assignment on an interface, defaulting to unicast unless anycast is specified. If lifetime is not specified, it is infinite. If only a valid lifetime is specified, then the preferred lifetime is equal to the valid lifetime. An infinite lifetime may be specified, or a finite value in seconds. The preferred lifetime must be less than or equal to the valid lifetime. Specifying a lifetime of zero causes the address to be removed.

```
ipv6 mipu [MN CN HA]
```

Set the operations on the node. You can set MN for mobile node, CN for correspondent node and HA for home agent.

```
ipv6 rt
```

Displays the current contents of the routing table. For each routing table entry, displays the route prefix, an on-link interface or a next-hop neighbor on an interface, a preference value (smaller is preferred), and a lifetime in seconds. Routing table entries might also have publish and aging attributes. By default, they age (the lifetime actually counts down, instead of remaining constant) and are not published (they are not used in constructing router advertisements). On hosts, routing table entries are normally auto-configured from router advertisements.

```
ipv6 rtu prefix if#[/nexthop] [lifetime L] [preference P] [publish] [age]
[spl site-prefix-length]
```

Adds or removes a route in the routing table. The route prefix is not optional. The prefix can either be on-link to a specified interface or it can be next-hop, specified with a neighbor address on an interface. The route can have a lifetime in seconds (the default is infinite) and a preference (the default is zero, or most preferred). Specifying a lifetime of zero causes the route to be deleted.

If the route is specified as published, meaning it will be used in constructing router advertisements, and then by default it does not age. The route's lifetime does not count down, so it is effectively infinite, but the value does get used in router advertisements. Optionally, a route can be specified as a published route that also ages. A non-published route by default always ages.

B.2. LAN Equipment List

- Two Dell Inspiron 3800 mobile nodes
- Two 3COM Megahertz 10/100 LAN PCMCIA adapters (Used in MNs)
- One Compaq Netelligent 1005 10BASE-T HUB (Hub 2)
- One CentreCOM MR820TR HUB (Hub 1)
- One CINET PPI-600 with 128MB RAM as HA and W2000 router

B.3. LAN Router Configuration Script

```
rem @echo off
rem ----------------------------------------------------
rem configure Windows 2000 router
rem ----------------------------------------------------

set H1=2000:1:1:1:a00:9ff:fea8:cc39  // The card connected to Hub1
set H2=2000:2:2:2:2a0:24ff:fef8:6e03 // The card connected to Hub2
set R2L0=250:4ff:feb5:17ca  // This is MN1
set R2L1=250:4ff:fef2:5d84  // This is MN2

set H1MN=2000:1:1:1::7 //Setting the prefix on the segment connected to Hub1
set H2MN=2000:2:2:2::7 //Setting the prefix on the segment connected to Hub2

rem -----------------------------------------------------
rem restart IPv6 stack so IF numbers assigned in expected order
rem note this script expects the IF numbers assigned as follows:
rem    IF        MAC        addr (assigned below)
rem    9         ..7f       2000:1:1:1
rem    8         ..68       2000:2:2:2
rem -----------------------------------------------------
net stop tcpip6
net start tcpip6

rem -----------------------------------------------------
rem enable Home Agent and Correspondent Node operation
rem -----------------------------------------------------
ipv6 mipu CN HA

rem -----------------------------------------------------
rem    configure interfaces for forwarding and router advertisments
rem    forwards means will forward packets not addressed to that IF
rem    advertises means will send router advertisements
rem    (do 9 twice, often first call is too soon)
rem -----------------------------------------------------

ipv6 ifc 9 forwards advertises
ipv6 ifc 8 forwards advertises
ipv6 ifc 9 forwards advertises

rem -----------------------------------------------------
rem assign static addresses to interfaces
rem     note default lifetime is infinite
rem -----------------------------------------------------

rem works fine but just use SAD pro tem to keep IPsec config simple
rem ipv6 adu 9/2000:1:1:1::9
rem ipv6 adu 8/2000:2:2:2::9

rem -----------------------------------------------------
rem add route to each link
rem -----------------------------------------------------
ipv6 rtu 2000:1:1:1::/64 9 pub
ipv6 rtu 2000:2:2:2::/64 8 pub

rem -----------------------------------------------------
rem    publish a default route
rem    note impl requires this else advertised router has
rem    a lifetime of zero
rem -----------------------------------------------------
ipv6 rtu ::/0 9/2000:1:1:1::9 pub
```

B.4. WLAN Equipment List

- Two Dell Inspiron 3800 mobile nodes
- One Breezecom WLAN C3 PCMCIA adapters (Used in MN2)
- One 3COM Megahertz 10/100 LAN PCMCIA adapters (Used in MN1)
- One Compaq Netelligent 1005 10BASE-T HUB (Hub 2)
- One CentreCOM MR820TR HUB (Hub 1)
- Two Breezecom WLAN AP A3 10 Pro.11
- One CINET PPI-600 with 128MB RAM as HA and W2000 router

B.5. WLAN Router Configuration Script

```
rem @echo off
rem --------------------------------------------------------
rem configure Windows 2000 router
rem --------------------------------------------------------

set H1=2000:1:1:1:a00:9ff:fea8:cc39  // The card connected to Hub1
set H2=2000:2:2:2:2a0:24ff:fef8:6e03 // The card connected to Hub2
set R2L0=220:d6ff:fe05:30f6  // This is MN1
set R2L1=220:d6ff:fe05:449e  // This is MN2

set H1MN=2000:1:1:1::7 //Setting the prefix on the segment connected to Hub1
set H2MN=2000:2:2:2::7 //Setting the prefix on the segment connected to Hub2

rem ---------------------------------------------------------
rem restart IPv6 stack so IF numbers assigned in expected order
rem note this script expects the IF numbers assigned as follows:
rem    IF        MAC        addr (assigned below)
rem    9         ..7f       2000:1:1:1
rem    8         ..68       2000:2:2:2
rem ---------------------------------------------------------
net stop tcpip6
net start tcpip6

rem ---------------------------------------------------------
rem enable Home Agent and Correspondent Node operation
rem ---------------------------------------------------------
ipv6 mipu CN HA

rem ---------------------------------------------------------
rem    configure interfaces for forwarding and router advertisments
rem    forwards means will forward packets not addressed to that IF
rem    advertises means will send router advertisements
rem    (do 9 twice, often first call is too soon)
rem ---------------------------------------------------------

ipv6 ifc 9 forwards advertises
ipv6 ifc 8 forwards advertises
ipv6 ifc 9 forwards advertises

rem ---------------------------------------------------------
rem assign static addresses to interfaces
rem     note default lifetime is infinite
rem ---------------------------------------------------------

rem works fine but just use SAD pro tem to keep IPsec config simple
rem ipv6 adu 9/2000:1:1:1::9
rem ipv6 adu 8/2000:2:2:2::9

rem ---------------------------------------------------------
rem add route to each link
rem ---------------------------------------------------------
ipv6 rtu 2000:1:1:1::/64 9 pub
ipv6 rtu 2000:2:2:2::/64 8 pub

rem ---------------------------------------------------------
rem    publish a default route
rem    note impl requires this else advertised router has
rem    a lifetime of zero
rem ---------------------------------------------------------
ipv6 rtu ::/0 9/2000:1:1:1::9 pub
```

Appendix C

This appendix captures important issues of how to configure application level services with MIPv6 support. It also describes compilation and debugging procedures of the MSRIPv6 stack.

