IPv6 Networking And Seamless Handover Study For Mobile Communications In Airplanes

By

Ignasi Esteva Gras

Tutor: Dr. Markus Wegner and Eriza Hafid Fazli

TriaGnoSys GmbH

Supervisor: Josep Paradells

UPC (Universitat Politècnica de Catalunya)

Escola Tècnica Superior d'Enginyeria de Telecomunicació de Barcelona

ETSETB

Munich (Germany)

February 2009 - August 2009

Abstract

Aeronautical communications have been evolved and changed radically during the last years and still doing so. Moreover, passenger communications are going to become widespread in the nearly days. The needs for data transmissions is increasing very fast, and the information exchange among all the parties involved in air traffic management are demanding better and faster ways to transmit. In the future there will be much more aircraft traffic compared to today, and the capacity provided by currently available communication technology will not be enough to manage these aircrafts. Furthermore, providing new link technologies with higher capacity will not be sufficient. In order to meet these future requirements, new networking concepts have to be developed.

Nowadays, one of the main problems in the aircraft traffic communications (ATC) is the delay caused by the long distance that the signal has to pass through from the airplane to ground networks and vice versa. Also, when an airplane is moving and has to change its point of attachment as it moves from one satellite cover zone to another, another delay is produced in the communication. Mobility handovers are just one aspect very important to bear in mind in the design of the future aeronautical communication.

These drawbacks can be solved using the new Internet Protocol version 6 (IPv6) which many networks are starting to use, replacing the current IPv4. Thanks to that new protocol and its extensions, it can be possible to create and perform new mechanisms that bring better benefits to mobile communications. Mobile protocols like Mobile IPv6 and NEMO, make possible the development of new technologies that permit entire networks to change their point of attachment through different networks without having to change their IP addresses.

Thus, this project deals the way to improve this mobility handlings and reduce the delays produced in the actual aircraft network handovers. To solve the problem, new ways will be studied and new applications developed to ameliorate the air traffic management (ATM) using the new IPv6 protocol and its mobile extensions that, as it will be seen, will permit to have multiple routes between the aircraft mobile network and the home base ground agent. That allows to share load and avoids to have to wait to make the transition from one point of attachment to another.

Within the mark of the NEWSKY project [1], we will design an aeronautical communication system using those new mechanisms, which will be tested through real aircraft communication applications to obtain conclusions of the new benefits. To have seamless handovers between a terrestrial link and a satellite link in an airplane-base ground communication is one of the main objectives.

This project describes the design of a network test-bed to simulate network handover between satellite and terrestrial communication links based on MIPv6, NEMO, Mobile IP handover achievement and TCP behaviour.

Also, to improve seamless handovers and to bring new features like load balancing and routing policies, various solutions are searched and discussed. The Multiple Care-of Addresses Registration extension protocol (MCoA) solution is the one that will be implemented and several test simulations will be done to characterize its behaviour in the test-bed.

Contents

List of Figures

Abbreviations

BGAN	Broadband Global Area Network
CN	Correspondent Node
ComMa	Communication Manager
DNS	Domain Name Server
FTP	File Transfer Protocol
HA	Home Agent
HTTP	HiperText Transfer Protocol
IFE	InFlight Entertainment
ICMP	Internet Control Message Protocol
IETF	Internet Engineering Task Force
INMARSAT	International Marine/Maritime Satellite (organization)
IP	Internet Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
ISP	Internet Service Provider
LAN	Local Area Network
MAC	Media Access Control
MTU	Maximum Transmission Unit
MN	Mobile Node
MR	Mobile Router
NAPT-PT	Network Address and Port Translation - Protocol Translation
NAT-PT	Network Address Translation - Protocol Translation
NEMO	Network Mobility
QoS	Quality of Service
RFC	Request for Comments
RO	Route Optimization
RTP	Real-time Transport Protocol
TCP	Transport Control Protocol
UDP	User Datagram Protocol

VoIP	Voice over IP
VPN	Virtual Private Network
WLAN	Wireless Local Area Network

Chapter 1

Introduction

Internet is being more and more ubiquitous everyday and it is becoming a very necessary and indispensable tool in our society. As a consequence of its spread, increasingly many entities and people tend to depend more on Internet and for different purposes: to work, for leisure, education, etc, what turns it into a powerful tool and very indispensable than ever at anytime and everywhere. No wonder then, that this technology is being expanded into new areas where its implementation is still lacking or difficult and costly to achieve.

In the recent days, Internet is beginning to be supported in platforms like planes, trains, cars and other vehicles that move fast, where Internet network mechanisms are difficult to implement as the points of attachment are not fixed [2]. In the aeronautical area, it will contribute on improvements for ATC, which will integrate all traffic generated between airplanes, satellites and base stations but also new service applications for on board passengers to a unique network. But to achieve this issue and to support Internet in vehicles moving rapidly, new protocols and technologies are needed to enable good, fast and safety connections with seamless handover.

This project has been developed in TriaGnoSys GmbH, in Munich, Germany, as part of the NEWSKY project. The goals are to study, develop and implement new mechanisms in airplane systems which will bring improvements in air traffic management and new aircraft on-board applications and services with better mobility connections. This project tries to take advantage of the new IPv6 protocols and all its mobility extensions (described in next chapters) for airplanes-base ground communications and study the results obtained to see its viability, give solutions for possible drawbacks and improve issues like reductions of the delays and the number of packets lost during the transmissions.

To explain how these improvements are applied, the project will describe a laboratory test-bed developed to simulate a real airplane communication system where the new protocols will be implemented to integrate new services and applications in the ATC. This project will study its viability and will give some solutions for the problems encountered.

Also, this document seeks to familiarize the reader with the new protocols implemented in NEWSKY project to facilitate the understanding of future projects in the fields of mobile communications.

In this introduction chapter a small review of the aeronautical traffic and communications situation studied in the project is exposed. Then, the scenario where the project will be developed is presented. Then, it is explained the main goals, proposals and requirements of the project. Finally, there is a description of this document structure.

1.1 Overview

The scenario which the project will refer consists of an airplane which has a network inside, a satellite which connects the aircraft to the ground networks and a ground station. This network comprises a WLAN used by passengers to access the Internet and a subnetwork used to share cockpit information. All these subnetworks are attached to a router, which has an antenna to connect to the stations on ground. Nowadays, satellite communications [3] are used because their coverage area is bigger than any other system, which means less handovers between access points. However, there are some problems during the communications like delays and packets lost due to the troposphere and the long distance that the signal must pass through. As an example of a coverage area of a satellite, the Inmarsat BGAN satellite, used in the test-bed, has a corevage like the one illustrated in the Figure 1.1.

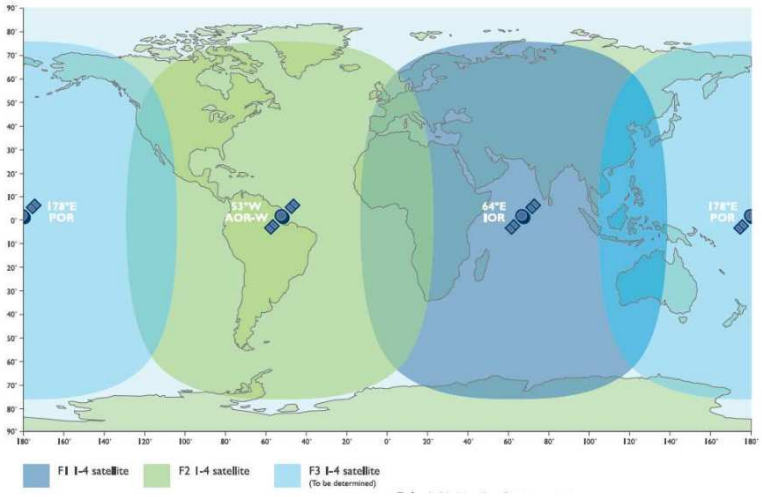

Figure 1.1: BGAN satellite coverage map

The router on board is connected to a satellite, but, while the airplane is moving and reaches the limit of the satellite coverage area, it has to change its point of attachment and connect to another satellite link (see Figure 1.2). When it happens, the router gets a new IP address, what causes a breakdown of the ongoing connections. This is because the TCP connections are characterized by the address and port of the source node and the address and port of the

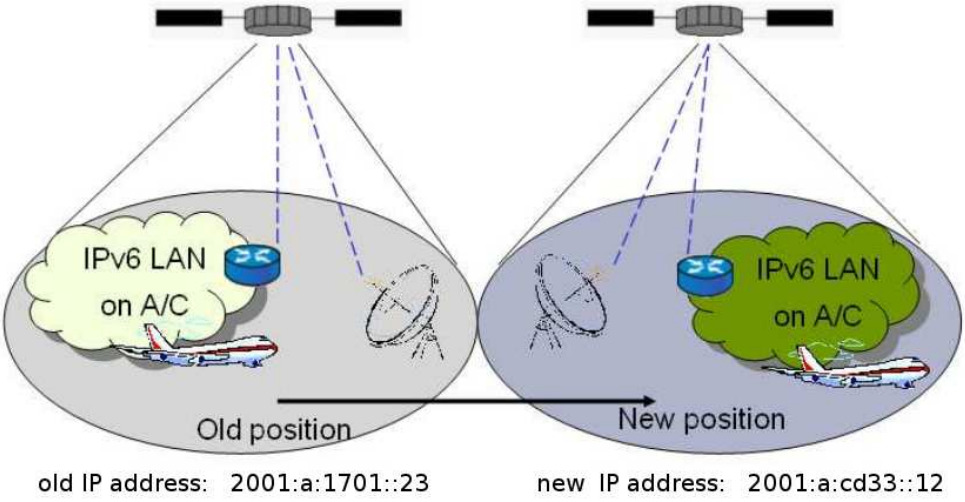

Figure 1.2: Aircraft Handover Process

destination. Then, if one of these changes, the connection breaks down and a new one must be established [4].

The plane can also change from one satellite system to another, if the signal is stronger in the new system at one point during the flight. That also means a change of address and the correspondent cut in the connections.

1.2 Background

TriaGnoSys Company is in charge of a laboratory test-bed within the NEWSKY project, which has the task to demonstrate network layer handovers in the middle of Voice over IP (VoIP) calls, and large data transfer. This project will describe the test-bed design and configuration and it will use it to make test simulations and to obtain conclusions from the results.

1.2.1 NEWSKY project

The NEWSKY project (NetWorking the SKY for Aeronautical Communications) is carried out by a group of European organizations with the aim to integrate different communication technologies and different application classes, into a global heterogeneous aeronautical communications network with appropriate priority properties.

As aeronautical communications are going to be radically changed in the future, the needs for data transmission will dramatically increase, with a view to share more and more information among all the parties involved in air traffic management. It is said that in the future there will be much more aircraft traffic compared to today, and these aircrafts can only be managed if

all information related to them (position, altitude, intended flight path, remaining fuel...) are properly disseminated to all parts involved in managing them (air traffic controllers). And this dissemination of information demands much more capacity compared to the one provided by currently available communication technology. Moreover, passenger communications are going to become widespread in a few years.

In order to meet these future requirements, NEWSKY project tries to solve this problem: instead of using individual communication systems for the various aviation control areas and applications, network solutions are proposed to integrate all these systems, using as often as possible Commercial-Off-The-Shelf components.

NEWSKY is a research project co-funded by the European Commission within its Sixth Framework Program (FP-6) that will enable to achieve improved communication capabilities and assists the expected paradigm shift in future ATM. Moreover, real air-ground integration is achieved and System Wide Information Management (SWIM) is made available to the aircraft. As a consequence, the NEWSKY approach supports the realization of the Single European Sky concept and helps to create a future European ATM system[1].

1.2.1.1 NEWSKY challenges, activities and benefits

The main objectives and activities carried out by NEWSKY are:

Goals

NEWSKY aims to offer global information availability and sharing that will solve the problem of ATM saturation. To pursue this global situation of "Networking the Sky", NEWSKY will integrate different communication links and technologies. Concerning data links, it is working in different links to fulfill the ATM communication requirements, and new future ones will be developed, with links communications like ground-based communications, satellite-based communications, air-air communications and communications in and around airports.

Also, the project will integrate different applications and services in the same network. It will try to put together up-to-date services and applications like ATS (Air Traffic Service), AOC (Airlane Operational Communications), Airline Administrative Communications (AAC) and APC (Air Passenger Communications) into a single, seamless aeronautical network.

This project tries to develop and design an integrated aeronautical communication network focusing on air-ground communication with IPv6 technologies, using well proven industry standards to enable a cost-efficient global provision of distributed services.

Activities

The first activity of NEWSKY is to identify application scenarios and service requirements. Then, a business case study is conducted, with the development of a transition roadmap and a long-term evolution. The main part and challenge of the project is the development of a networking concept, comprising:

- The protocol stack architecture development, including standard interfaces to radio link layers.

- The mobility management, including handover techniques, routing and multihoming.
- The Quality of Service management.
- The development of network security solutions.

At the end, the NEWSKY integrated airborne network design will be validated by means of computer simulations, and a laboratory test-bed.

Benefits

NEWSKY will bring significant benefits:

- An increased availability and reliability through the efficient use of different communication links.
- A globally optimized network performance, coming from the use of the right communication link.
- Technology at the right place and time.
- An interoperability between the different communication links resulting in a seamless system, which will be fully transparent to end users.
- A modular system concept, which enables the simple introduction of new technologies.
- An efficient and flexible utilization of the overall aeronautical frequency spectrum.

1.2.2 TriaGnoSys laboratory test-bed

The TriaGnoSys Company is one of the members of the NEWSKY project, whose contributions consist of a Business Case Study and a Laboratory Test-Bed demonstration, which are aimed at demonstrating handovers procedure between a terrestrial and a satellite links and validate seamless handovers. In order to make the demonstration as realistic as possible, typical aeronautical applications will be developed, like voice communication, pilot-controller messaging and data streaming information through advanced weather multicast applications [5].

1.2.2.1 Scenario

The scenario where the project will be developed represents a network inside an aircraft moving with different communication links and with different applications and mechanisms. The configuration of the test-bed and its components and protocols will be exposed in detail in chapter 5, but here a short description of them is exposed:

1. **New protocols**

   The test-bed is implemented with the new network-layer version protocol, the IPv6, as it is specified in the NEWSKY requirements. It offers more advantages compared to IPv4.

   To improve seamless handovers for mobile hosts and networks, IPv6 extensions are used to support mobility. That is why the Mobile IP protocol, and its extension to mobile networks NEMO (NEtwork MObility), has to be deployed. However, because the satellite network used is established over IPv4, it requires the use of a transition mechanism between our IPv6 network and the IPv4 satellite network. In Chapter 2 these protocols are described.

2. **Terrestrial and Satellite Links**

   In order to recreate the terrestrial and satellite links, there are two possibilities: either emulating the modem and link behaviour, or using a real modem and link.

   For the terrestrial link, a B-AMC (Broadband - Aeronautical Multi-carrier Communications) modem emulator will be used. Concerning the satellite link, the real Inmarsat BGAN modem will be employed. In Chapter 5, their main characteristics are detailed.

3. **Applications**

   During the handover demonstration, several applications (described in Section 5.3) will have to be running, successively or at the same time, in order to qualify the effect of the handover and traffic delays on these applications:

   - Voice over IP, to make possible the communication between the pilot and the controller.
   - Pilot - Controller messaging.
   - Weather streaming information, which represents data communications.

4. **Demonstration GUI**

   In order to make the presentation of the test-bed clearer and more convincing, a demonstration GUI (Graphical User Interface) program is developed, which displays a flight scenario, and some important information about the whole system, in order to have an idea of the test-bed system behaviour.

1.3 IPv6 and MIPv6

In the last few years, the IPv6 protocol has been designed as a new network-layer protocol version to improve and resolve several problems that appeared with the current IPv4. As it will be explained in Chapter 2, for example, the number of hosts to join Internet is increasing and the IPv4 space address is almost full, so a solution was required. The IPv6 brings this solution as it has more address space as the addresses length is increased, which reaches 128 bits, whereas in IPv4 only 32 bits are used.

From the beginning, many organizations, like the IETF group (Internet Engineering Task Force), have developed the IPv6 and several extensions that allow to solve the problems not only

of limited address space but also in many other areas where IPv4 could not face [6]. With IPv6, air traffic management will be improved, and thanks to its mobility extensions, mobile networks will take an important part in aircraft communications. Those are the reason NEWSKY wants to use it.

Due to the great importance IPv6 will have in the near future and because it is an important mechanism used in the test-bed of the project, in Chapters 2 and 3 the IPv6 protocol and its mobility extensions are described in detail.

1.4 Goals

The objective of this project is to design a new aeronautical system based on a mobile network inside planes with new applications and services which will be able to change its point of attachment with small delays.

A study of its behaviour is carried out to see the issues that can be improved as well as to ameliorate the protocols implemented to obtain better communications handovers. For that, several measurements using real communication application systems, like voice over IP, will be done to verify the behaviour of the system during simulated flights (where the mobile network changes its point of attachment over different links) and to obtain results with the aim to get conclusions for future improvements.

To reach this purpose, it will be used new mobile protocols, such as Mobile IPv6 protocol and NEMO extensions. After a theoretical definition of those protocols to get used to the new scenario, they will be implemented in the test-bed where simulations using typical aeronautical applications will be done to see the improvements of seamless handovers and routing processes in air traffic communications.

This project also discusses the possibility to improve seamless handovers using recent developed mechanisms like MCoA extension protocol, which finally will be implemented to the test-bed and tested to see the improvements or not regarding the first configuration.

1.5 System requirements

The NEWSKY test-bed should be able to provide the following requirements:

- Provide a mobile network transparent in front of the change of the points-of-attachment during the handovers.
- Users should not notice handovers between the different ground-satellite links.
- The Quality of Service should be maintained and not reduced.
- It should support Security Protocols and VPN.
- It should change the functionalities of existing protocols as less as possible.
- It should add small signaling and data overhead, when dealing with mobility as well as IPv6 over IPv4 links.

1.6 Structure of this document

The document begins with a theorical explanation of the protocols used in TriaGnoSys test-bed, as well as the MCoA registration protocol as will be also implemented to improve seamless handovers. Then, in Chapter 5, first the test-bed configuration is exposed and after that, it discusses the results of several VoIP calls in flight simulations to see the RTP packet delay during handovers. With this results, it compares the system behaviour between using only NEMO procotol and implementing the MCoA extension protocol. Finally, the document finishes with a conclusion.

Chapter 2

Internet Protocol version 6

Internet is almost everywhere so it is not strange that nowadays the majority of entities are getting into its world. The worst issue related to that growth has been the huge number of hosts that are connecting to the Internet, and the associated IP addresses that are being consumed by those hosts [6]. The most used IP protocol in our days, the IPv4, is having some problems with this growth, so the IETF and other organizations started to search for new solutions. The results of this research was the new network-layer protocol version IPv6 which have solved all IPv4 drawbacks and improved new features. That is the reason why this project will use this new protocol and its extensions in almost all its parts. The implementation of this new mechanism will allow to improve systems mobility and have better results in communication seamless handovers and for packet delays as it has better mobility extensions compared to IPv4.

In this chapter it is exposed the characteristics and the main functionalities of the IPv6 protocol and the most significant changes and improvements from its predecessor IPv4. First of all, there is an introduction about why it was necessary to design a new IP protocol. Secondly, the set of communications protocols used for the Internet and other similar networks known as TCP/IP stack, is described to introduce the IP protocol. Next, the new IPv6 protocol is explained focusing on the new Header and its Extension Header, and the addressing architecture with its new addressing types and functionalities [7][8]. Also the changes and improvements from IPv4 are discussed. Finally, the chapter ends with the description of different solutions to traverse IPv6 packets through IPv4 links.

