# IPv6 Routing for Mobility Environments

Tim Leinmüller (leinmuel@enst.fr)

Supervisor Motorola: Alexandru Petrescu

Encadrant ENST: Philippe Dax

Betreuung Universität Stuttgart:
Prof. Dr.-Ing. Dr. h. c. mult. Paul J. Kühn

Motorola Labs, Paris
Networking and Applications Lab (NAL)

École Nationale Supérieure
des Télécommunications (ENST), Paris

Institut für Kommunikationsnetze
und Rechnersysteme (IKR), Universität Stuttgart

July 29, 2003

**Acknowledgements**

First of all, I would like to thank the NAL staff (Networking and Applications Lab) for taking me on board, and also for the regular attention paid to the progress of my internship.

I am very thankful to Alexandru Petrescu, my supervisor, for his constant help and support. Likewise, I deeply thank Emmanuel Riou for the numerous questions he answered to and his availability, Christophe Beaujean for his constant care and attention, and all the others for the perfect atmosphere they created.

I shall not forget the Lab Manager, Hong-Yon Lach, for giving me the opportunity to work in such interesting domains.

Likewise I thank all the CRM members and especially the NAL team for their warm welcome, their kindness and sympathy.

Finally I want to thank Philippe Dax and Prof. Kühn for their support.

**Abstract**

This document describes the specification and the implementation of IPv6 routing for mobility environments, including the realization of Mobile IPv6 based network mobility.

Since standard Mobile IPv6 supports only host mobility, a solution to extend Mobile IPv6 to support network mobility is proposed. The primary aim of this solution is to introduce only few modifications compared to standard IPv6 and Mobile IPv6, to provide as much compatibility as possible. This results in the specification of two special nodes that handle network mobility transparent for normal, mobility unaware nodes, even those nodes residing in a mobile network.

Network mobility is realized by using a mobile router (MR) to establish connections to foreign networks. On the home network, a home agent (HA) acts on behalf of the MR, while this one is away from home. The HA follows the specification of a Mobile IPv6 HA, but is extended to support mobile routers and mobile networks.

To maintain the connectivity of the mobile network, the mobile router and the home agent establish a bi-directional MR-HA tunnel. All packets from and to the mobile network are sent through this tunnel.

In order to provide an adapted research environment, routing and the support for network mobility are implemented in the Motorola LIVSIX IPv6 stack. They have proven their usability in several tests, described in this document. The tests cover also interaction between standard mobile hosts and mobile networks, as well as interaction between two mobile networks (nested mobility scenarios).

LIVSIX is available from the Motorola LIVSIX homepage at
http://www.nal.motlabs.com/livsix/.

**Zusammenfassung**

Die Arbeit beschreibt die Spezifikation und Implementierung von IPv6 Routing für mobile Anwendungen. Dieses beinhaltet die Verwirklichung von Mobile IPv6 basierender Netzwerk Mobilität (*IPv6 based network mobility*).

Mobile IPv6 unterstützt standardmässig nur Host Mobilität. Aus diesem Grund wird ein Konzept vorgestellt, welches Mobile IPv6 erweitert und Netzwerk Mobilität ermöglicht. Primäres Ziel dieser Lösung ist es, möglichst wenige Veränderungen an IPv6 und Mobile IPv6 vorzunehmen, um möglichst kompatibel zu bestehenden Standards und vorhandenen Implementierungen zu bleiben. Daraus resultiert die Spezifikation von zwei Netzknoten (*nodes*), welche Netzwerkmobilität transparent für andere Netzknoten, die keine Mobilität unterstützen, ermöglichen (dies gilt auch für Netzknoten innerhalb eines mobilen Netzwerks).

Um das mobile Netzwerk an fremde Netze (*foreign networks*) anzukoppeln, wird ein mobiler Router (*mobile router, MR*) eingesetzt. Im Heimnetzwerk (*home network*) des mobilen Routers ist ein so genannter *home agent (HA)* dafür verantwortlich, den mobilen Router zu vertreten, so lange dieser nicht anwesend ist. Der *HA* entspricht dem aus der Mobile IPv6 Spezifikation, allerdings erweitert um die Unterstützung von mobilen Routern und mobilen Netzwerken.

Damit die ununterbrochene Anbindung des Mobilen Netzwerkes gewährleist ist, richten der *mobile router* und der *home agent* einen bi-direktionalen MR-HA Tunnel ein. Alle Pakete von und zu dem mobilen Netzwerk werden durch diesen Tunnel versandt.

Oben genannte Lösung wird in den Motorola LIVSIX IPv6 Stack implementiert, damit eine Forschungsumgebung und Experimentierplatform für Netzwerkmobilität zur Verfügung steht. Die Verwendbarkeit wird in mehreren Tests überprüft. Untersucht werden sowohl Wechselwirkungen zwischen mobilen Hosts und mobilen Netzwerken, als auch Wechselwirkungen zwischen zwei verschiedenen mobilen Netzwerken (*nested mobility scenarios*).

LIVSIX ist auf der Motorola LIVSIX Homepage erhältlich
http://www.nal.motlabs.com/livsix/.

# Contents

# List of Figures

# Chapter 1

# Introduction

## 1.1 IPv6 Mobility Environments

Today, more and more communication is handled over IP based networks, more and more devices implement IP to enable global connectivity. This includes fixed devices, e.g. computers, and mobile devices, for example cellular phones or PDAs. With the increasing amount of especially mobile devices, IP based mobility will play an important role, and IP version 4 (IPv4), today's commonly used version, will run out of addresses. Therefore, IP version 6 (IPv6), the successor of IPv4, with a lager address space and a better support for mobility, is about to be used in mobility environments. With IPv6, every device obtains a unique and global reachable address.

The number of IP ready devices used by a single person, or of small groups of persons, is steadily increasing. To maintain a connection to the Internet with all these devices, it seems reasonable to regroup the devices and form small networks to use a common uplink, instead of connecting every device separately. Since forming a small network does not and must not prevent the network from moving entirely, the ability to manage mobility of networks is required.

Such kind of network is referred to as a mobile network and includes at least one mobile router that assures the networks connection to the Internet. Nodes inside a mobile network might be either mobile nodes or fixed nodes (without support for mobility).

Examples for mobile networks are

- networks attached to people, so called Personal Area Networks (PANs), interconnecting a PDA, a laptop and a cellular phone, that acts as mobile router. PDA, laptop and cellular phone are interconnected via Bluetooth, UMTS is used as uplink technology.

- Networks in vehicles, providing a connection for the passengers (e.g. a passenger using a PDA) and for the vehicle's sensors.
- Networks in publics transports, e.g. trains or airplanes, providing instant access for passengers.

Network mobility can be enabled by combining two known functionalities of IP networks: routing and mobility. The result is the so called mobile router.

The aim of this work is to develop network mobility support for the Motorola LIVSIX IPv6 stack. Originally, this stack was meant to be used in (mobile) edge devices, not in intermediate routers, therefore the lack of any routing support. To provide network mobility through a mobile router, first a standard IPv6 router must be realized. Together with Mobile IPv6, this is the base for the implementation of mobile networks based on IPv6.

The document is structured as follows. Chapter 2 summarizes principles of IPv6, Mobile IPv6 and IPv6 routing. The last section of this chapter contains the resulting fundamentals and specification for IPv6 based mobile networks. Chapter 3 provides the specification for the LIVSIX router, the mobile router and the mobile router home agent implementation. The following chapter 4 explains their implementation, as well as the implementation of support for the routing protocol suite Zebra. Chapter 5 describes the performed tests and their results. The last chapter, chapter 6, gives a conclusion and ideas for future research directions.

# 1.2 Presentation of Motorola

## 1.2.1 Motorola in General

### 1.2.1.1 Overview

Motorola is one of the world's leading providers of wireless communications, semiconductors and advanced electronic systems, components and services. Major equipment businesses include cellular telephone, two-way radio, paging and data communications, personal communications, automotive, defence and space electronics and computers. Motorola semiconductors power communication devices, computers and millions of other products.

Motorola maintains sales, service and manufacturing facilities throughout the world, conducts business on six continents and employs more than 139,000 people worldwide. In France, the effective is about 3200 people.

### 1.2.1.2 History and Evolution

Paul V. Galvin founded the company in 1928 as the Galvin Manufacturing Corporation, in Chicago, Illinois. Its first product was a "battery eliminator,"allowing consumers to operate radios directly from household current instead of the batteries supplied with early models. In the 1930s, the company successfully commercialised car radios under the brand name "Motorola,"a new word suggesting sound in motion. During this period, the company also established home radio and police radio departments; instituted pioneering personnel programs; and began national advertising. The name of the company was changed to Motorola, Inc. in 1947. The decade of the 1940s also saw the company enter government work and open a research laboratory in Phoenix, Arizona, to explore solid-state electronics. By the time of Paul Galvin's death in 1959, Motorola was a leader in military, space and commercial communications, had built its first semiconductor facility and was a growing force in consumer electronics.

Under the leadership of Robert W. Galvin (Paul Galvin's son), Motorola expanded into international markets in the 1960s and began shifting its focus away from consumer electronics. The colour television receiver business was sold in the mid-1970s, allowing Motorola to concentrate its energies on high-technology markets in commercial, industrial and government fields. By the end of the 1980s, Motorola had become the premier worldwide supplier of cellular telephones.

## 1.2.2 Motorola in France

The first semiconductors manufacturing plant to be implanted outside the United States was implanted in France in 1967. Afterwards, Motorola has not stopped investing and

increasing its activities on the French territory: automobile electronic, cards and systems for embedded applications, professional radio communications, cellular phones, communication infrastructures, access control and identification products, Internet and networks, broadband communication.

The different groups and sectors head offices have been implanted in the Ile-de-France Region. The activities relative to the product distribution are also situated here: Les Ulis, Vélizy, Cachan, Antony and Paris. The total effective in this region is 528 people.

### 1.2.3 Motorola Labs

Motorola Labs is the research & development part of Motorola throughout the world; it regroups 850 researchers on 8 sites in the United States, France, United Kingdom, Japan and Australia.

Recently, the European Communication Research Lab (ECRL) has been created. It regroups UK Research Laboratory (UKRL) and Centre de Recherche de Motorola Paris (CRM).

### 1.2.4 The CRM (Centre de Recherche de Motorola - Paris)

#### 1.2.4.1 Overview

The Centre de Recherche de Motorola - Paris (CRM), located in Gif-sur-Yvette (Paris, France), is one of the research centres of the Motorola Labs. CRM started in 1996, and is growing rapidly to establish itself as a centre of excellence in the European research community.

Its mission is to conduct medium and long-term researches into communications, networking and applications for Motorola's future businesses, especially in the European context, by developing technology and participating in standards and regulatory developments.

The centre is working on various projects, involving the collaborations of researchers with expertise in the fields of Radio Technologies, Signal Processing and Speech, Telecommunication systems and Standards, Networks and Computer Science, Advanced Services and applications, Software and Hardware technologies, Devices and materials.

The CRM is located in the Saclay Scientipôle, just to the south of Paris in an area close to several distinguished engineering schools (Supelec, Ecole Polytechnique, ENST, ...), Universities (Orsay, ...) and public laboratories (CNRS, CEA, INRIA, ...). It has several cooperative projects started and under discussion, and expect this to become a significant part of our activity. It also hosts students for training periods, and has PhD students participating to its programs.

#### 1.2.4.2 Perspectives and Impact

The CRM is rapidly growing: at the beginning in 1996 there were only 4 researchers. One year later, they were 26 and five years later, the population had reached to 80 researchers.

Although it is quite recent, the CRM is already successful. Among the numerous works it realized, we'll note in particular that it:

- Developed a new methodology for estimating the interference between users of a wireless systems, and, more importantly, between users of different wireless systems. This methodology is now recognized as the reference methodology by major spectrum regulatory bodies (CEPT in Europe, ITU in the USA).
- Developed and transferred to businesses advanced algorithms for improving 2G cellular systems.
- Developed new algorithms for improvement of the capacity of 3G cellular systems. Those innovations have been recognized by the competition and included the UMTS standard.
- Developed intelligent algorithms for the location of users of cellular systems, offering better accuracy than the state of the art. These algorithms have been used in the Motorola demonstrator for 3G trials in Japan.
- Developed a new concept for the future of Personal Area Networks. Submitted this new concept as a candidate for the future evolution of Bluetooth towards more bandwidth, for multimedia Person-to-Person connection.
- Developed a Mobile Internet middleware. Demonstrated a transparent switching between GSM, wireless LAN and Ethernet in an Agent-based banking application. This work was completed in a European R&D consortium of R&D in the AMASE project
- Developed and demonstrated a new technology for Smart Internet Applications (Agent Technology)
- Participated into 7 European research consortia.
- Generated more than 100 inventions in the 4 years since the creation of the lab.

### 1.2.5 NAL (Networking and Applications Lab)

As part of CRM, the Networking and Applications Lab (NAL) researches into and develops technology, demonstrators and software prototypes to deliver advanced Internet experience. In particular, NAL develops:

- IPv6-based mobile multiparty multimedia networking technology for the edge of the Internet, with
  - Integrated support of mobility, multicast and QoS in Internet appliances and servers;
  - IPv6-based evolution of 3G systems; and
  - IETF standardisation.
- Intelligent personal proactive application technology for service access and provision, with
  - Intelligent agents;
  - Semantic communication, user modelling, and intelligent interface; and
  - FIPA standardisation.

NAL also actively collaborates with industrial and academic partners to advance technology; for example:

- European projects (Moby Dick, LEAP, and OverDRiVE);
- Close contacts with public laboratories and universities (INRIA, EPFL, Eurecom, etc);
- PhD thesis projects; and
- Internship projects.

# Chapter 2

# Fundamentals

## 2.1 IPv6

IPv6, short for IP version 6, is the designated successor of IPv4, IP version 4. The major advantages of IPv6 are the support of a larger address space (128bit instead of 32bit), the protocol extensibility, support for autonomous host configuration, support for Quality of Service (QoS) and improved support for mobility. These changes are the response to the demands of enormous growing worldwide networks.

The specification of IPv6 and related basic protocols can be found in the following RFCs

- 2460, Internet Protocol, Version 6 (IPv6) [RFC2460].
- 2461, Neighbor Discovery for IPv6 [RFC2461].
- 2462, IPv6 Stateless Address Autoconfiguration [RFC2462].
- 2463, Internet Control Message Protocol (ICMPv6) for IPv6 [RFC2463].

Since this work concentrates on IPv6 for mobility environments, especially on the implementation of support for routing and routing for mobile networks in the Motorola LIVSIX stack, the following sections summarize parts of these RFCs, with special regard to the requirements for this implementation.

### 2.1.1 Protocol Description

#### 2.1.1.1 Header Format

The structure of the IPv6 header shown in figure(2.1) is defined in [RFC2460].

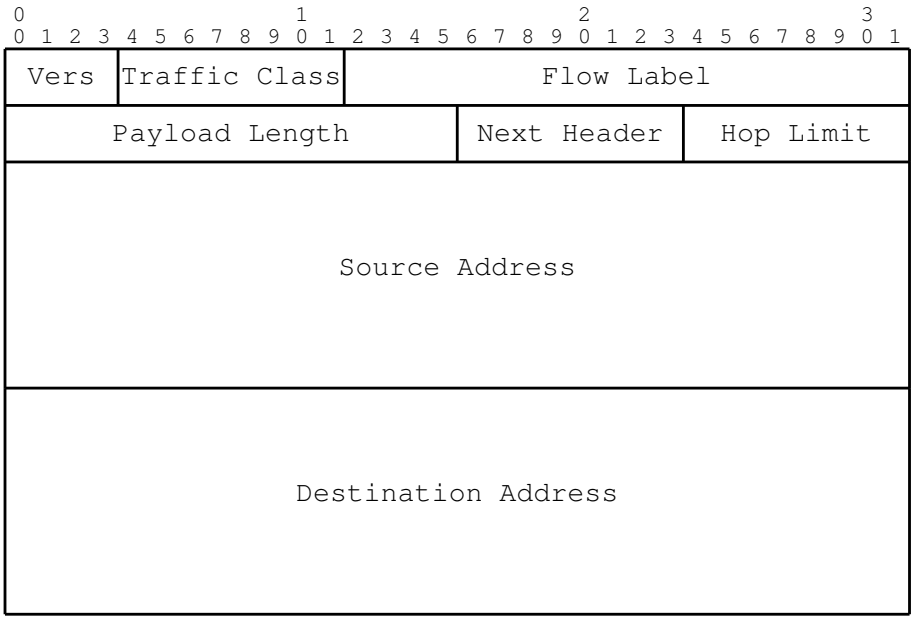



Figure 2.1: IPv6 header format

Traffic Class and Flow Label field are only of interest when dealing with Quality of Service (QoS), which is (not yet) implemented in, or used by the LIVSIX stack.

A router should decrement the Hop Limit of every packet before forwarding it to the selected next hop. On reception of a packet with a Hop Limit field set to zero, it should discard it and return a corresponding error message.

The Next Header field identifies the type of header immediately following the IP header. In most cases, this is the upper layer protocol header, but IPv6 also defines a small set of extension headers which might be located in between.

#### 2.1.1.2 Extension Header Format and Options

For intermediate nodes, as routers are in most cases, only one extension header type is of interest, the Hop-by-Hop Options header. If present, it must follow directly the IPv6 header. It contains information that has to be examined by every node on a packet's delivery path. As today, no relevant options are specified for this header type, it is sufficient to assure that new unknown options are ignored, but might easily be implemented later.

The second extension header that that has to be handled a router, but only in case of the IPv6 destination address field of the packet in question matches one of the router's IP addresses, is the routing header. If present, the router has to process it and act as specified for the corresponding routing header type. The behavior for routing header type zero, for example, is specified in [RFC2460].

For a (mobile) router, today, two routing header types are of interest. Routing header type zero [RFC2460] and type two [MIPv6]. The first is used to specify a packets path through several nodes on a network. The latter is described later in the Mobile IPv6 introduction chapter (see 2.2.2.4). Unknown types have to be treated in dependence of the Segments Left field, as specified in [RFC2460].

### 2.1.2 Neighbor Discovery

IPv6 introduces a set of functions, that provide methods to discover link and link-layer related information on a physical link. These mechanisms are called neighbor discovery (ND) [RFC2461].

Below these functions, router discovery is used to search for and discover routers on a local link, that are willing to forward a node's packets. The prefixes in the router advertisements (RAs) are used for host IP auto-configuration. Figure (2.2) shows an example of the router discovery process.

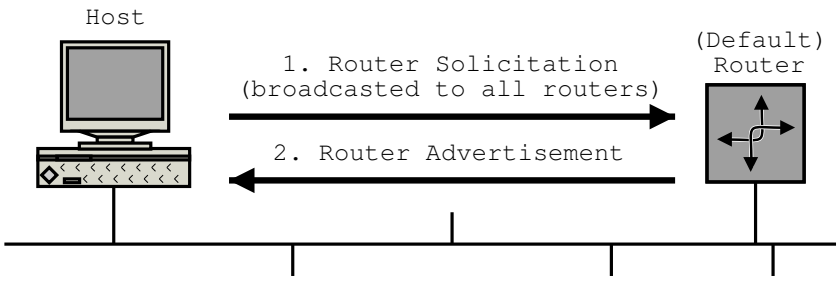



Figure 2.2: IPv6 Router Discovery

As described in [RFC2461] neighbor discovery defines five different types of ICMPv6 messages (for more ICMPv6 messages see [RFC2463]).

1. Router Solicitation (RS)
2. Router Advertisement (RA)
3. Neighbor Solicitation (NS)
4. Neighbor Advertisement (NA)

5. Redirect

While neighbor solicitation and neighbor advertisement are used between any two or more nodes, router solicitation, router advertisement and redirect are specific to communication between hosts and routers.

By using a hop limit set to 255 in all these messages, it is assured that they are discarded by a host, if ever they pass a network boundary (which would lead to a decrement of the value in the Hop Limit field).

#### 2.1.2.1 Router Solicitation

Router solicitation (figure (2.3)) is used by hosts to request a router advertisement from a neighboring router, immediately after one of the host's interfaces became enabled.

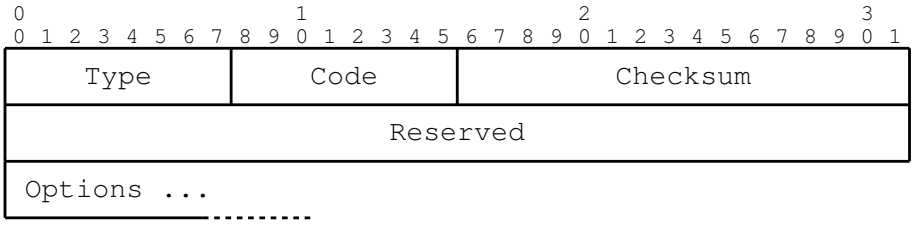



Figure 2.3: Router Solicitation message format

At the time of writing, the only valid option is the source link-layer address, that must not be included if the IP source address is the unspecified address (the host has not yet a proper IP address), otherwise it should be included.

Router solicitations are normally send to the all-routers multicast address.

#### 2.1.2.2 Router Advertisement

Router emit router advertisements periodically and upon explicit request (through a router solicitation) to propagate their presence on a link. These advertisements contain link related information, such as prefixes, a suggested hop limit value, etc. They may also be used to inform hosts on how to perform address autoconfiguration, or to submit link parameters such as the MTU.

The source address field of the IP header must be set to the link local address of the router interface, the message is sent from. The destination address is either the all-nodes

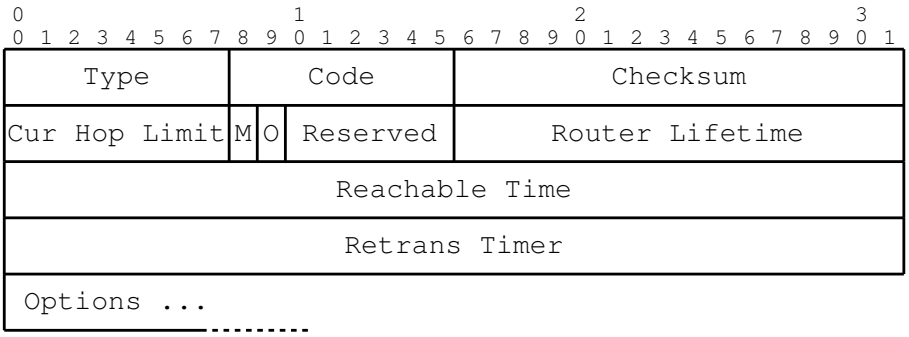



Figure 2.4: Router Advertisement message format

multicast address or, in case of a solicited advertisement, the source address of the invoking host.

Figure 2.4 shows the Router Advertisement message format. The M bit and the O bit are used to indicate the address autoconfiguration policy on the link (for information on stateless address autoconfiguration see [RFC2462]).

Router Lifetime indicates the time a router can be used as default router. If set to zero, the router is no default router. The Reachable Time field specifies the time, a node assumes a neighbor to be reachable after having received a reachability confirmation. Retrans Timer is the time between retransmitted neighbor solicitation messages. The latter two fields set to zero means that they are not specified by this router.

The options field may contain the options Source Link-layer Address (sending interface link layer address), MTU (necessary on links with variable MTU) and Prefix Information. Prefix Information is used to announce the prefixes used on a link (e.g. for address autoconfiguration).

A router advertisement contains a Prefix Information option (figure 2.5) for every prefix that is advertised. The option contains the prefix itself and a set of prefix related variables. The L bit flag, called on-link flag, indicates that a prefix can be used for on-link determination. The autonomous address-configuration flag A tells a host whether the prefix can be used for autonomous address configuration or not. Valid Lifetime contains the time, the prefix is valid for on-link determination and if it is usable for stateless address autoconfiguration, Preferred Lifetime the time, a prefix remains preferred for stateless autoconfiguration.

#### 2.1.2.3 Neighbor Solicitation

Neighbor solicitation messages are used to receive the link-layer address from a target node. They have no special function for routers. The soliciting host can include his own

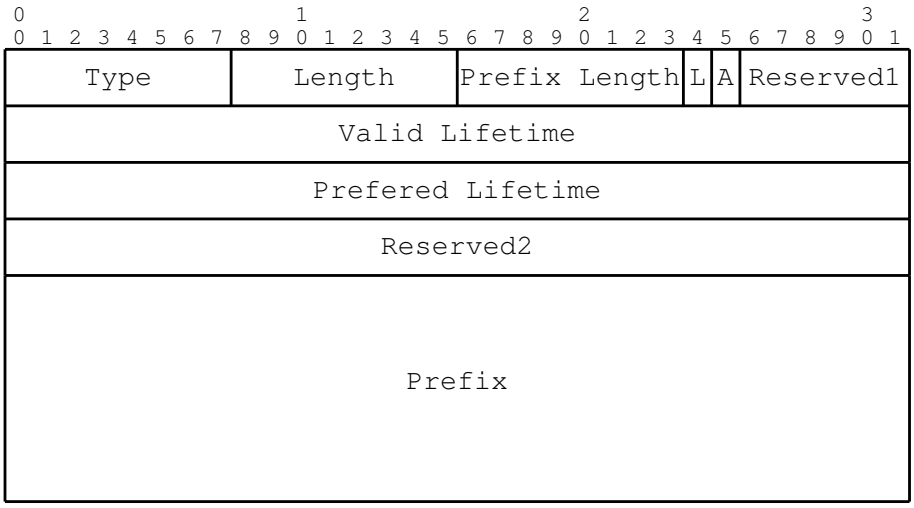



Figure 2.5: Prefix Information option

link-layer address as an option.