C.1. Machine Configuration

Configuration 1: To prepare a machine to run IE5.5 using the mobile IPv6 stack, proceed as follows:

- Install Windows2000 free (retail) OS
- Install Windows2000 Service Pack 1
- Install IE5.5 Service pack 1
- Install Windows2000 Technology Preview [32]
- Install mobile IPv6 stack (checked build) [32]

If incorrect versions of MSRIPv6 and the IPv6 technology preview are installed, or they are installed in incorrect order, it may lead to using library files that are not compatible. On completion, check that the system contains the expected version of *wininet.dll* (438KB, version 5.58.1.1) and *wship6.dll* (42KB). This machine can be used to run IE5.5 and *tlntsvr.exe*. If these files are not of right size the whole installation procedure can be done all over again or these files can be replaced manually as described later in this appendix (G.3).

Configuration 2: A second useful configuration provides an HTTP server and a telnet client:

- Install Windows2000 free (retail) OS
- Install Windows2000 Service Pack 1
- Install IE5.5 Service pack 1
- Install mobile IPv6 stack (checked build) [32]
- Install and configure Fnord! http server [35]
- Copy *telnet.exe* from technology preview machine (configuration 1)

C.2. Telnet Mobility Support

Telnet Server

In order to use this service on a Windows 2000 computer supporting MIPv6 and the telnet application, the following procedure must be carried out:

- Click on Start, Settings and Control Panel
- Double click on Administrative tools
- Double click on Telnet Server Administration.

The following menu will then appear:

```
Microsoft (R) Windows 2000 (TM) (Build 2195)
Telnet Server Admin (Build 5.00.99201.1)

Select one of the following options:


0) Quit this application
1) List the current users
2) Terminate a user session ...
3) Display / change registry settings ...
4) Start the service
5) Stop the service

Type an option number [0 - 5] to select that option:
```

Figure C.1 – Telnet server configuration menu.

To start the service *option 4* must be chosen. To let other users connect to your telnet service, *option 3* must also be chosen, followed by *option 7* NTLM, and set the NTLM value to zero.

Telnet Client

When IPv6 technology preview is installed, the default telnet client application is replaced with a version that can connect both to IPv4 and IPv6 servers. This means that users can use the standard command prompt with both IPv4 and IPv6 addresses, but this will not be elaborated here.

C.3. IPv6 Support in IE5.5

The file *wininet.dll* is normally write-protected because it is in use by Windows, but there is a work-around to replace the old file with another version manually. When a file is write-protected it cannot just be copied into the directory.

- If the IPv6 technology preview is used, do the following steps. If not move on to the next paragraph.
    - Extract *tpipv6-xxxxxx.exe* to a temporary directory (self-extracting archive)
    - Find *setup.exe* in the temporary files you just extracted and extract this file to a temporary folder as well.
- Locate *wininet.dll* and rename the file to *wininet.new* and copy it into the *%systemroot%\system32* folder.
- Start the command prompt and change the directory to *%systemroot%\system32* and execute the following commands:
    - *ren wininet.dll wininet.old*
    - *copy wininet.new wininet.dll* (accept the overwrite question)
- Check if the *wininet.dll* file is 438KB.
- Close all windows and restart the computer.

Visit http://ipv6.research.microsoft.com/default.htm to see if Internet Explorer is enabled for IPv6 addresses or connect to another IPv6 enable web server by using an IPv6 address directly. Remember to specify the IPv6 address within closed ([]) brackets, e.g. http://[fe80::2a0:24ff:fe5a:4c9f]/.

C.4. Connecting to 6bone

A computer set up with configuration 1 described in the implementation section 4.5 can quite easily be connected to the 6bone and access IPv6 enabled web pages.

6to4 is a method for connecting IPv6 hosts or sites over the existing IPv4 Internet infrastructure. It uses a unique address prefix to give isolated IPv6 sites their own IPv6 address space. 6to4 is like a "pseudo-ISP" providing IPv6 connectivity. You can use 6to4 to communicate directly with other 6to4 sites. You can also use a 6to4 relay to communicate with 6bone sites. 6to4 does not require the use of IPv6 routers and its IPv6 traffic is encapsulated with an IPv4 header.

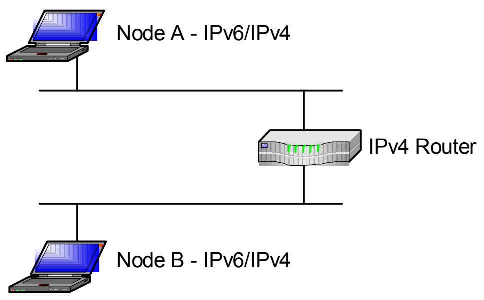


Figure C.2 – Configuration of two nodes on separate subnets using 6to4 to communicate across an IPv4 router.

The main requirement for using 6to4 is one globally routable IPv4 address for your site. Suppose that your site consists of a collection of IPv6 computers that you manage (some running the Microsoft IPv6 protocol and some running other IPv6 implementations). Assume also that all IPv6 computers are directly connected using Ethernet or 6-over-4. The globally routable IPv4 address must be assigned to one of your computers running the Microsoft IPv6 protocol. This computer will be your 6to4 gateway.

In this example the address of the 6to4 gateway is 128.39.202.214. You need your own globally routable IPv4 address to use 6to4.

Use *ipv6.exe* to enable 6to4 on your 6to4 gateway computer:

```
ipv6 rtu 2002::/16 2
```

The *ipv6 rtu* command performs a routing table update. It can be used to add, remove, or update a route. In this case it is enabling 6to4. The 2002::/16 argument is the prefix of the route, specifying the unique 6to4 prefix. The 2 argument specifies the on-link interface for this prefix. Interface #2 is the "pseudo-interface" used for configured tunnels, automatic tunneling, and 6to4. When an IPv6 destination address matches the 2002::/16 prefix, the 32 bits that follow the prefix in the destination address are extracted to form an IPv4 destination address. The packet is encapsulated with an IPv4 header and sent to the IPv4 destination address.

Configure a 6to4 address on your 6to4 gateway computer:

```
ipv6 adu 2/2002:8027:cad6::8027:cad6
```

The *ipv6 adu* command performs an address update. It can be used to add, remove, or update an address on an interface. In this instance, it is configuring the computer's 6to4 address. The 2/2002:8027:cad6::8027:cad6 argument specifies both the interface and the address. It requires configuring address 2002:8027:cad6::8027:cad6 on interface #2. The address is created using the site prefix 2002:8027:cad6::/48, subnet 0 to give a subnet prefix 2002:8027:cad6::/64, and a 64-bit interface identifier. The convention demonstrated uses the computer's IPv4 address for the interface identifier for addresses assigned to interface #2. For your use, 8027:cad6 should be replaced by the hexadecimal encoding of your own globally routable IPv4 address.