2.1 Introduction

Since the early 1990, the IETF has been developing the IPv6, a new network layer protocol which will substitute the IPv4. The issue that motivated its developing was mainly because the

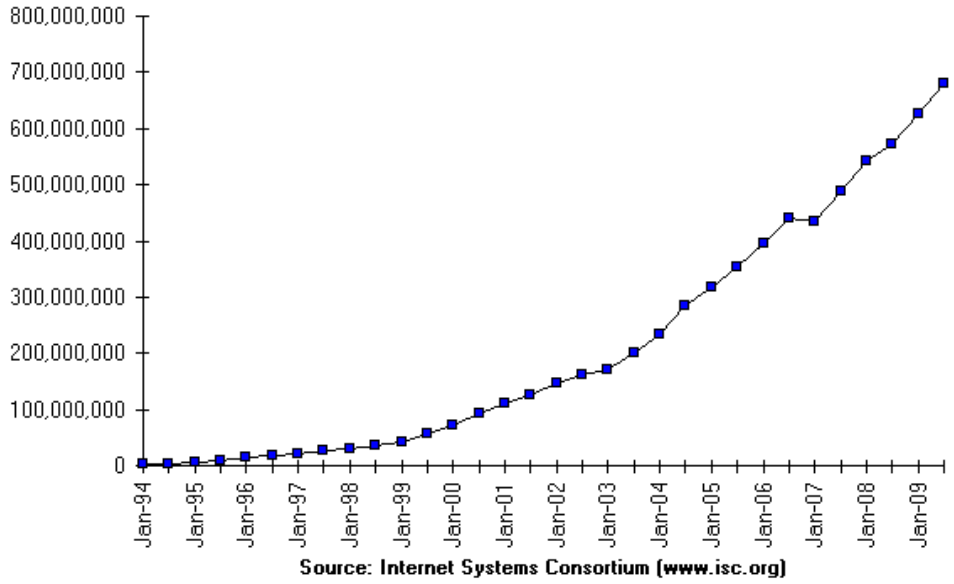

Figure 2.1: Growth of Internet hosts [9]

IPv4 addresses were almost all being used and new address space was required. As the Figure 2.1 illustrate, the number of hosts connected to Internet and connected domains are growing at exponential rates, with each of those hosts needing a unique IP identifier.

The current IPv4 address space can identify a theoretical 4.2 billion hosts (2^{32}), what imposes some additional constrains to its structure. In the other hand, IPv6 has increased it from 32 bits to 128 bits per address. There are other related problems, such as the limited addressing hierarchy that is possible within the confines of the 32-bit IPv4 address, plus the associated limitations on routing function scaling.

Besides those problems, IPv6 has added more functionalities and has improved some the extensions and other issues, such as new mobility extensions, security or real-time traffic handling. Also addresses can be autoconfigurated, and multicast routing has got better.

The Internet Protocol version 6 effort is dynamic and is being driven by worldwide sites currently implementing and testing its functionality. Multiple Request for Comments (RFC) and Internet Drafts have been written to support those interested in joining the testing effort.

2.2 The TCP/IP stack

The IPv6 is an Internet protocol that takes part of the TCP/IP stack model, which means Transmission Control Protocol/Internet Protocol. It describes a set of protocols to enable computers to communicate over a network, sending and receiving data. It is based on the notion

of IP addresses and provides end-to-end connectivity specifying how data should be formatted, addressed, transmitted, routed and received at the destination. This concept generates an IP address for each device connected to the Interned network.

This model is constituted by four different layers with their protocols, as we can see in the Figure 2.2.

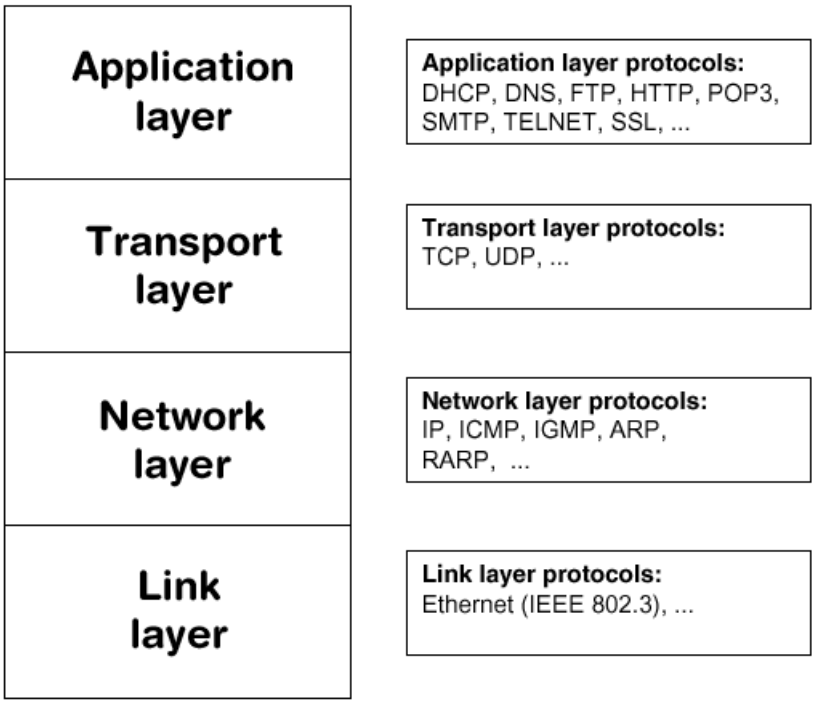

Figure 2.2: TCP/IP stack with some protocols

- **Link Layer**

  The Link Layer is the lowest component layer of the Internet protocols and is used to move packets from a host to another physically connected system placed on the same link. TCP/IP is designed to be hardware independent. As a result TCP/IP has been implemented on top of virtually any hardware networking technology in existence, what allows them to adapt themselves to every new technology. The TCP/IP model includes specifications of translating the network addressing methods used in the Internet Protocol to data link addressing, such as Media Access Control (MAC), however all other aspects below that level are implicitly assumed to exist in the Link Layer, but are not explicitly defined.

- **Network Layer**

  The Network Layer solves the problem of sending packets across one or more networks. It has to analyze the datagrams received in the lower layer to determine if they are addressed or not. If it is the case, it decapsulates the packet header and transmits the rest of the

datagram to the upper layer, but if not, it ignores them. It has the function to route the packets through the network and assures the addressing and the routing of the data. There are some protocols in this layer, which the most important one is the IP protocol. Some of the protocols carried by IP, such as ICMP (used to transmit diagnostic information about IP transmission) and IGMP (used to manage IP Multicast data) are layered on top of IP. The Internet Protocol performs two basic functions:

  - **Host addressing and identification:** This is accomplished with a hierarchical addressing system.
  - **Packet routing:** This is the basic task of getting packets of data from source to destination by sending them to the next network node (router) closer to the final destination.

  In the Section 2.4, IP protocol focused on the new version 6 will be defined in detail.

- **Transport Layer**

  The Transport Layer has to assure the reliability of the transfer and, in this case, it regulates the data flow. Thus, it includes end-to-end message transfer capabilities. This end-to-end message transmission or connecting applications at the transport layer can be categorized as either connection-oriented, implemented in Transmission Control Protocol (TCP), or connectionless, implemented in User Datagram Protocol (UDP).

- **Application Layer**

  The Application Layer refers to the higher-level protocols used by most applications for network communication. Examples of application layer protocols include the File Transfer Protocol (FTP) and the Simple Mail Transfer Protocol (SMTP).

2.2.1 Functionality

When a host wants to send a message to an other one, the data flow starts at Application Layer and goes over the different layers of the transmitter until the Link layer. In every level, a protocol-specific header is added to the data flow. The Figure 2.3 shows an illustrated description of this process, called encapsulation. Then, the frame goes through the network and arrives at its destination. Now, it goes up from the lowest layer, Link Layer, to the Application one. In each layer, the header is first read before being removed, what is called decapsulation, and then the remaining data is sent to the upper layers. At the end, the message becomes the original form in the Application Layer of the receiver host. This process is illustrated in the Figure 2.4.

The IP is the core protocol and has an important rule in the process because it allows the fragmentation and the transport of packets through two hosts. Nowadays there are two IP protocol versions in use (IPv4 and IPv6), where the second is now being implemented in devices more frequently.

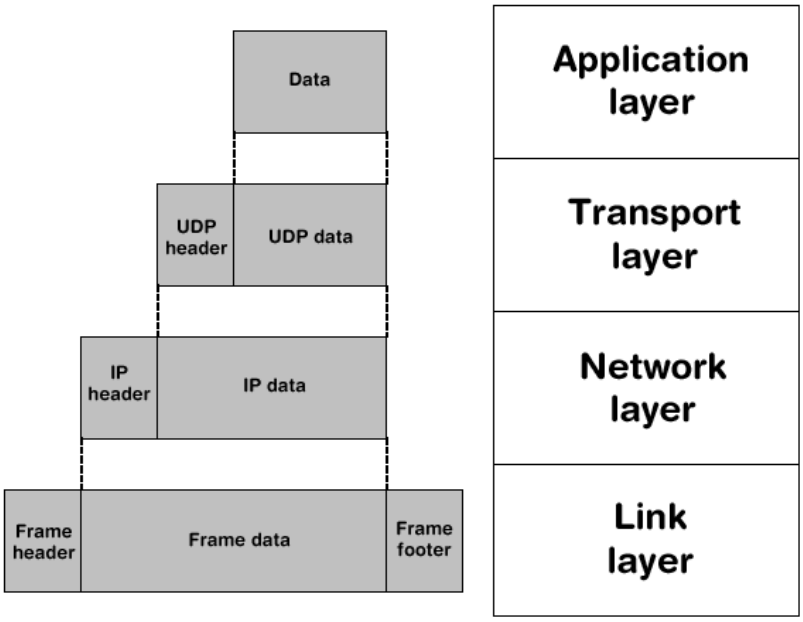

Figure 2.3: Example of data encapsulation within an UDP datagram

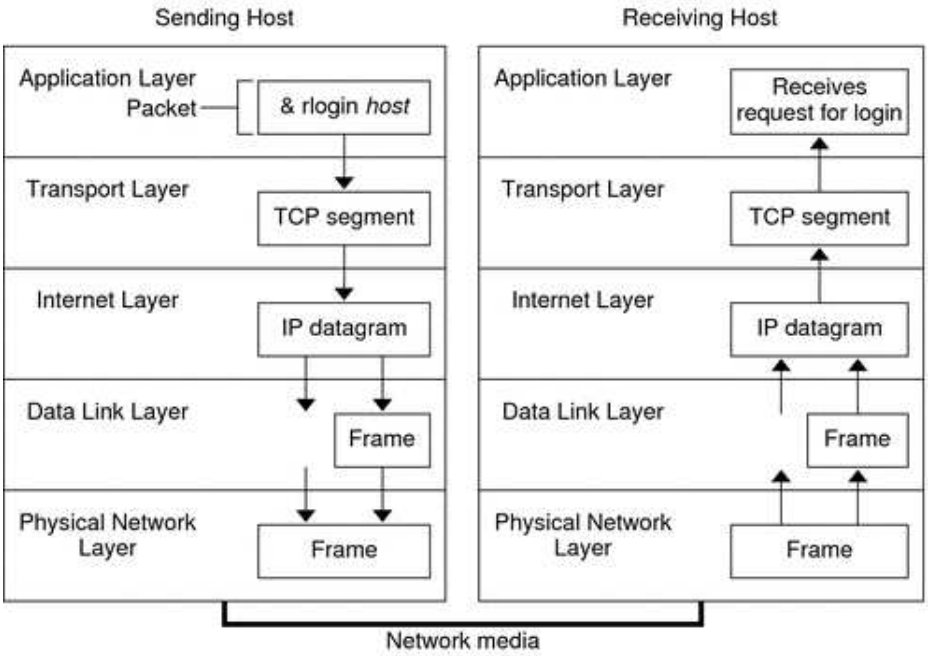

Figure 2.4: IP encapsulation and decapsulation process

2.3 Differences between IPv4 and IPv6

As the project's devices and systems will be implemented with the IPv6 protocol and its extensions, we will only define this IP version. But to understand the needs for a new Internet

Protocol and the consequent improvements from its predecessor, the most significant functional differences and improvements between IPv4 and the new IPv6 is described next.

- **Expanded the address space:** with IPv6, the IP address field increased from 32 bits to 128 bits in length, which allows more numbers of addressable nodes, more levels of addressing hierarchy, defining new types of addresses, etc.

- **Header Packet Format Simplification:** to reduce packet handling overhead, the new protocol eliminates some of the header fields IPv4 packets had. So, the IPv6 header processing overhead is reduced, what compensates for the larger addresses. Four fields of IPv4 have been removed from the header: Header Length (as it is fixed now), Identification Flags and Fragment Offset (which are now in the Fragment Extension Header) and the Header Checksum.

- **Extension headers and options:** Now, with IPv6 there are extensions headers that follow separately the main IPv6 header and which carry optional information, not like IPv4 whose headers are always compulsory. So, it permits not to carry so much unnecessarily information, and processing at every intermediate stop between source and destination may not be required. Those Extensions Headers are placed between the IP base header and the upper layer header. In this way, these optional headers are only added when required by a specific protocol function, such as fragmentation or packet routing.

- **Authentication and Privacy:** There are now extensions to support the authentication of the sender of a packet, data integrity, and optional data confidentiality.

- **Autoreconfiguration:** The IPv6 supports from node address assignments up to the use of the Dynamic Host Reconfiguration Protocol (DHCP), so routers do not have to configure them for each node in their network.

- **Quality of service capabilities:** A new capability is added to enable the labeling of packets belonging to particular traffic for which the sender has requested special handling, such as nondefault quality of service or real-time service.

Also the difference between the IPv4 and IPv6 architectures is that IPv4 is 32-bit aligned (the word has 32 bits) whereas IPv6 is 64-bit aligned (the word has 64 bits). That is because the original processors that implemented the IPv4 had 32-bits word lengths. IPv6 is built assuming 64-bit word, which accounts for some of the protocol processing improvements that IPv6 implementers are discovering [6].

As we can see in the Figure 2.5, IPv4 packet header contains a minimum of 20 octets of control information with ten fields and two addresses. An option field exists within the header that allows further optional bytes to be added. On the other hand, the IPv6 base header (with no extension headers) is 40 octets long with 8 fields and two addresses.

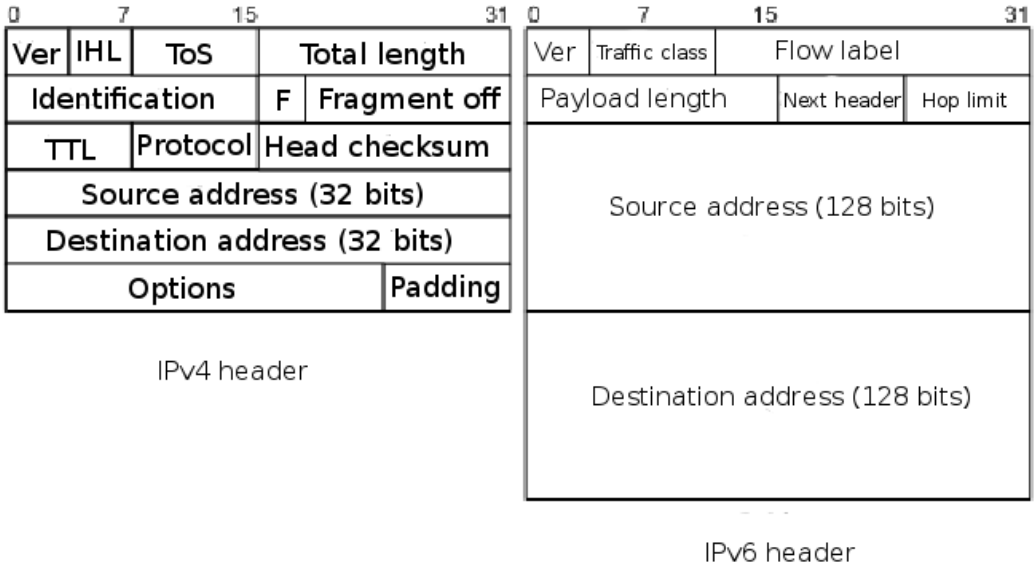

Figure 2.5: IPv4 and IPv6 headers

2.4 Basic IPv6 characteristics

2.4.1 IPv6 header structure

The IPv6 packet is carried within a local network frame like the IPv4 case (Figure 2.6), however the difference with IPv4, is that IPv6 packet consists in 2 parts: **IPv6 base header** and **optional IPv6 extension headers**. As we said before, extended headers are only appended when are needed to he mandatory IPv6 base header. As a consequence more flexibility and efficiency of the header information is obtained. With or without any optional extension headers, a fixed size constraint on the local network frame must be respected (like Ethernet frame). That is the same as IPv4.

The RFC 2460 [10] defines the structure of the IPv6 base header and its extensions headers. The fields that are implemented in the base header, as we can see in Figure 2.5, are:

- **Version field:** is 4 bits long. Identifies the version of the protocol. For IPv6, Version = 6.
- **Class field:** is 8 bits long. Organizes nodes and/or forwarding routers to identify and distinguish between different classes or priorities of IPv6 packets.
- **Flow Label Field:** is 20 bits long, and may be used by a host to request special handling for certain packets. For real-time quality of service or any other nondefault quality of service.
- **Playload Length field:** is 16 bits long. Measures the length, given in octets, of the payload (the balance of the IPv6 packet that follows the IPv6 base header, such any

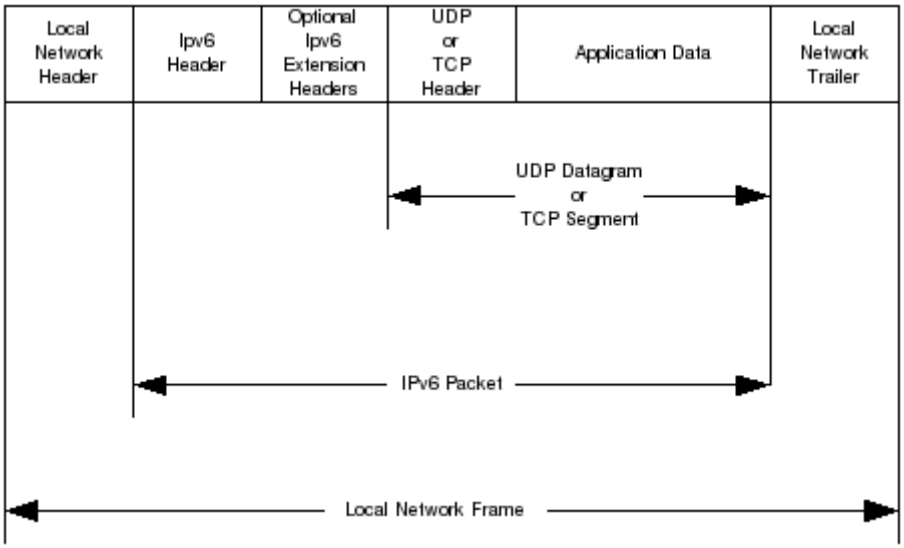

Figure 2.6: Internet transmission frame with IPv6

upper-layer protocols such TCP, FTP...) but not the IPv6 base header, different than the IPv4 that counted also the header. Optional extension IPv6 header is considered part of the payload.

- **Next Header field:** is 8 bits long. It identifies the header immediately following the IPv6 header.
- **Hop Limit field:** is 8 bits long. Is decremented by each node that forwards the packet. When it equals to zero, the packet is discarted and an error message is returned. In IPv6, there is no time basis.
- **Source Address Field:** The source address field is a 128-bit field that identifies the originator of the packet.
- **Destination Address Field:** is a 128-bit field that identifies the receivers address.

2.4.1.1 Extension Header

The IPv6 design simplified the existing IPv4 header by placing many of the existing fields in optional headers. In this way, the processing of ordinary packets is not complicated by excessive overhead, while the more complex conditions are still provided for. As it has been seen, an IPv6 packet, which consists of an IPv6 header plus its payload, may have zero, one or more options extension headers. Each of them is an integral multiple of eight octets in length to retain the eight-octet alignment for subsequent headers. For a better protocol performance, these extension headers are placed in a specific order (Figure 2.7).