#### 2.1.2.4 Neighbor Advertisement

Hosts send neighbor advertisement messages, either in response to a neighbor solicitation, or unsolicited to propagate new informations. The link-layer address is supplied in the Options field.

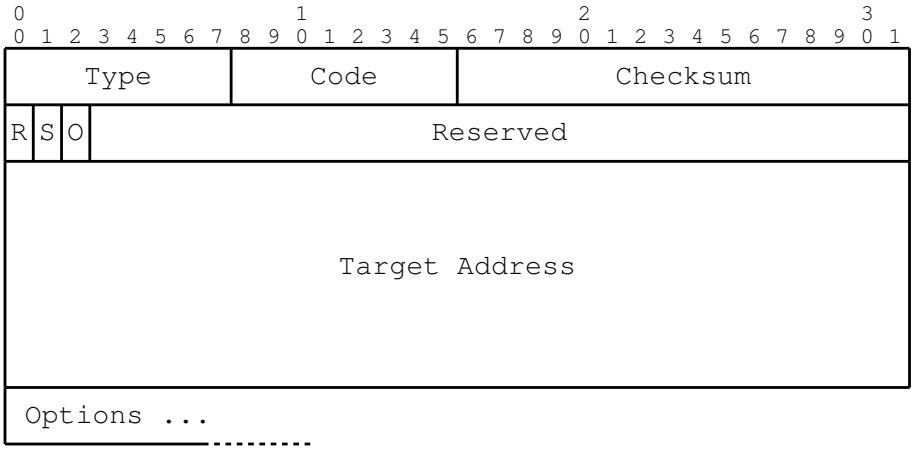



Figure 2.6: Neighbor Advertisement message format

A router has to set the R-bit (figure 2.6) in order to indicate that it is a router. When becoming a normal host it has to set it to zero.

#### 2.1.2.5 Redirect

Router use redirect messages (figure 2.7) to inform a host, that a better first hop for a destination exits, or even the destination can be reached without passing through a router.

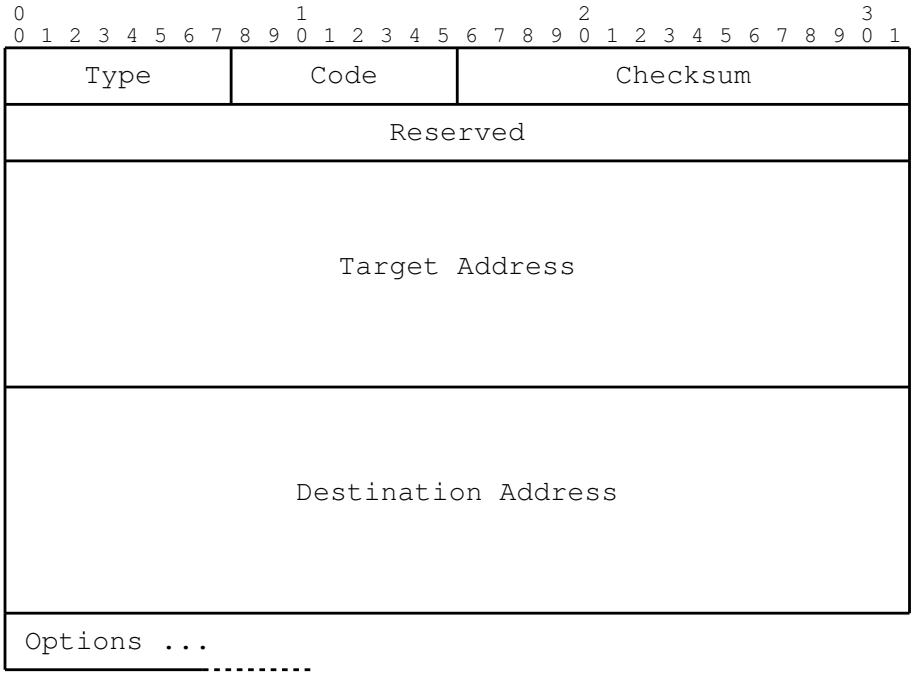



Figure 2.7: Redirect message format

As for the router advertisement, the IP header source address must be set to the link local address of the sending interface. Target Address is the better first hop, and Destination Address is set to the IP address of the destination, which can be reached via the proposed target. The Target Address has to be either the link-local address of a router, or, in case the better first hop is the destination, to be equal to the address in the Destination Address field.

Possible Options are Target Link-layer Address and Redirected Header. The latter contains as much as possible from the package that caused the redirection message, not depassing a maximum of 1280 octets.

## 2.2 Mobile IPv6

Mobile IPv6 (MIPv6) is an extension to IPv6 to support host mobility. Its current specification is supplied in form of an IETF working draft [MIPv6].

A mobile host can always be reached and identified by its home address, a global IPv6 address in its home network. A mobile host is away from home if it is connected to another network than its home network, a so called foreign network. While visiting a foreign network, the mobile host uses a care-of address, a global IPv6 address with a prefix belonging to the foreign network. The care-of address serves to assure the permanent reachability of the mobile host, as well as to identify its current location.

In order to be able to receive packets destined to its home address while visiting a foreign network, the mobile host registers its care-of address at a home agent (HA), located in the mobile host's home network.

A home agent is a proxy that intercepts all packets sent to a mobile host's home address while the host is away from home. The intercepted packets are forwarded (tunneled) to the mobile host's currently registered care-of address.

Packets sent from the mobile host to a correspondent node (CN) are also routed through the home agent. Thus, IPv6 mobility does not require MIPv6 support on the correspondent node side.

Alternatively, if the correspondent node does support IPv6 mobility, a direct connection between the correspondent node and the mobile host can be established. This process is called route optimization.

The figures 2.8-2.10 illustrate an example connection between a mobile host and a correspondent node. At the beginning (figure 2.8), the mobile host is attached to its home network. Then it moves to a foreign network (figure 2.9). Communication between the mobile host and the correspondent node is maintained via the home agent. Finally route optimization is used, which results in a direct mobile host - correspondent node connection (figure 2.10).

As already mentioned above, Mobile IPv6 does not require changes in other parts of the Internet than the home network. This means, visiting a foreign network does not require that the foreign network provides MIPv6 specific functionalities. The support of normal IPv6 (i.e. sending router advertisements for address autoconfiguration) is sufficient. On the home network side, only the installation of a home agent (and of course a mobile host) is required.

Mobile IPv6 does merely have any influence on upper-layer connections, since Mobile IPv6 in general is transparent for upper layers. Only short connection interruptions during network switch might be visible.

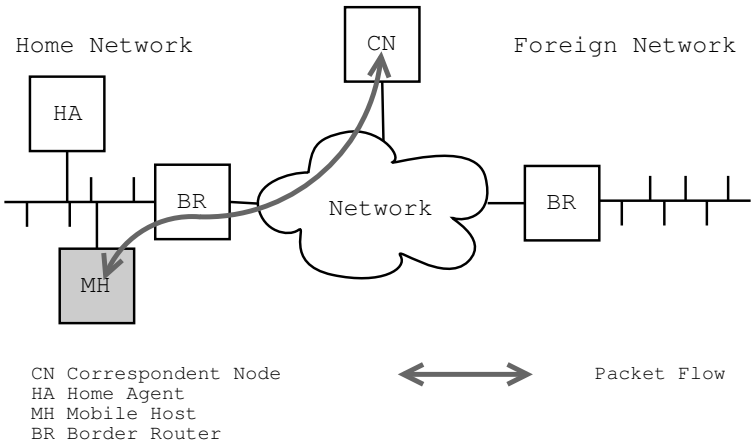



Figure 2.8: Mobile Host at home

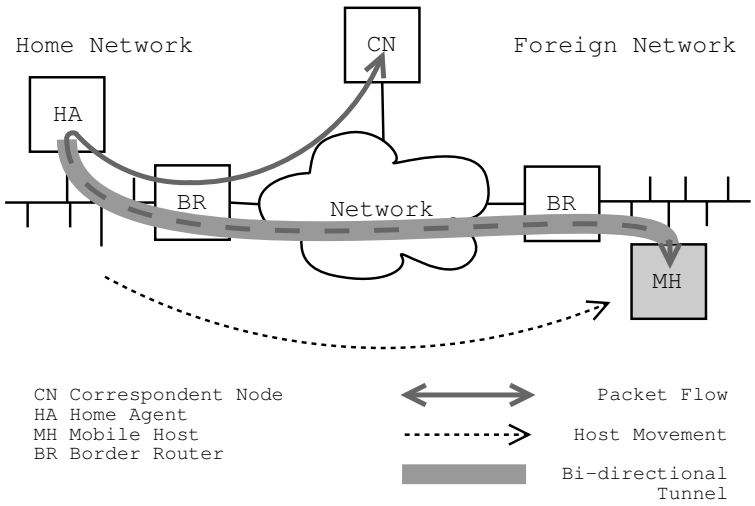



Figure 2.9: Mobile Host away from home, no route optimization

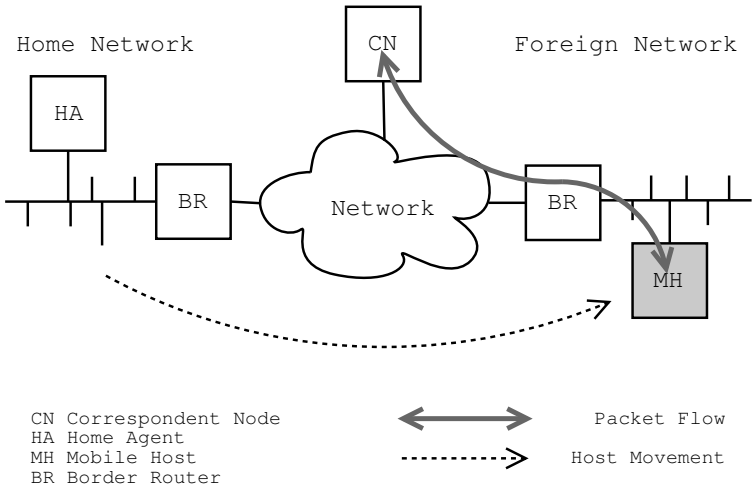



Figure 2.10: Mobile Host away from home, route optimization

### 2.2.1 Entities Description

This part gives a general overview on the entities that take part in a typical Mobile IPv6 scenario. It mentions the entities features without describing protocol details. Protocol details are covered in the following subsections. Further information can be found in [MIPv6].

#### 2.2.1.1 Mobile Host

While moving, a mobile host is able to connect to different access networks. Every time it connects to a new network, it has to update its bindings with other nodes. Bindings are used to establish a link between the mobile host's home address and its current care-of address.

Movement detection on IP level is done by examining the received router advertisements. A host has moved, when it receives router advertisements with prefixes that differ from the ones used before and no network renumbering took place.

To describe the binding management on the mobile host side, the authors of [MIPv6] introduce the binding update list. It contains one entry for every binding that a mobile host has, or is trying to establish, with a specific node. This includes correspondent registrations (route optimization with a correspondent node) and home registrations (binding with a home agent). Entries from the list are deleted, as the lifetime of the binding expires. Alternatively a binding may get replaced by another binding or removed upon explicit request.

A mobile host's home agent might either be configured manually or chosen automatically out of a list of possible home agents, build from the results of the dynamic home agent address discovery mechanism. This mechanism is described in section 2.2.1.2.

To maintain communication while visiting a foreign network, packets are either send and received via the bidirectional mobile host - home agent tunnel, or otherwise exchanged directly with the correspondent node in case route optimization was successful.

#### 2.2.1.2 Home Agent

A home agent is a special node in the home network of a mobile host. Since one of the home agent's tasks is to forward packets on IPv6 base, it can be called a router. But this does not mean, that it has to be a default router for normal machines on the home link at the same time, or that it must have more than one interface.

Mobile hosts, that are connected to a foreign network, register themselves at a home agent in their home network. The home agent has to verify the authenticity of the mobile hosts. This aspect is evaluated in section 2.2.3.

After a successful registration, the home agent defends the mobile host's home address. In other words it claims to own the home address and responds to neighbor solicitations as if it were one of its own addresses (for details on neighbor discovery see 2.1.2 or [RFC2461]).

Packets sent to the mobile host's home address are intercepted by the home agent and tunneled to the mobile host's currently registered care-of address. Vice versa, mobile hosts have the possibility to send packets via the home agent towards any other node.

To keep track on the currently registered mobile hosts and their care-of addresses, the authors of [MIPv6] introduce a conceptual data structure, called the binding cache (BC). The binding cache, contains one binding entry for every registered mobile host. It establishes the correspondence (the binding) between a mobile host's home address and its current care-of address.

The fields of an entry in the binding cache are described in [MIPv6]. Among others, there are the home address of the mobile host and its care-of address. The entries are created during the binding update process and are deleted upon lifetime expiration or explicit request (e.g. a host returns to its home link).

Another task of a home agent is to take part in the dynamic home agent address discovery mechanism. This mechanism permits mobile hosts to find home agents on their home network. The mobile host broadcasts a request to the Mobile IPv6 Home Agents Anycast address and a home agent should answer with a list of possible home agents. For this purpose, home agents maintain the home agents list, a list that contains global reachable IPv6 addresses from other home agents on the same link. The list is kept up-to-date from informations transmitted with and extracted from router advertisements (see 2.2.2.5 for the MIPv6 extension to router advertisements).

#### 2.2.1.3 Correspondent Node

Every node that supports standard IPv6 can be a correspondent node. Mobile IPv6 does not require any specific extension. Of course, route optimization is only possible if the correspondent node supports the necessary protocols.

To be able to use route optimization, a correspondent node has to implement the binding cache concept and must support the binding update process. If a correspondent node sends a packet to any other node, the binding cache must then be searched before the destination cache. This assures, that existing bindings to mobile host are used, instead of trying first to send it to the home address of the host.

The requirements concerning the maintenance of the binding cache are the same as for home agents.

#### 2.2.1.4 Access Router

In general, as for the correspondent node, access routers need no modification. But sending router advertisements at a higher rate than specified in [RFC2460] will fasten up the IP based movement detection. Hence, it is reasonable to redefine the minimum value and maximum value of the time interval between two unsolicited router advertisements, as suggested in [MIPv6].

### 2.2.2 Protocol Description

#### 2.2.2.1 MIPv6 Mobility Header

By using the Mobility Header, Mobile IPv6 defines a new IPv6 protocol. A complete description of this header is given in [MIPv6]. It is used for the transmission of messages related to the binding management.

The header is able to carry the following messages:

| | |
|---|---|
| Home Test Init (HoTI) | Sent during the Return Routability procedure from a mobile host to a correspondent node, through the mobile node - home agent tunnel (see 2.2.3.1). |
| Home Test (HoT) | Response to the HoTI message, send from the correspondent node to the mobile hosts home address, and forwarded from the home agent to the mobile hosts care-of address. |
| Care-of Test Init (CoTI) | Send during Return Routability procedure from a mobile host directly to a correspondent node (see 2.2.3.1). |
| Care-of Test (CoT) | Response to the CoTI message, send from the correspondent node to the mobile host's care-of address. |
| Binding Update (BU) | Sent from the mobile host to its home agent and to the correspondent nodes, as soon as the mobile host changes its care-of address. |
| Binding Acknowledgement | Sent in response to a Binding Update from the home agent and from the correspondent nodes to the mobile host |
| Binding Refresh Request | Requests a mobile host to send a Binding Update, normally sent by a correspondent node. |
| Binding Error | Sent from a correspondent node to the mobile host, in case of an error related to the binding process. |

The binding process and the usage of these messages are described in section 2.2.3.

#### 2.2.2.2 MIPv6 ICMP Messages

Mobile IPv6 defines the following ICMP messages:

| | |
|---|---|
| Home Agent Address Discovery Request | Sent from a mobile host to the Mobile IPv6 home agents anycast address to discover home agents on a local link (figure 2.11). |
| Home Agent Address Discovery Reply | Reply from a home agent to a Home Agent Discovery Request from a mobile host (figure 2.12). The home agent addresses field contains the list of all home agents known to the sending home agent. |
| Mobile Prefix Solicitation | Sent from a mobile host, while being away from home, to receive a Mobile Prefix Advertisement to be able to collect prefix information from its home network. |
| Mobile Prefix Advertisement | Sent from a home agent to mobile hosts that are away from home. It contains prefix informations from the mobile host's home link. It may be sent solicited (by a Mobile Prefix Solicitation from a mobile host), or unsolicited, if prefix modifications in the home network require it. |

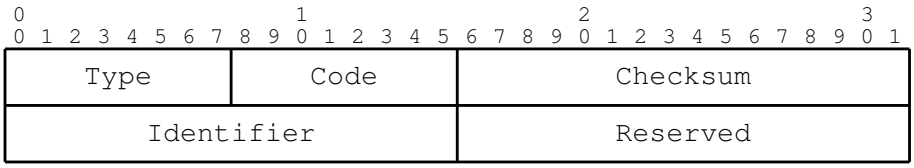



Figure 2.11: ICMP Home Agent Address Discovery Request Message

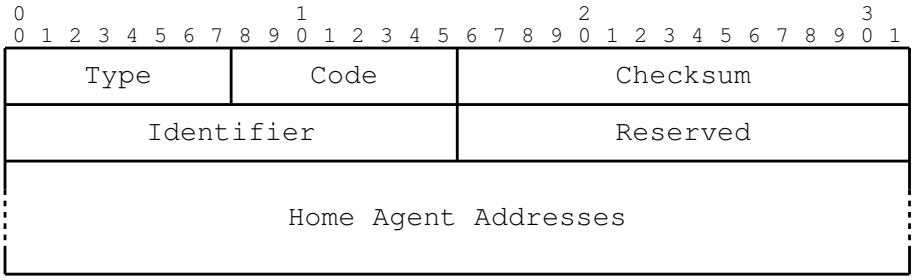



Figure 2.12: ICMP Home Agent Address Discovery Reply Message

#### 2.2.2.3 MIPv6 Destination Option

MIPv6 introduces a new destination option, the Home Address option. Mobile host's use it to transmit their home addresses in packets sent directly to a correspondent node while being away from home (only possible if route optimization between the mobile host and the correspondent node was successful).

The option is required, since a mobile host cannot put its home address in the source address field of the IPv6 header since the packets might get dropped during egress filtering of the access router in the visited network. As specified in [RFC2460], a destination option is only examined by the node, identified by the destination address in IPv6 header, not by intermediate routers. Therefore, the Home Address Option solves the potential problem of egress filtering.

#### 2.2.2.4 MIPv6 Routing Header

As equivalent to the Home Address option, only for the other direction, packets routed directly from a correspondent node to a mobile host's care-of address, Mobile IPv6 defines a new routing header, the type 2 routing header. The destination address field of the IPv6 header is filled with the care-of address, the type 2 routing header carries the home address. Upon packet reception, a mobile host replaces the care-of address with the home address and by this means, mobility remains transparent for upper layers.

#### 2.2.2.5 Extensions to the IPv6 Neighbor Discovery Process

Mobile IPv6 adds the H bit (see figure 2.13) to the IPv6 Router Advertisement message format, that serves to indicate that a router is serving as home agent on a local link. This is part of the dynamic home agent address discovery process.

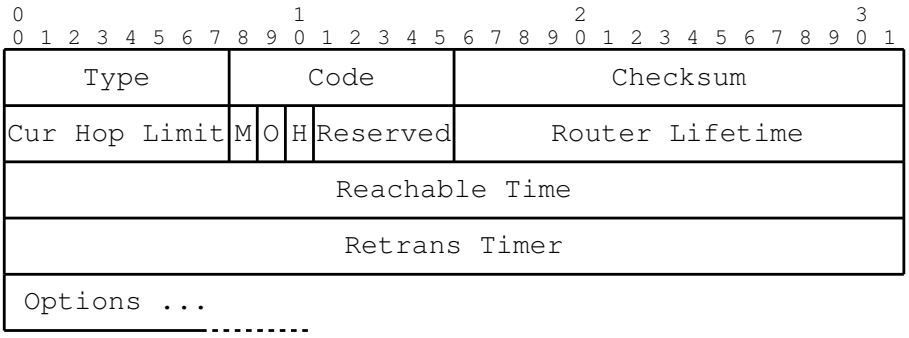



Figure 2.13: Extended Router Advertisement message format

All home agents, even those, that do not serve as default router (e.g. a home agent with only one interface), should, with respect to the home agent discovery process, send router advertisements containing the H bit set to one.

The second extension modifies the Prefix Information option format of IPv6. The Router Address bit (R) is added, to enable a router to communicate one of its complete global IPv6 addresses. If the R bit is set, the Prefix field in figure 2.14 contains a complete IPv6 address instead of only a prefix.

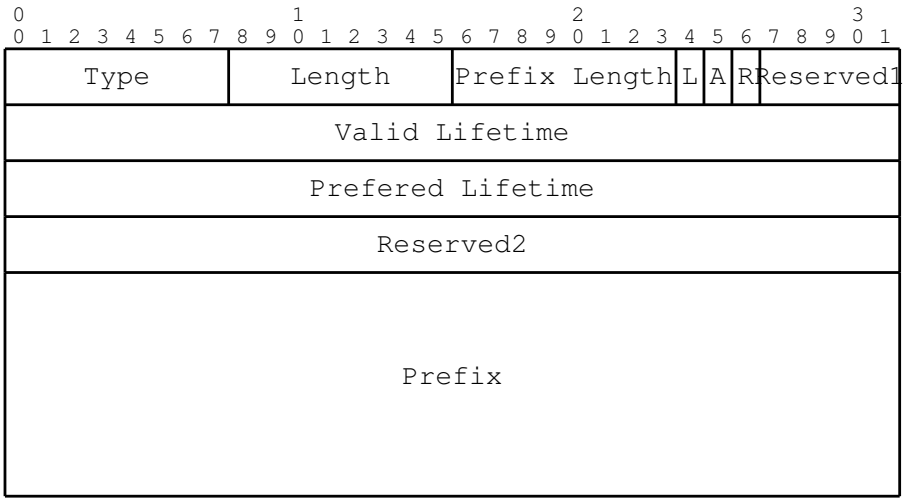



Figure 2.14: Extended Prefix Information option format

Finally, Mobile IPv6 introduces the home agent information option (figure 2.15), which is used by home agents to transmit additional home agent related information in router advertisements. The additional information consists in the home agent preference and the home agent lifetime. The higher the preference value, the more likely a home agent is used by a mobile host. The home agent lifetime indicates for which amount of time a router is to be considered usable as home agent.

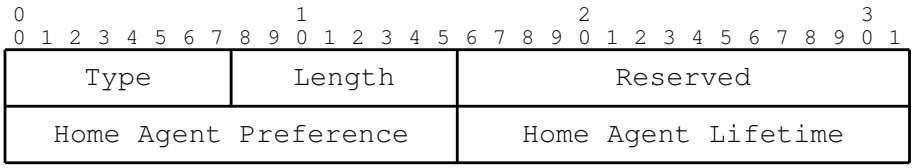



Figure 2.15: Home Agent Information option format

By using the home agent information option field, a home agent is capable to participate in the home agent discovery mechanism without having to be a default router at the same time. Again, as an example one might think of a node, that is a dedicated home agent (the node has only one interface). Such a home agent should send router advertisements with a Router Lifetime field set to zero and non zero value for the home agent lifetime.

As mentioned above (see 2.2.1.4), according to [MIPv6], the time between two unsolicited router advertisements should be reduced to smaller values than specified in [RFC2460], if the link is meant to be an access point for visiting nodes. Shorter time intervals increase the performance of the movement detection.

### 2.2.3 Security Aspects

In addition to the normal security issues in an IPv6 network, mobility introduces new, mobility specific security aspects. Apart from problems like the protection against replay and the assurance of data integrity, authentication during binding updates and during the Route Optimization process is very important.

Figure 2.16 shows the message flow, during a binding update, between a mobile host, that detects movement by reception of a router advertisement, and its home agent.

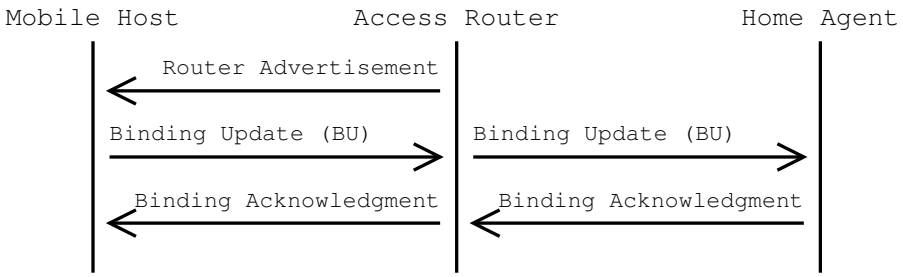



Figure 2.16: Binding Update message flow

Questions that do appear concerning a mobile node's binding update with a home agent on its network are:

- How can a mobile host assure, that it can trust a node on its home network that claims to be a home agent?
- How can a home agent verify, that any host that tries to register a care-of address for a specific home address, has the right to do so?