The above two commands are sufficient to allow communication with other 6to4 sites. For example, you can try pinging the Microsoft 6to4 site:

```
ping6 2002:836b:9820::836b:9820
```

To enable communication with the 6bone, you must create a default-configured tunnel to a 6to4 relay. You can use the Microsoft 6to4 relay router, 131.107.152.32:

```
ipv6 rtu ::/0 2/::131.107.152.32 pub life 1800
```

The *ipv6 rtu* command performs a routing table update, establishing, in this instance, a default route to the 6to4 relay. The ::/0 argument is the route prefix. The zero-length prefix indicates that it is a default route. The 2/::131.107.152.32 argument specifies the next-hop neighbor for this prefix. It requires that packets matching the prefix are forwarded to address ::131.107.152.32 using interface #2. Forwarding a packet to ::131.107.152.32 on interface #2 causes it to be encapsulated with a v4 header and sent to 131.107.152.32. The pub argument makes this a published route. Because this is only relevant for routers, it has no effect until routing is enabled. Similarly, the 30-minute lifetime pertains only if routing is enabled. You should be able to access 6bone sites as well as 6to4 sites. You can use the following command to test this:

```
ping6 3ffe:1cff:0:f5::1
```

Internet Explorer cannot browse IPv6 web sites if it is configured to use a proxy server. When Internet Explorer is configured to use a proxy server, all name resolution requests for web sites are forwarded to the proxy server. Until the proxy server is IPv6-enabled, proxy-based requests for local or remote IPv6 Web pages will not work. To disable Internet Explorer's use of a proxy server:

- In Internet Explorer, click Tools, then Internet Options.
- In the Internet Options dialog box, click the Connections tab, and then click LAN Settings.
- In the Local Area Network (LAN) Settings dialog box, clear the Use a proxy server checkbox and click OK.
- Click OK to save changes to Internet options.

Internet Explorer will use IPv6 to download web pages in the following circumstances:

- The Domain Name System (DNS) query for the name of the Web server in the URL returns an IPv6 address.
- The URL is using the format for literal IPv6 addresses as described in RFC 2732. A literal IPv6 address in a URL is the address enclosed in square brackets. For example, to reach the web server at the IPv6 address of 2010:836B:4179::836B:4179, the URL is http://[2010:836B:4179::836B:4179].

For a list of IPv6-only Web sites, see http://www.ipv6.org/other-sites.html. To access these sites using a literal IPv6 address, use the ping6 tool at the command prompt to ping the site DNS name and obtain the IPv6 address. Then substitute the DNS name portion of the URL with the IPv6 address enclosed in brackets.

C.5. Compilation and Debugging

In order to undertake a more thorough analysis of the MIPv6 implementation, it is necessary to modify the code to make improvements and detect scarcities, but the level of entrance compiling is quite high compared to utilizing the precompiled MSRIPv6 implementations.

To be able to compile a new stack, some applications available through the Microsoft Developer Network (MSDN) software distribution should be installed. MSDN contains among other things a checked build version of Windows2000 workstation, kernel debuggers, network monitor (NetMon), platform SDK and DDK. Some of these applications are necessary when rebuilding and testing the stack, while others are optional.

A *checked build version of Windows 2000* is much more rigorous in checking the consistency of kernel data structures than the standard "free" build, for example it will halt if a kernel assertion fails and it will halt if an application try to obtain a kernel lock that it already hold. In almost all cases the programmers are better off knowing about such things, as they are usually the result of a coding error that will probably matter soon enough. It should be noted, however, that a checked build is considerably larger and slower than a free build. Also, if you are running a checked build with kernel debugging enabled it will allow you to do something about any break points hit in the kernel (if the kernel stops on a debug break-point it will do nothing apart from wait for you to give it a debugging command).

When developing and compiling protocols in Windows 2000, Microsoft recommend using this checked build version, but it proved to be so slow and unstable that it in our case was more to loose than to gain. Our development therefore took place on a free build Windows 2000 version after experimenting with the checked build version.

In order to prepare a development environment, *MS Visual Studio*, *MS Platform SDK* and *MS Windows2000 DDK* must be installed on the development machine. You also need to have a working copy of the MSRIPv6 implementation available:

- Download a self-extracting precompiled release of the implementation [32]
- Execute the file and extract it to *d:\MIPv6*
- In this directory you will now find a new folder called *NT-bin* that will be your working copy

Once you have a working Windows2000 DDK you can rebuild the core stack by following these steps:

- Download the self-extracting source distribution of the MSRIPv6 implementation [32]
- Execute the file and extract it to *d:\MIPv6\NT-src*
- The DDK cannot default find the library files necessary for compilation, so the environment variables need to include this. DDK is in the following example installed in *d:\programs\NTDDK*, and the environment variables can then be found in *d:\programs\NTDDK\bin\setenv.bat* – open the file and find these lines:

  ```
  set BUILD_ALT_DIR=chk
  set NTDBGFILES=
  set NTDEBUG=ntsd
  set NTDEBUGTYPE=both
  set MSC_OPTIMIZATION=/Od /Oi
  :done
  ```

  Add the following line:

  ```
  set INCLUDE=D:\programs\NTDDK\inc\ddk;%INCLUDE%
  ```

  The next line will then be:

  ```
  set NEW_CRTS=1
  ```

- Start a DDK console windows (Start->Programs-> Development Kits->Windows 2000 DDK->Checked Build Environment). This command prompt loads the necessary environment variables for compiling.
- Change to the directory holding the core IPv6 stack, e.g. *cd d:\MIPv6\NT-src\tcpip6*
- Rebuild the stack using *build –cZ*
- If successful, the rebuilt stack appears in *d:\MIPv6\NT-src\tcpip6\common\objchk\i386\tcpip6.sys* (verify by checking date and time)
- Copy the new *tcpip6.sys* into your working copy MIPv6 directory
- Reinstall the stack on Windows 2000 from your working copy directory (see [MIPv6Inst] for details on how to install the MIPv6 stack)

MIPv6Conf.exe is a utility that can be used to control *MIPv6Service.exe*, the MIPv6 Configuration Service, and these can also be rebuilt in Visual Studio:

- Download the self-extracting source files [MSRDown#3]
- Execute the file and extract it to *d:\MIPv6\MIPv6Progs*
- The *MIPv6Progs\src* directory contains the source for *MIPv6Conf.exe* – copy the source to a directory in *d:\MIPv6* called *MIPv6Conf*
- The *MIPv6Progs\bin* directory contains the source for *MIPv6Service.exe* – copy the source to a directory in *d:\MIPv6* called *MIPv6Service*
- From Visual Studio select File->OpenWorkspace, browse to your target directory and select the *MIPv6Conf.dsw* (*MIPv6Service.dsw*) file
- Choose Project->Settings and adjust the C/C++ path for the location of the directory where you have installed your DDK environment (e.g. *d:\programs\NTDDK\inc*).