The Optional Extension IPv6 headers are:

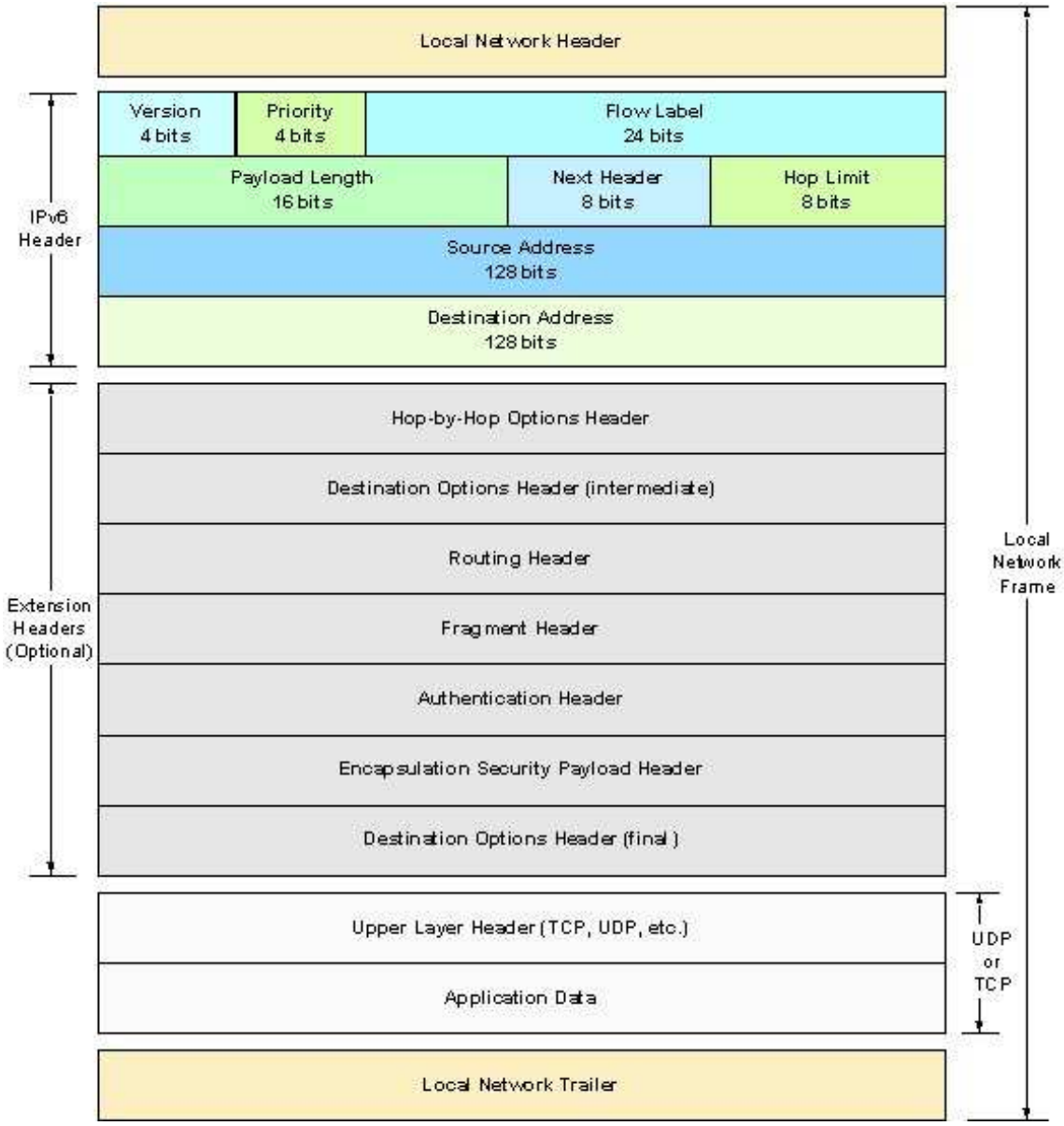

Figure 2.7: IPv6 basic and extension headers

- **Hop-by-Hop Options Header:** This option header carries optional information that must be examined by every node along a packet's delivery path. As a result, the Hop-by-Hop Option Header, when present, must immediately follow the IPv6 header. It is identified by the number 0 in the header's Next Header field.

- **Destination Option Header:** It carries optional information examined only by packet's destination nodes. An option now available is to insert octets of padding into the Option area of header.

- **Routing header:** This header lists one or more intermediate nodes that are "visited" on the path from the source to the destination.
- **Fragment header:** It is used by an IPv6 source to send packets that are larger than would fit in the path Maximum Transmission Unit (MTU) to their destinations. Fragmentation for IPv6 is only done at the source node, not at intermediate routers along the packet's delivery path (different from IPv4).
- **Encapsulating Security Payload Header:** It is designed to provide confidentiality, data origin authentication, connectionless integrity and limited traffic flow confidentiality.
- **No next header:** It indicates that nothing follows that header.

Extension headers may also employ a Next Header field, linking to a subsequent extension header. Figure 2.8 illustrates how the Next Header field locate the contents of the packet for the router to interpret and process.

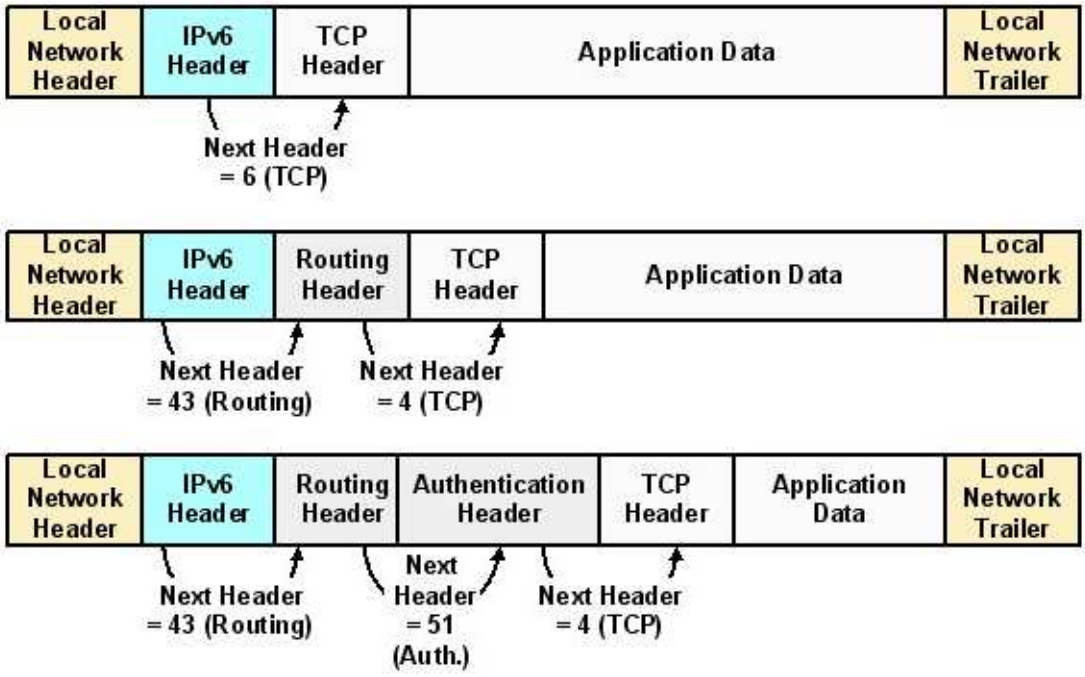

Figure 2.8: Next Header field example

2.4.2 The addressing architecture and representation

There are three different types of addresses, that varies somehow from the IPv4 addresses [11]:

- **Unicast:** is an identifier to a single interface. A packet sent to an unicast address is delivered to the interface identified by that address. There are 3 subcategories of unicast addresses:

  - **Link-Local Addresses:** are designed to be used on a single link for the purposes of address autoconfiguration and neighbour discovery, and for communication between hosts when no routers are present on the link. Packets containing link-local addresses, must never be forwarded by routers or travel outside the local link.
  - **Site-Local Addresses:** may be used by a organization that is not connected to the global Internet. Packets containing site-local addresses must never be forwarded by routers outside of the site in which these addresses are being used.
  - **Globally Routable Addresses:** must be used by nodes which wish to communicate with other nodes both outside of their own link and of their own site. A packet destined to a globally routable address may be forwarded by routers and may be sent from anywhere.

- **Anycast:** is an identifier for a set of interfaces (typically belonging to different nodes). A packet sent to an anycast address is delivered to one of the interfaces identified by that address (the nearest one, according to the routing protocol's measure of distance).
- **Multicast:** is an identifier for a set of interfaces. A packet sent to a multicast address is delivered to all interfaces identified by that address.

The main difference from IPv4 is that now the broadcast function is done by the multicast definition. Another difference is that in IPv6, addresses of all types are assigned to interfaces, not nodes (one node may have multiple interfaces).

2.4.2.1 Address representation

The format of the IPv6 addresses consists of three parts: an Address Prefix, a Site Identifier and a Host Identifier. IPv6 addresses are 128 bits long, represented by groups of 16 bits defined in hexadecimal and separated by colons:

$$x : x : x : x : x : x : x : x$$

where each x represents 16 bits. Each of those 16 bits sections is defined in hexadecimal.

An example could be:

$$2312 : 6453 : 3045 : 5500 : 6368 : 9463 : 0023 : 3344$$

If the address has groups of zeros, they can be eliminated:

$$1932 : 0000 : 0000 : 0000 : 4324 : 0430 : 0032 : 5543$$

$$1932 :: 4324 : 430 : 32 : 5543$$

The double colon is restricted to appearing only once in an address. In text, it is common to represent them by address and prefix length:

$$IPv6address/prefixlength$$

$$1234:0000:0454:6582:4332:4323:0000:0000/64$$

In that case, the prefix is: 1234000004546582

2.4.2.2 Addressing architecture

To provide maximum flexibility for both current and future address representation, the address may be divided into a number of subfields. The leading bits called the Format Prefix, define the specific type of IPv6 address. A multicast address begins with the binary value 11111111; any other prefix identifies a unicast address. Anycast addresses are part of the allocation for unicast addresses and are not given a unique identifier. The Figure 2.9 shows the list of different address prefixes.

2.4.2.3 Unicast addresses

There are many forms for IPv6 unicast addresses. The most simple form is a unicast address with no internal structure, with no address-defined hierarchy. The other possibility is to specify a Subnet Prefix within the 128-bit address, thus dividing the address into a Subnet Prefix (with n bits) and an interface ID (with 128-n bits)(Figure 2.10).

The address 0:0:0:0:0:0:0:0 is defined as the unspecified address, that indicates the absence of an address. It is used on startup when a node has not yet an address assigned. The address 0:0:0:0:0:0:0:1 is defined as the loopback address. It is used by a node to send a packet to itself.

2.4.2.4 Local Use Addresses

- The link-local address is used for a single link and is intended for auto-address configuration, neighbour discovery or for communication between hosts when no routers are present on the link. The link-local address begins with the Format Prefix 1111111010 and includes a 64-bit interface ID field. Routers never forward packets with link-local source or destination addresses to other links (Figure 2.11).
- The site-local address is used by organizations that have not yet connected to internet. Routers never forward packets with site-Local source addresses outside of that site (Figure 2.12).

2.4.2.5 Anycast addresses

An anycast address is one that is assigned to multiple interfaces, typically on different nodes. A packet with an anycast destination address is routed to the nearest interface that has that address. It is used to identify a set of routers attached to a particular subnet or for identifying a set of routers that provide entry to a particular routing domain. Those addresses must not be used as a source address for an IPv6 packet and may only be assigned to routers, not hosts (Figure 2.13).

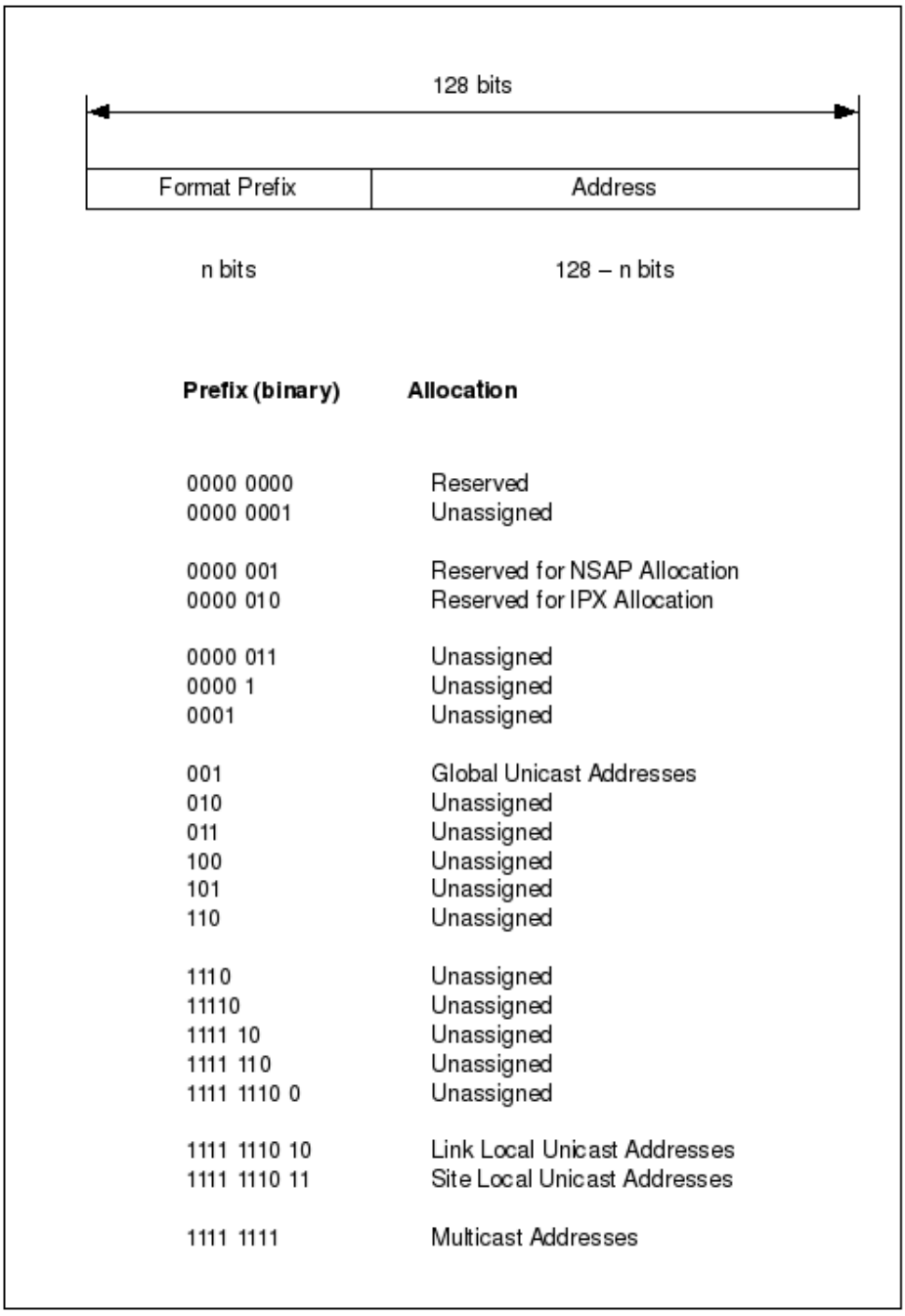

Figure 2.9: Addressing Architecture

2.4.2.6 Multicast addresses

The multicast address identifies a group of nodes and each of these nodes may belong to multiple multicast groups. The multicast address begins with the format prefix 1111111 and includes

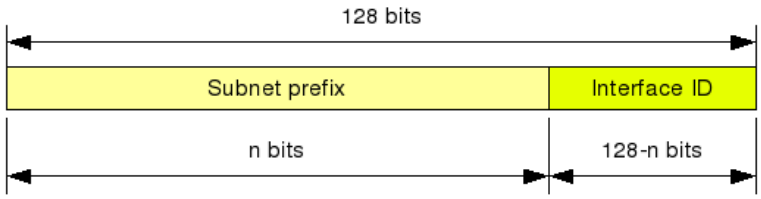

Figure 2.10: Unicast Addresses

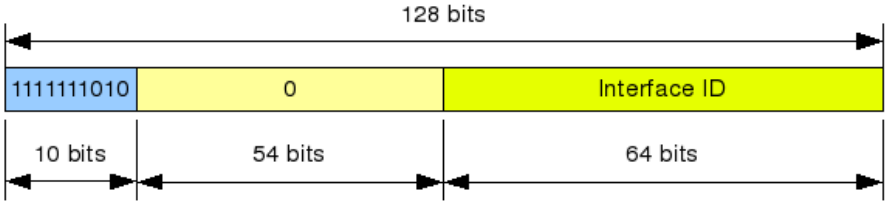

Figure 2.11: Link-local Addresses

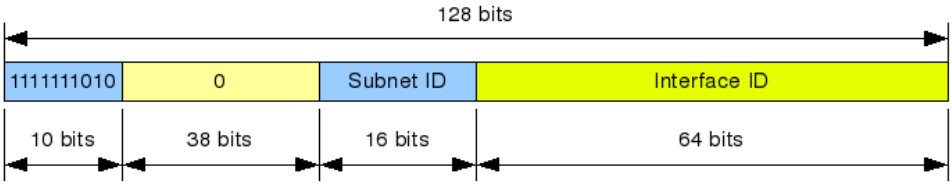

Figure 2.12: Site-local Addresses

three additional fields. The Flags field contains four one-bit flag. The Scop field is a four-bit field that is used to limit the scope of the multicast group. The Group ID field identifies the multicast group, either permanent or transient, within the given scope. Multicast addresses may not be used as source addresses in IPv6 datagrams or appear in any routing header (Figure 2.14).

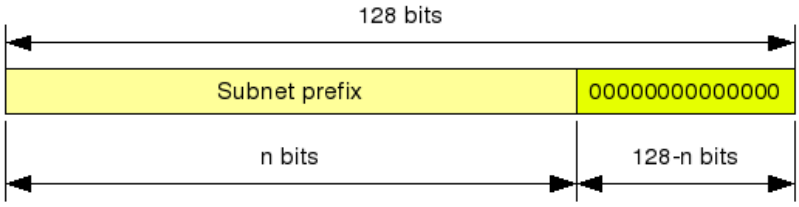

Figure 2.13: Anycast Addresses

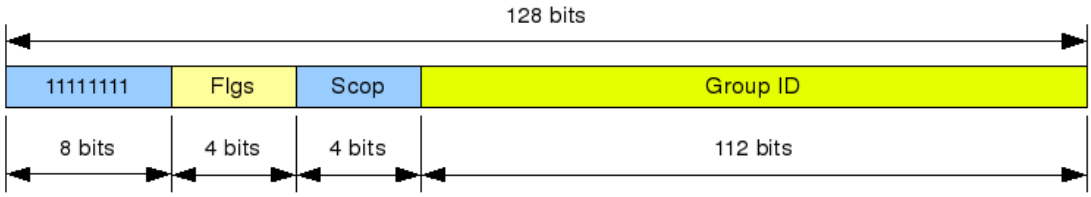

Figure 2.14: Multicast Addresses

2.5 Intranetwork communication: ICMPv6

The ICMPv6 messages (Internet Control Message Protocol) are used to report packet processing errors, intranetwork communication path diagnosis and multicast membership reporting. They are contained within IP datagrams, with the IP header preceding the ICMP message and ICMP data.

The new ICMP version 6 is an evolution of the ICMPv4, where the main functionalities of the old one have been kept and obsolete messages have been removed for simplification. In the RFC 1885, the protocol is defined [12].

ICMPv6 packets have three fields that are common to all messages: Type, Code and Checksum, plus a variable length message body whose contents depend on the type of the message being transmitted (Figure 2.15). The messages are grouped into 2 categories: **Error Messages** and **Informational Messages**. These two categories are identified by the high-order bit of the message Type field.

- The **Type** field is a 8-bit long which indicates the type of message. This field defines the ICMPv6 Message as either an error message or an informational message. A *0* in the high order bit of this field indicates that the message is an error message. A *1* in the high order bit indicates that the message is an informational message. In this fashion,

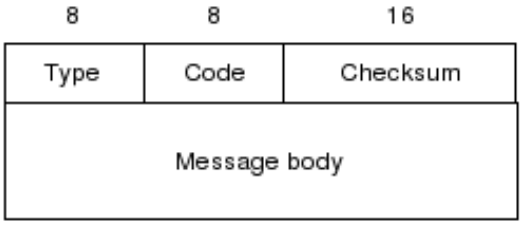

Figure 2.15: ICMPv6 packet structure

type field decimal values from 0 to 127 are error messages, and values from 128 to 255 are informational messages.

- The **Code** is a 8-bit field which creates an additional level of message.
- The **Checksum** is a 16-bit field used to detect errors in the ICMP message.

The Error Messages belong to 4 categories: Destination Unreachable, Packet Too Big, Time Exceeded and Parameter Problem. In the other hand, there exist eleven Informational Messages. The Figure 2.16 lists all error and informational messages with their respective Type value.

Type	Meaning
1	Destination Unreachable
2	Packet Too Big
3	Time Exceeded
4	Parameter Problem
128	Echo Request
129	Echo Reply
130	Group Membership Query
131	Group Membership Report
132	Group Membership Reduction
133	Router Solicitation
134	Router Advertisement
135	Neighbor Solicitation
136	Neighbor Advertisement
137	Redirect
138	Router Renumbering

Figure 2.16: ICMPv6 Error and Informational messages[13]

2.6 Autoconfiguration

Unlike IPv4, IPv6 protocol has two methods for obtaining addresses: the *Stateless* mechanism and a *Statefull* mechanism.