As a solution, the authors of [MIPv6] suggest that binding updates should be secured by an IPsec security association. This requires a security policy database on the home network, that has entries to unequivocally identify a single security association for any given home address and home agent. Alternatively, every home agent may have its proper security policy database. By this means, node authenticity is assured.

A similar problem occurs during route optimization:

- How can a correspondent node assure, that a node, using no matter what IPv6 address, is a mobile node having a specific home address?

In contrary to the binding updates, route optimization does not require the configuration of security associations and it does not require an authentication infrastructure between the mobile host and the correspondent node. Instead of, a mechanism, called the Return Routability procedure (see 2.2.3.1) is used.

#### 2.2.3.1 Return Routability Procedure

By the usage of this procedure, a correspondent node can determine whether a mobile host is really addressable at a given care-of address, or not. It makes use of the fact, that the home agent and the mobile host already have a security association, and thus the home agent can always identify an ill-behaving mobile host.

The message flow during the procedure is illustrated in figure 2.17. It is started by the mobile host that sends two messages to the correspondent node. The first via its home agent (Home Test Init (HoTI)) and the second directly to the correspondent node (Care-of Test Init (CoTI)).

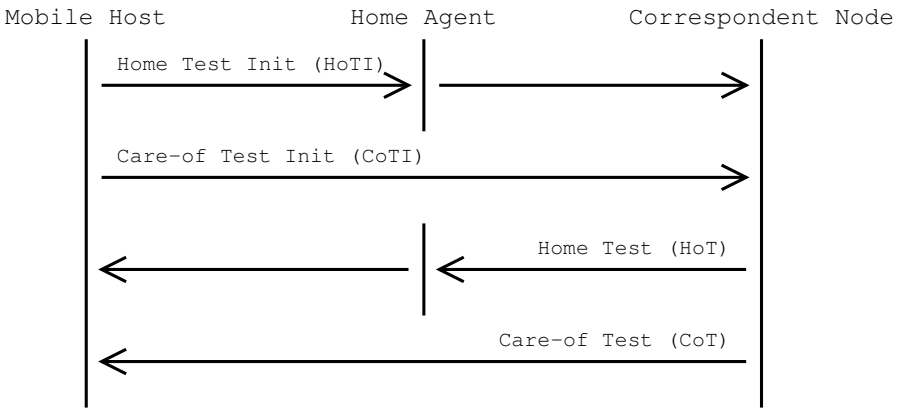



Figure 2.17: Message flow during Return Routability procedure

The correspondent node responds to these two messages using the same paths. The Home Test (HoT) message is send to the home address of the mobile host, the Care-of Test (CoT) message to the care-of address.

The Return Routability procedure is complete when the mobile host has received both answers, the HoT and the CoT message. Using the data contained in these two messages, it creates key to secure the binding updates with the correspondent node (see [MIPv6]).

## 2.3 IPv6 Routing

Routing is the process of forwarding packets for other network nodes. As described in section 2.1, the specification of IPv6 contains features, introduced to provide packet forwarding for nodes that are located on separate network segments within an IPv6 based network.

IPv6 packets contain a source address and a destination address, that are used to route data through a network. IPv6 layer services on a router examine the destination address of each packet, compare the address to a locally maintained routing table, and then determine how to deliver the packet.

### 2.3.1 IPv6 Router

A router is a node that forwards packets that contain a destination address not associated with any of the routers interfaces. In other words, routers deliver packets for other network nodes. One might also define a router as machine attached to two or more network segments, that is enabled to forward packets between them. While in the general case, the second definition is right, it does exclude a simple Mobile IPv6 home agent, having only one interface, that is covered by the first definition. An IPv6 router is a router that supports IPv6 and is compliant to [RFC2460] and [RFC2461].

Upon packet reception, an IPv6 node acts as illustrated in figure 2.18. The first step is to verify whether the packet's layer two address corresponds to one of the addresses the node is really listening to, because of using any software setting one of the node's interfaces to the promiscuous mode (e.g. ethereal), will make the IP stack receive all packets distributed on a shared media, which finally could result in unwanted packet duplication. If the layer two destination address does not match, the packet is silently deleted.

The second step is to examine the destination IP address. If the packet is destined to the host itself (this includes the host's multicast addresses), it acts as a normal host. If not and if the node is no router neither, the packet is discarded. Otherwise, the packet is processed by the router implementation (see figure 2.19).

### 2.3.2 Routing Table

Since a router requires information on how to deliver an IP packet with a destination address not being one of its own addresses, it maintains an address database, called routing table. By comparing data from the IPv6 packet with entries in the routing table, it tries to find a suitable next hop for the packet. That is either another intermediate router or the destination node itself.

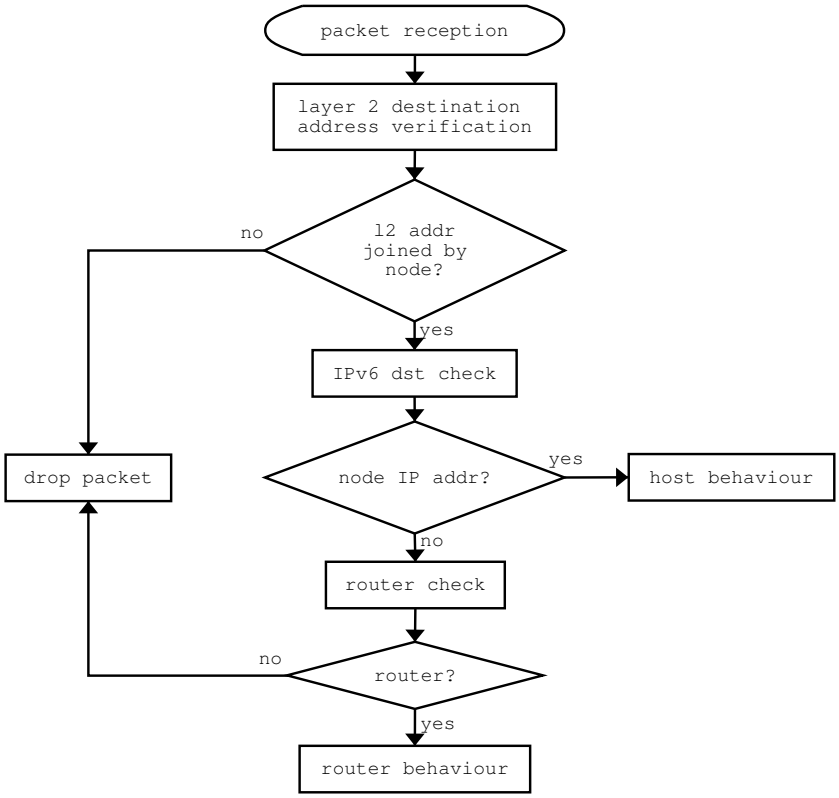



Figure 2.18: IPv6 router packet reception

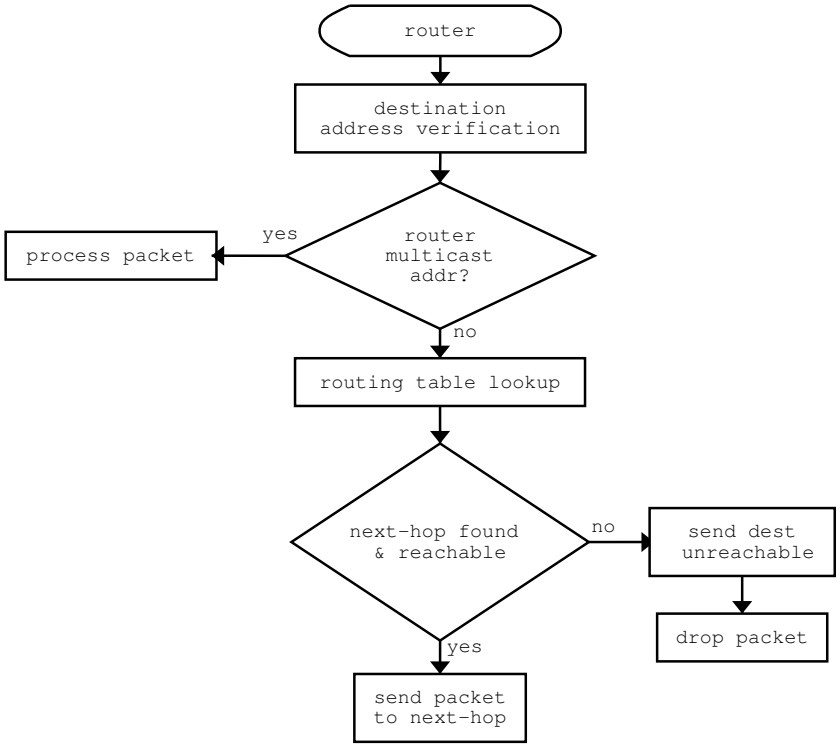



Figure 2.19: IPv6 router packet processing

The routing table can either be configured manually by an administrator or during runtime by usage of dynamic routing protocols.

A minimal routing table structure contains the following information

- Destination IP address or destination network address.
- Destination network prefix length (zero if the destination is a host).
- Gateway address, that is either a host address or zero. A host address requires a route to the host address or network. Zero means that the destination is located on a directly attached link.
- Destination (outgoing) device.

Additional items may be required if informations for routing protocols will be stored directly in the routing table, for example the metric or the link cost.

Any possible destination network can be identified by a prefix and an associated prefix length. Every IPv6 address that matches a certain prefix, virtually belongs to the related network segment and to its sub-network. To find the optimal destination network for an IP packet, a router has to compare the destination address field of the packet in question with its routing table. In general (not considering any routing policies), the optimal destination network is the one with the longest prefix, matching the packet's destination address.

The longest matching prefix problem is common to all router implementations, independent of the router architecture. The author of [FLPM] for example, proposes a set of advanced algorithms to solve the the problem in hardware and in software.

The simplest solution is an algorithm that searches in an unsorted list. As soon as it discovers a suitable prefix, the entry is kept until a longer matching prefix is found, or the last list element is reached. Obviously, this algorithm does not scale for large routing tables.

A better solution is to use a binary tree structure. Figure 2.20 shows an example of such a structure. Compared to the previous solution, a binary tree provides a significant performance improvement on bigger routing tables.

To optimize the binary tree algorithm, a radix tree structure similar to the one described in [RADIX] might be used.

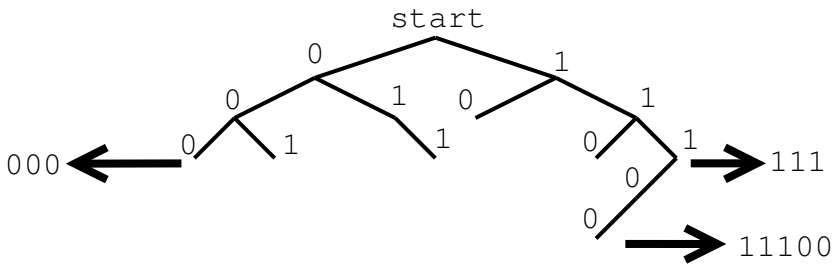



Figure 2.20: Binary routing tree example

### 2.3.3 Static Routing

Static routing means, that routing table entries are added manually. The table does not change without human interference. In case of a link failure, the router is does not modify routes itself, the destination becomes unreachable. Static routing requires a lot of administrative work and the maintenance of larger networks is nearly impossible.

### 2.3.4 Dynamic Routing

Dynamic routing is the result of the wish to solve the disadvantages and problems of static routing. Routing protocols are used to exchange messages containing details on the current network topology and current network status. The informations are used to dynamically adapt the local routing table.

Implementing a dynamic routing protocol provides the following advantages

- Automatic reconfiguration in case of a link failure.
- Alternative routing possibility on link overload.
- Automatic detection of topological changes.
- Simpler configuration of large networks.

Dynamic routing protocols are often realized as routing daemons. They run independent from the IP stack and and modify the IP stacks routing table through a predefined interface. Normally, the daemon uses the same interface as the utility used for static configuration, that means dynamic routing can be implemented transparent for the IP stack and does not require changes to the stack itself.

Routing protocols can be divided into two categories. Distance vector based routing protocols and link state based routing protocols.

#### 2.3.4.1 The Distance Vector Algorithm

Every router has a routing table that contains additionally, to the entries mentioned before, a metric. The distance vector is represented by the destinations and the metrics. The metric corresponds to the number of router separated links between this router and the final destination. As an alternative, different links might be represented with different metrics, for example a slow link is represented by a higher metric than a faster link.

The distance vector algorithm is based on message exchange between adjacent routers. The routers broadcast their distance vectors to their physical links. Every router that receives a broadcast from another router, compares the vectors to its own routing table. It decides to update its routing table, if the broadcast contained a new route to an unknown destination or a shorter route to a known destination.

Modifications to the own routing table are redistributed with the next broadcast. This continues until the algorithm converges. Routers continue to broadcast periodically, to assure to keep the routing tables up to date and to be able to detect a link or router failure. After failure detection, alternative routes diffused.

To avoid convergence problems on router failures, a maximum metric has to be defined. Even then, in general, a failure results in a long time of routing table reconfiguration.

An example for distance vector based IPv6 routing protocols is RIPng.

#### 2.3.4.2 The Link State Algorithm

The link state algorithm has been proposed to solve the convergence problem of distance vector based protocols.

Link state assumes that routers have a complete vision of the network topology (or at least a part of it).

The algorithm can be divided in the following steps:

- Every router discovers its neighboring routers.
- Every router creates the link state packet (LSP), a packet containing a list of neighboring routers and the link cost (metric) to reach them.
- The link state packets are distributed to all other routers.

- Based on the received information, every router calculates the shortest path towards all other routers.

In the initial phase, the first task of a router consists in discovering its neighbor routers. It broadcasts a special message, the so called HELLO message, on all its interfaces and the other routers respond.

After reception of the responses, the router builds its link state packet, that contains the identity of the source router, a sequence number, the packet age, the list of its neighbor routers and the link costs. The link state packets are recreated and distributed either periodically or after a topology change, caused for example by a link failure or router failure.

A router redistributes every received link state packet on all its interfaces, except on the one it received the packet. Before redistribution, the router decrements the sequence number. The router keeps track on received packets by storing pairs of source router identity and sequence number. When receiving a packet, it is compared to the known packets, and if unknown it is stored and redistributed. If it is known, which means, it contains the same router identity and the same sequence number, the router discards the received packet. In case of a lower sequence number, the packet is rejected.

The link state packet contains a second information that can force a router to destroy, the packet age. Every second, the packet age is decremented by one unit. As soon as the value reaches zero, the information of the packet is destroyed.

Finally, all routers will have the same information on the topology of the network. The routers then calculates routes to all connected subnetworks using the shortest path first (SPF) algorithm. The result is used to maintain the local IP stack routing table.

An example for a link state based routing protocol for IPv6 is OSPFv3 (Open shortest Path First, version 3).

## 2.4 IPv6 based Mobile Networks

This section discusses, how the Mobile IPv6 specification can be extended, to support not only host mobility, but also network mobility. The primary aim is to provide network mobility with as less modifications to standard Mobile IPv6 as possible.

As a preliminary requirement, network mobility has to be transparent for normal nodes in the mobile network, since they do not necessarily support mobility. This includes, that all nodes in a mobile network must always be reachable at their home address.

Thus, a mobile network requires a modified home agent, when being away from home, because instead of having only a home address, the network possess a home prefix. It has to intercept all packets with a destination address that matches the home prefix of the mobile network and then forward them to the mobile network.

Obviously, the node that provides mobility for a network has to be a special router, the so called mobile router (MR). Like a mobile host visiting a foreign network, the mobile router that attaches to a link other than its home link, auto-configures a care-of address in the foreign network and then starts the binding update process with a home agent of its choice.

After the establishment of the binding, it is the mobile router that receives tunneled packets from the home agent and delivers them to the destination nodes. Vice versa, packets from nodes within the mobile network are encapsulated by the mobile router, send through the tunnel, decapsulated and finally forwarded by the home agent.

The re-use of route optimization, as specified for Mobile IPv6, is impossible out of security reasons (see 2.4.3).

The figures 2.21-2.22 illustrate an example scenario, where a correspondent node communicates with a local fixed node (LFN) located in a mobile network. The mobile network starts at home (figure 2.21). Then it moves to a foreign link (figure 2.22). Communication is maintained through the bi-directional mobile router - home agent tunnel.

The packet flow for the example scenario is shown in figure 2.23. It also contains the messages exchanged between the mobile router and its home agent during the binding update process.

As well as Mobile IPv6, IPv6 network mobility has to be transparent for upper layers.

### 2.4.1 Entities Description

As in the introduction to Mobile IPv6, this part presents the entities that take part in a mobile network scenario. Compared to the Mobile IPv6 scenario, the mobile host is replaced by a mobile network, consisting of a mobile router and a local fixed node. Of

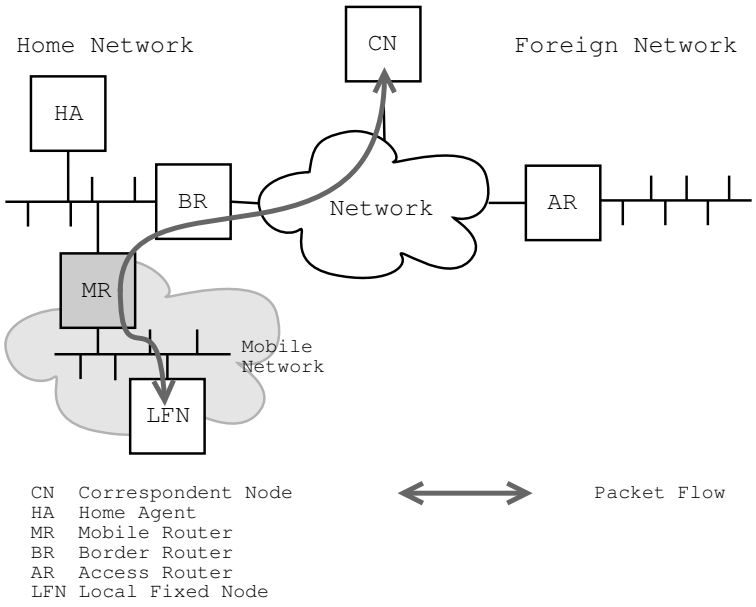



Figure 2.21: Mobile Router at home

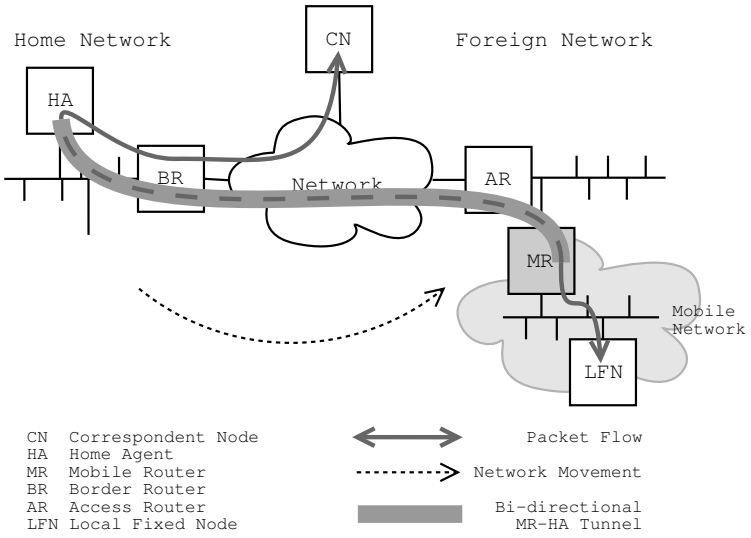



Figure 2.22: Mobile Router away from home

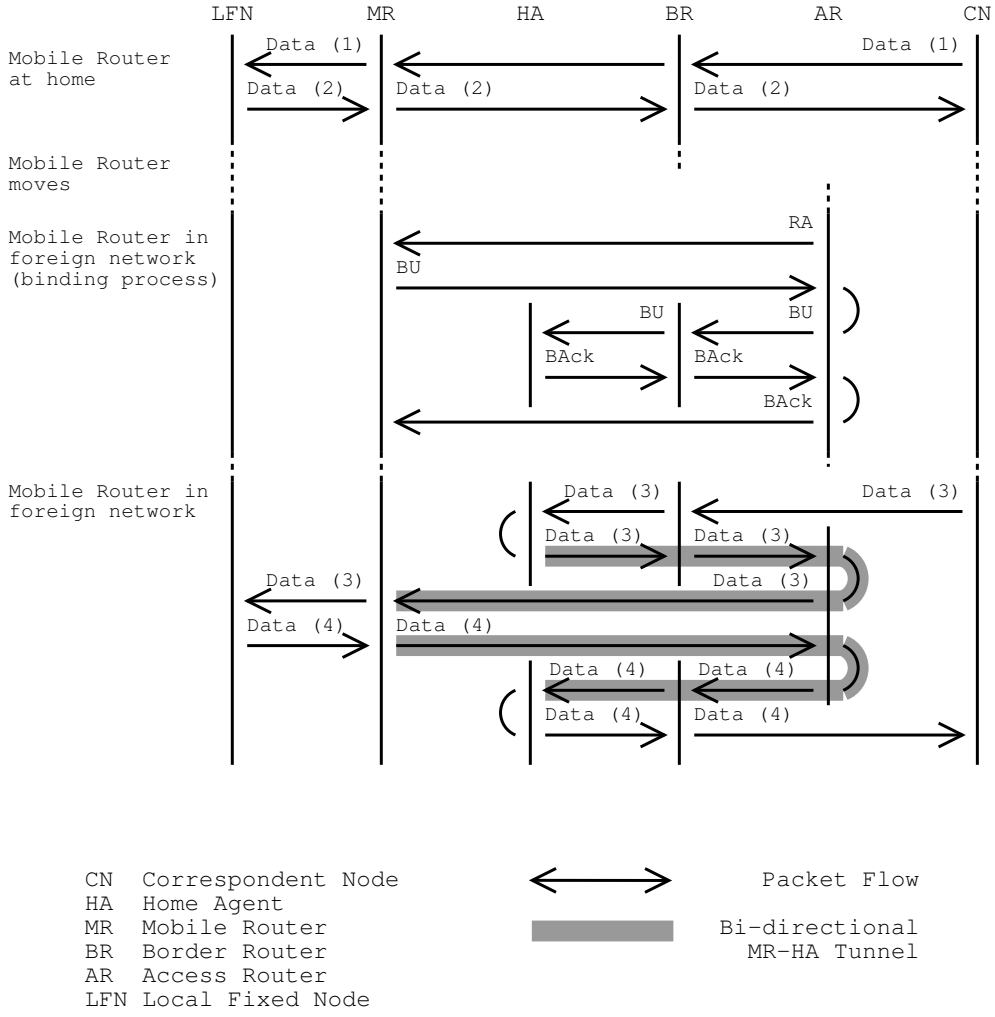



Figure 2.23: Mobile Router example scenario packet flow

course, the local fixed node could be replaced by any other combination of nodes, including mobile nodes and other mobile networks.

#### 2.4.1.1 Mobile Router

A mobile router is an extended mobile host, that is able to connect a local fixed network to different access networks, while the local fixed network does not have to change its network address. The network is fixed in terms of its permanent attachment to the mobile router.

As a mobile host, the mobile router detects movement by examining router advertisements from other routers. As soon as one of the advertisements contains only prefixes that are unknown to the mobile router, it deduces that it must have moved.

Upon detection of a movement, a mobile host auto-configures a care-of address belonging to a prefix of the visited network. For a mobile router, this behavior is not that obvious, since in general, one does not want a router to auto-configure interface addresses with prefixes obtained through router advertisements. Address auto-configuration on a router is a potential threat, as it could be abused by ill-behaving nodes, that advertise wrong prefixes. The problem is solved by defining an interface, or a set of interfaces, called mobile interface(s), on which the mobile router will auto-configure addresses, whereas on the other interfaces it does not. This solution limits the potential threat.

In a foreign network, the mobile router configures a care-of address on a mobile interface and than registers this address at a home agent in its home network. On a change of its point of attachment, it has to update its binding with the home agent, to point to its new care-of address. The binding management is the same as for mobile hosts.

A mobile router should always set the L bit in its binding update messages (for a closer description see [MIPv6]), to tell the home agent to defend the mobile router's link local address and intercept packages send to it. During the choice of the mobile router's home address, it has to be assured, that the link local address can be derived from this address. The home agent derives the link local address by replacing the subnet prefix in the mobile router's home address with the link local prefix. An extension of the binding update process to transmit the prefixes, that are used in the mobile network, is not required (see 2.4.3)

The choice of a home agent is nearly equivalent to the Mobile IPv6 scenario. Either it configured manually or automatically by an extended dynamic home agent discovery mechanism. The extension (see 2.4.2.1 for details) in comparison to Mobile IPv6 is required, since a mobile router needs a home agent that is capable to manage mobile routers.

Packets from and to the mobile network are always routed through the bi-directional mobile router - home agent tunnel. Route optimization, as for a mobile host, is impossible due to security reasons (see 2.4.3).

#### 2.4.1.2 Home Agent

The home agent for a mobile router is basically the same as for a mobile host, with the difference that it has to be able to manage a list of network prefixes that can be reached via the mobile router.

Mobile routers that are away from home, register their care-of addresses at the mobile router. The home agent has to verify the authenticity of a mobile router and which prefixes are attributed to the fixed network behind it.