This part can be a little tricky because the Project Options string is quite obscure. If you have used the same paths as described in these installation procedures, verify that the string looks like this:

```
/nologo /MD /W3 /GX /O2 /I "." /I "../MIPv6Service" /I
"d:/programs/ntddk/inc" /D "WIN32" /D "NDEBUG" /D "_WINDOWS" /D "_AFXDLL"
/D "_MBCS" /Fp"Release/MIPv6Conf.pch" /YX"stdafx.h" /Fo"Release/"
/Fd"Release/" /FD /I /NTDDK/inc " " /c
```

- Choose Project->Settings and adjust the Link path for the lib directory within the directory where you have installed the binary version of the distribution, e.g. *d:/MIPv6/NT-bin/lib/wship6.lib*

The same obscurity applies to the Project Options string here. Verify that the string looks like this:

```
ws2_32.lib shlwapi.lib d:/MIPv6/NT-bin/lib/wship6.lib /nologo
/subsystem:windows /incremental:no /pdb:"Release/MIPv6Conf.pdb"
/machine:I386 /out:"Release/MIPv6Conf.exe"
```

- Choose Build->BuildMIPv6Conf.exe (BuildMIPv6Service.exe);

For normal evaluation of the MSRIPv6 protocol it is not necessary to recompile the stack. We, however, were forced to do this because of the WLAN scenario described above did not behave as intended. By modifying the source, like adding new debug printouts, we made a better understanding of how the code is built up and how the dependencies between the different modules are. This was necessary to identify what went wrong in the WLAN scenarios.

Appendix D

These are the two html pages used to demonstrate transparent mobility with the http protocol.

D.1. page1.html

```
<html>
<head>
<meta http-equiv="refresh" content="1;url='page2.htm'">
</head>
<p><font color=blue size=50>Page 1.</font></p>
<p><a href="http://[2000:2:2:2:250:4ff:fef2:5d84]/page2.htm">Link to
Page2</a></p>
</html>
```

D.2. page2.html

```
<html>
<head>
<meta http-equiv="refresh" content="1;url='page1.htm'">
</head>
<p><font color=red size=50>Page 2.</font></p>
<p><a href="http://[2000:2:2:2:250:4ff:fef2:5d84]/page1.htm">Link to
Page1</a></p>
</html>
```

Appendix E

This appendix elaborates other message improvements to the Fast Handovers procedure than what are described in chapter 5. Changes that can be done, especially in order to simplify the specification and reduce the overhead are discussed.

E.1. PrRtAdv – New CoA Option

PrRtAdv is sent to the MN unsolicited if the handover is controlled by the network, or upon request if controlled by the MN. The message contains information about the newAR, such as the newCoA to the MN.

The newCoA can be inserted in an option field in the PrRtAdv message sent to the MN when moving to a new link, but this is not required in the stateless approach. If it does not contain this option, the mobile must construct a newCoA out of the interface ID and the prefix (extracted from the prefix information option – see section 3.2.1). If the option is included, this means that the MN must use the address attached and must be used when stateful address configuration is used. It can also be used in the stateless approach to be sure that the oldAR and the MN computes the same newCoA.

Attaching the newCoA in the stateful approach is understandable because this is the only way the MN can obtain its new address before changing to the newAR. In the stateless approach the MN and oldAR can generate the newCoA by using the link-layer address and the prefix information, but the new CoA option can also be included as in this approach. For the newAR this is a flexible solution because it is able to decide whether to compute the new address by itself or just send the prefix information so that the oldAR and the mobile node can do this themselves.

The reasons for the newAR to have to choose between these alternatives are, however, equivocal and could probably be limited only to function the way described in the stateful approach. This will increase the computing requirements on the newAR, but then computation is not needed at both the oldAR and the MN. The computation values are, however, so small that they almost can be neglected.

The network load will be almost the same in both approaches and is therefore not important either. The prefix information option will be included in the stateless approach while the new CoA option will be included in the stateful one, and these option fields are about the same sizes. There is however one point that goes in favor of including the newCoA instead of the prefix information. If this address is generated at the newAR, all involved parts (newAR, oldAR and MN) are informed about the new address that the MN is about to use – and this reduces the equivocation. By only allowing the latter approach the PrRtAdv message structure can also be simplified, but still providing the same functionality as before.

E.2. PrRtAdv – Movement Within the Same Network Segment

The PrRtAdv message can also inform the MN that the link-layer address in the corresponding RtSolPr message does not require change of CoA and that no (fast) handover has to be carried out.

This scenario is relevant when a MN operating in a wireless environment detects new radio links with better quality. The MN will then try to move to the new link, but both the old and the new link are connected through the same access router. No fast handovers or layer-3 information is therefore necessary, only reconnecting to the new access point in layer-2.

Implementing this feature in the fast handovers procedure is necessary to make the system support a diversity of link-layer technologies (as required in IP). If the link-layer is well designed in proportion to fast handovers, this functionality should be exploited rarely.

The MN may e.g. detect the link-layer address to the new access router and compare it with the one it is currently attached to. If they are equal, only a link-layer movement is necessary if the signal strength and load is below a threshold value and an RtSolPr message should not be sent to the oldAR. If the link-layer otherwise detects that the movement is between nodes located on the same network, this will have the same effect – fewer messages will be transmitted, thus reducing the network load and overhead.

As this discussion indicates, simple techniques can be used to reduce the number of messages exchanged. Some techniques are link-layer specific, while others depend on additional implementation in the MN. It is anyway important to find a solution that is flexible so that it will work on different access technologies, but also is able to take advantage of the mechanisms offered on the different access technologies.

E.3. HI – Flag Utilization

HI is a new ICMPv6 message sent by oldAR to the newAR to initiate a MNs handover from one network to another. Two flags are included in the header, one bit that requests a newCoA returned by the destination (S), and one bit that asks for buffering for the MN (U).

If the S flag is included it addresses some of the same issues as discussed under the PrRtAdv. This is whether the inclusion of the new CoA option should be mandatory or if it should include prefix information and the interface ID. This was for the purpose of notifying the MN and the oldAR about the newCoA.

The U bit, however, can probably be left out without affecting the result and the overall performance. Using this bit to request buffering in the newAR is initially a good idea, but the question is if this really is necessary. When an oldAR sends a HI message to the newAR, this is because it wants to start a fast handover procedure. Buffering packets in the newAR that is destined to the MN before it connects to the new link will reduce the packet loss, and therefore also improve the overall performance. The oldAR is supposed to set this bit if it wants better performance, and this is also the overall objective to the fast handovers procedure. Consequently, the U bit will probably be set every time to achieve the best possible performance. The specification can therefore require that all access routers, instead of including the bit in the HI message, must support buffering and execute it in every handover.

E.4. BU – Bicasting

When access routers duplicate packets and send them to the mobile node at both its previous and new points of attachment, it is called bicasting. This is semantically allowed by IP because it does not guarantee packet uniqueness, and higher-level protocols (such as TCP) are assumed to eliminate duplicates whenever this is important for the application. This mechanism can be used to improve the service quality for the MN when changing the point of attachment, but whether this is necessary depends on the link-layer functionality.

It may not be necessary for the oldAR to do bicasting if the link-layer can give a notification to layer-3 exactly when the MN moves to a new network. The oldAR may then forward packets to the MN through the newAR after it has changed the point of attachment. This will not generate unnecessary traffic over the radio interfaces and the handover procedure will therefore be more efficient than with the use of bicasting. The implementation can however be quite challenging because the fast handovers procedure is supposed to work despite of the underlying access technology. The network must therefore get notified on the IP layer whether or not to use bicasting.

If no such information is available, the oldAR can use bicasting if not otherwise is indicated (either by the link-layer or by the network components) to support improved service quality. This will generate more network traffic for the short period of time the bicasting is in progress, but for MNs using an access technology with a very limited bandwidth, or if the number of MNs exceeds the number the link is designed for, this can affect the overall performance. The advantages by far outweigh the disadvantages though.