2.6.1 Statefull Automatic Configuration

In a Statefull Autoconfiguration model, hosts obtain information of addresses configuration, parameters, etc, from a server. That server maintains a database containing the necessary information and keeps tight control over the address assignment. The Statefull Autoconfiguration model for IPv6 is defined by the proposed Dynamic Host Configuration Protocol for IPv6 (DHCPv6) like IPv4 [14]. This configuration mechanism is based on the version used for IPv4.

2.6.2 Stateless Automatic Configuration

Stateless Automatic Configuration is a new protocol designed for IPv6. This model requires no manual configuration or hosts, minimal configuration of routers and no additional servers. The stateless approach is used when a site is not concerned about the specific addresses that are used, as long as they are unique and routable.

With Stateless Autoconfiguration, a host generates its own address derived from the MAC address of the network interface, and using a specific algorithm. In IPv6, the first 64 bits correspond to the subnet prefix, and the last 64 bits are the local identifier. This local identifier will be built using the 48 bits-MAC address.

But Stateless Automatic Configuration can also be used to create global IPv6 addresses. This requires the use of ICMPv6 Router Advertisement messages. In a local network, the routers send Router Advertisement messages, with the ff02::1 destination address, which corresponds to the multicast group. These messages contain the global 64 bits-prefix, that will be used by the hosts of the network to create global addresses, appending their local identifier suffix. Because Router Advertisement messages are usually sent every 10 seconds, the hosts can stimulate the sending of Router Advertisements, using Router Solicitation messages, sent to the ff02::2 (all-router multicast group) destination address.

2.7 Transition mechanisms between IPv4 and IPv6

As it was said before, nowadays there are many devices and networks that have IPv6 protocol implemented, but IPv4 is still present everywhere. Thus, some mechanisms are needed to permit the interaction between IPv6 devices with the ones with IPv4 implemented.

In the test-bed of this project, there is a problem concerning that issue: the satellite link used may not be upgraded to IPv6 at the same time as the aircrafts, so only IPv4 is implemented. Then, ways to send version 6 packets over this link should be developed. In this study, a manual tunnel mechanism and a protocol translation are implemented in the test-bed in Linux.

At the time the new IP protocol was designed, a simple and flexible transition between IPv4 and IPv6 was required. By creating IPv6 in IPv4 mechanism, co-existence of the two protocols is possible, so users can create IPv6 links to take advantage of IPv6 while allowing the rest of their local network to upgrade as necessity arises.

There are several kinds of transition mechanism, that can be divided in two groups: the dual stack and the IPv4 link mechanism.

2.7.1 Dual Stack

In this case, a host or router have the two IP protocol versions. Each node has two addresses: an IPv4 and an IPv6 addresses. Then, the node can send and receive messages belonging either one protocol or the other.

This is the easiest method, but also the most expensive as it implies providing complete implementations of both versions of the Internet Protocol. Also, as packets are not modified nor added a header, the processing time and the bandwith used are less than other solutions.

2.7.2 IPv4 Link

There are two solutions in that case:

- **Tunneling mechanisms**
- **Translation mechanisms**

2.7.2.1 Tunneling mechanisms

Tunneling is a process where information from one protocol is encapsulated inside the frame or packet of another one, thus enabling the original data to be carried over that second architecture. Thus the tunneling mechanism for IPv4/IPv6 is designated to enable an IPv4 infrastructure to carry IPv6 packets by encapsulating the IPv6 information inside IPv4 datagrams (see Figure 2.17).

Examples of tunneling mechanisms are the 6to4 tunnel, 6over4 tunnel and L2TP tunnel. The L2TP tunnel mechanism is described next as it is the one used in the test-bed.

2.7.2.1.1 L2TP tunnel

The Layer 2 Tunneling Protocol is a tunneling protocol used to support Virtual Private Networks (VPN). Its name comes from the fact that L2TP allows to carry Point-to-Point Protocol (PPP) frames over an IP network, between two end points. It was published in 1999, as proposed standard RFC 2661 [15]. Originally, it was developed in order to bring together the functionalities of two older tunneling protocols for PPP : Cisco's Layer 2 Forwarding (L2F) and Microsoft's

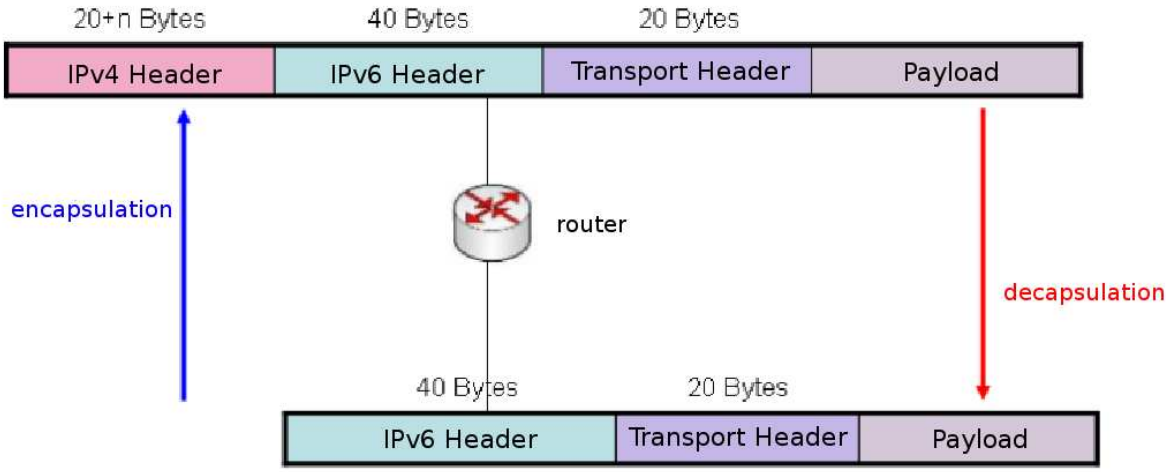

Figure 2.17: The tunneling mechanism

Point-to-Point Tunneling Protocol (PPTP). A new version of this protocol, L2TPv3, provides additional security features, improved encapsulation, and the ability to carry data links other than simply PPP over an IP network (Frame Relay, Ethernet, ATM, etc...). But L2TP is actually a session layer (layer 5) protocol, and uses the registered UDP port 1701.

The overhead induced by L2TP reaches 37 bytes, which is quite a big overhead. Moreover, L2TP does not provide confidentiality or strong authentication by itself. That is why IPsec (Internet Protocol Security) is often used to secure L2TP packets by providing confidentiality, authentication and integrity. But this makes the overhead even bigger.

The two endpoints of an L2TP tunnel are called the LAC (L2TP Access Concentrator) and the LNS (L2TP Network Server). The LAC is the initiator of the tunnel, whereas the LNS is the server which waits for tunnel requests. The necessary setup for tunneling a PPP session with L2TP consists of two steps:

1. The LAC sends a request to the LNS in order to create an L2TP Tunnel. Then, a Control Connection Establishment signaling is performed, and the L2TP tunnel is set up.

2. Once the tunnel is established, the network traffic between the two peers is bidirectional. In order to carry PPP frames through the tunnel, either the LAC or the LNS has to initiate a session, making a call through the tunnel. The traffic for each session is isolated by L2TP, so it is possible to establish multiple sessions across a single tunnel.

The packets exchanged within an L2TP tunnel are categorized as either control packets or data packets. L2TP provides reliability features for the control packets, but no reliability for data packets. Reliability for data packets, if desired, must be provided by the protocols running within the session of the L2TP tunnel.

2.7.2.2 Translation mechanism

By the time I was in TriaGnoSys, another mechanism was studied to improve the benefits of the L2TP mechanism or just to have another way to traverse IPv6 packets through IPv4 links. The translation mechanism used in the test-bed is called NAPT-PT.

NAPT-PT mechanism stands for Network Address Port Translation - Protocol Translation. This is a transition mechanism that allows IPv6 hosts to communicate with IPv4 hosts. It converts the IPv6 headers to IPv4 headers and vice versa, translating IPv6 addresses into IPv4 addresses and vice versa, and also using port translation, for more flexibility [16].

NAPT-PT is actually an improvement of NAT-PT (Network Address Translation - Protocol Translation). NAT-PT only converts IPv6 addresses to IPv4 addresses, but it doesn't change the TCP/UDP ports. Therefore, NAT-PT needs as many IPv4 addresses as there are IPv6 addresses. That is, once the IPv4 address pool affected to the NAT-PT entity is exhausted, newer IPv6 nodes cannot establish sessions with the outside world anymore. NAPT-PT, on the other hand, will allow for a maximum of 63K TCP and 63K UDP sessions per IPv4 address before having no TCP and UDP ports left to assign, thanks to the port translation mechanism. The limitations of NAPT-PT include well-known NAT limitations. For example, applications that carry the IP address in the higher layers will not work, because of the address translation. Moreover, end-to-end network layer security is not possible with NAPT-PT.

The main advantage of NAPT-PT compared to tunneling like L2TP is the absence of additional overhead. For this reason, this protocol has also been implemented in the test-bed after the first solution.

Chapter 3

Mobility scenario

Mobile computing has greatly increased in popularity over the past several years due to the rise in the number of portable computers and other mobile devices like personal laptops, PDA, mobile phones, sensor networks in vehicles... It is getting widespread, which explains that in recent years new systems have been developed to make possible to have continuous network connectivity to the Internet, irrespective of the physical location of the node.

Since mobility and ease of connection are crucial considerations for mobile device users, organizations that want to promote mobile communications are putting a great deal of effort into making mobile connection and uncomplicated for the user. They have developed or proposed several standards to address these needs, including Mobile IP and later enhancements, Mobile IP version 6 (MIPv6).

The IETF started to design a new mobile extension protocol for IPv6 that allows nodes and entire networks to change their point of attachment in order to continue communication in spite of its movement, without having to reboot their ongoing connections through seamless handovers, with small delays and with minimal loss of packets.

In this chapter it is explained why mobile nodes and networks are needed, and the problems encountered to implement them with old IP protocols. Next, it is described the new mobile IPv6 extension protocol for mobile nodes as well as the extension for network mobility called NEMO (NEtwork MObility). Those extensions will be the ones implemented in our test-bed, which at the end, we will study the results.

3.1 Introduction

Speaking in terms of computational communications, we define mobility as the ability of a node to change its point of attachment to a network from one link to another while maintaining all existing communications.

Initially, IP protocols were not designed for mobile environments, so when mobile applications were needed, many problems started to appear. Nowadays, different mechanisms are designed to solve them but many are not efficient enough or are not available anymore.

The best solution since now is the Mobile IP, which provides an efficient, scalable mechanism for roaming within the Internet. Using Mobile IP, nodes may change their point of attachment to the Internet without changing their home IP address. Node mobility is realized without the need to propagate host-specific routes throughout the Internet routing fabric.

Mobile IPv6 is an IETF standard that has added the roaming capabilities of mobile nodes in IPv6 network [17].

3.2 The mobility problem and first solutions

IP addresses play two different roles. On one hand, they are locators that specify how to reach the node that is using that address. The routing system keeps information about how to reach different sets of addresses that have a common network prefix. This address aggregation in the routing system satisfies scalability requirements. On the other hand, IP addresses are also part of the endpoint identifiers of a communication, and upper layers use the identifiers of the peers of a communication to identify them. For example, the Transmission Control Protocol (TCP), which is used to support most of the Internet applications, uses the IP address as part of the TCP connection identifier.

This dual role played by IP addresses imposes some restrictions on mobility, because when a terminal moves from one network (IP subnet) to another, we would like to maintain the IP address of the node that moves (associated to one of its network interfaces) in order not to change the identifier that upper layers are using in their ongoing sessions. However, we also would like to change the IP address to make it topologically correct in the new location of the terminal, allowing in this way the routing system to reach the terminal [18].

3.2.1 The solutions

Before the apparition of the Mobile IP protocol, some possible solutions were designed to support mobility on mobile nodes, but as was said before they have several lacks:

- **Host-Specific Routes:** this solution defines a directly route of the mobile node's position in the new network it is attached, without changing its IP address. But, it is not the best way and is an unworkable solution for Internet routing and in general for mobility because minimally, these host-specific routes must be propagated to all nodes along the path between a mobile node's home link and its foreign link. Some of these routes must be updated every time the node moves from one link to another and we expect millions of mobile nodes to be operating in the Internet. Also, there are serious security implications to using host-specific routes to accomplish node mobility in the Internet, which would require authentication and complicated key management protocols to address. Thus, host-specific routing has severe scaling, robustness and security problems which make it an unacceptable solution to node mobility in the global Internet.

- **Change the node's IP address:** A TCP connection is one of the most used transport-layer protocol in the Internet. This connection within a node is uniquely identified by the following four values: IP Source Address, IP Destination address, TCP source Port and TCP destination port. To a correct transmission, these four values must remain constant over the session of a TCP connection. If we use this solution, any host would simply drop its connections to a destination node whose IP address was to change, because as we said, this address must remain constant. Thus, all ongoing communications between a mobile node and any of these existing nodes would have to be finished, with new connections being initiated by the mobile node at its new address. So changing a mobile node's address as it moves does not solve the problem of node mobility.
- **Solve it at he link-layer:** There are indeed link-layers solutions to node movement that have been devised for use with Internet-related protocols like CDPD (Cellular Digital Packet Data) or IEEE 802.11. But they aren't sufficient solution to provide node mobility on the global Internet. First of all, link-layer solutions provide node mobility only in the context of a single type of medium. Another problem with link-layer solutions is that they need n different mobility solutions for each of n possible media over which nodes might want to send IP packets. A single solution which works over all media types is to be preferred over multiple medium-specific solutions, if such a thing is architecturally possible. Mobile IP is such a solution. Finally, link-layer solutions provide mobility within a limited geographic area and are unusable once the node leaves this area.

Mobile IP is unique in its ability to provide mobility over all types of media and therefore through an arbitrarily large geographic area. Using Mobile IP, a node can communicate using a fixed IP address wherever it can obtain a connection to the network.

3.3 Mobile IP

Mobile IP is a network-layer solution for mobility on the global Internet which is scalable, robust, secure and which allows nodes to maintain all ongoing communications while changing links. It is a mechanism for routing IP packets to mobile nodes which may be connected to any link while using their permanent IP address. Using this extension protocol, nodes are capable to change their point of attachment to the Internet without changing their home IP address. This allows them to maintain transport and higher-layer connections while roaming.

When a source computer wants to send a packet to a destination computer, the source does not know or care where the destination is located. It just wants its packets to be delivered to the proper recipient. This is the function of the network layer of the TCP/IP stack. The network layer is responsible for dynamically selecting a path from the original source of a packet to its destination.

3.3.1 Requirements for Mobile IP

The requirements which drove the design of Mobile IP are:

1. A mobile node must be able to communicate with other nodes after changing its link-layer point of attachment to the Internet.

2. A mobile node must be able to communicate with other nodes using only its home (permanent) IP address, regardless of its new current link-layer point of attachment to the Internet.

3. A mobile node must be able to communicate with other computers that do not implement the Mobile IP mobility functions.

4. A mobile node must not be exposed to any new security threats over and above those to which any fixed node on the Internet is exposed.

5. This new protocol may not change the existing fixed hosts and routers that don't need mobility mechanism. So it is required the Mobile IP implementation to be limited only to the mobile nodes themselves and the few nodes which provide special routing functions.

6. Mobile IP requires the transmission of routing updates between the various nodes in the network. In order to make this extension suitable for the use over a wide range of wireless links, one of the design goals was to make the size and the frequency of these updates as small as possible.

All Mobile IP requires an infrastructure of routers and links capable of routing packets to any node which is connected to its home link.

3.4 MIPv6

The Mobile IPv6 is the new extension protocol IETF has created in order to implement the mobile node mechanism in IPv6 protocol [17].

3.4.1 The scenario

The Mobile IPv6 protocol is used in nodes moving from between different networks. The usual process done during handovers from one link to another is described next, but to understand better the situation, an illustrated example pictured in Figure 3.1 and a previous vocabulary is exposed before all.

The different entities involved in Mobile IPv6 are:

- **Mobile Node (MN):** is the node that is moving through different networks and which can change its point of attachment while maintaining any ongoing communication and using only its IP Home Address. Home Address is the original IP address and signifies that the mobile node is logically connected to the home link. The Mobile Node is aware of mobility, which means that a specific Mobile IP software has to be run in it.

- **Home Network:** is the network where the Mobile Node is located at the beginning, when it is not moving to a Foreign Network.
- **Home Agent (HA):** usually is a router, which is located in the Home Network and is informed of the Mobile Node's current location. It is responsible for tunneling and addressing the packets to the Mobile Node. The Home Agent is also aware of mobility, so a dedicated Mobile IP software has to be run in it, too.
- **Foreign Network:** is a new network visited by the Mobile Node.
- **Care-of Address (CoA):** is the new IPv6 address that is acquired by the Mobile Node in the Foreign Network and identifies it. The Home Agent has to be informed about this address, in order to reach the Mobile Node when it is in a Foreign Network. A Mobile Node's Care-of Address generally changes every time the Mobile Node moves from one foreign link to another.
- **Access Router (AR):** is the default router for the Mobile Node in the foreign network and assists it in informing its Home Agent of its current Care-of Address. It gives a new Care-of Address to the visiting Mobile Node. The Access Router doesn't have to be aware of mobility. It de-tunnels packets for the Mobile Node that have been tunneled by its Home Agent and it serves as a default router for packets generated by the Mobile Node while connected to this foreign link. The only requirement is that a Router Advertisement Daemon process has to send Router Advertisement messages through the interface corresponding to the Mobile Node's location.
- **Correspondent Node (CN):** is any computer in the Internet that wants to communicate with the Mobile Node. The Correspondent Node normally is not aware of mobility, but with route optimization (explained in Section 3.4.2) it needs to have the mobile software.

3.4.2 Mobile IPv6 operation

As it is illustrated in the Figure 3.1, before any handover process, the Mobile Node is at its home network and it is expected to be addressable at its Home Address (2001:b:1::3 in this example). The Home Address doesn't change and remains permanent despite the Mobile Node moves from link to link. When a Mobile Node is at home, packets addressed to its Home Address are routed through the Home Agent, which has at least one interface on the Mobile Node's home link (in the example, address 2001:b:1::2) and is the one that will take care of the Mobile Node when being out.

Home Agents and Access Routers advertise their presence on any attached link by periodically multicasting or broadcasting special Mobile IP messages called Router Advertisements (RA). Those messages are used for Mobile Nodes to examine their contents and determine whether they are connected to their home link or a foreign link. While connected to their respective home link, Mobile Nodes act just like stationary nodes, so they make use of no other Mobile IP functionality.

With very few exceptions, a Mobile Node communicates with all other nodes using only its Home Address as the IP Source Address and it is also the IP Destination Address of all packets

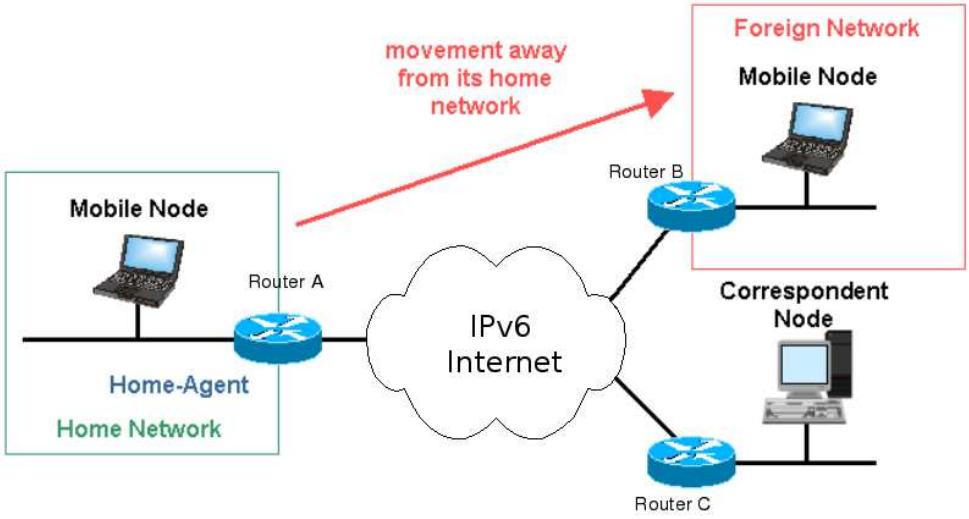

Figure 3.1: Mobile IPv6 scenario

sent to the Mobile Node. Consequently, the Home Address has to be placed in the IP address field of its entry in the Domain Name System, so that other nodes looking up the Mobile Node's hostname will find it.