If the registration is successful, the home agent defends the mobile routers home address and link local address. Defending the mobile router's link local address is important for the routing process. Routers usually have routing tables configured with link local addresses instead of global addresses, so defending the link local address of a mobile router and thus responding to neighbor discovery messages is essential for a mobile router's home agent. Otherwise, other routers on a mobile router's home link would have to use the mobile router's global home address in their routing tables, which, for example, would force a modification of all routers routing tables, in case of the renumbering of a network. Likewise, dynamic routing protocols normally use link local addresses and not global IPv6 addresses.

Whereas the home agent forwards all packets destined to the home address of the mobile router, whether to forward packets with a destination address field containing the link local address is a difficult question. On the one hand, it is the home agent that has to participate in the neighbor discovery process on behalf of the mobile router, on the other hand the mobile router may run a routing protocol, that uses link local addresses to exchange data (e.g. RIPng). The simplest solution consist in forwarding all packets but ICMPv6 packets.

The home agent has to know the prefixes of the fixed networks attached the mobile router, because packets sent to a node in these networks, and thus packets meant to pass through the mobile router and the home agent, are not send to the IPv6 address of the mobile router (neither the home address, nor the local link address) but directly to the destination node's global IP address. As described in section 2.3.1 it is only on the home link, where other nodes set the layer 2 destination address to the one of the mobile router, respectively to the one of its home agent, using neighbor discovery.

To store the associations mobile network prefix - mobile router care-of address, mainly two conceptual solutions can be considered. The first consists in defining a modification of the binding cache, to provide support for prefixes. The second solution is that the home agents keep a separate list containing the mobile router's home address and the associated prefixes.

Both solutions require a security mechanism, that unequivocally determines, which prefixes are associated to the home address of which mobile router. This mechanism might either be implemented locally on every mobile router supporting home agent, or on a central node in the home network.

Since the binding update process was not modified so far, it is desirable to use the concept of a separate list. This concept has also the advantage, that if a mobile router changes its care-of address, the home agent needs only to modify one entry in its binding cache, not all entries for the different prefixes.

Apart from the normal Mobile IPv6 dynamic home agent address discovery mechanism, a home agent for mobile routers should also support the extended home agent address discovery process. It has to maintain a second list of home agents, that all support mobile routers. Upon receiving an extended home agent address discovery request, the home agent replies by sending only the contents of this second list. For a detailed specification of the extension, see 2.4.2.1.

#### 2.4.1.3 Local Fixed Node

As illustrated in figure 2.21 and figure 2.22, a local fixed node is any node, located in a mobile network. This might be a simple host, a router, a mobile host or even another mobile router. No matter what kind of node it is, network movement of its (home) network is always transparent for this node.

#### 2.4.1.4 Correspondent Node

Similar to the correspondent node in Mobile IPv6, there is no extension to IPv6 required. When a mobile router is away from home, all packets are tunneled though its home agent, unremarkable for the correspondent node.

Route optimization with a mobile router and its local fixed network is not specified by this document.

#### 2.4.1.5 Access Router

The access router is identical to the one in the Mobile IPv6 scenario. Out of the same reason, sending router advertisements in short intervals is desired, but not mandatory (see 2.2.1.4).

### 2.4.2 Protocol Extensions to Mobile IPv6

#### 2.4.2.1 Additions to Dynamic Home Agent Address Discovery

Since mobile routers need home agents with support for network mobility, it is necessary, to extend the existing mechanism, to enable mobile routers to discover home agents that support mobile networks.

The modification consists in adding the mobile router home agent bit M to the ICMP messages for home agent discovery (figures 2.24 and 2.25) and to the home agent information option (figure 2.26).

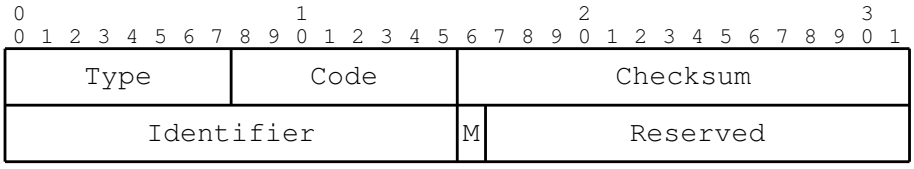



Figure 2.24: Extended ICMP Home Agent Address Discovery Request Message

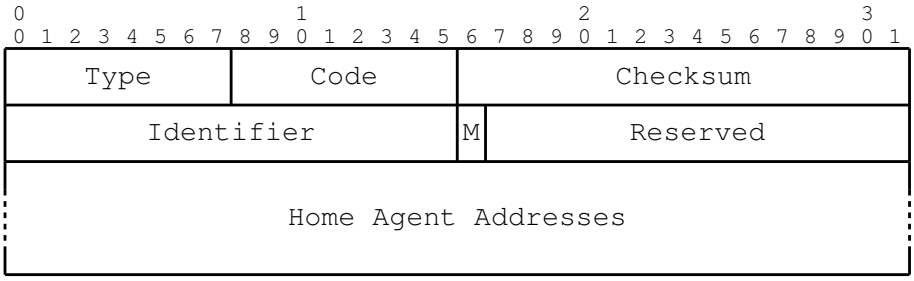



Figure 2.25: Extended ICMP Home Agent Address Discovery Reply Message

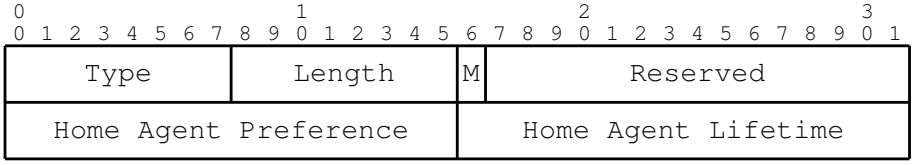



Figure 2.26: Extended Home Agent Information Option Format

Added to the ICMP home agent address discovery request message, the M bit serves to indicate, that a mobile router is searching for suitable home agents with mobile router support. If this request is received by a normal home agent, it will ignore the M bit and send an ICMP home agent information discovery reply message as specified in [MIPv6], without the M bit set. This reply includes all known home agents on the home network, regardless of the support for mobile routers. Otherwise, if the recipient is a mobile router supporting home agent, it will discover the M bit, and send an extended ICMP home agent address discovery reply message as response (with the M bit set to one), containing only home agents with support for mobile networks.

The M bit in the home agent information option is used to indicate to other home agents, whether the sending home agent is capable and willing to act as home agent for mobile

routers. A home agent, that does not support the extension, ignores the M bit and adds the supplied information to its normal home agent list, while a home agent that does recognize the M bit, will add an entry to its mobile router supporting home agent list.

### 2.4.3 Security Considerations

Apart from inheriting the security problems of host mobility, network mobility introduces new security issues, that do not exist in the mobile host scenario.

The binding process between the mobile host (substituted by the mobile router) and the home agent remains untouched, which means security problems and solutions as discussed in section 2.2.3 are still valuable.

The separate list containing mobile router home addresses and associated prefixes, can either be configured by hand on every home agent, or on a central secured data base server. By using this list and the information of the binding process, a home agent is capable to establish and maintain a secure association of a mobile network prefix and a mobile router's current care-of address. The manual configuration of the separate list is no disadvantage or additional work, since it corresponds to the configuration of routing table entries on routers located within the home network.

While for a mobile host, it is possible to establish a secured direct route between the correspondent node and the mobile host in a foreign network by using the return routablility procedure, the procedure can not be modified to support route optimization for mobile networks.

The mobile host in the return routablility procedure can't simply be replaced by the mobile router, because the correspond node is unaware of the mobile routers existence. The correspondent node is not communicating directly with the mobile router (as it would do with a mobile host), but only with a local fixed node, located in the mobile router's fixed network.

So if a mobile router would send a binding update for a network prefix, the correspondent node could not verify whether the mobile router has the right to claim this network prefix or not.

For a mobile host, the correspondent node can verify the binding by sending the responses to the home test init and care-of test init messages (see figure 2.17). In the mobile network scenario, the same responses would only approve the mobile router's address, not the prefixes. The correspondent node has no possibility to verify the association of the home address to the prefix.

To use the home agent's list of associations is also not an option, since otherwise, a hacker might easily deploy pairs of home agent and mobile router to hijack connections to any desired subnetwork.

# Chapter 3

# Specification

## 3.1 IPv6 Router

The following subsections analyse the requirements on an IPv6 and Mobile IPv6 compatible router, to result in a standard compliant specification. Basically, the requirements can be separated into two different categories, requirements related to the neighbor discovery process (router discovery) and requirements with regard to IPv6 packet routing.

The implementation of the resulting router specification is described in the next chapter.

### 3.1.1 Router Advertisements

Most informations on router advertisements are listed in [RFC2461] and have been summarized in section 2.1. Additional requirements for Mobile IPv6 can be found in [MIPv6] (summarized in section 2.2). Some details will be added or modified, to provide more flexibility, even if they would permit to violate the standard specification( This does not mean, that they do this by default, but that violation can be achieved by setting the corresponding configuration options).

The authors of [RFC2461] describe a set of configurable options, a router should have. For this purpose they define variables, that help to provide the requested behavior. This document reuses the names of these variables, to simplify comparison to related literature. Most of the options must be configurable separately for each interface.

A router needs a variable (AdvSendAdvertisements, default: false) to indicate whether it does, or it does not, send unsolicited advertisements on an interface. It is reasonable that the default should be, not to send them, to prevent accidental emission, while not acting as a router.

Two parameters are used to specify the range for the time intervals between two unsolicited router advertisements.

| | |
|---|---|
| MaxRtrAdvInterval | The maximum time allowed between sending unsolicited multicast router advertisements from the interface. Default: 60000ms. |
| MinRtrAdvInterval | The minimum time allowed between sending unsolicited multicast router advertisements from the interface. Default: (0.33 * MaxRtrAdvInterval). |

In [MIPv6], the authors suggest to use shorter time intervals as the ones proposed in [RFC2460], when acting as an access router for mobile nodes. The reason is, as mentioned before, that sending router advertisements more frequently will allow mobile nodes to determine faster, that their point of attachment has changed. The new values and implemented parameter ranges are (the default values remain the same):

| | |
|---|---|
| MaxRtrAdvInterval | [7:180000]ms |
| MinRtrAdvInterval | [3:180000]ms |

After sending an unsolicited advertisements, the schedule time for the next advertisement is randomly chosen in between the specified range. In contrary to [RFC2461] and [MIPv6], the implementation must allow equal values for both parameters, to simplify testing. Solicited router advertisements have to be unicasted (if possible), and therefore, the timer for unsolicited advertisement must not be reset when responding to a router solicitation.

Another group of variables must contain the contents of the router advertisements on a specific link.

| | |
|---|---|
| CurrentHopLimit | The hop limit on the link. Default: 64. |
| Managed | Tells the hosts to use stateful address autoconfiguration in addition to stateless autoconfiguration. Default: false. |
| OtherConfig | When set, hosts use the stateful protocol for autoconfiguration of non-address information. Default: false. |
| RouterLifetime | Lifetime of a default router. Default (3*MaxRtrAdvInterval). |
| Reachable Timer | The time, a node assumes a neighbor to be reachable after a reachability confirmation. Default 0ms. |
| Retrans Timer | The time between retransmitted neighbor solicitations. Default 0ms. |
| MTU | The maximum transferable unit on the link. Default 0. |

The parameters to fill the Prefix Information options of a router advertisement can be described with the following set of parameters.

| | |
|---|---|
| Prefix | The prefix itself, or in case of a home agent a complete IPv6 address. Default: ff02::1. |
| Prefix Prefix Length | The prefix length. Default: 0. |
| Prefix On Link | When set, the prefix can be used for on-link determination. Default: true. |
| Prefix Autonomous | Indicate, whether a prefix can be used for autonomous address configuration. Default: true. |
| Prefix Valid Lifetime | The time, the prefix is considered to be valid for on-link determination. Default: 2592000s. |
| Prefix Prefered Lifetime | Indicates the time, that an address generated from this prefix advertisement, remains preferred. Default: 604800s. |

When becoming a router, the node has to join the all-routers multicast address on the interface in question, to be able to receive router solicitations. This implies not only the join on the IP layer, but also to join on a lower level, e.g. to join the all-routers Ethernet multicast address 33-33-00-00-00-02 on Ethernet links.

On incoming router solicitations, the router uses the values of the specified variables to send a response to the soliciting node. If possible, the response is unicasted directly to the link local address of the node, otherwise, the response is broadcasted. When unicasting the neighbor advertisement, the timer between used to separate two unsolicited advertisements is not reset.

Additionally, a router has to verify the consistency of router advertisements sent by neighboring routers, but it should never do any auto-configuration of IPv6 addresses based on these router advertisements. This is a decision out of security considerations, since otherwise, ill behaving nodes could announce wrong prefixes, which would cause strong perturbation of the link (because not only all host would be affected, but also the routers).

Before terminating to act as a router, the node should send one last unsolicited router advertisements to the link, with a router lifetime field at zero and then leave the corresponding all-routers multicast address. As a result of the last advertisements, other nodes should remove the router from their default router list.

If a home agent is meant to support the dynamic home agent discovery mechanism, it needs an additional set of parameters, that represent the values in the home agent information option. It also needs to support the addition made to router advertisement messages as well as the router bit in the prefix information option.

| | |
|---|---|
| HomeAgent | If set, indicates that this router serves as home agent. Default 0. |
| Prefix RouterAddress | If set, it indicates that the Prefix field contains a complete global IPv6 address of the home agent that sends this router advertisement. Default 0. |
| HA Preference | The preference of the home agent. Default 0. |
| HA Lifetime | The lifetime of the home agent. Default 0ms. |

Such a home agent must also be capable to maintain a list of other home agents on the same link to support the dynamic home agent discovery mechanism. The informations are gathered by examining router advertisements from other routers.

### 3.1.2 Packet Routing and Routing Table Management

To maintain and manage a routing table, the following functionalities are required

- create a new table
- delete a table
- add an entry
- delete an entry
- search for an entry
- lookup for a destination

The implementation of these functions depends on the selected lookup algorithm and the routing table structure.

The minimal routing table, as discussed in section 2.3.2, is extended to provide more compatibility with the Linux kernel. In addition to the already mentioned fields, the destination IP, the destination network prefix length, the gateway address and the destination device, the table contains three new fields. Their definitions are the same as in the normal Linux kernel IPv6 stack.

1. The flags field is used to supply additional information about the table entry. The following flags are defined
   - U (route is up)
   - H (target is a host)

- G (use gateway)
- R (reinstate route for dynamic routing)
- D (dynamically installed by daemon or redirect)
- M (modified from routing daemon or redirect)
- A (installed by addrconf)
- C (cache entry)
- ! (reject route)

2. The metric field is used to save the "distance"to the target (usually counted in hops). This field is needed by routing daemons that use directly the routing daemon to store a links metric.

3. The use field contains the count of lookups for the route.

Since a node must be configurable as host and as router, it requires a variable (IsRouter) to switch between the two modes. By default, this variable has to be set to host mode.

Packet reception and routing of a router must be done as described in section 2.3.1. Figure 3.1 shows a simplified scheme of this process (without layer two address checks). First the router verifies, whether the packet has to be processed as host or as router. In the router case, the next step consists in finding a next-hop in the routing table to forward the packet to. If no suitable next-hop is found, the packet must be dropped.

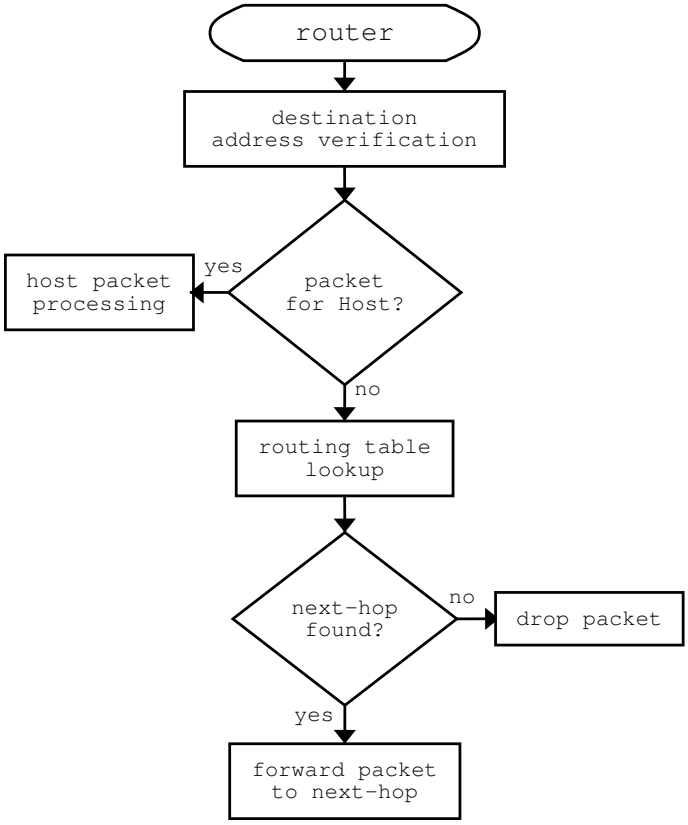



Figure 3.1: Router packet processing

## 3.2 Mobile Router

A mobile router combines the features of a mobile host and a normal router to support network mobility. Since the behavior of a mobile router that is connected to a foreign link differs widely from the behavior in its home network, it makes sense to specify the different behaviors in two separate subsections, while general requirements are discussed in this common section.

For movement detection and address autoconfiguration, a variable (MobileInterface) is required to identify a mobile interface. As already mentioned in 2.4.1.1, it is only the mobile interface, on which the mobile router is accepting autoconfiguration of care-of addresses based on prefix informations, received from router advertisements. The binding process is managed as it is in a mobile host.

Extended dynamic home agent discovery is not discussed, since, at the time being, not even dynamic home agent discovery for mobile host is implemented in the used IPv6 stack.

### 3.2.1 Mobile Router at Home

Connected to its home link, a mobile router has to act just like a vanilla router, with the exception that it must assure not to become default router for any host connected to this link. This means, if sending router advertisements on the home link, the default router lifetime field has to contain a zero value.

In case the mobile router just returned from being away from home, it detects being at home by recognizing its home prefix in the router advertisements. After updating its home agent and stopping bi-directional tunneling, the mobile router can restart sending router advertisements on the mobile interface.

### 3.2.2 Mobile Router in a Foreign Network

As soon as the mobile router discovers that it is connected to a foreign network, immediately it stops sending router advertisements on its mobile interface. Then a binding with a home agent in its home network is established. This procedure is the same as for mobile hosts. After this procedure, the bi-directional mobile router - home agent tunnel is ready.

When receiving a packet (assuming layer 2 address verification has already been done), the mobile router examines the destination IPv6 address of the packet at first, as it would do, if it were at home. If the IP does not match one of the mobile routers addresses, it has to try to route the packet. The routing process is illustrated in figure 3.2.

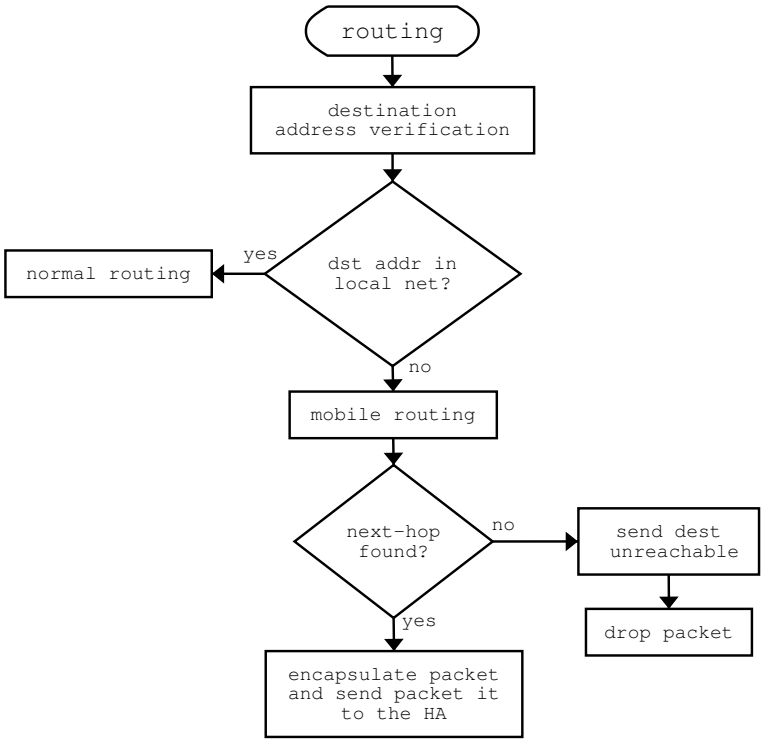



Figure 3.2: Mobile Router routing, while being away from home

Routing towards the local fixed networks is done the usual way (as while being at home). The lookup mechanism determines the next-hop IPv6 address, which is either an intermediate router/gateway, or the destination host itself. If necessary, the router starts the neighbor discovery process to receive the layer 2 address of the host and finally it delivers the packet.

Packets with a destination located in another network will be sent via the bi-directional mobile router - home agent tunnel. As for destinations in the local fixed networks, the router starts the lookup mechanism. But instead of finding a "normal"entry, the result will point to the tunnel. The mobile router has to encapsulate the packet and send it via the access router of the visited network to its home agent.

In case the packet destination IPv6 address is associated with the mobile router, this one must analyze the packets next header. If it is an encapsulated packet, the mobile router decapsulates it and then re-runs the reception procedure. Else the mobile router must treat the packet as being normal host (or router when responding to packets that had been unicasted to its address).

Packets from the mobile router itself are either send directly to the attached fixed networks, or in case of the destination is located outside of the local fixed networks, tunneled via the mobile router's home agent.

## 3.3 Mobile Router Home Agent

For nodes on its local link, except for the home agent(s), it must not make any difference, whether a mobile router is at or away from home. This requires that the home agent acts unnoticeable as proxy on behalf of the mobile router, while this one is away from home. The only difference is, that a home agent cannot respond to router solicitations, since it does not support the router's functions of the router discovery process.

The packet reception process (figure 3.3) is similar to the one of a normal home agent. The first step consists in looking at the packet's destination IPv6 address. If it is one of the home agent's addresses, and the next header value indicates that it is not an encapsulated packet, the home agent continues processing the packet as it would do a normal host. Otherwise, if it is an encapsulated packet, the home agent decapsulates the packet and delivers it to the new destination.

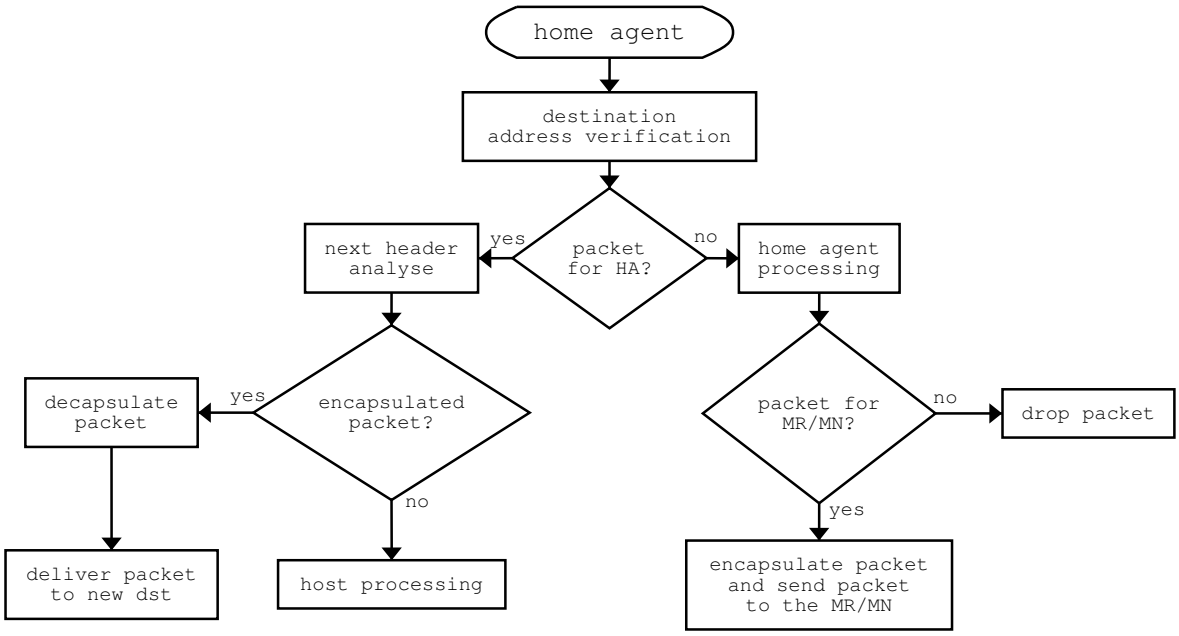



Figure 3.3: Mobile router home agent packet reception

In case of the destination is not the home agent itself, the second step is to verify, if the packet is either for one of the home agent's mobile hosts, or for a mobile network attached to one of the home agent's mobile routers. In this case, the packet is encapsulated and forwarded to the mobile host's or the mobile router's currently registered care-of address.