E.5. Movement Controlling Mechanisms

Both the network and the MN itself can control the movement between different physical links, and a solution somewhere in between can also be used. The fast handovers specification accepts all types of controlling mechanisms, but the final decision has to be made by the MN by sending a BU message to the oldAR.

A system with network determined handovers could more easily be designed and deployed in a predictable manner because of prior experiences with similar systems, such as GSM. Even though there are many differences, the level of entrance will be lower because of the knowledge that many people possess with this technology compared to the mobile determined solution. There are today several indications, and among these UMTS, that future telecommunication networks will migrate from smart networks and dumb mobile nodes to smart mobile nodes and dumb networks. In other words network determined handovers might be said to represent the old school, while the mobile determined handovers represents a new way of thinking. But using the new technology is not necessarily the best solution because there are few, or no, experiences using it. This can affect the expected performance as well as the calculated costs in both a negative and positive direction, but we believe that it will be a more cost efficient approach due to the reduced network complexity and cost.

The performance issues are important when deciding how to implement low latency handovers. If it is too much network traffic it will be easier to implement load-balancing algorithms in the network to exploit the resources as good as possible. The only way a MN can determine the workload on the access routers is by measuring the difference in delays, but this is not a very accurate way of calculating the load. The access routers may for example know how many MNs that are attached to it, and how many it can handle without compromising the network performance. This is not the same thing as having a totally network determined handover approach, but the network must be able to pass this information, and which access router it should move to, on to the MNs. Consequently, it will be easier to guarantee services and performance when the network is a participating in the fast handovers process. This will also add complexity to the system because several messages must be exchanged between the overlapping access routers so that they are able to distribute the load between them.

If fast handovers between different access-technologies are executed, it can be implemented quite easy in a mobile determined approach. The MNs must check the link quality and workload on the different access points of different technologies, and make a decision on its own based on the information received by each of them. If this is to be implemented in a network-determined approach the complexity will increase immediately. The access routers using different access technologies have to communicate through a predefined protocol, so that they are able to distribute their workload.

Making a decision whether to use a network or mobile determined handover is difficult and maybe it should not be taken at all. The fast handovers procedure is designed to make these approaches co-exist, and the regional administrators may decide what is most suited within their own domain.

Appendix F

A mathematical analysis of efficient handovers using hierarchical and nonhierarchical approaches is elaborated in this section.

F.1. General Proposal

The following network entities are defined:

- **HA** – Home Agent
- **MN** – Mobile Node
- **CN(i)** – ith Corresponding Node
- **m** – total number of CN(i)
- **LMM(k)** – kth Localized Mobility Manager (equals foreign access router)
- **LMM-T** – Top level Localized Mobility Manager (equals top level foreign access router (a MAP) if a hierarchy is involved)

Let:

- **z** – signaling delay between HA and MN
- **x(i)** – signaling delay between MN and CN(i)
- **y(k)** – signaling delay between MN and LMM(k)
- **w(k)** – signaling delay between LMM-T and LMM(k)

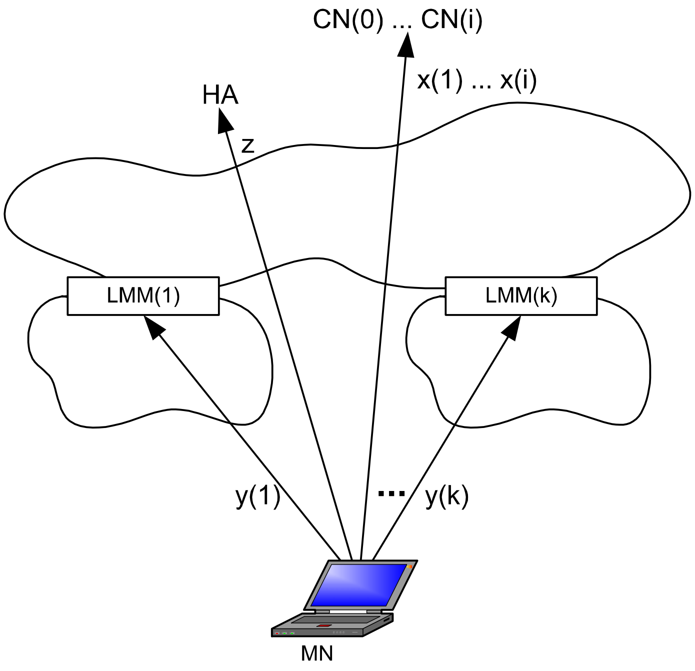


Figure F.1 – Signaling in the nonhierarchical approach.

We observe that deploying LMM only makes sense if the following inequalities hold:

$$\forall i,k : x(i) << y(k)$$

$$\forall k : z >> y(k)$$

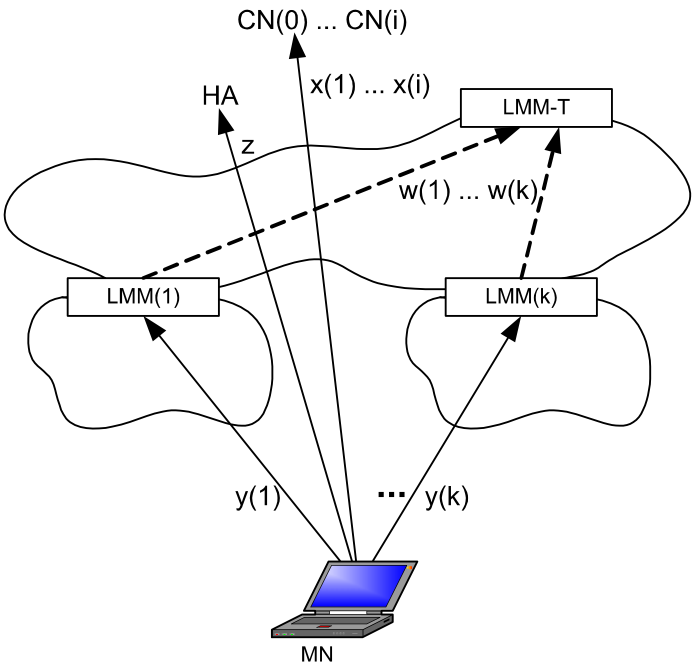


Figure F.1 – Signaling in the hierarchical approach.

We observe that deploying hierarchical LMM only makes sense if the following inequalities hold:

$$1)\forall k : w(k) \le y(k), z >> w(k)$$
$$2)\forall i,k : x(k) >> w(k)$$

We now calculate the signaling overhead for both techniques and compare.