When the Mobile Node moves from the home network to a foreign link, the following operations are done:

1. The Mobile Node needs to obtain a new address to join the foreign network, but should not loose his Home Address to prevent a disconnection of the current sessions. Then, when a Mobile Node joins to the new link, it gets a new address a well as the Home Address, called Care-of Address (CoA). This new address associated with the moving host while visiting a foreign link, has the foreign subnet prefix (of the foreign network) as its subnet prefix. To get it, the Mobile Node performs an address autoconfiguration, as we will explain in Section 3.4.2.1. This new address is advertised in one of the fields of the Router Advertisement messages sent by the Access Router. As we can see in Figure 3.2, the prefix of the Foreign Network is 2001:3::/64 and the new Care-of Address is 2001:3::230:5ff:fed3:51c8.

2. Consequently, the Mobile Node registers the Care-of Address acquired with its Home Agent, using a message-exchange defined by Mobile IPv6 protocol. In the registration procedure, the Mobile Node asks for service from the Access Router of the new link. In order to prevent remote denial-of-service attacks, the registration messages are required to be authenticated.

The association between a Mobile Node's Home Address and Care-of Address is known as a **binding**. The moving host performs this binding registration by sending a Binding Update (BU) message to the HA as we can see in Figure 3.3. It replies to the Mobile Node by returning

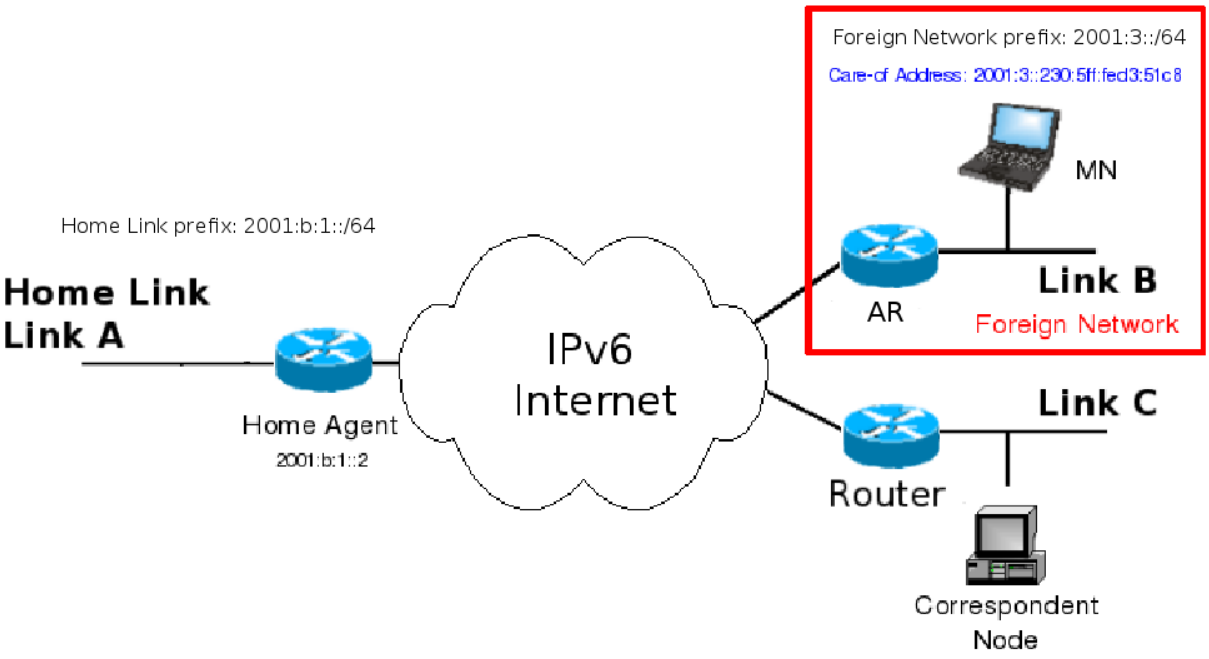

Figure 3.2: Mobile Node visiting a foreign network

a Binding Acknowledgment (BA) message saying that it has registered the Care-of Address in its binding cache. The binding cache is an address entry table used to associate the CoA with the Home Address of every Mobile Node visiting a foreign network. On the other side, the Mobile Node has also a Binding Update list, where it has an item for every binding that the node has or is trying to establish.

Communication between Mobile Node and Correspondent Node

A Correspondent Node (CN) is any foreign network host that wants to communicate with the Mobile Node. When they try to establish a connection, the following operations are done:

1. The CN sends an IP packet addressed to the permanent IP address of the Mobile Node, that is, the Home Address.

2. The Home Agent intercepts the packet as it is sent to the MN's home link and consults the mobility binding cache table to find out if the MN is already visiting any other network or otherwise is at "home".

3. In case that the MN is in a foreign network, the Home Agent finds out the MN's Care-of Address and constructs a new IP header that contains the MN's Care-of Address as the destination IP address. The original IP packet is put into the payload of this IP packet and ready to be sent. This process of encapsulating one IP packet into the payload of another one is known as **tunneling**. In the example being used, the Home Agent would tunnel the IP packet like the way shown in the Figure 3.4

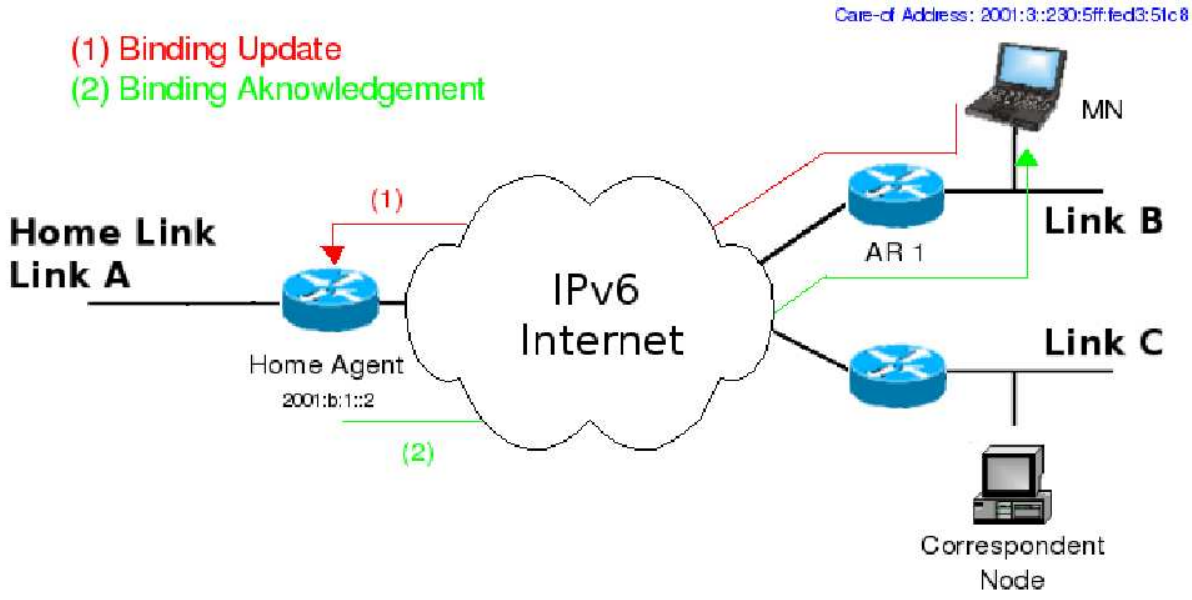

Figure 3.3: The binding operation between Mobile Node and Home Agent

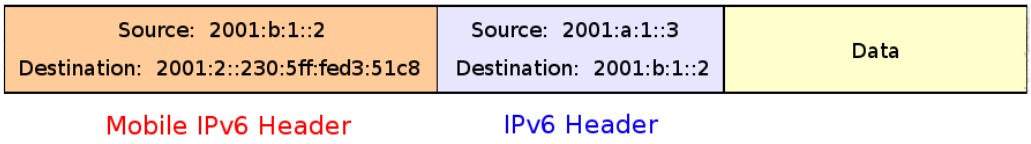

Figure 3.4: IP packet tunneling

4. When the encapsulated packet reaches the Access Router of the foreign network, the original packet is extracted from the tunnel and then delivered to the MN. It means that in both directions, packets are encapsulated in the Home Agent and decapsulated in the Access Router. In the Figure 3.5 we can see the path that a packet sent by the Correspondent Node to the Mobile Router would follow.

There are two possible modes to establish communication between Mobile Node and a Correspondent Node.

- The first mode, **bidirectional tunneling**, does not require Mobile IPv6 support from the CN and is available even if the MN has not registered its current binding with the CN. As we can see in Picture 3.6, packets from the CN are routed to the Home Agent and then tunneled to the MN and, in the other side, packets to the CN are tunneled from MN to the Home Agent ("reverse tunneled") and then routed normally from the home network to the CN. This tunneling is performed using IPv6 encapsulation.

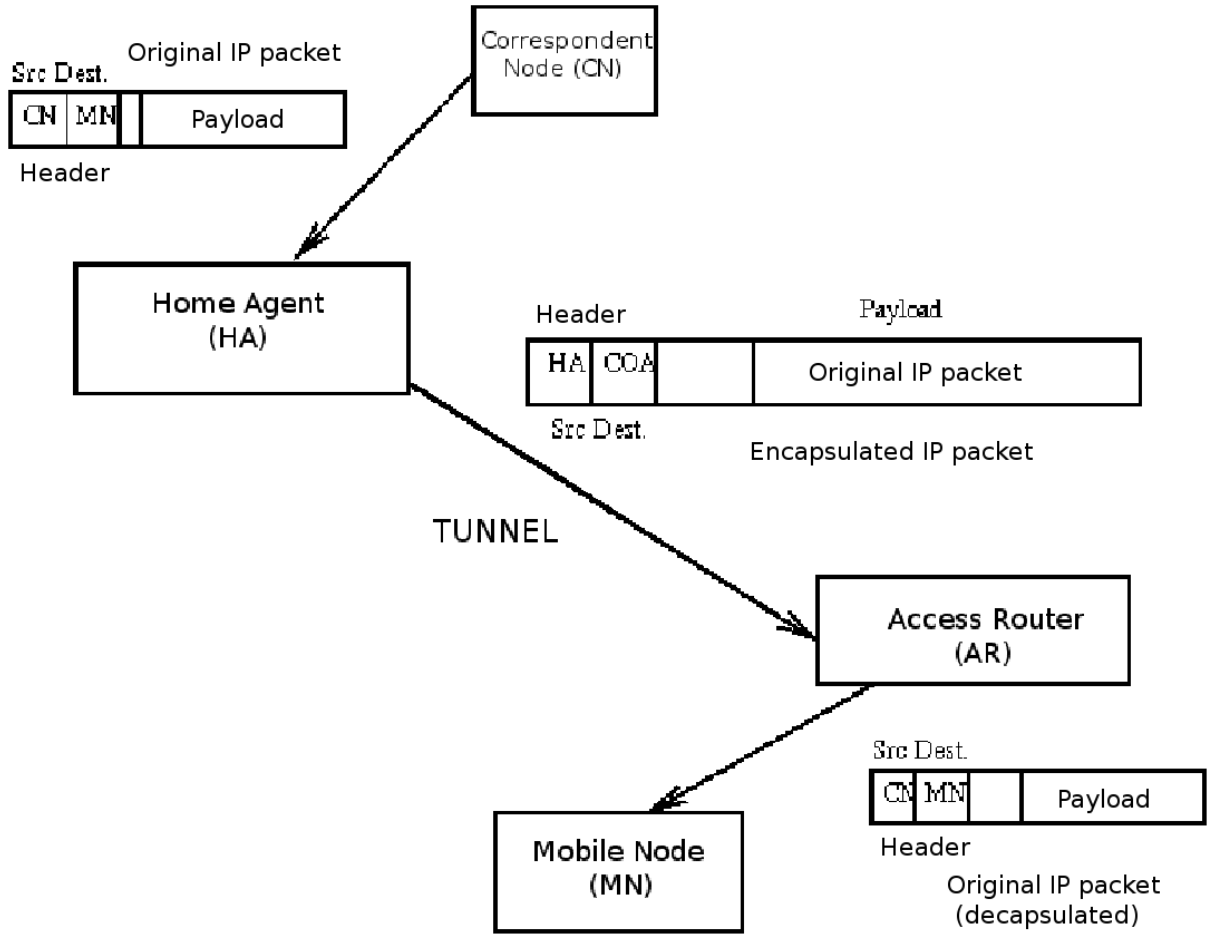

Figure 3.5: IP packet tunneling process in MIPv6

- The second mode, **route optimization**, requires the Mobile Node to register its current binding at the CN. As illustrated in Figure 3.7, packets from CN can be routed directly to the Care-of Address of the MN without visiting the Home Agent. When sending a packet to any IPv6 destination, the CN checks its cached bindings for an entry for the packet's destination address. If a cached binding for this destination address is found, the node uses a new type of IPv6 routing header to route the packet to the Care-of Address indicated in this binding.

  Routing packets directly to the MN's Care-of Address allows the shortest communications path to be used. It also eliminates congestion at the Mobile Node's Home Agent and home link. In addition, the impact of any possible failure of the Home Agent or networks on the path to or from it is reduced. But on the other hand, the CN has to be implemented with Mobile IPv6 technology.

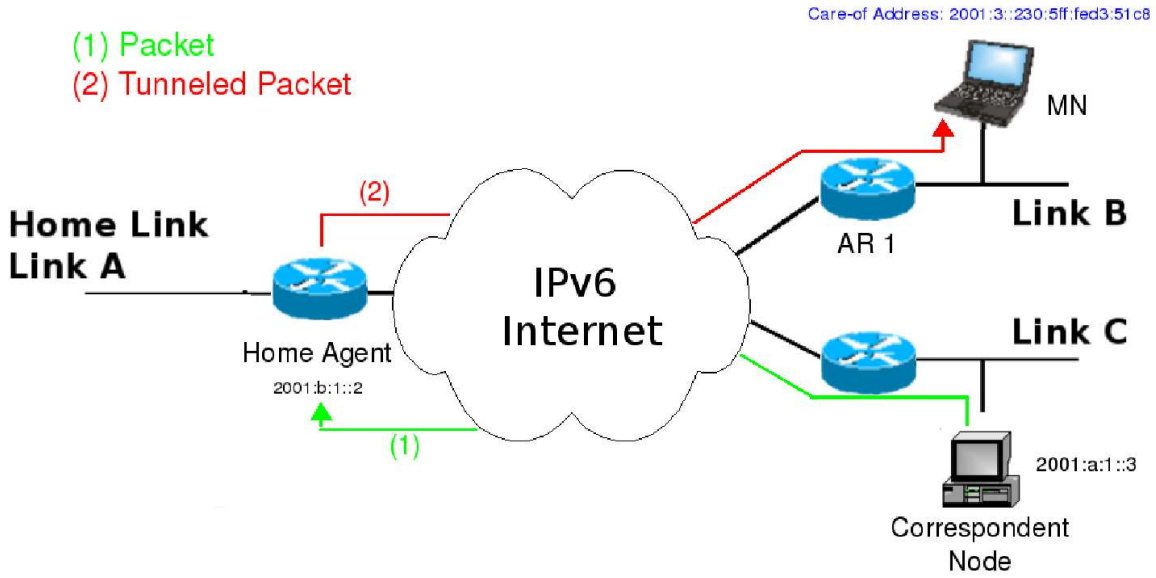

Figure 3.6: Bidirectional tunneling

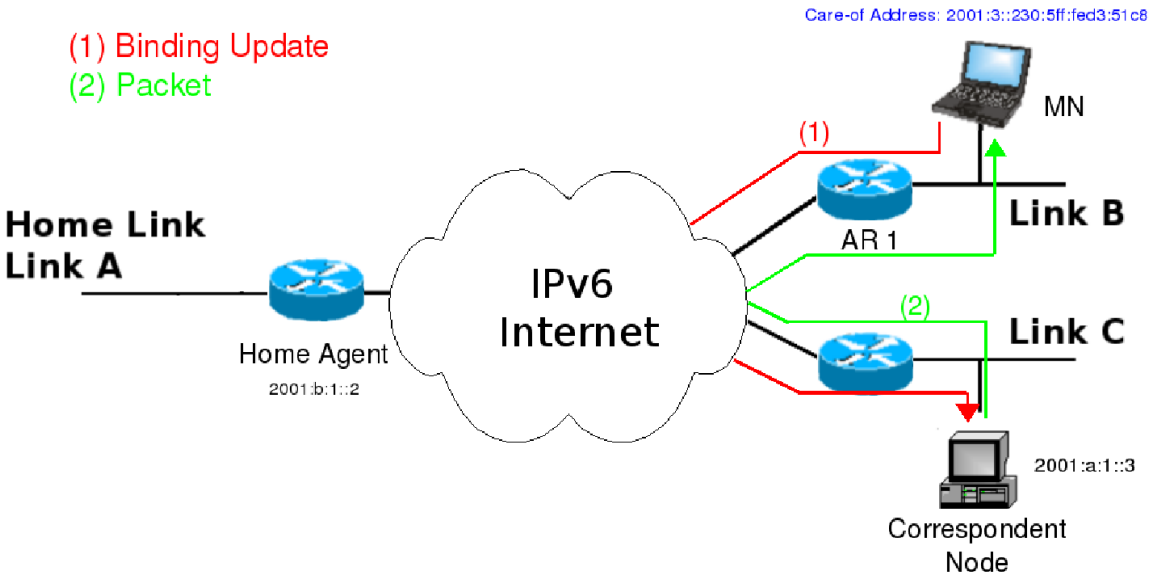

Figure 3.7: Route optimization

3.4.2.1 How does a Mobile Node obtain a Care-of Address?

There are two methods in order to acquire a Care-of Address: the Stateful and the Statless Address Autoconfiguration. The Router Advertisements sent by the Access Router and received by the Mobile Node, carries the information that determine which method should be used [19].

Stateful Address Autoconfiguration

In this method, the Mobile Node simply asks a server for an address and uses it as a Care-of Address. In that case, it also uses the protocol for stateful address assignment for normal IPv6 protocol: the Dynamic Host Configuration Protocol for IPv6 (DHCPv6).

Stateless address autoconfiguration

Mobile nodes can also acquire a Care-of Address by Stateless Address Autoconfiguration. It works as follows:

- The Mobile Node first forms a link-dependent identifier for the interface by which it connects to the foreign link. This identifier is typically the node's link-layer address on that interface. For example, on Ethernet, the identifier would be the Mobile Node's 48-bit Ethernet address.
- The Mobile Node examines the Prefix Information Options that are contained within Router Advertisements to determine the valid network-prefixes on the current link.
- The Mobile Node forms a care-of address by concatenating one of the valid network-prefixes with the identifier. Address autoconfiguration (stateful and stateless ones) contains mechanisms by which a node can determine whether the address it has acquired is identical to an address being used by any other node on the link. If there is such a duplicate address, then the autoconfiguration protocols define ways in which a unique address can be acquired by the node.

3.4.3 Basic MIPv6 characteristics

3.4.3.1 The Mobility Header

The new feature that Mobile IPv6 brings to the IPv6 protocol architecture is the Mobility Header. It is a new IPv6 extension header designed to contain the MIPv6 signaling messages which makes possible the mobility mechanism. The Mobility Header is used by Mobile Nodes, Home Agents and Correspondent Nodes (in case of route optimization) for all messaging related on binding creation and management. The format of this header is illustrated in the Figure 3.8.

The fields are:

- **Next Header:** 8-bit selector. Identifies the type of header immediately following the Mobility Header. Uses the same values as the IPv6 Next Header field.