The support for mobile networks requires the implementation of a list of prefixes that are associated with a mobile router. Since the LIVSIX mobile networks implementation is based on a Mobile IPv6 implementation, the simplest solution is to have a list with prefixes

and mobile router home addresses. As stated before, this list does not have to be modified, when the mobile router changes its care-of address.

# Chapter 4

# Implementation

## 4.1 LIVSIX IPv6 Stack

This section describes the Motorola LIVSIX IPv6 stack, that is used for the implementation part and test part of this work.

LIVSIX is an IPv6 stack designed entirely from scratch, with special regard to high portability and support for mobility environments. Portability means portability across operating systems and C compilers, portability in terms of code size, and portability in the sense of minimal modifications to existing IPv6 applications and to TCP.

LIVSIX sources are available under the Motorola LIVSIX Public License and can be downloaded from the LIVSIX homepage, http://www.nal.motlabs.com/livsix/.

Today, LIVSIX is developed as a Linux kernel module and meant to be used predominantly in mobility environments. A port to FreeBSD is planned.

The LIVSIX architecture is separated into three abstraction layers (see figure 4.1)

- plt → platform dependent layer, kernel interface
- sfk → software framework and adaptation layer between plt and sfk
- csx → platform independent protocol implementations

Code modularity is enforced by considering the sfk part as a "thick interface"between csx and plt. All functions in csx or in plt can only call functions in sfk. The sfk is the only place where both, functions from plt and csx, can be called.

The csx layer holds the major part of the source code, e.g. the routing algorithm, the packet construction and the mobility management. The biggest part of the sfk-layer consists in

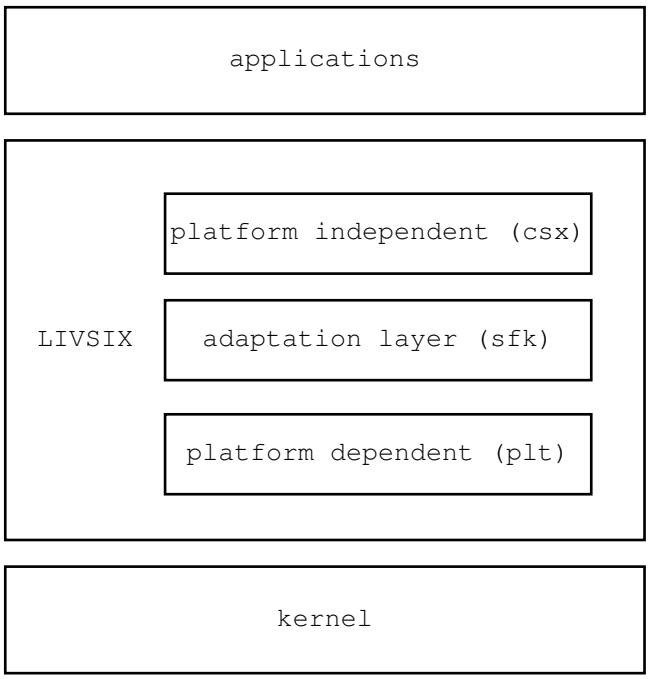



Figure 4.1: LIVSIX architecture

wrapper functions between the other layers and the plt layer is responsible for the signal and buffer exchange between the LIVSIX module and the kernel.

As a consequence, when porting to an other operating system, the csx layer remains untouched, the sfk layer requires minor changes and the plt layer needs full adaptation.

Due to the concentration on mobility environments, the first release of LIVSIX implemented only rudimentary host functionality, e.g. only partial support for UDP and no TCP. This enabled the usage of simple applications as ping6 or tftp. Subsequent, host mobility has been added as well as simple TCP support.

Current development introduces router functionality, improved TCP and UDP support, and provides a mobile router implementation for mobile networks.

## 4.2 IPv6 Router

Router functionalities for the LIVSIX IPv6 stack are implemented in separate modules, to avoid too much modifications of the existing code. The router advertisement procedure is embedded in the existing code for neighbor discovery, the routing table, its management and the packet routing itself are located in a separate part. This separation guaranties independent development and test of the modules.

According to the LIVSIX architecture, most of the code, that realizes the router implementation, is located in the platform independent layer. Only packet sending and receiving packets, as well as passing configuration options to the kernel module require platform dependent code and wrapper functions.

The existing function that handles the packet reception is extended. Before dropping the packet, if routing is enabled, LIVSIX tries now to route the packet. This results in the following packet processing procedure:

1. packet reception as home agent (if enabled)
2. packet reception as host
3. packet routing (if enabled)
4. drop packet

As specified, router functionality is only enabled if, the global IsRouter variable is set. Setting this variable causes LIVSIX to ignore router advertisements from other routers, as required for a router. The existing default router list is cleared and only routes, that have been add administratively to the routing table, remain valid. In other words, everything, that has been learned through router advertisements from other routers, is deleted. The router joins the all routers IPv6 multicast address (FF02::2) and the all routers Ethernet multicast address.

### 4.2.1 Router Advertisement Module

Both, the Linux kernel IPv6 stack, as well as the KAME reference stack, do not implement sending router advertisements directly in the kernel. They make use of external router advertisement daemons (rtadvd or radvd).

LIVSIX realizes an implementation in the stack itself. The decision not to use an external daemon, was taken out of several reasons. First, since sending router advertisements is a part of neighbor discovery, and most functionality of neighbor discovery has to be

implemented directly within the stack, it seems reasonable to implement the entire mechanism in the stack. Furthermore a stack implementation simplifies monitoring of router advertisements from other routers and the comparison to the own advertisements. And finally, the integration in the stack can be used to support special functionality for mobile routers (like blocking outgoing router advertisements on the mobile interface, while being connected to a foreign network).

The drawback of a kernel module implementation is the variety of possible configuration options that have to be passed to the module, to allow a RFC conform implementation, even though normally few of them are really used.

However, having a kernel implementation does not prevent administrators to use external router advertisement daemons.

To pass the configuration options to the kernel module, the Linux sysctl interface is used. Every option described in section 3.1.1 is represented by an entry (this means, options that apply to an interface are represented once per interface, options for a prefix once per prefix). Since dynamic home agent discovery is not supported by LIVSIX, the related options are not included.

Advertising a prefix is started as soon as the minimum set of options is configured. This set includes a prefix, the prefix length, the AdvSendAdvertisements variable on the interface and the IsRouter flag. All other values are initialized by the suggested defaults.

After setting the AdvSendAdvertisements variable to "true", a timer is scheduled with a randomized value, as defined in the specification in section 3.1.1. Upon its expiration, an unsolicited router advertisement is sent and the next timer is initialized.

Modified options are taken into account upon next timer expiration. Every interface is associated an individual timer, as soon as the AdvSendAdvertisements variable is set. Multiple prefixes on the same interface are aggregated to one advertisement. Currently, a maximum of ten prefixes per interface is defined and supported.

The prefix advertisement data is stored in an internal data structure. As mentioned above, data, that is not modified by the user, is initialized with predefined default values. The data structure is organized as a list, that contains one element for every interface. Within these elements, options for router advertisements are stored as well as a pointer to a list of prefixes and prefix related options that are advertised on the interface.

### 4.2.2 Routing Module

#### 4.2.2.1 Routing Table Structure

LIVSIX is not meant to be used in a backbone network, but to be used in access networks. That means, future routing tables will hold approximately five entries, max. twenty entries.

As a consequence, a simple table structure is sufficient.

This leads towards the simplest implementation, a double chained unsorted list. Research in this table is done, by always examining the whole list to find the optimal route for a given destination. Every list element contains the fields specified in section 3.1.2.

Nevertheless, to be able to switch to a different structure and more performant lookup algorithm, the table structure as well as the related functions are implemented well separated from other code, to be easily exchangeable.

#### 4.2.2.2 Routing Table Management

The routing table is internally managed by the functions listed in section 3.1.2. Access from outside of the stack is provided by two interfaces, a sysctl interface and a clone of the Linux IPv6 stack ioctl interface. The ioctl interface was implemented to enable compatibility to existing applications, such as the "route"command or routing daemons, while the sysctl interface provides the advantage to be usable on every operating system, that supports a /proc filesystem. Both interfaces use the internal functions for modifications of the routing table. The current routing table is permanently accessible in "/proc/net/livsix_route"or "/proc/net/ipv6_route", the latter is implemented for Linux IPv6 stack compatibility. Since the routing table fields "flags", "metric"and "use"are not used or modified by the stack itself, the ioclt interface is the only way to modify them.

The most important support function is the function to find a specific route in the routing table. Routes are identified by the destination prefix or address and its corresponding prefix length. Having two or more routing table entries with the same combination of both of these values is not allowed. The function browses the routing table for a given combination and returns a pointer to the entry, in case of it exists, or otherwise a zero pointer when reaching the end of the table.

To add a new entry to the routing table, the responsible procedure requires at least the destination prefix, the prefix length, the gateway address and the device as parameter. Other fields of the table, that are omitted, are initialized by default values. The first task is to verify, that no other entry with the same prefix and prefix length exists, by using the above research function. If there is already such an entry, adding the new one is refused. If nothing is found, memory for a new table entry is reserved, filled with the data and linked to the existing routing table.

Deleting a route, requires only the two parameters that are sufficient to identify and find the route, the destination prefix and the prefix length. The corresponding table entry is removed from the list and the associated memory is freed.

The function that is used to create a new table initializes an empty table with a newly created entry.

Since all routing table modifications are done by using these four functions, switching to another table structure is possible by changing the contents of the functions (plus adapting the lookup function), while keeping the prototypes. No additional changes to the stack are required.

#### 4.2.2.3 Packet Routing

Destination research in the routing table is done by a separate lookup function. The function requires a destination address as input parameter and returns the next hop address and the outgoing interface. The longest matching prefix is searched examining all routing table entries and comparing bitwise the destination address to the prefix. If the address matches the prefix, the table entry is kept as a possible destination, as long as another entry with a larger prefix length matches. At the end, the function returns the data from the kept entry and thus the longest matching prefix.

For packet delivery, the routing procedure calls the lookup function to get information on how to route the packet. In case of the lookup returns a suitable result, a lower level packet delivery function is called, with the parameters outgoing interface, next-hop and packet. Otherwise, the packet is discarded.

So far, neither the destination unreachable message nor a redirect message are implemented. Packets that could not be routed are dropped silently and packets with a better next hop on the same local link are routed without sending a redirect message.

## 4.3 Mobile Router

As mentioned in the last chapter, when acting as router, the LIVSIX implementation ignores other router's router advertisements by default. To enable network mobility, the MobileInterface variable is introduced, that is controlled through the Linux sysctl interface. By setting this variable, an administrator can define one interface, on which the router accepts router advertisements, detects movements and autoconfigures interface addresses. For mobility management, the existing Mobile IPv6 implementation is reused.

### 4.3.1 Mobile Router at Home

Since, a mobile router behaves as a normal router while it is connected to its home network, this part does not require any additional implementation.

### 4.3.2 Mobile Router in a Foreign Network

#### 4.3.2.1 Router Advertisements on the Mobile Interface

As soon as a movement to a foreign network is detected, the Mobile IPv6 implementation manages prefix autoconfiguration and the binding process with a home agent. Binding updates with correspondent nodes are disabled.

Immediately after the movement is detected, an internal variable is set, that indicates, that the mobile router is not attached to its home link. As a consequence, the router advertisement module does not send any router advertisements on the mobile interface any more.

#### 4.3.2.2 Routing Table Adaptation

The only problem that remains are existing entries in the routing table. While at home, all entries are valid, and the default route generally points to a neighbor router in the home network, in a foreign network, all packets send via routes towards the mobile routers home network must be tunneled through the mobile router - home agent bi-directional tunnel. In other words, all routes that go by the mobile interface are invalid, and can not be used by the normal packet routing module. On the other hand, as soon as the mobile router returns to its home network, the routes become valid again.

To solve this problem, a second routing table, named shadow routing table, is implemented, that holds all routing information that has the mobile interface as destination interface, while the mobile router is connected to a foreign network. As soon as movement from

the home network to any foreign network is detected, a function is invoked, to move all the routing table entries, that contain the mobile interface, to the shadow routing table. Vice versa, as soon as a movement from any foreign network to the mobile router's home network is detected, all entries from the shadow routing table are moved back to the normal routing table.

The word "shadow"in the name of the second routing table is significant for two properties of this table. The first one is of course, that the entries are hidden from the normal routing algorithm. The second one is, that the separation into two routing tables is transparent for normal applications, that modify the routing table. Normal applications means applications, that do not support network mobility, e.g. the standard Linux route command (which uses the Linux kernel ioctl interface). Modifications that are caused by such an application are automatically executed on the right table.

To achieve the transparent behavior, compared to the normal router, the routing table output is modified.

- "/proc/net/livsix_route"represents the normal routing table,
- "/proc/net/livsix_route_shadow"contains the entries of the shadow routing table
- and "/proc/net/ipv6_route"holds the entries from both routing tables.

Applications, that do not have particular support for LIVSIX network mobility, only use the third table.

#### 4.3.2.3 Packet Routing and Packet Reception

The usage of the shadow routing table assures that the routing function works unmodified at home, and only slightly modified in a foreign network. In a foreign network, first the lookup function on the normal routing table is executed (that contains all entries towards local fixed networks) and only in case this table does not contain a suitable route, the lookup function is called a second time, but now on the shadow routing table. If the second lookup is successful, the packets have to be tunneled through the mobile router - home agent bi-directional tunnel. The packet tunneling function of the Mobile IPv6 module is invoked and the packet, that has to be routed, is sent through the bi-directional tunnel, via the access router in the visited network, to the home agent.

Packet reception on the mobile router is implemented as a combination of what is done by a router and a mobile host. All packets are processed by the generic packet reception function. For packets that have to be routed, the enhanced routing function, as described above, is executed. Packets that are received through the mobile router - home agent tunnel are unpacked by the mobile host packet decapsulation function, which recalls the packet reception function after decapsulation.

## 4.4 Mobile Router Home Agent

Since the functionality of a Mobile IPv6 home agent is already available, the main task during the implementation of the mobile router supporting home agent, is the realization of the list of network prefixes, that can be reached via the mobile router. For this purpose, the standard routing table is used. For every prefix in a fixed network of a mobile router, the routing table contains one entry with the following fields.

- Destination network address = prefix of the mobile network.
- Destination network prefix length = prefix length of the mobile network.
- Gateway address = home address of the corresponding mobile router.

Upon reception of a packet, the Mobile IPv6 home agent implementation compares the packet destination address to the entries in its binding cache. In case of a matching entry, the packet is tunneled to the current care-of address of the mobile node. To take advantage of this implementation, the comparison of the destination IP address to the network prefixes that are reachable via a mobile router, has to be done before. That's why before searching the binding cache, the routing table lookup function is called with the destination IP address as parameter. If the result of the lookup returns a gateway (possibly the IP address of a mobile router), the binding cache is searched for this gateway, otherwise the home agent tries to find a binding cache entry directly for the destination IP address (as a normal home agent does). In both cases, if a care-of address is found, the packet is tunneled to this care-of address.

Encapsulated packets with a destination IP address that belongs to the home agent are processed the same manner as they are processed by a standard home agent. After decapsulation the home agent delivers them either directly if the destination node is on the same link, or via the home agent's default router.

The above mentioned modifications and additions enable the support of mobile networks. The binding process remains entirely untouched.

## 4.5 Dynamic Routing with Zebra

GNU Zebra (http://www.zebra.org) is a routing software suite that provides IP (IPv4 and IPv6) based routing services with multiple routing protocol support. For IPv6, implementations of RIPng, OSPFv3 and BGP-4+ are available. Adapting the interfaces of Zebra and LIVSIX, to work together, provides the possibility to use dynamic routing protocols on LIVSIX routers.

The advantage of the Zebra suite is its unified architecture (4.2). All communication with the IP stack is done via the central zebra daemon, which means, once it is adapted to the stack (or the stack to the zebra daemon), one is able to use every routing protocol supplied by the Zebra package.

The above mentioned communication, between the stack and the daemon, includes adding and deleting routes from the kernel routing table, adding and deleting IP addresses from/to an interface, and changing interface flags. Routing protocol specific data exchanges are done directly between the corresponding routing daemons.

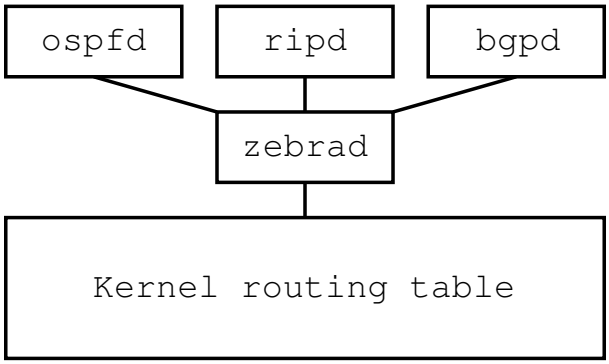



Figure 4.2: GNU Zebra system architecture

Zebra offers a set of interesting features:

- It supplies a management console for every daemon (including the zebra daemon itself). By this means, it is possible to modify settings and routing tables at distance even without having root access on the machine in question (a password for zebra is required).
- Zebra maintains its own routing table, apart from the kernel routing table.
- Each routing daemon uses its data base to maintain routing related information.
- It is possible to run more than one (different) routing daemon on the same router at the same time.

- The daemons are able do exchange and redistribute routing information.

A detailed documentation of Zebra can be found on the following homepage: http://www.ipv6peer.net/zebra/.

### 4.5.1 Zebra RIPng

The RIPng daemon, that is part of the Zebra routing suite is compliant to RIPng as specified in [RFC2080]. RIPng is the RIP (Routing Information Protocol) implementation for IPv6, a distance vector based routing protocol. It is intended to be used in small and medium size networks.

RIPng is an UDP based protocol. Messages are normally send from one of the hosts link local addresses, to either another link local address or to the RIPng all-rip-routers multicast group address FF02::9. The Hop Count field is set to 255, which guarantees, that the messages rest restricted to local links (the same principle is used for the neighbor discovery process).

RIPng specifies two message types, Request and Response. A Request message is send after a node became router, to be able to fill its routing table quickly. A Response packet is used to answer to a Request message, to send periodically unsolicited route updates, or to announce a route update. Every 30 seconds a router sends Response messages containing all its routing table entries to its neighboring routers. A detailed message format description is given in [RFC2080]. The Response messages contain one or more entries of the type prefix, prefix length and metric.

The distance or the cost of a route is expressed by the metric. It is specified in the range from 1 to 15, 16 means infinite or unreachable. In the most simple scenario, local destination networks have a metric of 1, a destination network which can be reached through N routers has a metric of $MIN(N + 1, 16)$. In more complex scenarios the link cost (the value to be added to the metric on a specific link) can vary.

The expiration of routes is based on two time intervals. The timeout time and the garbage-collection time. If there is no Response message for a certain prefix within the timeout interval, the route towards this prefix will be marked as invalid and the deletion process is started. The route is still advertised to the neighboring routers, but with a metric of 16. With the start of the deletion process, the garbage-collection timer is triggered. On its expiration the route is finally deleted from the routing table and the Response messages.

The deletion process is stopped and the timeout timer is reset, if, within the deletion process, an update message for the concerning prefix is received.

The second reason for a route deletion is the reception of an advertisement of with a metric of 16.

Upon reception of a Response message, the router verifies the validity for each routing entry. Then it calculates the metric for the local routing table entry ($metric = MIN(metric + linkcost, infinity)$. The next step is to check the local routing table, whether there is already a route to this prefix. If not, the new route is added with the calculated metric, unless the metric is infinite. Otherwise, if there is already an entry, update it or leave it untouched, according to the rules mentioned in [RFC2080]. For example, a route through a new router with a lower metric value would cause an update, whereas the same message with a higher metric would leave it untouched. When having the same metric, the behavior depends on the implementation.

Additional Control functions as well as the usage of a split horizon are described in [RFC2080]

### 4.5.2 RIPng for Mobile Routers and Mobile Networks

The advantages of RIPng are, that it is small and simple, which is extremely useful in a mobility scenario. The drawback are regular broadcasts. Since Mobile networks are in general small and not too complex, the restrictions of RIPng in terms of size an complexity are no problem.

Network mobility introduces an interesting aspect. While the mobile router is attached to its home link, it participates in the exchange of RIPng messages, like every other router does. But as soon as the mobile router is connected to a foreign network, there is a problem. Regular RIPng messages on the home link, send by the mobile router, are required, to make the other routers in the home network maintain their routes towards the mobile networks, attached to the mobile router.

Basically, there are two possible approaches as solution. Either the home agent starts to turn RIPng on behalf of the mobile router, or the RIPng packages are tunneled through the mobile router - home agent tunnel.

The first solution would have the advantage, that the traffic through the tunnel might be minimized, in terms of only sending update messages in case of a topology change in either the home network or in the mobile network (the mobile network itself might contain several routers turning RIPng). But the disadvantage is the necessary synchronization of the RIPng daemons turning on the mobile router and on the home agent. Additionally, in order to emulate the mobile router on the home link, the home agent has to send its RIPng messages with the link local address of the mobile router.

The second solution is much more convenient, especially with regards to the LIVSIX design philosophy. RIPng is tunneled. The home agent joins the RIPng routers multicast address and forwards RIPng packets to the mobile router. Packets from the mobile router towards the home network are delivered as every other packet from a node withing the mobile network. Mobility remains transparent for Zebra and the RIPng daemons. Only the home

agent has to be modified to forward the RIPng messages. The downside of this approach is the higher traffic through the tunnel, caused by the periodically distributed RIPng messages.

The usage of OSPFv6 should also be possible by using the second approach. Only the home agent would have to forward packets send to the ospf6d router multicast address.

### 4.5.3 Zebra for LIVSIX

According to the LIVSIX philosophy, to provide as much compatibility as possible, most adaptations, that are required to be able to use Zebra on a LIVSIX router, are done done in the stack. Only one parameter and few procedures are changed in Zebra itself, and even those modifications should disappear with the evolution of LIVSIX.

The zebra routing suite is used in version 0.93b. The following changes apply to this version.

In the source code of zebrad, it is only necessary to change the path to the /proc file system entry, that enables router functionality (packet forwarding) on all interfaces. The Linux IPv6 stack uses "/proc/sys/net/ipv6/conf/all/forwarding", LIVSIX uses "/proc/sys/net/livsix/isrouter". This modification can be easily implemented in future versions of zebra, that do have explicit support for LIVSIX.

Ripngd has to be patched, since for sending the RIPng messages, a not supported kernel function is used. As soon as this feature is implemented in LIVSIX, no more modifications are required.

Ospf6d requires this kernel function, so at the moment, it is not usable with LIVSIX.

Zebra has to be compiled with the following commands:

```
./configure --disable-netlink --enable-ipv6 --disable-ospfd
make
su root
make install
```

This compiles the zebra routing suite with support for IPv6, without support for IPv4 ospfd (errors during compilation) and with support for the Linux ioctl kernel interface (the netlink interface is not supported by LIVSIX).

# Chapter 5

# Tests Scenarios

## 5.1 Router Tests

The tests, that are described in the following subsections, followed subsequently on the implementation of the corresponding router functions.

### 5.1.1 Router Advertisement Test

This test is used to verify, that the implementation of the router discovery mechanism and the resulting router advertisement messages are conform to the specified standards in [RFC2461] and [MIPv6].

Figure 5.1 shows the test setup. Two machines are connected via a 100Mbit hub. To assure optimal compatibility, the tests are run twice, the first time with a host that uses the Linux IPv6 stack, and the second time with a host running on FreeBSD v4.6 using the KAME IPv6 stack.

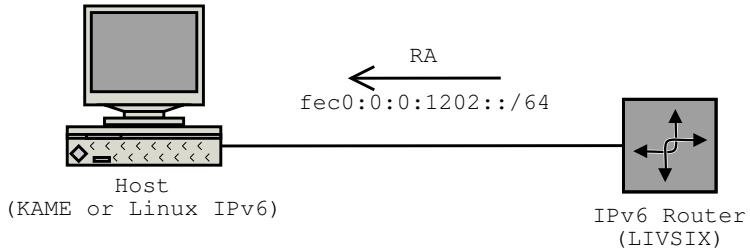



Figure 5.1: Router Advertisement test

The LIVSIX machine is configured with the following commands:

```
echo fec0:0:0:1202::  > \
   /proc/sys/net/livsix/conf/eth0/ra_prefix_00/ra_pfl_prefix
echo 64 > \
   /proc/sys/net/livsix/conf/eth0/ra_prefix_00/ra_pfl_prefixlen
echo 1 > /proc/sys/net/livsix/conf/eth0/ra_send_ras
echo 1 > /proc/sys/net/livsix/isrouter
```

The packet flow scheme in figure 5.2 visualizes the packet flow in in the case, where the host is up before the LIVSIX machine starts sending unsolicited router advertisements. The host does address autoconfiguration from the prefixes in these advertisements.