Nonhierarchical Case

Signaling delay per handoff at 5th LMM domain:

$$\sum_{i=0}^{m} x(i) + z + y(k)$$

Suppose MN traverses across n LMM domains. Total signaling overhead is:

$$\sum_{k=0}^{n}\left(\sum_{i=0}^{m} x(i) + z + y(k)\right) = n\sum_{i=0}^{m} x(i) + nz + \sum_{k=0}^{n} y(k)$$

Hierarchical Case

We assume here one full set of CNs and HA BUs at the beginning to register the external CoA. The signaling delay per handoff at the 5th LMM domain:

$$w(k) + y(k)$$

Suppose MN traverses across n LMM domains. Total signaling overhead:

$$\sum_{i=0}^{m} x(i) + z + \sum_{k=0}^{n}\left(y(k) + w(k)\right)$$

Analysis

Let:

$$A = \sum_{i=0}^{m} x(i)$$

$$B = \sum_{k=0}^{n} y(k)$$

$$C = \sum_{k=0}^{n} w(k)$$

In order for the nonhierarchical overhead to be less than the hierarchical overhead, the following would have to be true:

$$nA + nz + B < A + z + B + C$$

$$\Leftrightarrow nA - A + nz - z + B - B < C$$

$$\Leftrightarrow (n-1)A + (n-1)z < C$$

$$\Leftrightarrow (n-1)\sum_{i=0}^{m} x(i) + (n-1)z < \sum_{k=0}^{n} w(k)$$

But this is not possible by the inequalities in equations 1 and 2 above. From 1 you can see that *z* is supposed to be much bigger than *w(k)* – that is the time it takes to send a BU to the HA is much bigger than the time it takes to update LMM-T. From the equation above, however, *z* is supposed to be less than sum of all *w(k)*'s – that is the time it takes to send a BU to the HA (in addition to the time it takes to send BUs to all CNs) shall be less than time it takes to send BUs all LMM's. This is of course impossible and it is therefore a contradiction. Thus, the signaling latency in the hierarchical case will be a better solution when considering the signaling load.

Terminology

General

- **Host** - Any node that is not a router.
- **Interface** - A node's attachment to a link.
- **Interface identifier** - A number used to identify a node's interface on a link. The interface identifier is the remaining low-order bits in the node's IP address after the subnet prefix.
- **IP** - Internet Protocol Version 6 (IPv6).
- **Link** - A communication facility or medium over which nodes can communicate at the link layer, such as an Ethernet (simple or bridged). A link is the layer immediately below IP.
- **Link-layer address** - A link-layer identifier for an interface, such as IEEE 802 addresses on Ethernet links.
- **Node** - A device that implements IP.
- **Packet** - An IP header plus payload.
- **Router** - A node that forwards IP packets not explicitly addressed to itself.
- **Subnet prefix** - A bit string that consists of some number of initial bits of an IP address.

MIPv6 Technology

- **Binding** - The association of the home address of a mobile node with a care-of address for that mobile node, along with the remaining lifetime of that association.
- **Care-of address** - An IP address associated with a mobile node while visiting a foreign link; the subnet prefix of this IP address is a foreign subnet prefix. Among the multiple care-of addresses that a mobile node may have at a time (e.g., with different subnet prefixes), the one registered with the mobile node's home agent is called its "primary" care-of address.
- **Correspondent node** - A peer node with which a mobile node is communicating. The correspondent node may be either mobile or stationary.
- **Foreign link** - Any link other than the mobile node's home link.
- **Foreign subnet prefix** - Any IP subnet prefix other than the mobile node's home subnet prefix.
- **Home address** - An IP address assigned to a mobile node within its home link.
- **Home agent** - A router on a mobile node's home link with which the mobile node has registered its current care-of address. While the mobile node is away from home, the home agent intercepts packets on the home link destined to the mobile node's home address, encapsulates them, and tunnels them to the mobile node's registered care-of address.
- **Home link** - The link on which a mobile node's home subnet prefix is defined. Standard IP routing mechanisms will deliver packets destined for a mobile node's home address to its home link.
- **Home subnet prefix** - The IP subnet prefix corresponding to a mobile node's home address.
- **Mobile node** - A node that can change its point of attachment from one link to another, while still being reachable via its home address.

- **Movement** - A change in a mobile node's point of attachment to the Internet such that it is no longer connected to the same link as it was previously. If a mobile node is not currently attached to its home link, the mobile node is said to be "away from home".

Efficient Handovers

- **Access router (AR)** - The last router between the network and the mobile node
- **New access router (newAR)** - When a mobile node moves from one physical sub-network to another, this is the next point of attachment.
- **New care-of address (newCoA)** - The CoA on the newAR.
- **Old access router (oldAR) -** When a mobile node moves from one physical sub-network to another, this is the previous point of attachment.
- **Old care-of address (oldCoA)** - The CoA on the oldAR.

AAA

- **AAA credential** - Data provided by a client to the AAA in an authorization request. For example, this can be a message authentication code constructed using a secret shared between the host and AAAH.
- **AAAH** - The AAA server in the home domain, which is able to authorize its clients.
- **AAAL** - The AAA server in the foreign domain that mediates local access to the AAA infrastructure.
- **Attendant** - The attendant is the entity that extracts identification and authorization data sent by the client and forwards them to AAAL for verification. It is also responsible for making the necessary configuration updates (e.g., to the packet filter, and the router's neighbor cache) so that only authorized clients can access the network.
- **Client** - The client is the entity whose authorization is checked. The client resides on the host system.
- **Controlled and uncontrolled access** - Each network interface of the router can be configured to provide AAA services. When an interface is configured, all transiting packets are subject to controlled access. If a packet does not pass access control but is an AAA message addressed to the router, it is given to the attendant in the uncontrolled access part.
- **Host System** - The host system is the node requesting access to the network.
- **Packet filter** - A packet filter (also called firewall/security gateway) is the entity responsible for disallowing unauthorized datagram traffic. When a client is authorized, the access control list of the filter is updated with the corresponding host's IP address(es).
- **Router System** - The router is the node that provides network access to the host. In addition to the usual packet forwarding functionality, the router system typically consists of other functional blocks like the attendant and the packet filter.

Glossary

A: Attendant.

AAA: Accounting, Authentication, and Authorization.

AAAH: AAA Home Domain Authority.

AAAL: AAA Local (Foreign) Domain Authority.

AP: Access Point.

AR: Access Router.

ARP: Address Resolution Protocol.

AuC: Authentication Center.

BACK: Binding Acknowledgement.

BC: Binding Cache.

BR: Binding Request.

BU: Binding Update

BUL: Binding Update List.

CDMA: Code Division Multiple Access.

CIDR: Classless Interdomain Routing.

CN: Correspondent Node.

CoA: Care-of Address.

CR: Credential.

DAD: Duplicate Address Detection.

DDK: Driver Development Kits.

DHCP: Dynamic Host Configuration Protocol.

DHCPv6: Dynamic Host Configuration Protocol version 6.

DLL: Dynamic link library.

DNS: Domain Name System.

DoS: Denial-of Service.

EDO: Embedded Data Option.

EUI: Ethernet Universal Identifier.

FA: Foreign Agent.

FBACK: Fast Handover Binding Acknowledgement.

FBU: Fast Handover Binding Update.

FDD: Frequency Division Duplex.

GPRS: General Packet Radio Services.

GSM AKA: GSM Authentication Key Agreement.

GSM: Global System for Mobile Communication.

HA: Home Agents.

HACK: Handover Acknowledgement.

Haddr: Home Address.

HAE: Home Address Entries.

HAL: Home Agent List.

HI: Handover Initiation.

HLR: Home Location Register.

HMIPv6: Hierarchical Mobile Internet Protocol version 6.

HP: Hewlett Packard.

HTML: Hyper Text Markup Language.

HTTP: Hyper Text Transfer Protocol.

ICMP: Internet Control Message Protocol.

ICMPv6: Internet Control Message Protocol version 6.

ID: Identification.

IE: Internet Explorer.

IEEE: Institute of Electrical and Electronics Engineers.