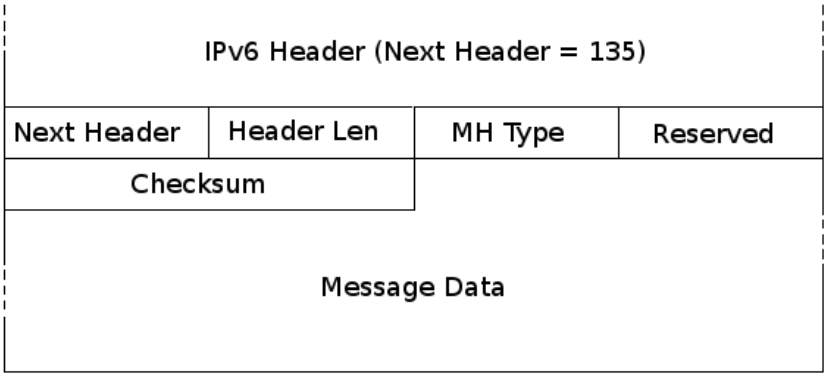

Figure 3.8: The Mobility Header Format

- **Header Len:** 8-bit unsigned integer, representing the length of the Mobility Header in units of 8 octets, excluding the first 8 octets. The length of the Mobility Header has to be a multiple of 8 octets.
- **MH Type:** 8-bit selector. Identifies the particular mobility message in question.
- **Reserved:** 8-bit field reserved for future use.
- **Checksum:** 16-bit unsigned integer. This field contains the checksum of the Mobility Header. It is calculated from the octet string consisting of a "pseudo-header" followed by the entire Mobility Header starting with the Payload Proto field.
- **Message Data:** A variable length field containing the data specific to the indicated Mobility Header type.

The Mobility Header is identified by a next header value of 135 (decimal) in the IPv6 base header (or an alternative preceding optional header if there is one). The MH Type field identifies the specific mobility message in question and can be one of the following:

- Home Test Init (HoTI)
- Care-of Test Init (CoTI)
- Home Test (HoT)
- Care-of Test (CoT)
- Binding Request (BR)
- Binding Update (BU)
- Binding Acknowledgement (BA)
- Binding Missing (BM)

The **Home Test Init, Home Test, Care-of Test Init** and **Care-of Test** are four messages used to perform the return routability procedure from the Mobile Node to a Correspondent Node.

Binding Update: as was said before, a Binding Update is used by a Mobile Node to notify the Mobile Node's Home Agent or a correspondent node of its current binding.

Binding Acknowledgement: A Binding Acknowledgement is used to acknowledge receipt of a Binding Update, if an acknowledgement was requested in the Binding Update, the binding update was sent to a Home Agent, or an error occurred.

Binding Refresh Request: A Binding Refresh Request is used by a Correspondent Node to request a Mobile Node to re-establish its binding with the correspondent node. This message is typically used when the cached binding is in active use but the binding's lifetime is close to expiration. The Correspondent Node may use, for instance, recent traffic and open transport layer connections as an indication of active use.

Binding Error: The Binding Error is used by the Correspondent Node to signal an error related to mobility, such as an inappropriate attempt to use the Home Address destination option without an existing binding.

3.5 IPv6 Mobile Networks: the NEMO protocol

Nowadays, it exists many mobile platforms that provision Internet access like planes, trains, cars, etc, making it necessary to support the mobility of complete networks inside those vehicles and not only of one host. As said in Chapter 1, the scenario studied comprises a network inside an aircraft moving, which means that many users should connect to the Internet.

A first solution of that new situation could be enabling node mobility support in all hosts of the moving network, so they could independently manage their mobility. However, it implies several problems: it would require all devices to be capable to support Mobile IP what would generate excess overhead as every device has to perform Mobile IP functions. Moreover, it would not work because of the limited capacities of the nodes (such as in sensors or embedded devices in the vehicles) or because it is not possible to update the software in some older devices.

Because of these problems, it was needed to standardize a solution enabling network mobility at the IPv6 layer. Nowadays, a good solution would be the NEtwork MObility (NEMO) Basic Support Protocol [20].

The basic new feature of this mobile extension is the mobile router, a single entity that manages the mobility of the complete network. Nodes of the mobile network will gain access to the Internet through the mobile device, using cheaper and widely available access technologies (for example, WLAN technology or Bluetooth).

3.5.1 Introduction

Network Mobility Basic Support protocol (NEMO) is an extension of Mobile IPv6 that enables to support the movement of a complete network that changes its point of attachment to the fixed infrastructure, maintaining the sessions of every device of the network.

The most important difference between the standard Mobile IPv6 protocol and NEMO is that in that case, instead of having a Mobile Node, we have a Mobile Router (MR) with an entire network behind it. This router is the only entity aware of mobility in the Mobile Network and which connects this network to the fixed infrastructure. This is now the end-point of the mobility tunnel, which is established between this Mobile Router and the Home Agent. The nodes inside the Mobile Network are called Mobile Network Nodes (MNN) and they are not aware of mobility. The end of the bidirectional tunnel at the side of the Mobile Router needs to be updated each time the Mobile Network moves (and also periodically to refresh the binding update at the home agent), to reflect the current location of the Mobile Router.

As we will see in Section 3.5.2, where the NEMO protocol operation is described, there are many similarities between the normal handover process of a simple mobile node and a complete moving network. As we can see in the Figure 3.9, which shows an example of a mobile network situation, the devices that take part of the scenario are all almost the same as if we had just a simple Mobile Node.

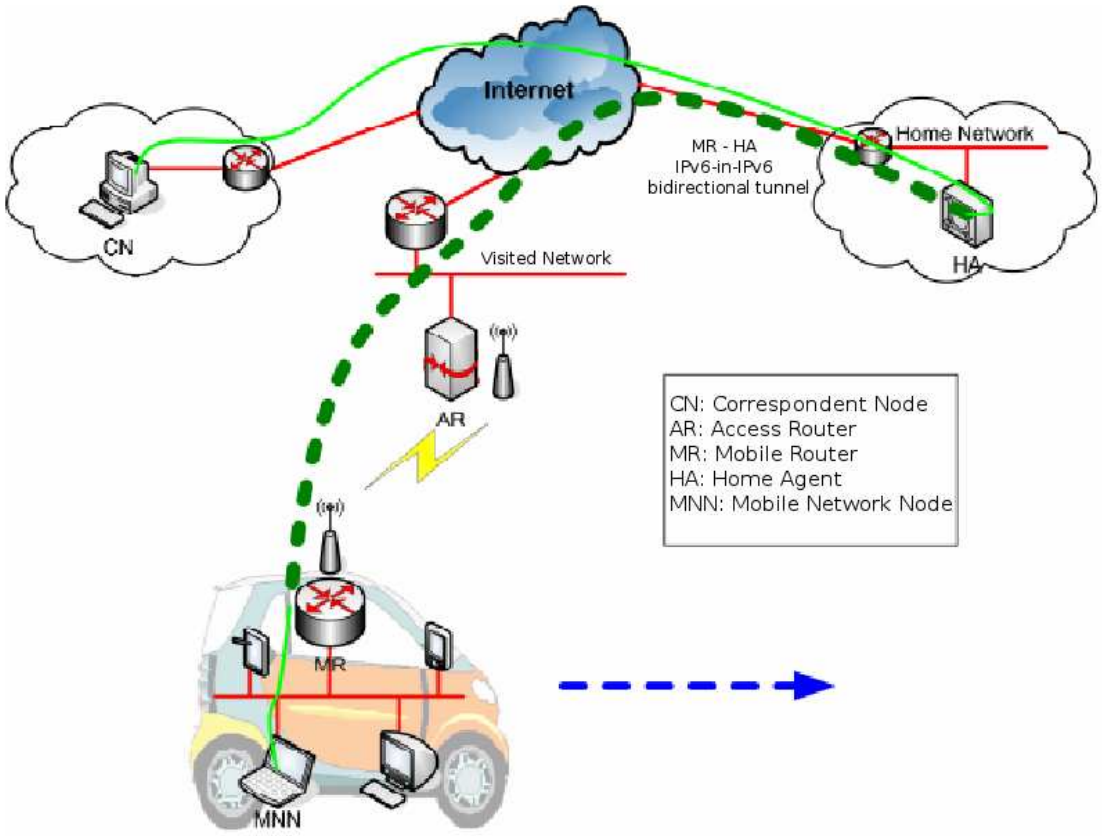

Figure 3.9: Mobile Network scenario

3.5.2 Operation of the NEMO Basic Support Protocol

To make easier to understand the function of the protocol I will describe a network-mobility scenario and the handovers process of a mobile network through an example.

A typical network mobility situation could be the one the Figure 3.10 depicts, where there is a complete network moving with a router and nodes attached to this mobile network. In that case, only the router supports NEMO protocol and the rest of the nodes in the mobile network don't have any mobility implementation, so they see the network always in the same way like if they were not moving. The router connects the mobile network to the Internet and plays the same role more or less as the Mobile Node in Mobile IPv6 protocol. It is assumed that this NEMO network is assigned to a particular network where it resides when it is not moving, known as its Home Network.

The mobile network has a permanent prefix address assigned to its home network: the Mobile Network Prefix (MNP). Looking at the example Figure 3.10, it would be 2001:b:1:1::/64. This address remains assigned to the NEMO network when it is away from home. All addresses beginning with this prefix have to be routed in the Internet toward the Home Network. Of course, these addresses have topological meaning only when the NEMO network is at home.

The handover operation consists:

- The Mobile Router acquires a home address before moving away to a foreign network.
- When the mobile network is away from home, only the Mobile Router acquires the Care-of Address (CoA) from the Access Router (AR) of the foreign network through Router Advertisements.
- When the Mobile Router gets the Care-of Address it starts to send Binding Updates (BU) to the Home Agent at the home network to register it in the binding cache. This Binding Update message still contains the new primary Care-Of Address, but now also the Mobile Network Prefixes. This is because the Home Agent has to know for which destination addresses it has to send the packets to the Mobile Router.
- Then, the Mobile Router waits for Binding Acknowledges sent by the Home Agent.
- As in an usual Mobile IPv6 protocol scenario, the Home Agent and the Access Routers are always sending Router Advertisements to inform Mobile Routers of its position (if they are at home or away).

Now, when any Correspondent Node (CN) wants to exchange information with a Mobile Network Node (MNN), the following operations are involved in the communication:

1. The packet sent by the Correspondent Node is routed to the Home Agent of the mobile network where it encapsulates the datagram to a new message and tunnels it to the mobile router. This datagram sent by the CN carries as its destination address the IPv6 address of the MNN, which belongs to the MNP of the NEMO.

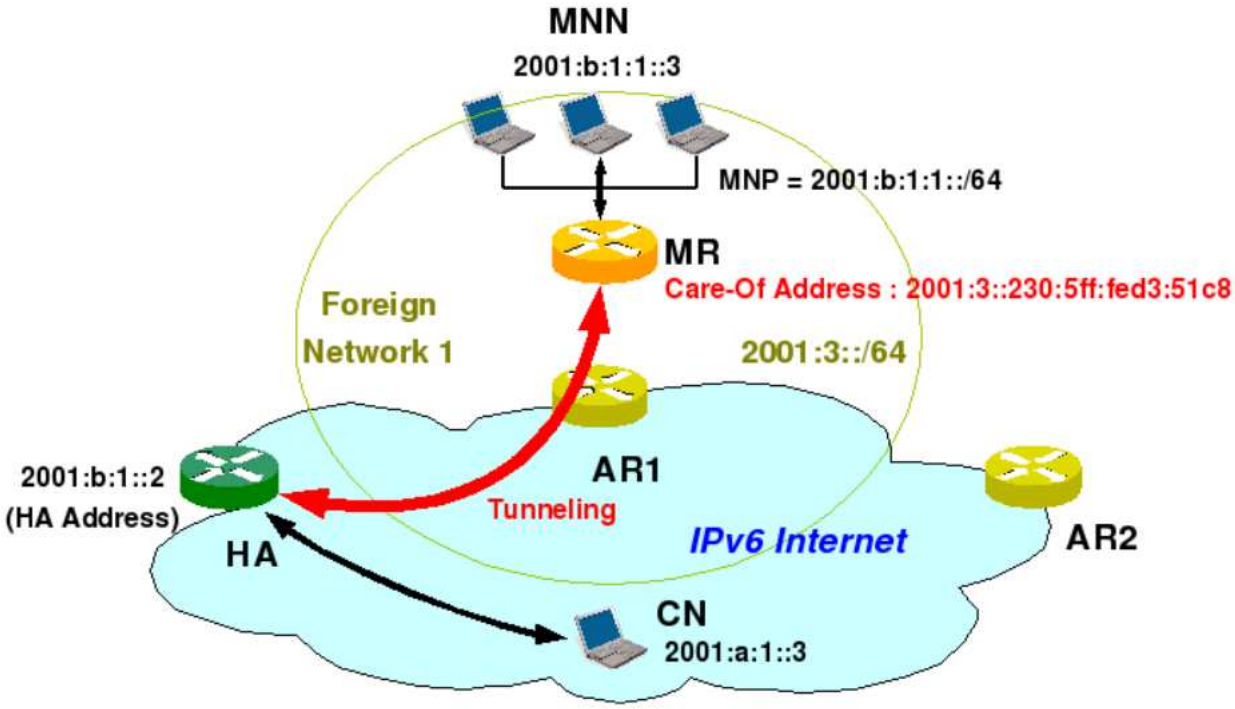

Figure 3.10: NEMO operation process

2. Now, the new datagram is sent by Home Agent to the CoA of the Mobile Router, with the IP address of the Home Agent as source address, like the Figure 3.11 depicts. This encapsulation preserves mobility transparency (neither the MNN nor the Correspondent Node are aware of the mobility of the mobile network) while maintaining the established Internet connections of the MNN.

3. The Mobile Router receives the encapsulated IP datagram, removes the outer IPv6 header, and delivers the original datagram to the MNN.

4. The operation is analogous in the opposite direction. The Mobile Router encapsulates the IP datagrams sent by a MNN toward its Home Agent, which then forwards the original datagram toward its destination. As in mobility for a node, the encapsulation is required to avoid problems with ingress filtering, because many routers implement security policies which do not allow the forwarding of packets with a source address that appears topologically incorrect.

3.5.3 NEMO Protocol Details

The NEMO Basic Support Protocol is an extension of the solution proposed for host mobility support, Mobile IPv6[20].

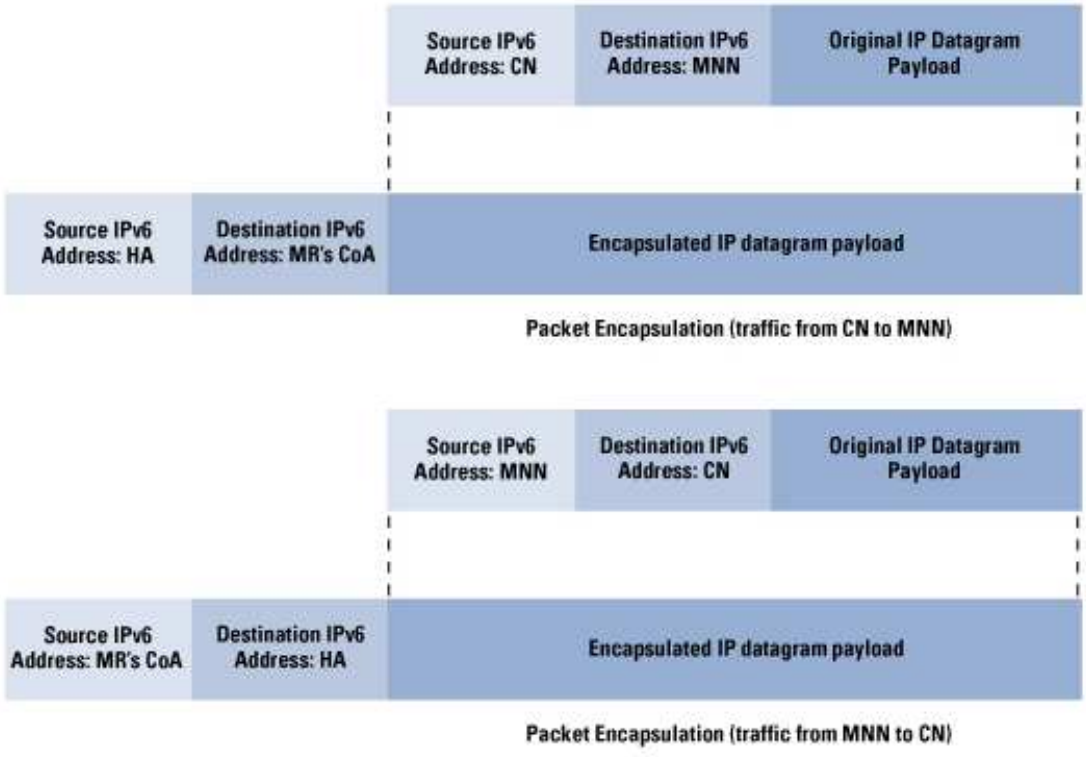

Figure 3.11: Overview of NEMO Basic Support Protocol Encapsulation

The Mobile Router can act at any time either as a Mobile Host or as a Mobile Router. What decides which performance it has to do, is a new flag placed in the Binding Update message header. If it acts as a Mobile Host, then it becomes a normal Mobile Node with the normal Mobile IPv6 protocol.

If the Mobile Router seeks to act as a Mobile Router and provide connectivity to nodes in the mobile network, it indicates this to the Home Agent by setting this new flag in the Binding Update. Then it also include information about the Mobile Network Prefix in the Binding Update message, so that the Home Agent can forward packets meant for nodes in the Mobile Network to the Mobile Router.

The most significant changes from the MIPv6 protocol are described next:

3.5.3.1 New Mobility Header

- **Binding Update (BU):** a new flag (R) is included in the Binding Update to indicate to the Home Agent whether the Binding Update is coming from a Mobile Router and not from a mobile node. The rest of the Binding Update format remains the same as Mobile IPv6 protocol.

- **Binding Acknowledgement (BA):** as a request for the new flag of the binding update header, in the BA a new flag is also included to indicate that the Home Agent processed the corresponding Binding Update and that it supports Mobile Routers. The flag is set only if the corresponding Binding Update had the Mobile Router Flag set to 1. The rest of the Binding Acknowledgement format remains the same as for Mobile Nodes.
- **Mobile Network Prefix Option:** The Mobile Network Prefix Option is included in the Binding Update to indicate the prefix information for the Mobile Network to the Home Agent. There could be multiple Mobile Network Prefix Options if the Mobile Router has more than one IPv6 prefix in the Mobile Network and wants the Home Agent to forward packets for each of these prefixes to the Mobile Router's current location.

3.5.3.2 New Home Agent Binding Cache

As it was defined before, the Binding Cache is a conceptual data structure of the Home Agent used to register the Mobile Routers' home address and the corresponding CoA's. The difference between the Binding Cache used in MIPv6 protocol is that now the Home Agent might need to store the Mobile Network Prefixes associated with a Mobile Router in the corresponding Binding Cache Entry. This is required if the Binding Update that created the Binding Cache Entry contained explicit prefix information. The Home Agent also stores the status of the Mobile Router Flag (R) in the Binding Cache entry. Also there is the Prefix Table which contains the following fields:

- The Home Address of the Mobile Router. This field is used as the key for searching the pre-configured Prefix Table.
- The Mobile Network Prefix of the Mobile Router associated with the Home Address.

3.5.4 Applications of NEMO

The different scenarios and advantages that network mobility offers are:

- **Public transportation systems:** These systems will let passengers in trains, airplanes, ships, etc. to accede to the Internet from terminals onboard (i.e. laptops, cellular phones, PDAs...) through a Mobile Router located in the transport vehicle that connects to the fixed infrastructure. Future vehicles will benefit from having Internet connectivity, not only to enhance safety but also to provide personal communication, entertainment, and Internet-based services to passengers.
  - Airplanes: Until recently, wireless devices have been prohibited on commercial airline flights due to the risk of interference with airplanes electrical systems. However, in June of 2005, the Federal Aviation Administration (FAA) gave permission to United Airlines to install Wi-Fi (802.11) wireless network equipment on some of its aircraft [21]. This new regulation will open the door for in-flight Internet service and invite NEMO as a solution to provide uninterrupted Internet connectivity to multiple passengers.

  - Automobiles: It is not difficult to image networked systems or even Internet enabled navigation, multimedia, or driving system on automobiles. In the case of critical driving systems, NEMO would be essential in order to maintain continuous connectivity and availability [22].

- **Personal Area Networks (PANs):** People are beginning to carry multiple Internet enabled devices such as cell phones, PDAs, laptop computes, and music players. Instead of each device connecting to the Internet separate, all of devices could connect the Internet through a PAN. Using NEMO, one device, such as a cell phone, would act as the mobile router providing continuous access to the rest of the devices.