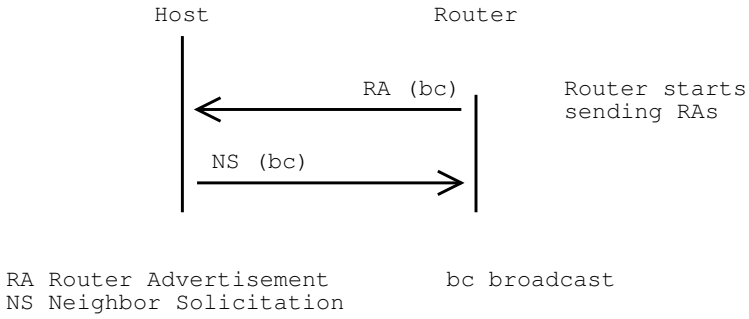



Figure 5.2: Unsolicited Router Advertisement test

In the second scheme (figure 5.3) the LIVSIX router already started to send router advertisements. When the host gets up, it broadcast a router solicitation to receive an router advertisement from a router on the same link. The LIVSIX router responds with a unicasted router advertisement.

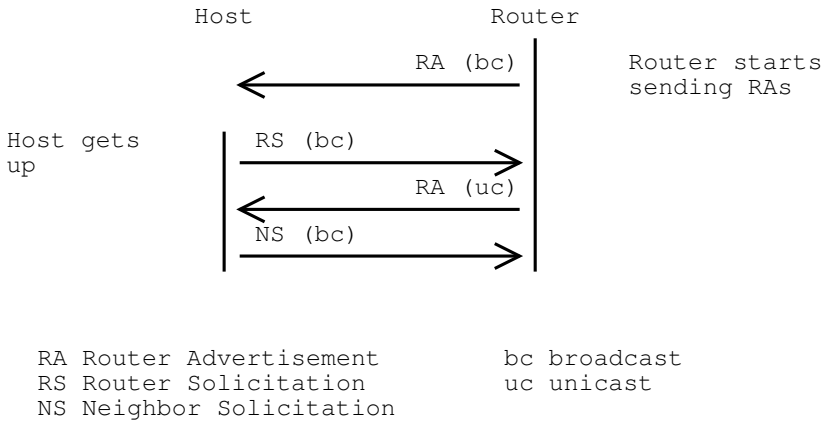



Figure 5.3: Solicited Router Advertisement test

The tests demonstrate, address autoconfiguration from LIVSIX router advertisements works. The packet format analysis with ethereal (for more information on ethereal, see C.1) shows, that the packets are compliant to the standards. Modifications the router advertisement parameters are possible and result in the desired changes of the router advertisement messages.

### 5.1.2 Simple Routing Tests with multiple Hosts

The prerequisites for the following tests are, that routing table, routing table management, destination lookup and packet delivery function are implemented. The tests are used to verify simple router functionality: add a routing table entry, receive a packet, do a destination lookup and deliver the packet directly to the destination host.

During the tests, the hosts are connected to links directly attached to the router (packets will always be delivered directly to the node indicated by the packet header's destination IP address, no intermediate routers are present). The hosts use address autoconfiguration from router advertisements sent by the LIVSIX router.

The functionality of the routing table management is verified by adding and deleting several routes on the LIVSIX router. For this purpose, both, the sysctl and the ioctl interface are used.

The first routing test (see figure 5.4) uses a LIVSIX router with two network interfaces, interconnecting two separate networks. Within each network, there is one host, using the Linux IPv6 stack.

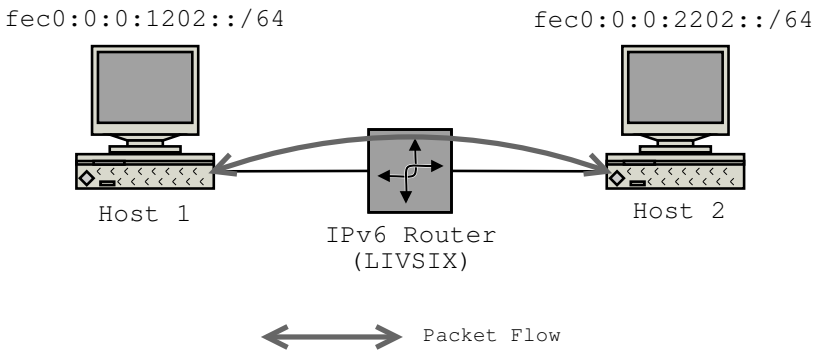



Figure 5.4: Simple routing test

The second routing test extends the first, by adding another link, attached to the third interface of the router (figure 5.5).

The stability of the router implementation is tested by running different applications on the hosts. Ping (12h flood-ping) and VIC, that use the UDP protocol, Ssh (remote connection to another host) and scp (file transfer) as applications using TCP.

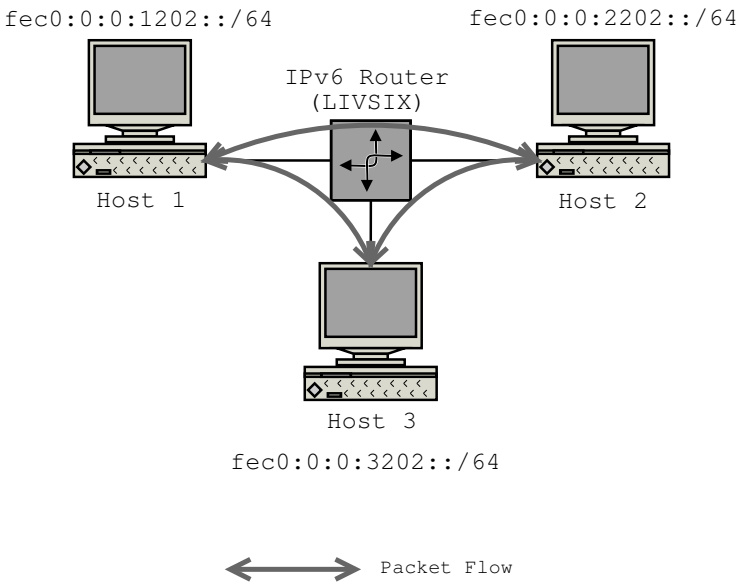



Figure 5.5: LIVSIX Router connected to three different subnets

The tests prove, that the routing table management and the destination lookup work as expected.

### 5.1.3 Two Host on the same Link

This test serves to verify, that the following problem does not appear: A router, that is connected to a shared link, duplicates all packets on the link.

This misbehavior happens, if the routers interface is set to promiscuous mode (e.g. by an application such as ethereal) and the router does no verification of the destination layer two (mac-) address.

Figure 5.6 shows the test setup. If running ethereal in promiscuous mode on the router without layer 2 destination address verification, one single ICMP echo request sent from Host 1 to Host 2 results in four echo replies. Figure 5.7 illustrates this. With layer 2 destination address verification no packet duplication occurs and the router shows the desired behavior.

### 5.1.4 Routing Test with multiple Routers

The difference compared to the previous tests is, that this time, the destination for the packets is not on the same link, but reachable via a second router (or even a network of routers). Figure 5.8 shows the testbed with two LIVSIX routers and two Linux hosts.

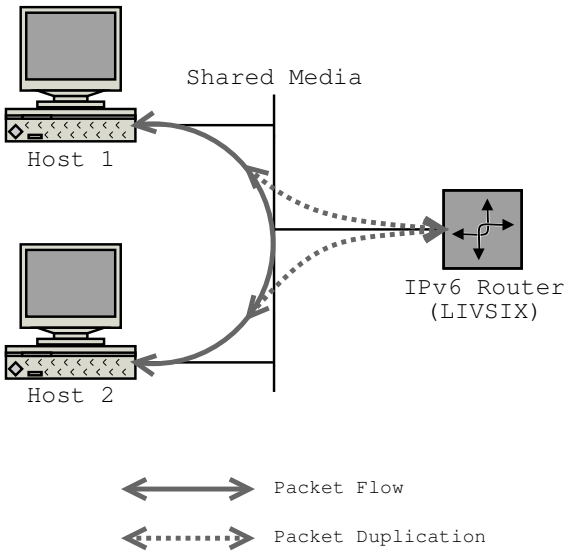



Figure 5.6: LIVSIX router packet duplication test

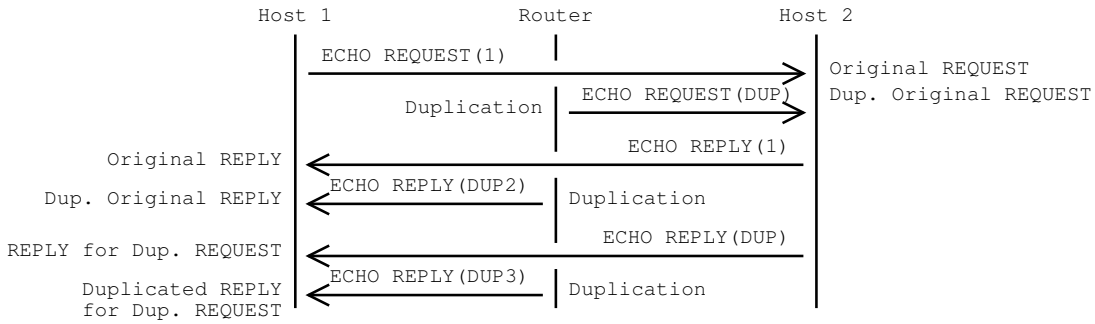



Figure 5.7: Packet duplication: packet flow

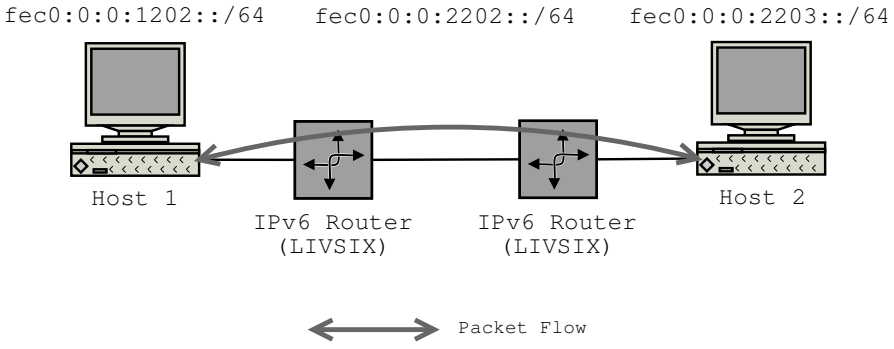



Figure 5.8: Routing test with two routers

During this test, the routing table lookup functions returns an outgoing interface and a next hop, instead of only the interface, as in the previous test cases. The packet delivery function must now forward the packet to the next hop's layer 2 address, instead of to the layer 2 address belonging to the packet's destination IP address.

As before, the applications, that are used on the hosts are ping (flood-ping), VIC, ssh and scp. Then, the second router is replaced by an entire network and on the Linux node, the Mozilla web browser is used to browse the KAME project web-site. Finally the latest KAME snapshot is downloaded from the site.

### 5.1.5 Simulation of the NAL Testbed Core Network

This final router test is a combination of all previous tests. The aim is to build a triangular core network with attached leaf networks (similar to the NAL testbed). The resulting testbed is shown in figure 5.9.

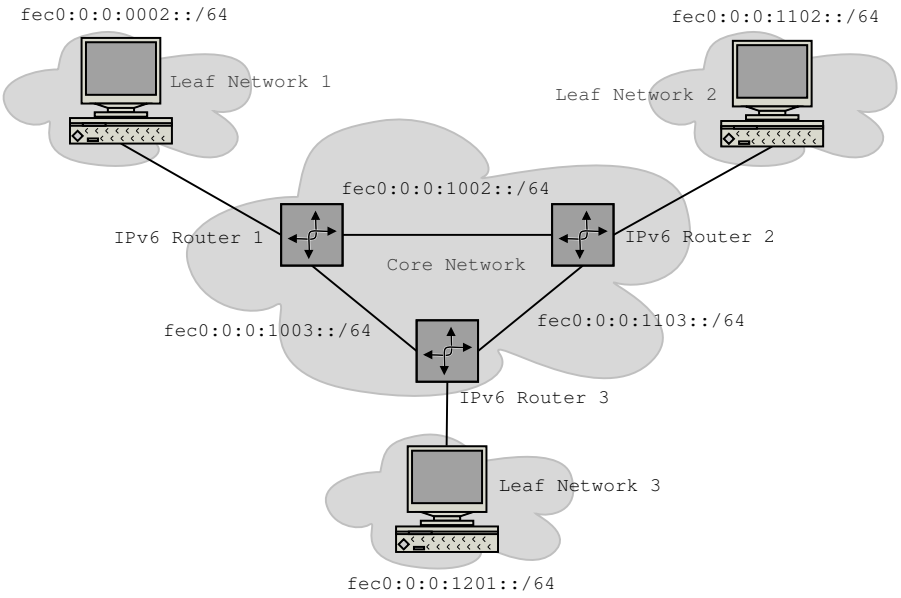



Figure 5.9: NAL testbed core network clone with LIVSIX routers

In a first series of tests, all routers are configured with static routes, since support for dynamic routing protocols is not yet implemented. The router are configured to send router advertisements on all links.

Randomly attached nodes on all sub-networks (also "in"the core) autoconfigure addresses and are able to communicate with nodes on other sub-nets.

With the support for dynamic routing protocols (namely zebra with RIPng), the testbed is reused to examine the router's routing table reconfiguration in case of link failures.

At the beginning, all router's routing tables contain only routes to networks directly attached the respective router. The routing daemons on every router obtain the same information through their configuration files. As expected, the routing daemons then exchange routing protocol messages and adapt their router's routing tables. This results in approximately the same routing tables as in the static routing case.

Cutting the link between two core routers, to simulate link failure, results in an automatic reconfiguration of the core. After a short period of time (depending on the RIPng configuration), all networks, except the one with the cut link, are reachable again from within the whole testbed.

Upon the re-establishment of the link, the routes are only adapted in case the new path is "shorter"than the current one.

Multiple tests in this testbed with different applications show, that the LIVSIX router implementation is usable and stable.

## 5.2 Mobile Network Tests

This section summarizes the different tests, that have been made to verify the LIVSIX realization of mobile networks. While the first test assures, that basic mobile networking works, the ensuing tests are performed to examine the interaction between vanilla Mobile IPv6 and mobile networks, as well as between different mobile networks.

### 5.2.1 Basic Mobile Network

#### 5.2.1.1 Usability Test

Once the home agent is extended to support mobile networks, and the mobile router combines the router implementation with Mobile IPv6, the first test is obviously the one, that is illustrated in the figures 2.21 and 2.22 in chapter 2, section 2.4.

The scenario consist of the mobile network with the mobile router (MR) and the local fixed node (LFN), the home network with the home agent (HA) and the border router (BR), an access router (AR) and a correspondent node (CN). Access router, border router and correspondent node are interconnected via a random IPv6 network. The MR is configured to send router advertisements to the mobile network and to accept router advertisements on its Mobile Interface.

The CN pings continuously the LFN. The mobile network disconnects from its home link and reconnects on the AR's subnet. After the reception of a router advertisement and the completion of the binding update process between MR and HA, the CN restarts to receive echo replies for its echo requests.

The packet flow is recorded on the home agent and the mobile router, using ethereal. The examination of the log files shows, that the packet flow is similar to the one depicted in figure 2.23.

To complete this test, the mobile network moves back to its home link. After movement detection and the exchange of the binding messages, everything is as it was before the first movement.

#### 5.2.1.2 Advanced Tests

Following the first simple test, more advanced tests are made. These tests introduce other applications on the LFN and more network movements. This means, the mobile network does not move from foreign network one back to its home network, but to foreign network two, three, etc., back home, to network X, etc.. (see figure 5.10).

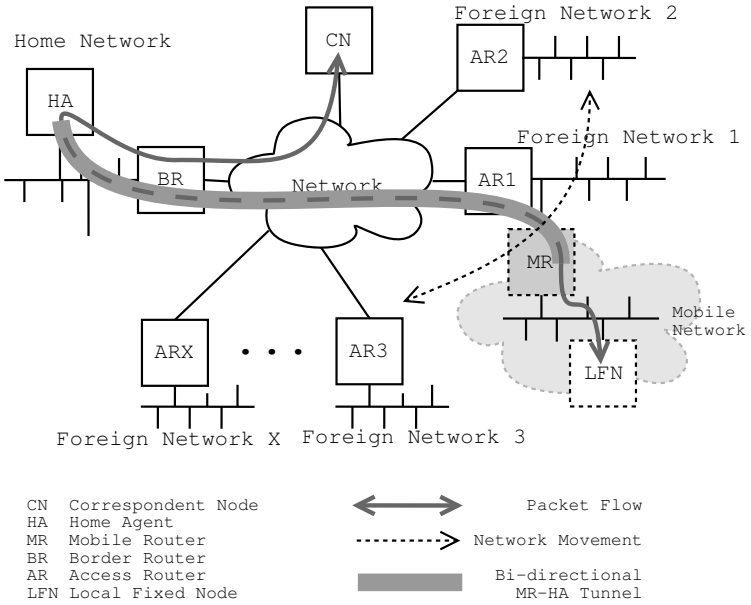



Figure 5.10: Mobile Network movements

The mobile router is now also configured to send router advertisements on its Mobile Interface. On the home network, these advertisements are sent, but as soon as the mobile router detects that it has left the home link, it stops sending router advertisements on this interface. The MR restarts, when returning to its home link. The router advertisements on the interface towards the mobile network are not influenced by this behavior.

On the other hand, the local fixed node now uses different applications to establish connections to different host outside of the mobile network.

The tests prove, that network mobility can be nearly transparent for the LFN. Only a link failure for a short period of time, resulting in packet loss, during the movement and the binding update procedure is visible.

As expected the handover time strongly depends on the frequency of the router advertisements on the visited links (and of course, the home link). While on links with approximately second-by-second RAs, the handover is nearly seamless, on other links with larger time intervals, the handover can result in a total de-synchronization of communication.

The network throughput towards and from the mobile network depends on the used home agent and mobile router, as well as on the connection between HA and MR. With a 100Mbit connection between HA and MR (MR connected to a foreign network) and a LFN communication with a node somewhere else on the Internet, no additional delays occur.

The influence on different applications can be characterized as follows. Ping shows X echo

requests that are not answered during handover, then the replies are received as before the movement. Video Conferences with VIC show strong interruptions during handover. This is a result from the mpeg compression. TCP transfers, e.g. http file transfers are interrupted during handover, but after the successful handover, the transfer restarts immediately.

### 5.2.2 Mobile Host and Mobile Network

These tests serve to study the the interaction between Mobile IPv6 and extended MobileIPv6 for mobile networks. For this purpose, testbeds similar to the one depicted in figure 5.11 are used. Ping is used as test application.

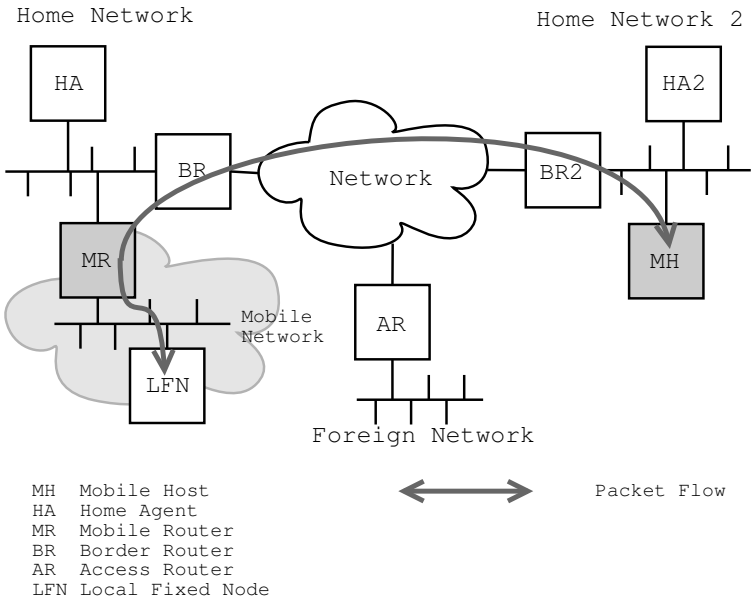



Figure 5.11: Mobile Host and Mobile Network testbed

The test setup is varied, the communication between LFN and Mobile Host remains. For example, in one setup, MR and the mobile host are on the same home link and share the home agent, in another one Home Agent 2 and Mobile Host are part of the mobile network.

Both, the mobile host and the mobile network disconnect and reconnect on different attachment points. E.g., the mobile host disconnects, reconnects in the mobile network and then, the entire mobile network, including the mobile host, moves.

During all tests, no interference between network and host mobility are encountered. Even in scenarios with double tunneling (Mobile Host connected to the mobile network, the mobile network connected to any foreign network).

### 5.2.3 Two Mobile Networks

In the next step, the mobile node used in the previous test, is replaced by a second mobile network (see 5.12). Packets are exchanged between the local fixed nodes.

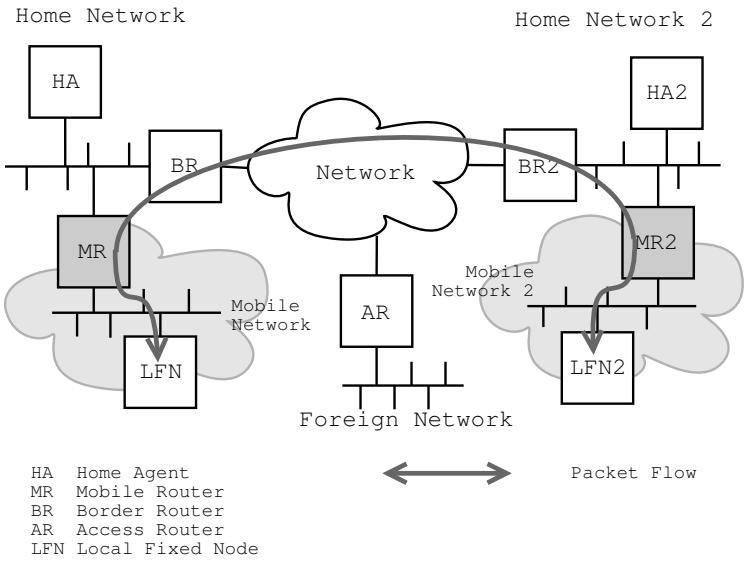



Figure 5.12: Two Mobile Networks Testbed

As during the tests with the mobile host, the test setup is modified, to be able to cover as much as possible different scenarios. The setup, in which the second mobile network is a sub-net of the first one, is discussed in the next section, since it introduces a special problem of nested mobility. Though being part of the same home network and sharing the same home agent is possible.

Again as much as possible network movements are simulated. For example, Mobile Network 1 leaves its home link, attaches to Mobile Network 2, and then Mobile Network 2 (including Mobile Network 1) moves.

The setup with a shared home agent and the movement given in the example above (shown in figures 5.13 to 5.14), introduces an interesting effect.

In the network state depicted in figure 5.14, apart from the double bi-directional tunnel, a ping-pong effect (not shown in the figure) between the home agent and the border router occurs, during communication between LFN and LFN2 (see figure 5.16). This is the result of the double encapsulation on the home agent. After the first encapsulation, the HA tries directly to deliver the packet, without re-regarding its binding cache.