IETF: Internet Engineering Task Force.

IMSI: International Mobile Subscriber Identity.

IOCTL: Input-Output Control.

IP: Internet Protocol.

IPsec: Internet Protocol Security.

IPv4: Internet Protocol version 4.

IPv6: Internet Protocol version 6.

ISP: Internet Service Provider.

LAN: Local Area Network.

LC: Local Challenge Option.

LCoA: On-link Care-of Address.

LMM: Localized Mobility Manager.

LMM-T: Top Level Localized Mobility Manager.

MAC: Media Access Control.

MAP: Mobility Anchor Point.

MCC: Mobile Country Code.

MIP: Mobile Internet Protocol.

MIPv4: Mobile Internet Protocol version 4.

MIPv6: Mobile Internet Protocol version 6.

MN: Mobile Node.

MNC: Mobile Network Code.

MN-NAI: Mobile Node Network Access Identifier.

MS: Microsoft.

MSDN: Microsoft Developer Network.

MSIN: Mobile Subscriber Identification Number.

MSR: Microsoft Research.

MSRIPv6: Microsoft Research Internet Protocol version 6.

MTU: Maximum Transmission Unit.

NA: Neighbor Advertisement.

NAI: Network Access Identifier.

ND: Neighbor Discovery.

NetMon: Network Monitor.

NewAR: New Access Router.

NewCoA: New Care-of Address.

NS: Neighbor Solicitation.

NTE: Net Table Entry.

NUD: Neighbor Unreachability Detection.

OldAR: Old Access Router.

OldCoA: Old Care-of Address.

OS: Operating Systems.

OSIRM: Open Systems Interconnection Reference Model.

PAE: Proxy Address Entry.

PC: Personal Computer.

PCMCIA: Personal Computer Memory Card International Association.

PDA: Personal Desktop Assistant.

PF: Packet Filter.

PHY: Physical.

PPA: Provide Poll Address.

PrRtAdv: Proxy Router Advertisement.

QoS: Quality of Service.

RA: Router Advertisements.

RCE: Route Cache Entry.

RCoA: Regional Care-of Address.

RFC: Request For Comments.

RPA: Request Pool Address.

RPI: Replay Protection Indicator.

RS: Router Solicitation.

RtSolPr: Router Solicitation for Proxy.

SA: Security Association.

SDK: Software Development Kits.

SIM: Subscriber Identity Module.

SMS: System Management Services.

TCP: Transport Control Protocol.

TDMA: Time Division Multiple Access.

UDP: User Datagram Protocol.

UMTS: Universal Mobile Telecommunications System.

USB: Universal Serial Bus.

WCDMA: Wideband Code Division Multiple Access.

WLAN: Wireless Local Area Network.

References

[1] Perkins, Charles E (1998): “Mobile Networking through Mobile IP”, *IEEE Internet Computing*, Vol. 2 (No1), pp 58-69.

[2] Johnson D & Perkins C/IETF Mobile IP Working Group (2000): *Mobility Support in IPv6* [online]. Available from: http://www.ietf.org/internet-drafts/draft-ietf-mobileip-ipv6-13.txt [Accessed 10 May 2001].

[3] Perkins, Charles E/IPng Working Group (2000): *AAA for IPv6 Network Access* [online]. Available from: http://www.ietf.org/internet-drafts/draft-perkins-aaav6-03.txt [Accessed 10 May 2001].

[4] Peterson L & Davie B (2000): *Computer Networks – A Systems Approach*, 2nd ed, pp 284-301. San Francisco: Morgan Kaufman Publishers.

[5] Peterson L & Davie B (2000): *Computer Networks – A Systems Approach*, 2nd ed, pp 316-319. San Francisco: Morgan Kaufman Publishers.

[6] Narten T, Nordmark E & Simpson W/IETF Network Working Group (1998): *Neighbor Discovery for IP Version 6 (IPv6)* [online], Internet RFC 2461. Available from: http://www.ietf.org/rfc/rfc2461.txt [Accessed 10 May 2001].

[7] Thomson S & Narten T /IETF Network Working Group (1998): *IPv6 Stateless Address Autoconfiguration* [online], Internet RFC 2462. Available from: http://www.ietf.org/rfc/rfc2462.txt [Accessed 10 May 2001].

[8] Bound J, Carney M & Perkins C/IETF DHC Working Group (2001): *Dynamic Host Configuration Protocol for IPv6 (DHCPv6)* [online]. Available from: http://www.ietf.org/internet-drafts/draft-ietf-dhc-dhcpv6-18.txt [Accessed 10 May 2001].

[9] Plummer, David C/IETF Network Working Group (1982): *An Ethernet Address Resolution Protocol* [online], Internet RFC 826. Available from: http://www.ietf.org/rfc/rfc826.txt [Accessed 10 May 2001].

[10] Deering, S/IETF Network Working Group (1991): *ICMP Router Discovery Messages* [online], Internet RFC 1256. Available from: http://www.ietf.org/rfc/rfc1256.txt [Accessed 10 May 2001].

[11] Postel, J/IETF Network Working Group (1981): *Internet Control Message Protocol* [online], Internet RFC 792. Available from: http://www.ietf.org/rfc/rfc792.txt [Accessed 10 May 2001].

[12] Kent S & Atkinson R/IETF Network Working Group (1998): *Security Architecture for the Internet Protocol* [online], Internet RFC 2401. Available from: http://www.ietf.org/rfc/rfc2401.txt [Accessed 11 May 2001].

[13] KAME Project (2001): *Free Reference Implementations of IPv6* [online]. Available from: http://www.kame.net [Accessed 11 May 2001].

[14] Lunde T A, Haslestad T & Sollund A (2000): *Mobile IPv6 WLAN Internetworking*, R&D N 60/2000. Kjeller: Telenor Forskning og Utvikling.

[15] Microsoft Corporation (2001), *Sample Router Config Script* [online]. Available from: http://research.microsoft.com/programs/europe/projects/r2config.txt [Accessed 11 May 2001].

[16] Microsoft Corporation (2001), *Configuring the IPv6 stack* [online]. Available from: http://research.microsoft.com/programs/europe/projects/MIPv6Conf.asp [Accessed 11 May 2001].

[17] OperatingSystems.net (1998), *Reliability* [online]. Available from: http://www.operatingsystems.net/holistic/reliable/reliable.htm [Accessed 11 May 2001].

[18] Peterson L & Davie B (2000): *Computer Networks – A Systems Approach*, 2nd ed, pp 185-186. San Francisco: Morgan Kaufman Publishers.

[19] Johnson D & Perkins C/IETF Mobile IP Working Group (2000): *Mobility Support in IPv6* [online]. Available from: http://research.microsoft.com/programs/europe/projects/draft-ietf-mobileip-ipv6-12.txt [Accessed 11 May 2001].

[20] RAD Data Communications (1994): *The Seven Layers Model* [online]. Available from: http://www.rad.com/networks/1994/osi/layers.htm [Accessed 11 May 2001].

[21] Lancaster University & Microsoft Research (2001): *LandMARC* [online]. Available from: http://www.landmarc.net [Accessed 11 May 2001].

[22] Finney, Joe (2000): *LandMARC Project Report: Adapting the MSR IPv6 Stack to Support Mobile IPv6 Functionality* [online]. http://www.la.net [Accessed 11 May 2001].