As stated previously, only one exchange (two packets, one in each direction) is required per movement, regardless of the number of MNNs that are attached to the mobile router one of the main advantages of using the NEMO Basic Support Protocol on the Mobile Router instead of Mobile IPv6 on every node of the mobile network, because the signaling generated by a complete moving network (composed of numerous nodes) is the same as the one generated by a single moving node.

However, there are still many problems and issues that can be improved in NEMO protocol mechanisms. Next, the main features and problems that can be improved are exposed and some possible solutions, that are still being tested, are described.

3.5.5 Sub-optimality with NEMO protocol

With NEMO Basic Support, all packets sent between a Mobile Network Node and its Correspondent Node are forwarded through the Mobile Router-Home Agent tunnel, resulting in a pinball route between the two nodes. This has the following sub-optimal effects:

- **Longer route leading to increased delay and additional infrastructure load**

  Because a packet must transit from a mobile network to the Home Agent then to the Correspondent Node, the transit time of the packet is usually longer than if the packet could go straight from the mobile network to the Correspondent Node. When the Correspondent Node (or the mobile network) resides near the Home Agent, the increase in packet delays can be very small. However when the mobile network and the Correspondent Node are relatively near to one another but far away from the Home Agent on the Internet, the increase in delay is very large. Applications such as real-time multimedia streaming may not be able to tolerate such increase in packet delay. In general, the increase in delay may also impact the performance of transport protocols such as TCP. Moreover, by using a longer route, the total resource utilization for the traffic would be much higher than if the packets were to follow a direct path between the Mobile Network Node and Correspondent Node. This would result in additional load in the infrastructure.

- **Increased packet overhead**

  The NEMO Basic Support Protocol relies on the creation of a bidirectional tunnel between the Mobile Router and the home agent to provide transparent mobility support to a complete network. The use of this tunnel causes an additional overhead of 40 bytes per packet,

because of the extra IPv6 header added by the encapsulation. The effect of this overhead might be relevant for applications that generate small packets, such as voice-over-IP (VoIP) packets, because the 40-byte added overhead may be even bigger than the actual VoIP payload.

- **Increased processing delay**

  The encapsulation of packets in the Mobile Router-Home Agent tunnel also results in increased processing delay at the points of encapsulation and decapsulation. Such increased processing may include encryption/decryption, MTU computation, fragmentation and reassemble and other mechanisms.

- **Increased chances of packet fragmentation**

  The augmentation in packet size due to packet encapsulation may increase the chances of the packet being fragmented along the MRHA tunnel. This can occur if there is no prior path MTU discovery conducted, or if the MTU discovery mechanism did not take into account the encapsulation of packets. Packets fragmentation will result in a further increase in packet delays, and further reduction of bandwidth efficiency.

- **Increased susceptibility to link failure**

  Under the assumption that each link has the same probability of link failure, a longer routing path would be more susceptibility to link failure. Thus, packets routed through the Mobile Router-Home Agent tunnel may be subjected to a higher probability of being lost or delayed due to link failure, compared to packets that traverse directly between the Mobile Network Node (MNN) and its Correspondent Node.

- **Unique Care-of Address**

  Mobile Routers only can register one Care-of Address at each time in the Binding Cache of the Home Agent. This means that when the mobile network makes a handover between two foreign networks, first the Mobile Router has to de-register the old Care-of Address from the Binding Cache and then get another Car-of Address of the new visiting network and register it. That produces unnecessary delays while performing the handovers in ongoing communication sessions.

3.5.6 Future Work for NEMO

The NEMO Working Group was established to find a basic solution to network mobility. Unfortunately, many of the performance enhancing features of Mobile IP do not work when using basic NEMO. There are many mechanism that can still be improved to achieve better performance in terms of seamless handovers, traffic delays, etc. The IETF WG itself has made official statements and analyzes of the problems at hand including route optimization to increase routing efficiency and multihoming to increase fault tolerance and capacity. Those two mechanisms are explained in the next Sections (3.5.6.1 and 3.5.6.2). There are also projects outside of the IETF Working Group, such as Nautilus6, that are also working on implementing these features in hopes of getting them incorporated into future standards. In Chapter 4.1 a new mechanism of registering multiple Care-of Addresses, designed by Nautilus6 project, is explained and widely described.

3.5.6.1 Route Optimization (RO)

Route optimization (RO) provides a mechanism to eliminate the inefficiency in tunneling packets from Mobile Routers to their Home Agent before being sent to Correspondent Nodes over the Internet. RO would allow a way for Mobile Routers or Mobile Network Nodes to send packets directly to Correspondent Nodes. Figure 3.12 demonstrates this direct communication between Mobile Network Nodes and Correspondent Nodes via a tunnel. RO could decrease path delay and network load and avoid bottlenecks at Home Agents. However, the NEMO Basic Support Protocol does not address this issue and the NEMO Working Group is not currently chartered to define a standard for RO.

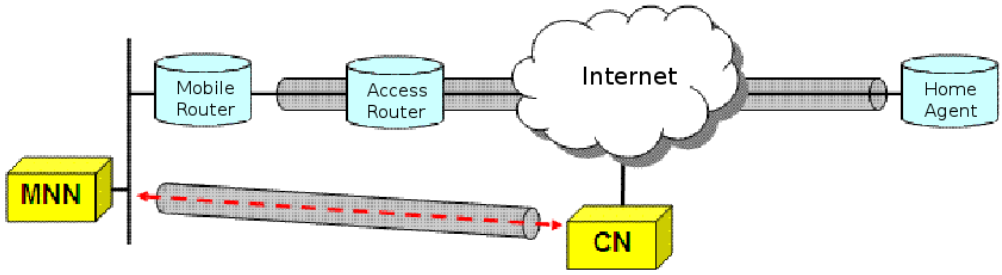

Figure 3.12: Route Optimization

As we can see at the example Figure 3.12, Mobile Network Nodes are able to send packets directly to Correspondent Nodes without tunneling to the Home Agent. Mobile IP performs RO by utilizing a Binding Cache on the Correspondent Node. Mobile IP nodes send Binding Updates with current CoAs to their Correspondent Nodes as they change attachment points to the Internet. Mobile nodes and Correspondent Nodes are then able to directly communicate using the CoA of the Mobile Node [23]. There are several possible approaches to the NEMO RO problem; however, each has its own trade-offs. Such trade-offs include increased signaling overhead, longer handover delay, and the need to make additional devices such as CNs and MMNs aware of NEMO.

3.5.6.2 Multihoming

In this proposed mechanism, Mobile Router has multiple access links, thus multiple Care-of Addresses, one for each link. In a general sense, multihoming is a technique of increasing reliability and performance by providing redundant links. Under NEMO, multihoming takes the form of multiple Home Agents, Mobile Routers, access links and network prefixes. Multihoming has the potential to provide load balancing, seamless handovers, fault tolerance and increased bandwidth. Figure 3.13 shows an example of a multihoming Mobile Router with multiple access interfaces. This Mobile Router could be using multihoming for redundancy as well as increased bandwidth.

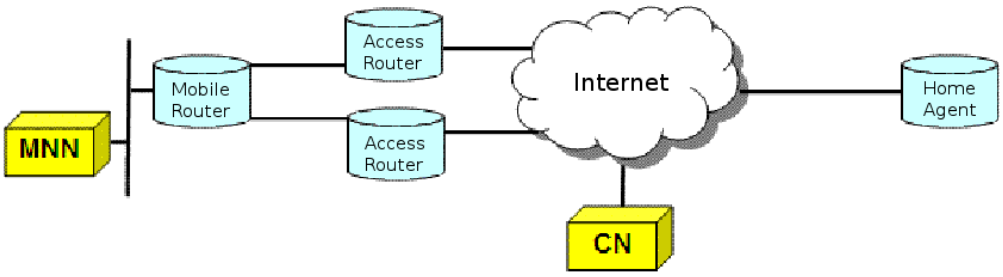

Figure 3.13: Multihoming

3.5.6.3 Multiple Care-of Addresses

This mechanism extension for NEMO protocol, proposed by Nautilus6 group, allows a Mobile Router to register several Care-of Addresses at the same time to its Home Agent. Main benefits, among others, are policy routing, fault tolerance for the Mobile Router, seamless handovers and load balancing. In Mobile IPv6 and NEMO protocols, the Mobile Router can only register one Care-of Address every time, so it has just one point of attachment. The Home Agent can only associate one CoA for every Mobile Router's home address and network prefix in the Binding Cache. That is translated in delays during handovers of the mobile network, because every time that it moves, it has to de-register the old Care-of Address from the Binding Cache, get another CoA of the new Access Router and then register the new CoA again. This new mechanism permits the Mobile Router to be connected at the same time in two or more foreign networks using only one Home Address, so it permits to make seamless handovers and also balance the information through different Access Routers.

Chapter 4

Multiple Care-of Address registration extension protocol

NEMO Basic Support (NEMO BS) adds a mobility function to IPv6 routers and permits them to have a network behind, the mobile network, that becomes logically static. This function is considered useful when a network has a lot of nodes that do not have a mobility capacity, so they can move with the network. As was explained in the previous chapter, for this function, a temporal address is used, called Care-of Address. According to the Mobile IPv6 and NEMO specification, a Mobile Node or Mobile Router is not allowed to register more than one Care-of Addresses to a single home address. If a Mobile Node or Router sends Binding Updates with more than one Care-of Address, Home Agents would always overwrite the Care-of Address recorded in the binding cache with the one contained in the latest received Binding Update. So, it is impossible for a both mobile entities to register multiple Care-of Addresses in the Home Agent's binding cache.

The first big problem with that specification is that the Mobile Router and its mobile network will face service disruption of network connectivity while the Mobile Router is moving from one network to another one. Thus, solutions are needed to support continuous connectivity allowing to have seamless handover.

The test-bed developed in TriaGnoSys has initially been implemented using only NEMO extension protocol in the Mobile Routers and Home Agents. But in order to obtain better seamless handovers for the mobile network communications and not the disruptions mentioned before, some solutions were studied that could face those drawbacks. Thus, a new mechanism had been implemented from a new mobile protocol that has been created in the early days: the Multiple Care-of Address registration mobile extension.

The Multiple Care-of Address registration mechanism is a mobile extension protocol of IPv6 still being developed that tries to solve the problem of connectivity disruptions in handovers, as it permit to register more than one Care-of Address at the same time with a unique Mobile

Router and home address. I implemented it to the test-bed system to ameliorate the handovers delays but also to study its behaviour and see if it could be a good mechanism for the future of aeronautic communications through test simulation results.

So, in this chapter I will explain the basic characteristics and functions of the Multiple Care-of Address registration protocol [25] and the advantages with respect to the NEMO implementation.

4.1 Multiple Care-of Address Registration mechanism

Multiple CoAs Registration mechanism makes possible to use multiple network interfaces concurrently at the Mobile Router in a mobile network as it permits to register Multiple Care-of Addresses bound to a single Home Address instead of the unique primary Care-of Address. Without MCoA, it is impossible to avoid the service disruption during handovers. If multiple network interfaces concurrently can be used in a Mobile Route, then it can prepare a network interface for a new foreign network to where the it is going to move, before disconnecting from the old foreign network. This extension is targeted to NEMO (NEtwork MObility) Basic Support as well as to Mobile IPv6, what it means that can be applied to a Mobile Router and to a Mobile Node. [26]

4.1.1 Protocol characteristics

The basic new addition brought by this new mobile extension that permits Mobile Nodes and Mobile Routers to register multiple Care-of Addresses to a single home address and create multiple binding cache entries in the Home Agent, is a new identification number called Binding Unique Identification number (BID). It is introduced to distinguish between multiple bindings pertaining to the same home address. This number is created for every Care-of Address that the Mobile Node and Router wants to register and is sent within the Binding Update message. The Home Agent that receives this message creates a separate binding for each BID in its binding cache list. As consequence, also a new binding cache management to store the BID and a new sub-option for binding update to carry the BID is created. As this extensions works for both Mobile Router and Mobile Node in the same way, I will explain the mechanism for the Mobile Router case, as the NEWSKY test-bed will deal with it.

The new binding process is quite the same as a normal MIPv6/NEMO operation: when a mobile network visits a foreign network, the Mobile Router obtains a Care-of Address from the Access Router and register it with the home address in the binding cache of the Home Agent. In that case, the Binding Update message includes a Binding Identifier mobility option which carries the BID. When the Home Agent receives the Binding Update, it copies the BID to the corresponding field in the binding cache entry. If the mobile network detects another foreign network signal and wants to access it, the Mobile Router can get another Care-of Address from the new Access Router and send it within another Binding Update message through the new link. This time, the Home Agent detects that there is another identical home address registered with different Care-of Address, but as it doesn't match with the one with existing entry, the Home Agent creates a new binding cache entry for the new Care-of Address and BID.

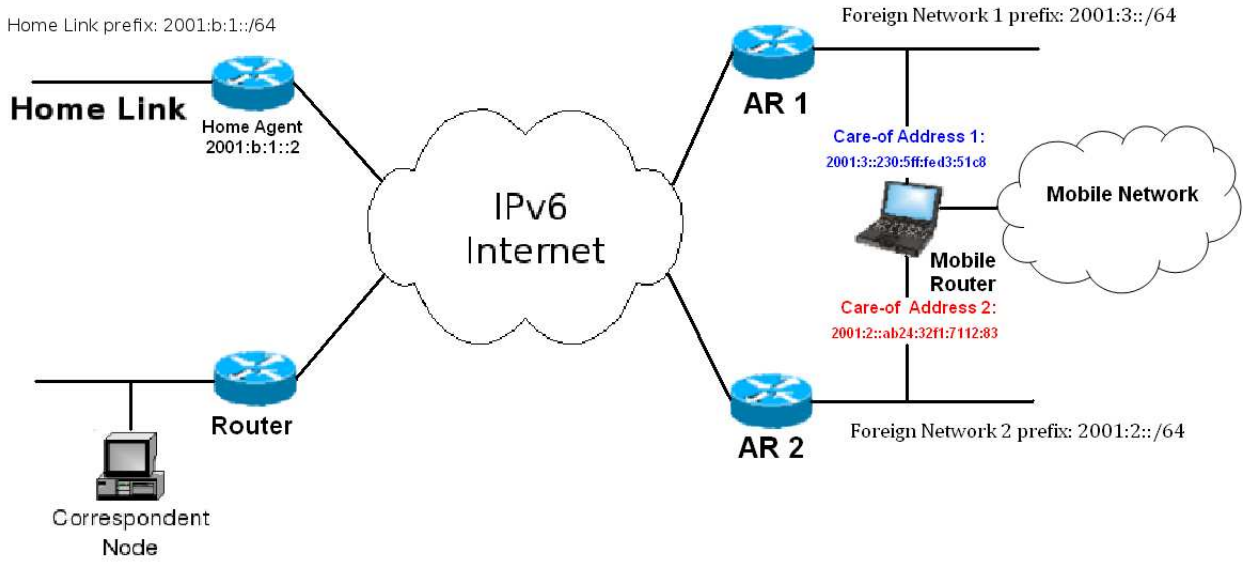

Figure 4.1: Example of a mobile network with MCoA configuration

With this extension, the Mobile Route can either register multiple Care-of Addresses at once in a single binding or in individual Binding Updates. The Home Agent has to support also the new extension protocol so can register the BID in the binding cache.

For example, in Figure 4.1 there is a Mobile Router that obtains two Care-of Addresses at two foreign links. In that case, the Mobile Router needs to have two different interfaces. The Mobile Route registers its CoAs (CoA1 and CoA2) at the same time to its Home Agent. As a result, two bi-directional tunnels are established between them. The traffic can be distributed between the two tunnels. The distribution policy depends on the local policy of the network operator, as will be explained in Section 4.4. In that case, the home address of the Mobile Router would be for example 2001:b:1::3/64 and it acquires the Care-of Address 2001:3::230:5ff:fed3:51c8 for the first foreign link and the 2001:2:ab:24:32f1:7112:83 for the second foreign link. The Mobile Router assigns two different Binding Identifier numbers for each link attached: BID1 and BID2.

If the Mobile Router decides to act as a regular Mobile Router, with no multiple Care-of Addresses, it sends the Binding Update without no binding identifier mobility options. Then, the receiver of the Bindig Update deletes all the bindings registered with a BID and registers only a single binding for the Mobile Router.

The binding cache is created this time based on the home address and the BID information if a BID is available. This is different from using just MIPv6/NEMO protocols where only the home address is used for binding cache lookup. Also the BID is required to be stored in the Binding Update List structure of Mobile Routes so they can know through which tunnel have to send the different kind of packets.

When a Mobile Router decides to delete all the bindings for its home address, it sends a regular de-registration Binding Update with lifetime set to zero as defined in RFC3775. In that case, the Binding Identifier mobility option is not required.

If the Mobile Router decides to register only a single binding, it just sends a Binding Update without a Binding Unique Identifier sub-option (i.e. normal Binding Update). The receiver of the Binding Update registers only a single binding for the mobile node. If the receiver has multiple bindings, one binding is registered without BID and the rest of bindings are deleted.

If a Mobile Router wants to delete a particular binding from its Home Agent, the Mobile Router sends a Binding Update with lifetime set to zero and includes a Binding Identifier mobility option with the BID it wants to de-register. The receiver will remove only the Care-of Address that match the specified BID.

4.2 Binding Update message architecture

The only feature added in the message structure of the Binding Update of MIPv6/NEMO protocols is a new flag, (the "O" flag that appear in the message structure of the Figure 4.2). When this flag is set, all the binding cache entries for a Mobile Route are replaced by new entries registering with this Binding Update message. This flag is only used when the BID mobility option is carried with the Binding Update.

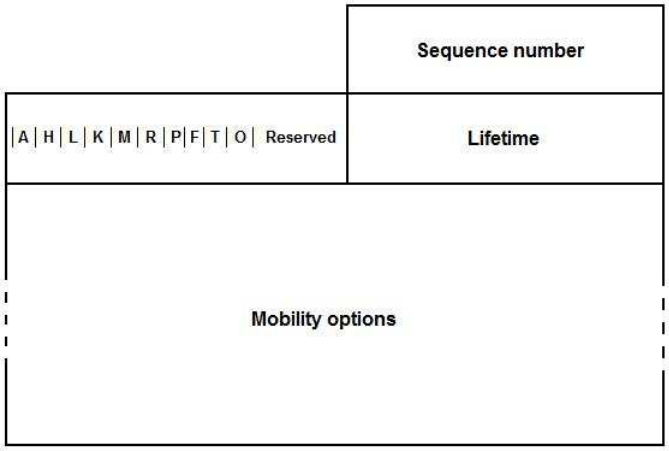

Figure 4.2: Binding Update Header

4.3 Binding Identifier mobility option

This new option is included in the Binding Update, Binding Acknowledgement, Binding Refresh Request, and Care-of Test Init and Care-of Test messages, explained in the chapter before.

The fields that are integrated in the message are:

1. **Type:** the value for Binding Identifier is 35.

2. **Length:** is a 8-bit unsigned integer field. Determines the length of the option payload, in octets, excluding the Type and Length fields.

3. **Binding ID (BID):** carries the BID that is assigned to the binding indicated by the care-of address in the Binding Update or the Binding Identifier mobility option. The BID is a 16-bit unsigned integer.

4. **Status:** is a 8-bit unsigned integer. When the Binding Identifier mobility option is included in a Binding Acknowledgement, this field overwrites the Status field in the Binding Acknowledgement only for this BID. This Status field is also used to carry error information related to the care-of address test in the Care-of Test message.

5. **Simultaneous Home and Foreign Binding (H) flag:** this flag indicates that the mobile node registers multiple bindings to the home agent while it is attached to the home link. This flag is valid only for a Binding Update sent to the home agent.

6. **Reserved:** is a 7-bit Reserved field.

7. **Care-of Address:** If a Binding Identifier mobility option is included in a Binding Update for the home registration, either IPv4 or IPv6 care-of addresses for the corresponding BID can be stored in this field. For the binding registration to correspondent nodes (i.e., route optimization), only IPv6 care-of addresses can be stored in this field. If no address is specified in this field, the length of this field must be zero (i.e., not appear in the option). If the option is included in any messages other than a Binding Update, the length of this field has to be also zero.

When a Mobile Router registers a given BID for the first time, it must include the Care-of Address filed in the Binding Identifier mobility option. For any subsequent registrations that either re-register or de-register the same BID, the Mobile router need not no include the Care-of Address field in the binding identifier option.