The overall result from these tests: Ssh connections between the two LFNs and pinging the one LFN from the other is always possible. But in the discussed setup, the double

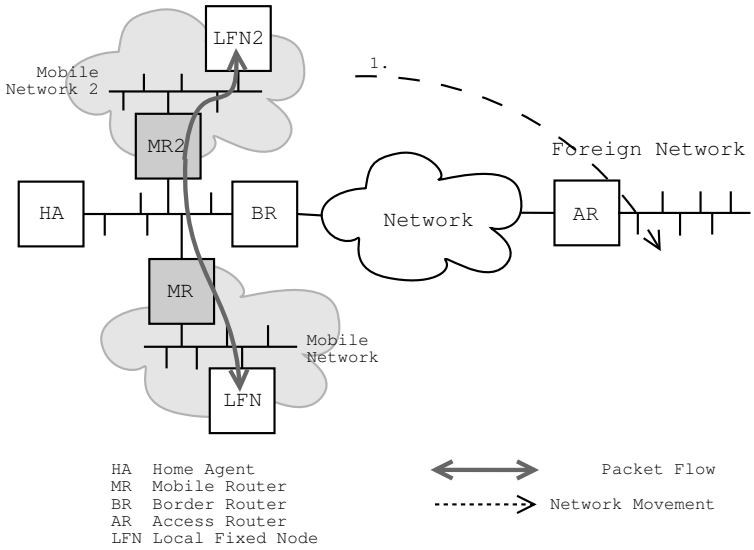



Figure 5.13: Two Mobile Networks, one Home Agent (1/3)

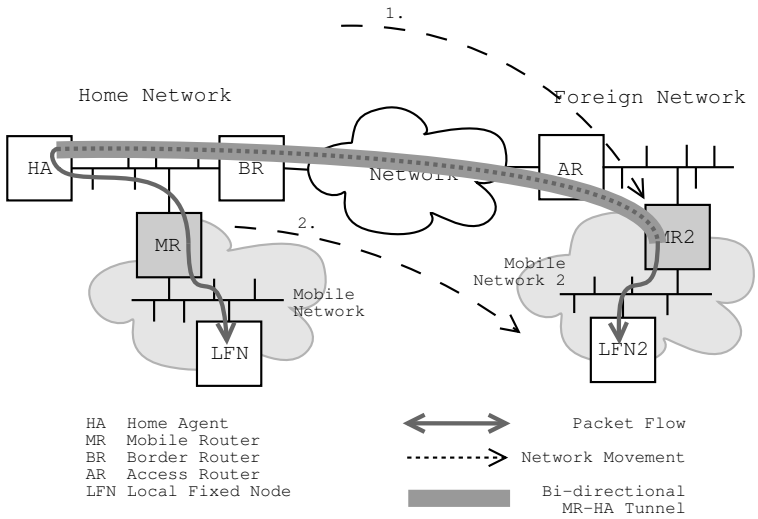



Figure 5.14: Two Mobile Networks, one Home Agent (2/3)

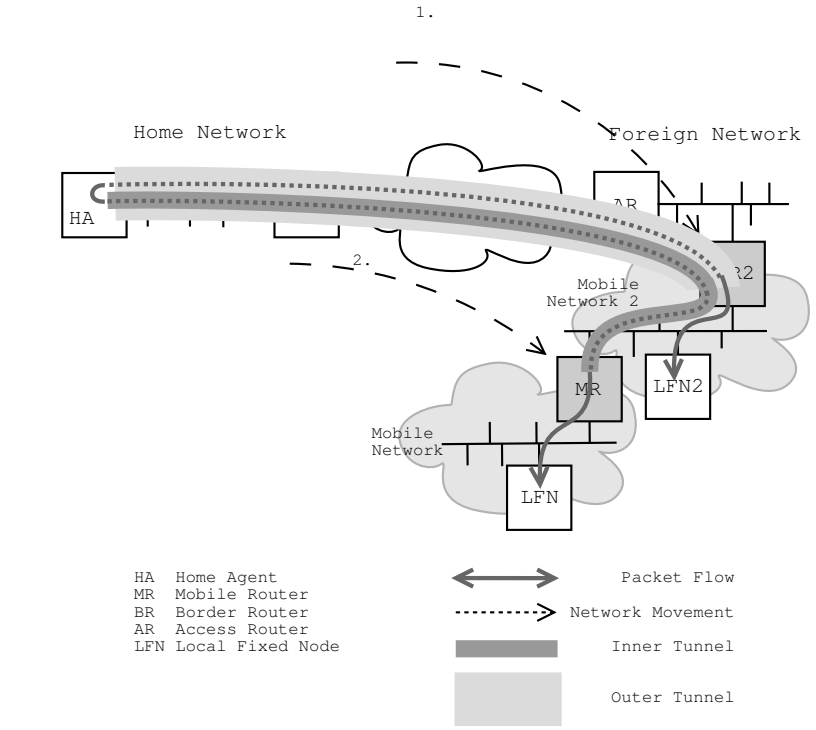



Figure 5.15: Two Mobile Networks, one Home Agent (3/3)

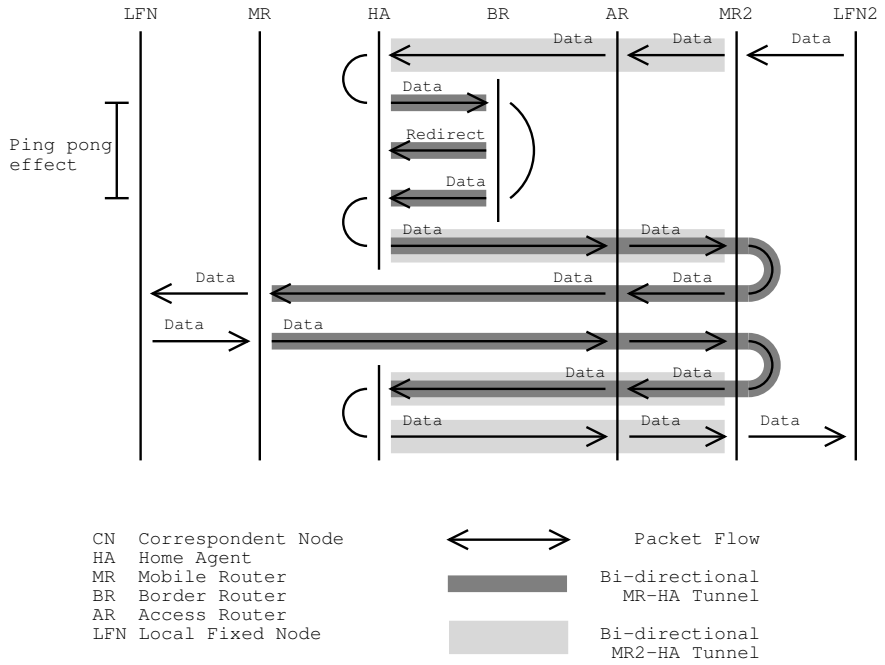



Figure 5.16: Two Mobile Networks, one Home Agent: packet flow

bi-directional tunnel introduces a noticeable delay and large file transfers suffer from i/o-performance problems on the machine that serves as home agent (on a 100 Mbit link). Sometimes this even leads to complete de-synchronization of the TCP connection between the two LFNs.

### 5.2.4 Nested Mobility

As mentioned in the previous subsection, in case of Mobile Network 2 being a subnet of Mobile Network 1 (see figure 5.17), a special problem of nested mobility may occur. The first movement of Mobile Network 2, as shown in figure 5.18 does not cause any problems, communication between LFN1 and LFN2 remains possible (see first part of figure 5.20), both nodes remain reachable from any other node on the network.

But the second network movement, as shown in figure 5.19, where Mobile Network 1 disconnects from its home link and re-attaches in Mobile Network 2, results in total unavailability of both networks.

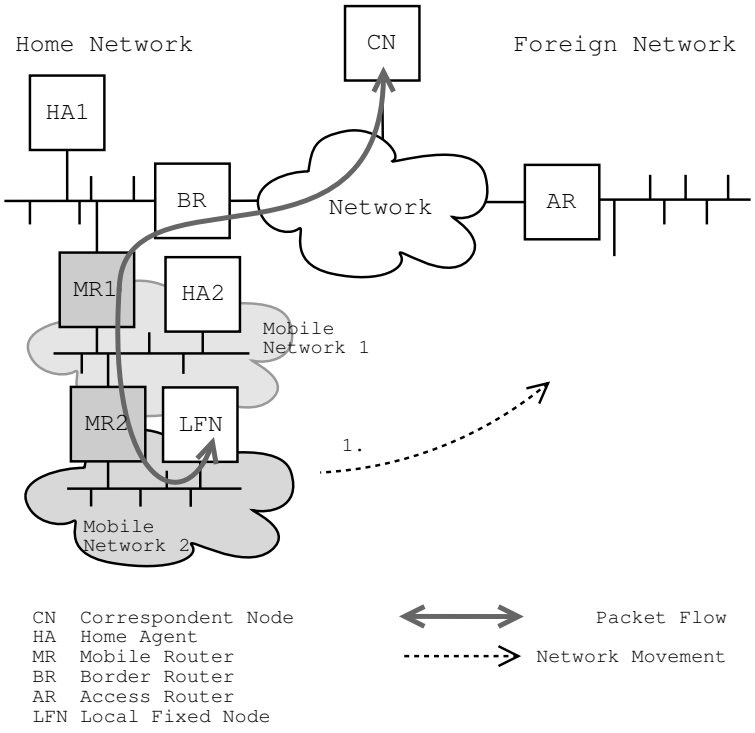



Figure 5.17: IPv6 Mobile Networks nested mobility problem (1/3)

Figure 5.20 shows the packet flow of this scenario, beginning with the Mobile Network 2 connecting to a foreign link. In the last part of the diagram, Mobile Network 1 joins Mobile Network 2, but the binding update, sent from Mobile Router 1 to its home agent never

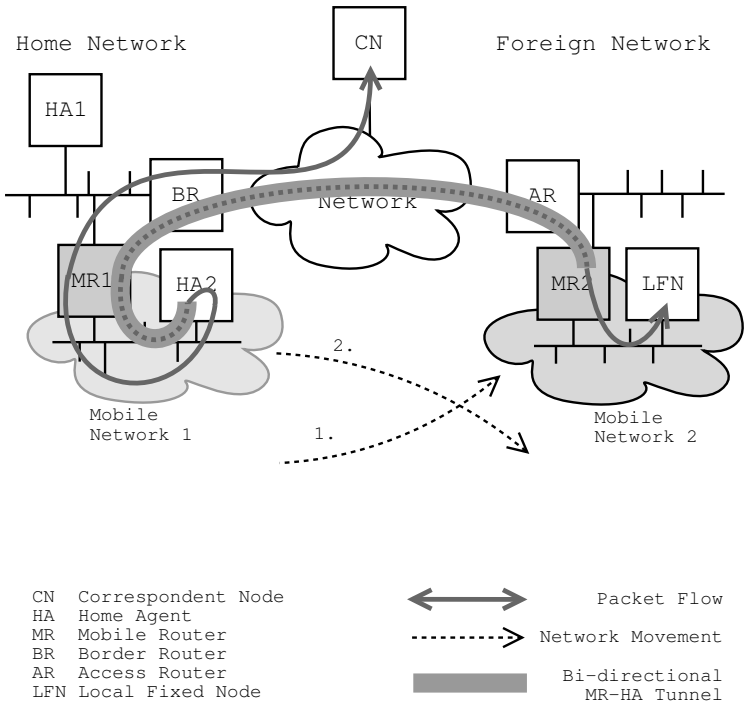



Figure 5.18: IPv6 Mobile Networks nested mobility problem (2/3)

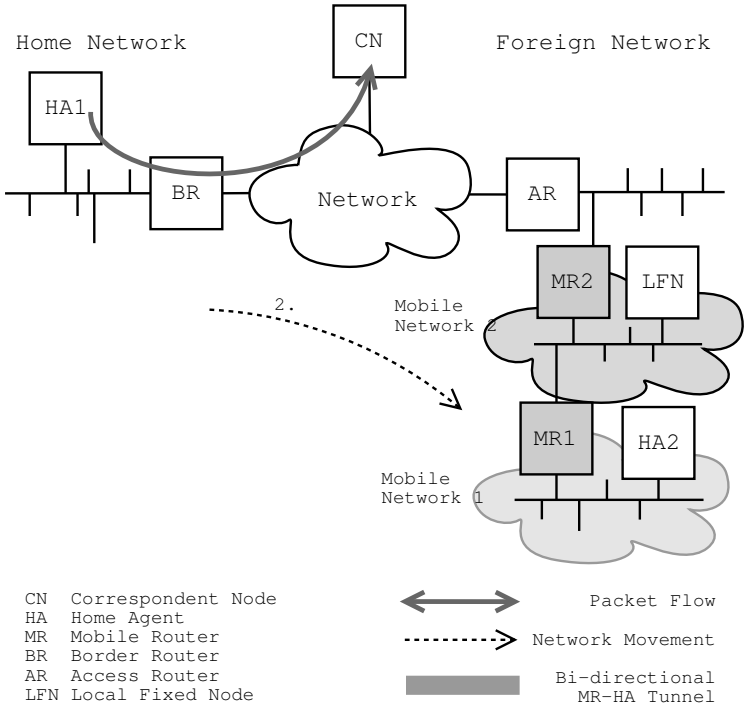



Figure 5.19: IPv6 Mobile Networks nested mobility problem (3/3)

arrives, because the bi-directional tunnel between MR2 and HA2 is not available anymore. The border router in the home network receives the tunneled binding update and tries to deliver it to HA2 via MR1. But MR1 has moved and HA1 does not know it.

With the current approach and implementation, this problem is unsolvable.

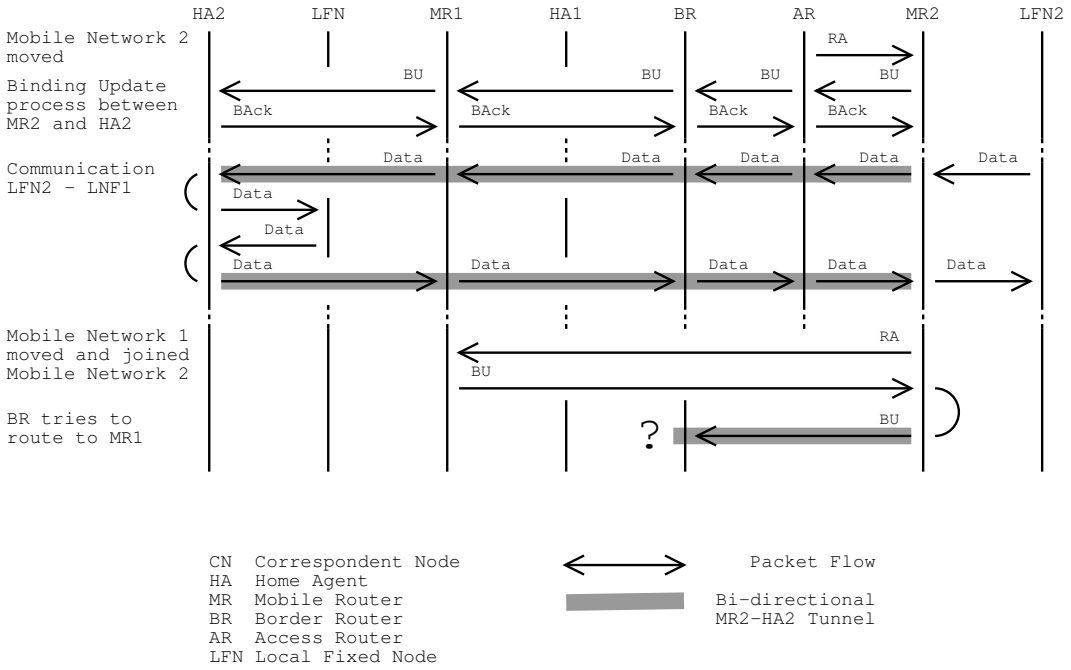



Figure 5.20: IPv6 mobile router nested mobility problem: packet flow

# Chapter 6

# Conclusion

## 6.1 Results

This work shows the development of IPv6 routing for mobility environments, to provide global mobility. Network mobility is realized, based on standard IPv6 routing and Mobile IPv6. The proposed solution is based on the principal idea, that network mobility should be as transparent as possible and realized with as less modifications to existing protocols and standards as possible.

The result is network mobility, managed by two special nodes, the mobile router and home agent, modified to support Mobile Networks. While the mobile network is attached to a foreign link, connectivity is maintained by using the bi-directional tunnel between the mobile router and the home agent.

To realize the solution in the Motorola LIVSIX IPv6 stack, first standard IPv6 routing had to be implemented, since the stack did not provide it before. This includes the capability to take part in the router discovery process, i.e. to send Router Advertisements, periodically and/or solicited. The next steps were to combine routing and host mobility, to add the necessary modifications, which finally resulted in the proposed solution.

Various tests have proven the usability of this solution of network mobility, and showed interesting directions for future work.

LIVSIX provides an adapted research platform, Open Source and distributed under the Motorola LIVSIX Public License. It is used in other research projects, e.g. the IST project OverDRiVE.

LIVSIX homepage: http://www.nal.motlabs.com/livsix/
IST OverDRiVE homepage: http://www.ist-overdrive.org/

## 6.2 Future Directions

Future work has to examine possibilities of Route Optimization. Using a bi-directional tunnel is not an ideal solution and generates unnecessary traffic in the network. Also, a disconnection of a mobile network's the home network results in the unavailability of the mobile network, even if this one is physically reachable, since it is connected to a foreign link.

The impact of the network movements on the used applications and protocols might also be a research direction. Packet loss during handover and varying delays while being attached to different foreign networks have a non neglectable influence on data communication.

The optimization of the handover is also of interest. Discovering the reconnection to another link by other means than the reception of a Router Advertisement with a differing network prefix, might significantly shorten the handover time.

Network mobility is an active research area. Its standardization activities are coordinated by the IETF NEtwork MObility (NEMO) Working Group.

NEMO Working Group homepage: http://www.nal.motlabs.com/nemo/

# Bibliography


[RFC2080] G. Malkin and R. Minnear. RIPng for IPv6. RFC (Standards Track) 2080, IETF, January 1997.

[RFC2081] G. Malkin. RIPng Protocol Applicability Statement. RFC (Informational) 2081, IETF, January 1997.

[RFC2460] S. Deering and R. Hinden. Internet Protocol, Version 6 (IPv6) Specification. RFC (Standards Track) 2460, IETF, December 1998.

[RFC2461] T. Narten, E. Nordmark, and W. Simpson. Neighbor Discovery for IP Version 6 (IPv6). RFC (Standards Track) 2461, IETF, December 1998.

[RFC2462] S. Thomson and T. Narten. IPv6 Address Autoconfiguration. RFC 2462 (Standards Track), IETF, December 1998.

[RFC2463] A. Conta and S. Deering. Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6). RFC 2463 (Standards Track), IETF, December 1998.

[MIPv6] David B. Johnson, Charles E. Perkins and Jari Arkko. Mobility Support in IPv6. Internet Draft draft-ietf-mobileip-ipv6-21.txt (Work in Progress), IETF, February 2003.

[RADIX] Keith Sklower. A tree-based packet routing table for Berkley Unix. Technical report, University of California, Berkeley, 1993.

[FLPM] Marcel Waldvogel. Fast Longest Prefix Matching, Algorithms, Analysis and Application. Swiss Federal Institute of Technology, Zurich, 2002.

# Appendix A

# Glossary

| | |
|---|---|
| IP | Internet Protocol Version 6 (IPv6), Version 4 (IPv4). |
| Node | A device that implements IP. |
| Router | A node that forwards IP packets not explicitly addressed to itself. |
| Host | Any node that is not a router. |
| Subnet prefix | A bit string that consists of some number of initial bits of an IP address. |
| Link-layer address | A link-layer identifier for an interface, such as IEEE 802 addresses on Ethernet links. |
| Destination option | Destination options are carried by the IPv6 Destination Options extension header. Mobile IPv6 defines one new destination option, the Home Address destination option. |
| Home address | An IP address assigned to a mobile node, used as the permanent address of the mobile node. This address is within the mobile node's home link. |
| 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". |
| Mobile host | A host that can change its point of attachment from one link to another, while still being reachable via its home address. |
| Correspondent Node (CN) | A peer node with which a mobile node is communicating. The correspondent node may be either mobile or stationary. |
| Mobile Router (MR) | A router, that is at the same time a mobile node, wich provides mobility for an entire network. |

| | |
|---|---|
| Home Agent (HA) | A router on a mobile node's home link with which the mobile node has registered its current care-of address. |
| Home Address | A (global) IP address associated with a mobile node in its home network. |
| Care-of Address (CoA) | An IP address associated with a mobile node while visiting a foreign link. |
| 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. |
| Binding procedure | A binding procedure is initiated by the mobile node to inform either a correspondent node or the mobile node's home agent of the current binding of the mobile node. |

# Appendix B

# Linux Kernel Programming

During the implementation, the following internet sites provided usefull information on the Linux kernel and the progamation of Linux kernel modules. They provide general information as well as example code, howto use the kernel module interfaces.

Today, LIVSIX uses the /proc-filesytem and ioctl to cummunicate information to the user and applications, ioctl and sysctl to receive information (for furhter information, please consult the following web-sites, especially the Linux Kernel Module Programming Guide).

Kernel Newbies:
http://www.kernelnewbies.org/

Kernel Hacking HOWTO:
http://www.kernelhacking.org/docs/kernelhacking-HOWTO/index.html

The Linux Kernel Module Programming Guide:
http://www.tldp.org/LDP/lkmpg/index.html

# Appendix C

# Test Software

## C.1 Ethereal Packet Analyser

Ethereal is a free network protocol analyzer. During the tests of the implementation it is used to examine the packet contents and to verify their conformity to the existing standarts. Ethereal provides a powerful graphical interface that enables the user to easily separate the packets into different protocols, to have a look at the different protocol headers.

Ethereal can be found on the Ethereal homepage at http://www.ethereal.com/.

## C.2 Ping6

Sending ICMP echo requests with ping6 is the simplest application to test a new feature. Flood-pinging (ping6 -f) is a good possibility to test a router's or Mobile Router's ability to handle large amount of packets in a short time.

## C.3 Vic

Vic is a multimedia application for video conferencing over IP networks, that supports IPv6. Showing live video streaming from one node to a Mobile Node or a Local Fixed Node within a Mobile Network proves LIVSIX's ability to support realtime multimedia applications.

Vic can be downloaded from http://www-nrg.ee.lbl.gov/vic/

## C.4 Quake2

Quake2 is a multi-player network game, a first person shooter. Having a well known real time based game, provides an ideal demonstration platform for LIVSIX. Quake2 is tested on both, a Mobile Host and Local Fixed Node in a Mobile Network.

The IPv6 supporting version of Quake2 can be found at
http://www.viagenie.qc.ca/en/ipv6/quake2/ipv6-quake2.shtml

## C.5 Mozilla

Mozilla, the well known web browser is used to browse web sites and download files. As example site, we used the KAME project site , http://www.kame.net, where one can see a dancing kame, when connecting via IPv6.

Mozilla project page: http://www.mozilla.org/.

## C.6 Ssh and Scp

On our lan, we use ssh to remotely connect to other machines. Handovers, taken place during a connection, show, that session continuity is given. Scp is used to verify stability while transfering large amount of data (we used cd images of 600MByte). This proves the stability of the router and the Mobile Router implementation.

# Appendix D

# LIVSIX HOWTOs

The latest versions of these HOWTOs can always be found on the Motorola LIVSIX homepage: http://www.nal.motlabs.com/livsix/ (in the LIVSIX cvs repository, docs directory).

## D.1 Router Advertisement HOWTO

```
------------------------------------------------------------------------
         HOWTO use LIVSIX Router Advertisement (RA) Functions
               Tim Leinmueller leinmuel@nal.motlabs.com
------------------------------------------------------------------------

This HOWTO describes the configuration and usage of the RA
functionality of LIVSIX. In contrast to other Open Source IPv6 stacks
(namely the KAME IPv6 stack and the Linux IPv6 stack) LIVSIX
implements this functionality within the kernel space and not as a
user level daemon.



Introduction
------------
The LIVSIX source code contains two #defines (see
$livsix/include/sfk/router.h), which specify the maximum number of
supported interfaces (MAX_INTERFACES) and the maximum number of
supported prefixes per interface (MAX_PREFIXES). The MAX_INTERFACES
constant is also used for default interface determination.
```

By default, all router functionalities (this includes sending RAs) are disabled. To change this, you can modify the corresponding variable by the usage of LIVSIX's sysctl interface. Moreover using this interface, you can modify the contents of the RAs, and their periodicity.

For a detailed specification of the router advertisement related LIVSIX sysctl interface have a look at the "LIVSIX sysctl interface" section. In case you just want to discover its functionality, you might start directly with the "First steps / Examples" section, but it is recommended to read the other section at first.

LIVSIX sysctl interface
---

LIVSIX's behavior is controlled through a set of entries in the sysctl structure. After having launched LIVSIX, they are located in "proc/sys/net/livsix/". In this section we will explain the entries related to RAs.

"/proc/sys/net/livsix"

| entry | values | description |
|---|---|---|
| isrouter | 0/1 | disable/enable router functionality |

"/proc/sys/net/livsix/conf"

| entry | description |
|---|---|
| default/ | default values for an interface, if an interface gets "up" (at the moment, used only at startup, so modifying values, won't change anything. |
| ethX/ | this directory exists for every interface ethX (X = 0...?), which was "up" at the startup of LIVSIX. |

"/proc/sys/net/livsix/conf/ethX"

| entry | values | description |
|---|---|---|
| ra_curhop_limit | 0-255 | Cur Hop Limit field |
| ra_def_lifetime | 0-65535 | Router Lifetime field |
| ra_managed | 0/1 | Managed flag 0/1 |
| ra_maxra_interval | 4-1800 | The maximum time allowed between sending unsolicited multicast RAs from the interface. |

| | | |
|---|---|---|
| ra_minra_interval | 3-1800 | The minimum time time allowed between sending unsolicited multicast RAs from the interface. |
| ra_mtu | 0-(32bit) | The MTU to be placed in the MTU option (entered in decimal, 0 means MTU option is not send with the RA). |
| ra_otherconfig | 0/1 | Other Config flag 0/1 |
| ra_prefix_XX/ | | Configuration of the prefix option to be send with the RAs. The directories are always there, even if you don't use them. (XX = 00...MAX_PREFIXES). |
| ra_reachable_time | 0-(32bit) | Reachable Time field (values are entered decimal!) |
| ra_retrans_timer | 0-(32bit) | Retrans Timer field (decimal!) |
| ra_send_ras | 0/1 | Do not/do send RAs on this interface. |

NOTE: The maximum value for ra_min_ra_interval is always inferior or equal to the current ra_maxra_interval, as well as the minimum value for ra_maxra_interval is always superior or equal to the current value of ra_min_ra_interval. Wrong values will be ignored. The time between two unsolicited RAs is randomly distributed in the interval [ra_min_ra_interval,ra_maxra_interval].