[23] Microsoft Corporation (2001), *Microsoft Windows Driver Development Kits* [online]. Available from: http://www.microsoft.com/ddk [Accessed 11 May 2001].

[24] Microsoft Corporation (2001), *Mobile IPv6 Setup* [online]. Available from: http://research.microsoft.com/programs/europe/projects/MIPv6Setup.asp [Accessed 11 May 2001].

[25] Microsoft Corporation (2001), *Configuring the IPv6 stack* [online]. Available from: http://research.microsoft.com/programs/europe/projects/MIPv6Conf.asp [Accessed 11 May 2001].

[26] Microsoft Corporation (2001), *Systems Management Server 2.0* [online]. Available from: http://www.microsoft.com/smsmgmt/default.asp [Accessed 11 May 2001].

[27] Microsoft Corporation (2001), *Microsoft Developer Network* [online]. Available from: http://msdn.microsoft.com [Accessed 11 May 2001].

[28] Sysinternals (2001), *DebugView v4.11* [online]. Available from: http://www.sysinternals.com/ntw2k/freeware/debugview.shtml [Accessed 11 May 2001].

[29] Bruno R, Conti M & Gregori E (2001): *WLAN technologies for Mobile ad hoc Networks* [online]. Available from: http://www.computer.org/proceedings/hicss/0981/volume%209/09819003abs.htm [Accessed 11 May 2001].

[30] Microsoft Corporation (2001), *Configuring the IPv6 stack* [online]. Available from: http://research.microsoft.com/programs/europe/projects/MIPv6Conf.asp [Accessed 11 May 2001].

[31] Hunskaar, Jørn (12 January 2001): "Duplicate address when using MIPv6 in WLAN", *Microsoft Research IPv6* [online]. Available from: msripv6-users@list.research.microsoft.com [Accessed 12 January 2001].

[32] Microsoft Corporation (2001), *MIPv6 Download* [online]. Available from: http://research.microsoft.com/programs/europe/projects/MIPv6Download.asp [Accessed 12 May 2001].

[33] Microsoft Corporation (2001), *Mobile IPv6* [online]. Available from: http://research.microsoft.com/programs/europe/projects/MIPv6.asp [Accessed 12 May 2001].

[34] Microsoft Corporation (2001), *Getting started with MIPv6* [online]. Available from: http://research.microsoft.com/programs/europe/projects/MIPv6Start.asp [Accessed 12 May 2001].

[35] Morin, Brian (2001), *Fnord! webserver* [online]. Available from: ftp://ftp.research.microsoft.com/users/msripv6/msripv6-fnord-1.5.exe [Accessed 12 May 2001].

[36] Peterson L & Davie B (2000): *Computer Networks – A Systems Approach*, 2nd ed, pp 489-494. San Francisco: Morgan Kaufman Publishers.

[37] Tsirtsis G, Yegen A, Perkins C, Dommety G, El-Malki K & Khalil M/IETF Mobile IP Working Group (2001): *Fast Handovers for Mobile IPv6* [online]. Available from: http://www.ietf.org/internet-drafts/draft-designteam-fast-mipv6-01.txt [Accessed 12 May 2001].

[38] Soliman H, Castelluccia C, El-Malki K & Bellier L/IETF Mobile IP Working Group (2001): *Hierarchical MIPv6 mobility management* [online]. Available from: http://www.ietf.org/internet-drafts/draft-ietf-mobileip-hmipv6-01.txt [Accessed 1 February 2001].

[39] IETF Mobile IP Working Group (2001): *IP Routing for Wireless/Mobile Hosts (mobileip)* [online]. Available from: http://www.ietf.org/html.charters/mobileip-charter.html [Accessed 12 May 2001].

[40] Connectathon (2001): *A network providing ground for allowing vendors test their interoperability solutions* [online]. Available from: http://www.connectathon.org [Accessed 12 May 2001].

[41] Perkins, Charles E/IPng Working Group (2000): *Fast Handovers for Mobile IPv6* [online]. Available from: http://www.ietf.org/internet-drafts/draft-perkins-mobileip-handover-00.txt [Accessed 12 May 2001].

[42] Bradner, S/IETF Network Working Group (1996): *The Internet Standards Process -- Revision 3* [online]. Available from: http://www.ietf.org/rfc/rfc2026.txt [Accessed 12 May 2001].

[43] Gwon Y & Takeshita A/IETF Mobile-IP Working Group (2001): *Network Layer Triggered Mobile IPv4 Predictive Handoff* [online]. Available from: http://www.ietf.org/internet-drafts/draft-gwon-mobileip-l3mp-mipv4-00.txt [Accessed 12 May 2001].

[44] Koodli R, Tiwari M and Perkins C/IETF Context and Micro-mobility Routing (seamoby) Working Group (2001): *Context Relocation for Seamless Header Compression in IP Networks* [online]. Available from:

http://www.ietf.org/internet-drafts/draft-koodli-seamoby-hc-relocate-00.txt [Accessed 12 May 2001].

[45] Asokan N, Flykt P, Perkins C & Eklund T/IETF IPng Working Group (2000): *AAA for IPv6 Network Access* [online]. Available from: http://www.ietf.org/internet-drafts/draft-perkins-aaav6-03.txt [Accessed 12 May 2001].

[46] Ptacek T & Newsham T (1998): *Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection* [online]. Available from http://www.snort.org/idspaper.html [Accessed 12 May 2001].

[47] Aboba B & Beadles/IETF Network Working Group (1999): *The Network Access Identifier* [online], Internet RFC 2486. Available from: http://www.ietf.org/rfc/rfc2486.txt [Accessed 12 May 2001].

[48] Coulouris G, Dollimore J & Kindberg T (2001): *Distributed Systems – concepts and design*, 3rd ed, pp 354-371. Harlow: Addison-Wesley.

[49] Haverinen H/IETF Mobile IP Working Group (2001): *GSM SIM Authentication and Key Generation for Mobile IP* [online]. Available from: http://search.ietf.org/internet-drafts/draft-haverinen-mobileip-gsmsim-02.txt [Accessed 12 May 2001].

[50] ETSI European Telecommunications Standards Institute (1999): *GSM AKA* [online]. Available from: http://194.248.43.201:8080/upload/TS0320_v800.pdf [Accessed 12 May 2001].

[51] Calhoun P & Perkins C/IETF Network Working Group (2000): *Mobile IP Network Access Identifier Extension for IPv4* [online], Internet RFC 2794. Available from: http://www.ietf.org/rfc/rfc2794.txt [Accessed 12 May 2001].

[52] Perkins C & Calhoun P/IETF Network Working Group (2000): *Mobile IPv4 Challenge/Response Extensions* [online], Internet RFC 3012. Available from: http://www.ietf.org/rfc/rfc3012.txt [Accessed 12 May 2001].

[53] Le F, Faccin S & Patil B/IETF Mobile IP Working Group (2001): *Challenge-Response Authentication Request* [online]. Available from: http://www.ietf.org/internet-drafts/draft-le-mobileip-authreq-00.txt [Accessed 12 May 2001].

[54] Agder College (2001), *Mobility in IPv6* [online]. Available from: http://www.siving.hia.no/ikt01/ikt6400/jhunsk99/index.htm [Accessible from 28 May 2001].