"/proc/sys/net/livsix/conf/ethX/ra_prefix_XX"

| entry | values | description |
|---|---|---|
| ra_pfl_autonomous | 0/1 | Autonomous flag 0/1 |
| ra_pfl_onlink | 0/1 | Onlink flag 0/1 |
| ra_pfl_preferred_lifetime | 0-(32bit) | Prefered Lifetime field (decimal) |
| ra_pfl_prefix | IPv6addr | The prefix. A prefix that is NOT a valid IPv6 address will erase this entry (if present in the prefix-list for this interface). (hexadecimal!) |
| ra_pfl_prefixlen | 0-127 | Prefix Length field |
| ra_pfl_valid_lifetime | 0-(32bit) | Valid Lifetime field (decimal) |

For a detailed explanation of the RA fields, take a look at the corresponding rfc (rfc 2461 - Neighbor Discovery for IP Version 6 (IPv6)). There you will also find the default values for the fields.

```
First steps / Examples
----------------------
Start LIVSIX as usual.

To start advertising a prefix:
  echo $PREFIX > \
    /proc/sys/net/livsix/conf/ethX/ra_prefix_XX/ra_pfl_prefix
  echo $PREFIXLENGTH \
    > /proc/sys/net/livsix/conf/ethX/ra_prefix_XX/ra_pfl_prefixlen
  echo 1 > /proc/sys/net/livsix/conf/ethX/ra_send_ras
  echo 1 > /proc/sys/net/livsix/isrouter

NOTE: these are only the minimum options that have to be set to
      advertise a VALID prefix on a link (omitting the prefix length
      will send a prefix option with a 0 in the prefix length field)

EXAMPLE (send on eth0):
  echo 2002:c3d4:6ffd:1:: > \
    /proc/sys/net/livsix/conf/eth0/ra_prefix_00/ra_pfl_prefix
  echo 64 > /proc/sys/net/livsix/conf/eth0/ra_prefix_00/ra_pfl_prefixlen
  echo 1 > /proc/sys/net/livsix/conf/eth0/ra_send_ras
  echo 1 > /proc/sys/net/livsix/isrouter


Add another prefix (on the same link):
  echo $OTHERPREFIX > \
    /proc/sys/net/livsix/conf/ethX/ra_prefix_XX/ra_pfl_prefix
  echo $PREFIXLENGTH \
    > /proc/sys/net/livsix/conf/ethX/ra_prefix_XX/ra_pfl_prefixlen
(ethX should be the same as above :), ra_prefix_XX should be different)

EXAMPLE:
  echo 2002:c3d4:6ffd:2:: > \
    /proc/sys/net/livsix/conf/eth0/ra_prefix_02/ra_pfl_prefix
  echo 64 > /proc/sys/net/livsix/conf/eth0/ra_prefix_02/ra_pfl_prefixlen


Modify a prefix:
  echo $NEWPREFIX > \
    /proc/sys/net/livsix/conf/ethX/ra_prefix_XX/ra_pfl_prefix

EXAMPLE:
```

```
  echo 2002:c3d4:6ffd:3:: > \
    /proc/sys/net/livsix/conf/eth0/ra_prefix_02/ra_pfl_prefix


Stop advertising a prefix:
  echo $<anything that does not match an IPv6 address/prefix> > \
    /proc/sys/net/livsix/conf/ethX/ra_prefix_XX/ra_pfl_prefix

NOTE: The all the options for this prefix will be deleted, even if
      this is NOT reflected to the sysctl entries. This means, if you
      first delete a prefix and then add a prefix to the same table,
      the new entry will use default values!

EXAMPLE:
  echo x > /proc/sys/net/livsix/conf/eth0/ra_prefix_00/ra_pfl_prefix


Stop sending RAs on a special link, and send a last RA with
lifetime 0:
  echo 0 > /proc/sys/net/livsix/conf/ethX/ra_send_ras

EXAMPLE:
  echo 0 > /proc/sys/net/livsix/conf/eth0/ra_send_ras


Stop being router (implies stop sending RAs on all interfaces, at the
moment):

EXAMPLE:
  echo 0 > /proc/sys/net/livsix/isrouter

NOTE: It is a good idea, to stop adverting prefixes some seconds
      before, since when doing a
        echo 0 > /proc/sys/net/livsix/conf/ethX/ra_send_ras
      a last router advertisement is sent, containing a zero in the
      Router Lifetime field, which will cause other nodes to remove
      this router from their default router list.



Final Words
-----------
For comments, suggestions, questions and bug reports, please post on
```

the LIVSIX mailing list or contact the author.

## D.2 LIVSIX Router Setup HOWTO

```
------------------------------------------------------------------------
                HOWTO use LIVSIX Routing Functionality
               Tim Leinmueller leinmuel@nal.motlabs.com
------------------------------------------------------------------------

This HOWTO describes the configuration and usage of the a LIVSIX based
router. The configuration of router advertisements (RAs) is described
in a separate HOWTO, so please refer to this one, if you need
information about RA configuration.



Introduction
------------
As already mentioned in the RA HOWTO, by default, all router
functionalities are disabled. To change this, you can modify the
corresponding variable by the usage of LIVSIX's sysctl interface.

When becoming router, the node's (auto created) defaultrouterlist is
cleared. Then existing routers (gateways) are copied from the routing
table. Future changes of the routing table will also be taken into
account (adding routers as well as deleting routers).  From now on
incoming RAs from other routers will be ignored.

This HOWTO is separated into three sections, the first one that gives
a detailed explanation of the routing table related sysctl variables,
the second one describing the routing table output and the last one
suppling examples.


LIVSIX sysctl interface
-----------------------
LIVSIX's behavior is controlled through a set of entries in the
sysctl structure. After having launched LIVSIX, they are located in
"proc/sys/net/livsix/". In this section we will explain the entry
related to routing table manipulation.

Additional information (as the current routing table) can be found
in "/proc/net/livsix-X" (e.g. "/proc/net/livsix-route").
```

```
"/proc/sys/net/livsix"
entry                 values    description
isrouter              0/1       disable/enable router functionality
routing_information   *         information to add/delete from the
                                routing table.



Routing table output
--------------------
The current routing table is always (even if you disabled routing)
stored in "/proc/net/livsix_route".

The file has the following format:
$DEST_IP_OR_NETWORK $PREFIXLENGTH $GATEWAY_IP $INTERFACE

NEW:  LIVSIX now support vanilla linux route command. To show the
      current IPv6 routing table, enter
        route -A inet 6


First steps / Examples
----------------------

To add an entry to the routing table, enter
  echo add $DEST_IP_OR_NETWORK/$PREFIXLENGTH $GATEWAY_IP $INTERFACE > \
    /proc/sys/net/livsix/routing_information
NEW:
  route -A inet6 add $DEST_IP_OR_NETWORK/$PREFIXLENGTH \
    gw $GATEWAY_IP $INTERFACE

EXAMPLES:
  echo add  fec0:0:0:1202::/64 :: eth0 > \
    /proc/sys/net/livsix/routing_information
  echo add  fec0:0:0:3202::/64 fec0:0:0:1202:2D0:59FF:FECC:A1EB \
    eth0 > /proc/sys/net/livsix/routing_information

NOTE: You always have to enter all options!
      The first example adds an entry without a gateway (net is on the
      same link as the router), in the second example the destination
      network is connected via a router.
```

```
NEW:
  route -A inet6 add fec0:0:0:1202::/64 eth0
  route -A inet6 add fec0:0:0:3202::/64 \
    gw  fec0:0:0:1202:2D0:59FF:FECC:A1EB eth0


To delete an entry from the routing table, enter
  echo del $DEST_IP/$NETMASK > \
    /proc/sys/net/livsix/routing_information
NEW:
  route -A inet6 del $DEST_IP/$NETMASK $INTERFACE

EXAMPLE:
  echo del fec0:0:0:1202::/64 > \
    /proc/sys/net/livsix/routing_information
NEW:
  route -A inet6 del fec0:0:0:1202::/64 eth0

To show the current routing table:
  cat /proc/net/livsix-route
NEW:
  route -A inet6
```

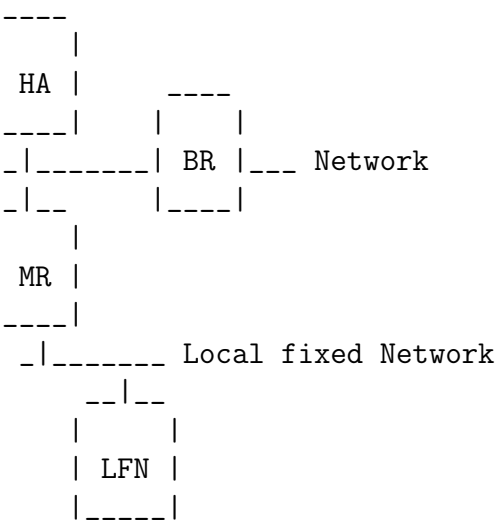
## D.3 Mobile Router HOWTO

```
------------------------------------------------------------------------
                  HOWTO use LIVSIX as Mobile Router
              Tim Leinmueller leinmuel@nal.motlabs.com
------------------------------------------------------------------------

This HOWTO describes the configuration and usage of a Mobile Router
(MR) using LIVSIX. The configuration of router advertisements (RAs) as
well as the configuration of a normal router are described in separate
HOWTOs, so please refer to these ones, if you need additional
informations.



Introduction
------------
Although it is not absolutely necessary, it's strongly recommended to
read the two howtos mentioned above at first. This will help you to
understand how to setup the Mobile Router and his Homeagent.

The main difference between a normal router and a MR (based on LIVSIX)
is, that the Mobile router will accept router advertisements (RAs) on
ONE single interface, his so called mobile or egress interface. By
accepting the RAs he is able to determine whether he is at or away
from home. While being at home he acts as a normal router, while
visiting a foreign network, he establishes a bidirectional tunnel to
his homeagent (HA).

On the HA side, we "abuse" the routing table to enter routes to the
fixed networks behind the MR. If the MR is away from home, which means
the HA has to intercept all packets for the MR, the HA acts as
follows. It does a lookup in his routing table and if he finds a next
hop, he searches his binding cache for this nexthop to finally tunnel
the packet to the MRs care of address.

This HOWTO is separated into three sections, the first one that gives
a detailed explanation of the MR related sysctl variables, the second
one describing the output and the contents of the modified routing
table and the last one suppling examples.
```

```
LIVSIX sysctl interface
-----------------------
The following entries are required to configure a MR. In addition,
router advertisements (RAs) should be configured as described in the
corresponding howto. Please note, that at the moment you should not
advertise any prefix on your mobile interface, since it is not assured
that advertisements are suspended while visiting a foreign network.

NEW:  Livsix now assures NOT to send RAs on a visited link, so you can
      now configure it to send RAs on the mobile interface. But you
      should set the Default Router Lifetime on this interface to zero
      since the mobile router hardly wants to be default router in his
      home network.

"/proc/sys/net/livsix"
entry                 values    description
isrouter              0/1       disable/enable router functionality
routing_information   *         information to add/delete from the
                                routing table.
mr_mobile_interface   0/ethX    defines the MRs mobile interface

For more information concerning the routing table, please refer to the
routing table howto.


Routing table output
--------------------
In addition to the normal routing table, a MR uses a so called Shadow
Routing Table. It is stored in "/proc/net/livsix-route_shadow".
It's functionality is to store all routes related to the mobile
interface when being away from home. By this means the MR is able to
determine the routes for which a packet has to be sent through the
bidirectional tunnel.

The files have the following format:
$DEST_IP_OR_NETWORK $PREFIXLENGTH $GATEWAY_IP $INTERFACE

NOTE:   route -A inet6 prints the entries from both routing tables
        (first the normal than the shadow, not separated).


First steps / Examples
----------------------
```

```
To set a HA, a MR and a Local Fixed Node (LFN) in the fix network of a
MR, execute the following steps

On the HA:
- Add a route to the local fixed network
    echo add $DEST_NETWORK/$PREFIXLENGTH $MR_HOMEADDRESS $INTERFACE > \
      /proc/sys/net/livsix/routing_information
NEW:  you should use
        route -A inet6 add $DEST_NETWORK/$PREFIXLENGTH \
          gw $MR_HOMEADDRESS $INTERFACE
      instead.

- Modify $LIVSIX/userspace/livsix.sh that LIVSIX is started in HA
  mode

On the MR:
- Modify your $LIVSIX/userspace/livsix.sh that it contains the right
  HA, that LIVSIX is started in HA mode and that Route Optimization
  (RO) is disabled.

- Add the prefixes to advertise on the local fixed network as
  described in the corresponding howto (or see the supplied example)

- Enable the router
    echo 1 > /proc/sys/net/livsix/isrouter

- Add the (default) route to the Border Router (BR)
    echo add :: $BR_ADDRESS $EGRESS_INTERFACE > \
      /proc/sys/net/livsix/routing_information
NEW:  you should use
        route -A inet6 add :: gw $BR_ADDRESS $EGRESS_INTERFACE
      instead.

- Add a route to the local fixed network
    echo add $DEST_NETWORK/$PREFIXLENGTH :: $INGRES_INTERFACE > \
      /proc/sys/net/livsix/routing_information
NEW:  you should use
        route -A inet6 add $DEST_NETWORK/$PREFIXLENGTH \\
          $INGRES_INTERFACE
      instead.

- Add an address on the $INGRES_INTERFACE which is in the associated
```

```
  $DEST_NETWORK
    $LIVSIX/userspace/livconfig add $DEST_NETWORK_IP
NEW:  you must use
        ifconfig $INGRES_INTERFACE add $DEST_NETWORK_IP/$PREFIX
      instead.

- Set the mobile interface:
    echo ethX > /proc/sys/net/livsix/mr_mobile_interface

- To show the current routing tables:
    cat /proc/net/livsix-route
    cat /proc/net/livsix-route_shadow # (empty when at home)
NOTE:   route -A inet6 prints the entries from both routing tables
        (first the normal than the shadow, not separated).

Now you have a network that should like this
```




```
Example
(warning: all 2002:c3d4:6ffd:xxxx:/64 prefixes belong to Motorola Labs
 in Paris, so please don't use those prefixes, use your local prefixes
 instead.)
Do the changes in the livsix.sh files. Then execute the following

# HA
  echo add 2002:C3D4:6FFD:1206::/64 \
    2002:C3D4:6FFD:1202:202:2DFF:FE49:D97D eth0 > \
    /proc/sys/net/livsix/routing_information
#NEW:
```

```
  route -A inet6 add 2002:C3D4:6FFD:1206::/64 \
    gw 2002:C3D4:6FFD:1202:202:2DFF:FE49:D97D eth0

# MR
  echo 2002:C3D4:6FFD:1206:: > \
    /proc/sys/net/livsix/conf/eth1/ra_prefix_00/ra_pfl_prefix
  echo 64 > /proc/sys/net/livsix/conf/eth1/ra_prefix_00/ra_pfl_prefixlen
  echo 1 > /proc/sys/net/livsix/conf/eth1/ra_send_ras

  echo 1 > /proc/sys/net/livsix/isrouter

  echo add ::/0 2002:C3D4:6FFD:1201:210:4bff:FE4d:735a eth0 > \
    /proc/sys/net/livsix/routing_information
#NEW:
  route -A inet6 add ::/0 \
    gw 2002:C3D4:6FFD:1201:210:4bff:FE4d:735a eth0

  echo add 2002:C3D4:6FFD:1206::/64 :: eth1 > \
    /proc/sys/net/livsix/routing_information
#NEW:
  route -A inet6 add 2002:C3D4:6FFD:1206::/64 eth1

  $LIVSIX/userspace/livconfig eth1 add  \
    2002:C3D4:6FFD:1206:204:75FF:FE80:ACF9
#NEW:
  ifconfig eth1 add 2002:C3D4:6FFD:1206:204:75FF:FE80:ACF9

  echo eth0 > /proc/sys/net/livsix/mr_mobile_interface
```

# Appendix E

# Bluetooth HOWTO

This HOWTO represents the experimentations to use the Linux BlueZ Bluetooth stack together with an AXIS access point. The aim of the experimentations is to simulate a PAN, i.e. a PDA that connects to a laptop using Bluetooth (via the access point), and this laptop acts as Mobile Router.

Since BlueZ changed a lot since the writing of this document, it is a bit out-of-date. Nevertheless, this document provides important information on the interaction of BlueZ with the AXIS access point (and thus with the AXIS linux Bluetooth stack, since the AXIS access point is based on their own stack).

With the recent updates, connecting a laptop using the USB stick and the LAN profile is possible. Also using a Compaq IPAQ together with the access point works. Browsing the web is possible from the PDA is possible.

## E.1 BlueZ - AXIS HOWTO

```
-------------------------------------------------------------------------
NOTE:   This HOWTO has to be considered as work in progress. It may
therefore be incomplete and contain errors. Comments, help and
suggestions are appreciated. I was able to connect to the AXIS
9010, but after the ppp link establishment the connected computer
freezes when accessing the AXIS web-interface (I did not try to
connect to a LAN).
-------------------------------------------------------------------------

HOWTO connect to an AXIS 9010 using BlueZ
-----------------------------------------
The AXIS 9010 implements the Bluetooth LAN profile, which means it
```

```
provides a connection using ppp for Bluetooth devices to a LAN.

This text shows how to connect to this access point using the BlueZ
Bluetooth stack.


Hardware:
AXIS 9010 access point
Compaq Armada M700
3COM Bluetooth USB Adapter 3CREB96
(3COM 3CRWB6096 PC Card - froze system after "hciconfig up")

Software:
Linux kernel 2.4.20-pre4
bluez-bluefw-0.5 (http://bluez.sourceforge.net/)
bluez-hcidump-1.3 (http://bluez.sourceforge.net/)
bluez-libs-2.0 (http://bluez.sourceforge.net/)
bluez-sdp-0.7 (http://bluez.sourceforge.net/)
bluez-utils-2.0 (http://bluez.sourceforge.net/)
bt3c-0.4 (http://www.holtmann.org/linux/bluetooth/ - for the 3COM PC Card)
rfcomm-0.10 (http://www.holtmann.org/linux/bluetooth/)
pppd version 2.4.1


General Setup
--------------
Compile the kernel with the following options:
CONFIG_BLUEZ=m
CONFIG_BLUEZ_L2CAP=m
CONFIG_BLUEZ_HCIUSB=m
CONFIG_BLUEZ_HCIBT3C=m  #if you want to try to use the 3COM PC Card.
#use other drivers, depending on your hardware :)

Don't forget to include ppp support (I used the modules)!

Unpack, build and install the BlueZ packages mentioned above. To build
the packages please refer to the supplied READMEs (in general
"./configure", "make", "make install" should do it).  If you have any
problems with dependencies, try to install the other packages first.

Install the rfcomm package, as described in the supplied README/INSTALL.

If you want (try) to use the 3COM PC Card install bt3c-0.4, for any
```

```
other PC Card use the corresponding driver.



First Steps
-----------
Note: The following steps assume that you use a USB Bluetooth device.
You should easily adapt them for other devices.

Insert the USB-Stick. Execute the following commands:
"modprobe bluez"
"modprobe hci_usb" #differs if you use a PC Card
"modprobe uhci" #or usb-uhci, depending on your USB Host
#Controller Driver
"modprobe l2cap"

Now execute "hciconfig"
The output should look like this:
hci0:   Type: USB
        BD Address: 00:00:00:00:00:00 ACL MTU: 0:0  SCO MTU: 0:0
        DOWN
        RX bytes:0 acl:0 sco:0 events:0 errors:0
        TX bytes:0 acl:0 sco:0 commands:0 errors:0

To bring up the interface, do an "hciconfig hci0 up" :).
Executing "hciconfig" again, you should receive:
hci0:   Type: USB
        BD Address: 00:04:76:F2:C3:2E ACL MTU: 128:8  SCO MTU: 64:8
        UP RUNNING PSCAN ISCAN
        RX bytes:69 acl:0 sco:0 events:8 errors:0
        TX bytes:27 acl:0 sco:0 commands:7 errors:0

Do a "hcitool scan" to scan for Bluetooth devices. As a result, you
should find the AXIS 9010.
Scanning ...
        00:30:8E:4F:A9:BA       AXIS 9010 (192.36.253.80)

You should now be able to ping the AXIS using l2ping ("l2ping
<bd_addr>"). You may also make use "hcidump" to monitor the
connection.
```

```
Connect to the AXIS 9010
------------------------
Edit /etc/bluetooth/pin that it contains "123456" (I'm not sure if you
really have to :)).

Now you have to set the Bluetooth PIN on the AXIS to the same value.
According to the manual, by default it is set to an empty string.
This seems not to be supported by BlueZ. (Do your own test, and
monitor them with hcidump - seems that BlueZ requires at least 5
chars).  To set the PIN, branch the AXIS to an Ethernet port and use
the web-interface as described in the manual. Once using the
web-interface, you might also want to set up the Bluetooth-IP-settings
for the Clients. In case you want to test the AXIS without a
connection to a DHCP-server, you have setup the client IPs manually,
either by choosing "In a Private network Masquerading range..." or
"Manual range from... to...".  The first option "Automatically" won't
work.

Now start "hcid" and "sdpd".

Run "rfcomm 0 <bd_addr>" to open a serial connection to the AXIS.  A
window should pop up and ask you for a PIN. Enter "123456" :).  You
should receive:
Connected /dev/ttyU0 to <bd_addr> on channel 1

Once you got this serial connection, you can use ppp. Execute
"pppd /dev/ttyU0 noauth". Take a look at /var/log/messages to see, that
you got a connection.

NOTES:
1. After establishing the ppp-link my machine freezes when
   using the ppp-link (ping works, but accessing the
   web-interfaces causes the freezing).
2. If your ppp-deomon fails, you may have to modprobe
   some missing ppp-modules


Author:
Tim Leinmueller
leinmuel@nal.motlabs.com
```

# Appendix F

# Mobile Network Test Scenarios

Figure F.1 is included, to try to give an overview on the multiple test and testbed setups, that are used during the mobile network tests. Due to the tremedous amount of different setups, providing separate diagrams for every setup is out of scope for this document.

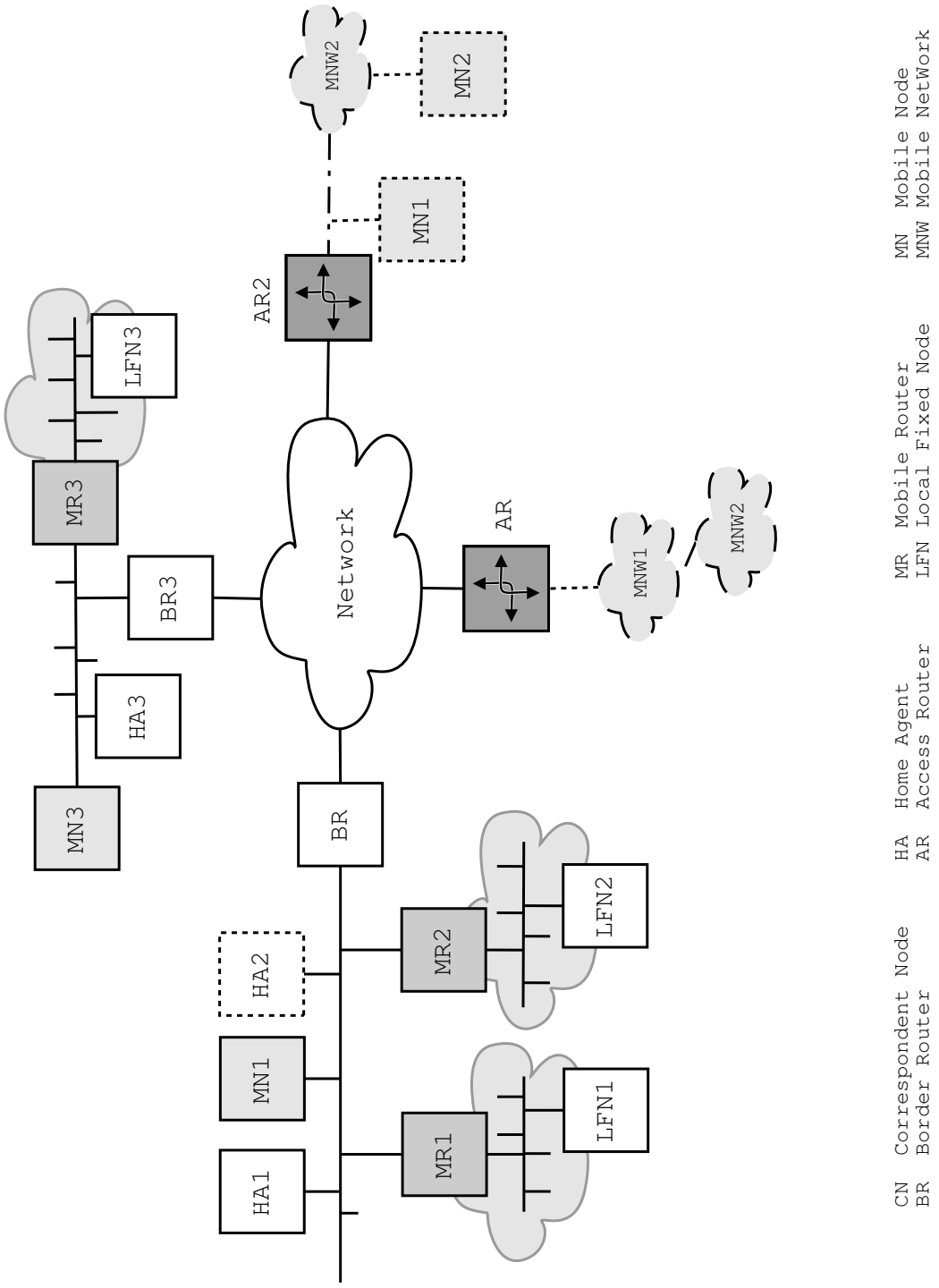



Figure F.1: IPv6 Mobile Network test scenarios