4.4 Multiple Bindings Management and policy routing

Some policies may be bind to a BID in the Mobile Router. Those policies are used to divide the packet flows to multiple network interfaces that the Mobile Router uses. The flows can be divided by flow type, port number, destination address, etc.

In addition, each communication flow sent to a distinct network interface, provides efficient network bandwidth consumption. It becomes possible for users to select the most appropriate network interface depending on a visiting network environment, since wireless networks are mutable and less reliable than wired networks and since each network interface has different cost, performance, bandwidth, access range, and reliability. Thanks to this mechanism, users are able to select the most appropriate interface per communication type. For example, TCP traffic should be transmitted over the wireless interface, whereas UDP traffic should be transmitted over the wired interface to avoid disturbing TCP connections, using different binding tunnels.

In the other hand, the BID is used as a search key for a corresponding entry in the Home Agent's binding cache in addition to the Home Address. When the Home Agent checks the binding cache database for the Mobile Routes, it searches a corresponding binding entry with the Home Address and BID of the desired binding.

To select a specific binding, the Home Agent uses policies and filter informations. If a Mobile Route registers a binding with priority value, the priority can be a key to select a binding. If there is no desired binding, it searches the binding cache list with the Home Address as specified in Mobile IPv6 and the first matched binding entry may be the one used.

If multiple bindings exists, when packets from or to a mobile node are not delivered correctly, the Home Agent and the Mobile Router can change the tunnel used, so what also means to change the binding entry in order to recover the connection immediately.

In the next chapter, the way to assign the priorities is explained, as it will be used for the test-bed configuration when using the MCoA extension protocol.

Chapter 5

NEWSKY Test-bed

As said in the first chapter, TriaGnoSys contribution to NEWSKY project consists of a laboratory test-bed with the aim to integrate a mobile network inside aircrafts for new services and applications and validate its function in terms of traffic delays and packets lost to improve handovers between different networks.

As part of the TriaGnoSys team that was in charge of the NEWSKY test-bed, I helped on building the system, configuring the machines used and implement them with the protocols described in past chapters as well as being the one in charge to find out a way to improve the test-bed's mobile network handovers to obtain seamless point-of-attachment changes.

In this chapter, the test-bed architecture configuration is described as well as the different applications we developed and which will be able to be simulated in the system. Then I will expose some results of various tests I made to validate the test-bed behaviour. At the end, I will explain a way to improve the handovers delays of the system, studying and implementing the Multiple Care-of Address registration protocol. Also, some results of several tests will be exposed to compare the two behaviours.

5.1 The real architecture

In this section, the real architecture that the test-bed wants to recreate is described. As the Figure 5.1 shows, it would consist on a Mobile Network, with its respectives Mobile Router and Mobile Networks Nodes, located in the airplane and which will have connection with the Correspondent Node situated in the ground through the terrestrial-link Access Router (AR1) or the satellite-link Access Router (AR2). As we will see in next sections, the terrestrial link is better than the satellite link in terms of bandwidth, packet delays and packet loss during transmission,

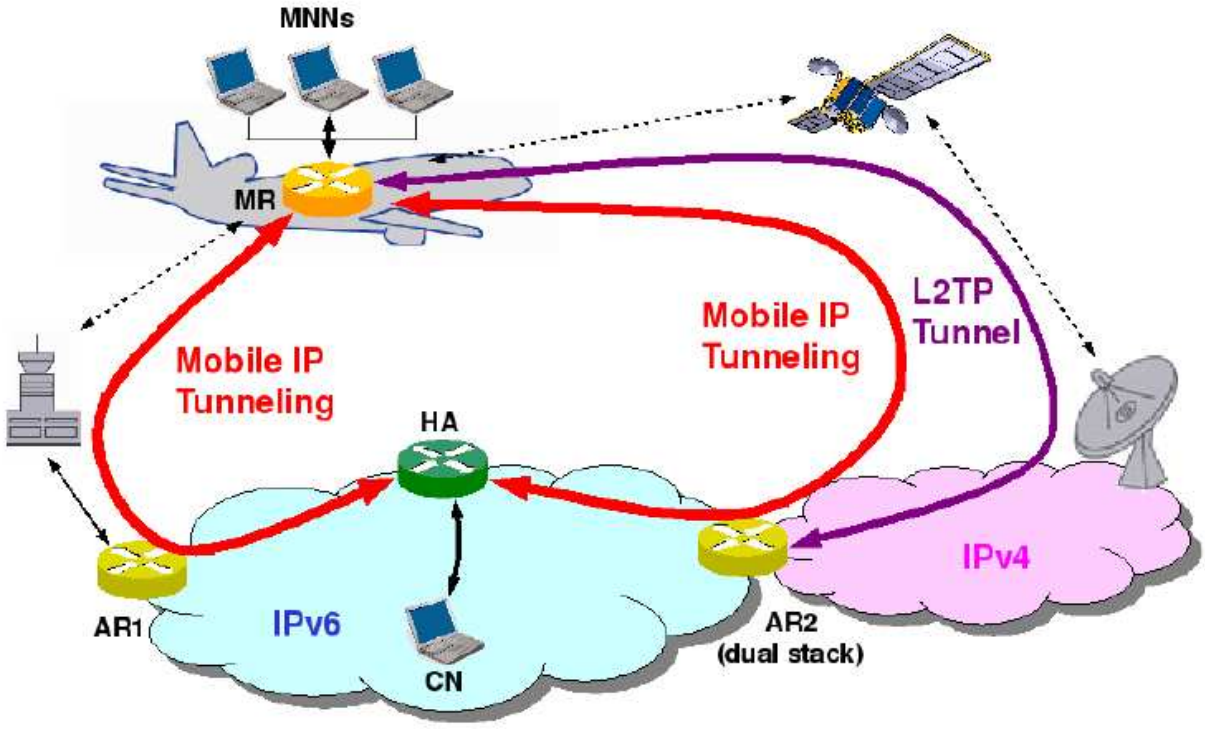

Figure 5.1: Real architecture design

but its cover area is smaller, just around the base-ground station, while the satellite link covers all over the world. These two links use different protocols in the communications and will be the ones I will use to simulate handovers.

In that scenario, the Mobile Network Node will probably be a terminal in the cockpit which provides interface to the pilot to make a phone (VoIP) call, download data, etc. or any passenger using his mobile device to connect to the Internet. The Mobile Router will be an entity in the aircraft which controls all communication traffic going in and out the aircraft. The Home Agent will be a node in the Internet which provides mobility service. This could be provided by the airline, an aeronautical communication service provider or any other dedicated organization. The Correspondent Node will be, for instance, the air traffic controller which provides information to the cockpit.

With this architecture, the Mobile Network Nodes in the airplane are able to communicate with any Correspondent Node in the IPv6 internet, and this without being aware of mobility, in a completely transparent way. As we can see, in this scenario, there is no home link, so it means that the mobile network is always away from its home network. So, there will always exist a tunnel between the Mobile Router and the Home Agent through an Access Router.

The test-bed will try to recreate this scenario and demonstrate that a VoIP call (or a data transfer) can be made between MNN and CN without breaking the call (or the data transfer)

through seamless handovers.

5.2 Configuration of the test-bed: protocols and mechanisms

In this section, the test-bed we developed in TriaGnoSys is described. It tries to simulate the real aeronautical architecture described just before using the protocols and mechanisms explained in previous chapters. After setting all the machines that take part of this test-bed, the simulated network architecture has the structure depicted in the Figure 5.2.

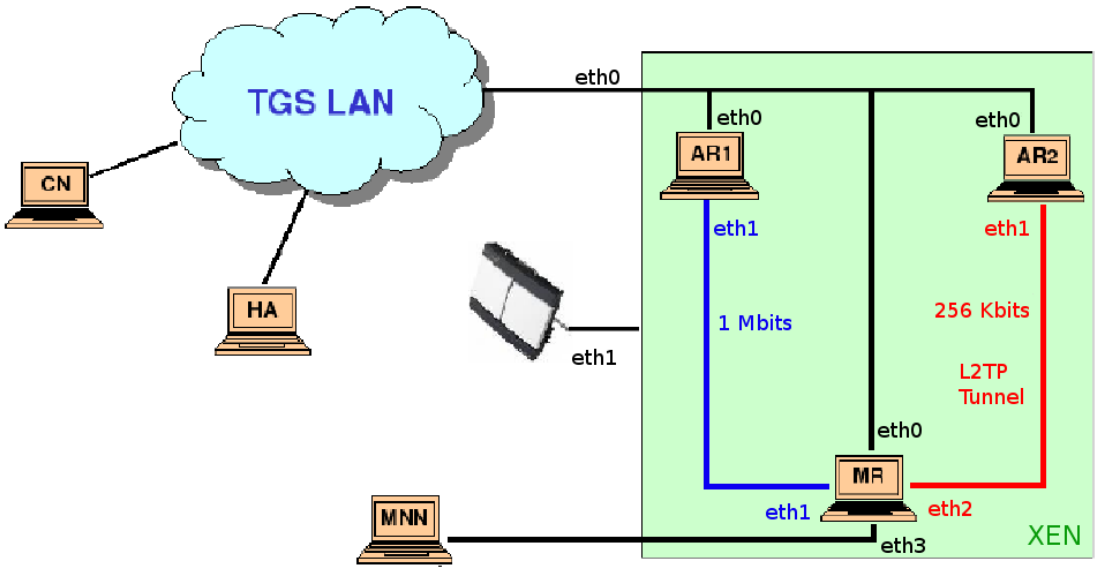

Figure 5.2: test-bed network architecture

5.2.1 Network architecture and XEN machines

All the machines are configurated under Linux based OS. At first, the Mobile Router and Home Agent machines will support only NEMO protocol (explained in Section 3.5) to allow to have a mobile network behind. Later, I will implement and develop a new mechanism for those two machines to improve handover delays and communications disruptions. The mobile network nodes in the mobile network support only the IPv6 protocol and they are connected to the Mobile Router. As the Figure 5.2 depicts, this router has three interfaces: one is used for the link between the Mobile Router and the Access Router 1 (the blue link in the figure) that simulates the terrestrial link, a second one for the link between the Mobile Router and the Access Router 2 (in red) that corresponds to the satellite link and a third one used to connect the mobile

network node. It has also a interface that will be used to connect the real satellite BGAN modem. In Section 5.2.2 the links are described.

As we can see in the previous figure 5.2, for this network mobility scenario, at least 6 computers are required (which will simulate the Mobile Network Node, Mobile Router, Access Router 1, Access Router 2, Home Agent and the Correspondent Node). As it was difficult to ask for 6 real computers of the company, we used only 4, where 1 of them is used to build 3 virtual machines (Access Router 1, Access Router 2 and Mobile Router, that are the ones illustrated inside the XEN square of the Figure 5.2). For these virtual computers, we employed the XEN software, which is a virtual machine monitor and allows the creation of up to 4 virtual machines in the same computer.

Nodes in Access Networks 1, Access Network 2 and in the Home Network get IPv6 address via stateful autoconfiguration. All other IPv6 addresses are configured manually. The TGS LAN is an emulating Air Traffic Control (ATC) ground network.

For the Mobile Router, Home Agent and Correspondent Node we used the NEMO implementation based on NEPL (NEMO Platform for Linux). In the Correspondent Node case, only it is necessary if we want to use route optimization. The MIPv6/NEMO protocol implementation consists in two parts: a kernel patch and a user-space program. NEPL is a free available NEMO implementation for Linux, which the original release was based on MIPL2 (Mobile IPv6 for Linux) and has been developed and tested in cooperation between the Go-Core Project and the Nautilus6 Project (WIDE) [24].

In the Appendix A, the interfaces configuration and addresses of the Home Agent and the Mobile Router machines are exposed as well as the final architecture configuration of the test-bed with all IPv6 addresses of all machines and networks involved.

5.2.2 Communication Links

As we saw before, there are two links considered in NEWSKY project between the mobile network and the base-station:

1. **Inmarsat BGAN:** for the satellite-link, the final NEWSKY test-bed will use a real BGAN satellite of Inmarsat. To access the BGAN service it is used a satellite terminal Thrane&Thrane 500[27]. The Inmsartat BGAN cover area is showed in the Figure 5.3.

2. **B-AMC:** for the terrestrial-link an emulator which will simulate the link characteristic of B-AMC is used.

But for the time I was working on the test-bed, B-AMC modem emulator was not operative and the real Inmarsat satellite link was quite expensive to be used all the time. As a result, I have simulated the behaviour of both satellite and terrestrial links within the Ethernet local network. To simulate the links, I used the Linux traffic control (*tc*) tool which provides all kinds of traffic shaping and realistic delays and packet losses simulation that will perform the two links.

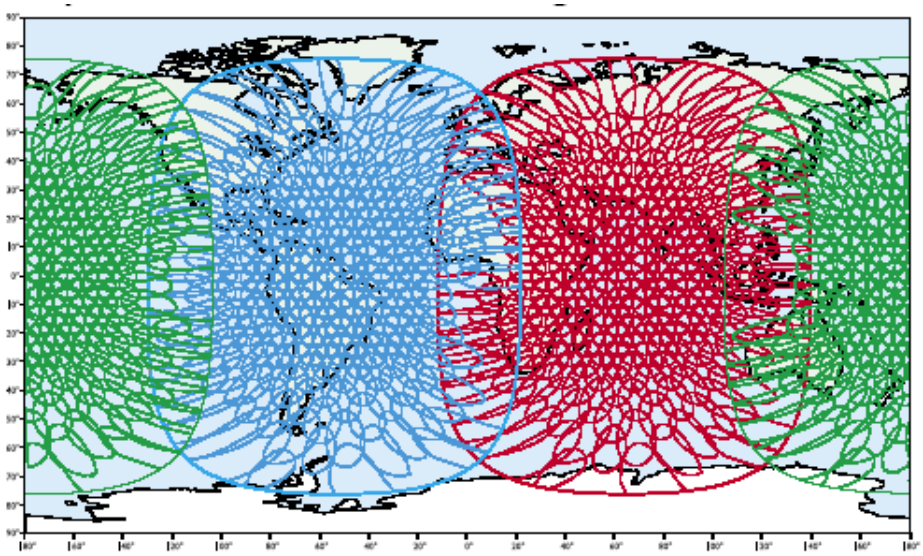

Figure 5.3: Beams of Inmarsat BGAN System [27]

5.2.2.1 IPv6 to IPv4 traversal mechanism

As it was explained in Section 2.7, the satellite network (BGAN) used, works over IPv4 protocol. As the project focuses mainly on IPv6 network, a way to traverse to IPv4 is needed. In that case, the L2TP mechanism will be used for the simulations.

When the Mobile Route detects that it moves to an IPv4 network, it tries to establish an L2TP tunnel to Access Router 2. It is assumed that the Mobile Router has the knowledge of Access Router 2's IPv4 address. Access Router 2 has a pool of IPv6 address subnet that it advertises in the tunnel interface such that the Mobile Router may perform IPv6 stateless autoconfiguration to construct IPv6 Care-of Address. Once the tunnel is established, the MIPv6/NEMO signalling can then run as in the normal complete IPv6 network case.

For the test-bed, the l2tpd GNU software is downloaded, to be run under Linux.

Once the software is correctly configured in both the Mobile Router and the Access Router 2, the following commands are used to set up the tunnel :

echo "t 192.168.9.155" >/var/run/l2tp-control

This command, which is run in the Mobile Router, sends a tunnel creation request to the Access Router 2. The IPv4 address is the Access Router 2's IPv4 address.

echo "c 28775" >/var/run/l2tp-control

Once the tunnel is established, a call has to be made in order to create a session which will carry the PPP frames. This command makes a call to the other end point of the tunnel, identified with the tunnel ID (28775 in this case).

If the session establishment is a success, a new ppp0 interface appears in both the Mobile Router and Access Router 2.

First of all, in the IPv4 satellite network, the dual-stack Access Router keeps a permanent IPv4 address, whereas the Mobile Router, in the plane, acquires its IPv4 address with DHCP. That's why at the beginning of the satellite connection, the Mobile Router knows the Access Router's IPv4 address, whereas the Access Router doesn't know the Mobile Router's IPv4 address. Therefore, the Mobile Router has to create the L2TP tunnel, and the Access Router is waiting for the Mobile Router's tunnel creation request. As a result, the Mobile Router has to be the LAC, and the Access Router has to be the LNS. When the session is established between the Mobile Router and the dual-stack Access Router, a ppp0 interface appears in both routers. This ppp0 interface has to be added in the MR's MobileIP configuration file, but also in the AR2's Router Advertisement Daemon configuration file, because Router Advertisement from the AR2 to the MR have to be sent now though this interface. Then, a link local address must be added to this interface in both computers, in order for MobileIPv6 and the Router Advertisement Daemon to be able to use the interface. Moreover, a global IPv6 address belonging to the Foreign Network has to be added to the ppp0 interface in the Access Router. This address must have the same prefix as the one advertised by the Access Router to the Mobile Router.

In order to reduce the overhead, another IPv6 to IPv4 traversal mechanism is currently being studied for the test-bed: the NAPT-PT protocol, which performs basically a header conversion and that I explained in section 2.7.2.2.

5.3 Applications

Some applications are done for the test-bed to make simulations, so I used them to test the behaviour of the system and to obtain results with the aim to characterize the effects of the handovers. The applications are: Voice over IP transmission and Weather Streaming Information transmission.

5.3.1 Voice over IP (VoIP)

The test-bed offers a VoIP transmissions. It consists of carrying digital audio, reduced in data rate using speech data compression techniques, and the encapsulated in a data-packet stream over IP. This mechanism uses some protocols as SIP (Session Initiation Protocol)[28] and RTP (Real-time Transport Protocol). The first is a signaling protocol used to create, modify and terminate call sessions, while the second one is used to deliver audio in streaming mode. I will use this application to see how are the delays in terrestrial-satellite links handovers during phone call conversations between the Mobile Network Node and the Correspondent Node. For that application, linphone software is used which is a softphone that supports IPv6.

5.3.2 Weather Streaming Information

This application is used to create data communications, which uses a FTP protocol to provide instantaneous weather maps and weather information between Correspondent Node and Mobile

Network Node. But no specific application was developed yet, so to simulate this system in the test-bed, we just sent map pictures from the Correspondent Node to the Mobile Network Node to study the TCP packets data transmissions during handovers between the two links.

5.4 Graphical User Interface

In order to have a clear idea of the demonstration scenario and to better understand the system behaviour, I have designed a GUI (Graphical User Interface) program with JAVA language that shows an animated flight of an airplane between Europe to USA. I will use it to simulate the handovers and configure the machines in every link connection during the tests.

The program depicts a map with the moving airplane changing from terrestrial to satellite link. As we can see in the Figure 5.4, it consists on the map and different buttons. The map has different elements and regions circles representing the reachability range of every network. When the GUI starts, the airplane beggins to move through the line that goes from Paris to Atlanta. In a normal mode, when the airplane is inside a circle, it is connected to the base-station through the terrestrial-link, but when it surpasses the area of the circle, then the handover is done and the satellite link is used (then, the GUI traces a line between airplane-satellite-base-station to show the connection path). The different buttons permit different kinds of simulations, for example the possibility to change the link automatically or manually, stop the flight, increase the speed of the aircraft, change the coverage regions area, etc.

The GUI triggers handovers while the airplane flies over the border of the terrestrial link coverage. The handover is triggered by changing the *ip6tables* rules in the Mobile Router when detects a change of network, so sets the Mobile Router links up or down.

The satellite link is available during the whole flight, the link is simply opened at the beginning of the flight to be more realistic. Then, a script is run in the Mobile Router to create the L2TP tunnel with the Access Router. Afterwards, a script is run also in the Access Router to send Router Advertisement messages. As a result, the Mobile Router adds a Care-Of Address assigned to the satellite link during the whole flight. For instance, when the plane flies over the ocean and needs to make the handover toward the satellite link, nothing special is done. In case the satellite link is available only when needed, the whole initialization process has to be achieved when the handover toward the satellite link is required, but early enough to prevent the plane to become out of reach of terrestrial base stations. When the plane flies again near ground-base networks, the satellite link is switched off. If the handover cannot be anticipated, a big interruption of communication can result of this situation, which can be annoying.

In the Appendix B, the GUI java code is exposed and also a flow chart of the different steps that the airplane follows during the flight with respective connection links is described.

5.5 Simulations and study of the system using NEMO protocol supported in MR and HA

In order to check the good behaviour of the system, I have done several tests of some simulations. In this section I will expose the procedures I used to get results. First, the handover process is

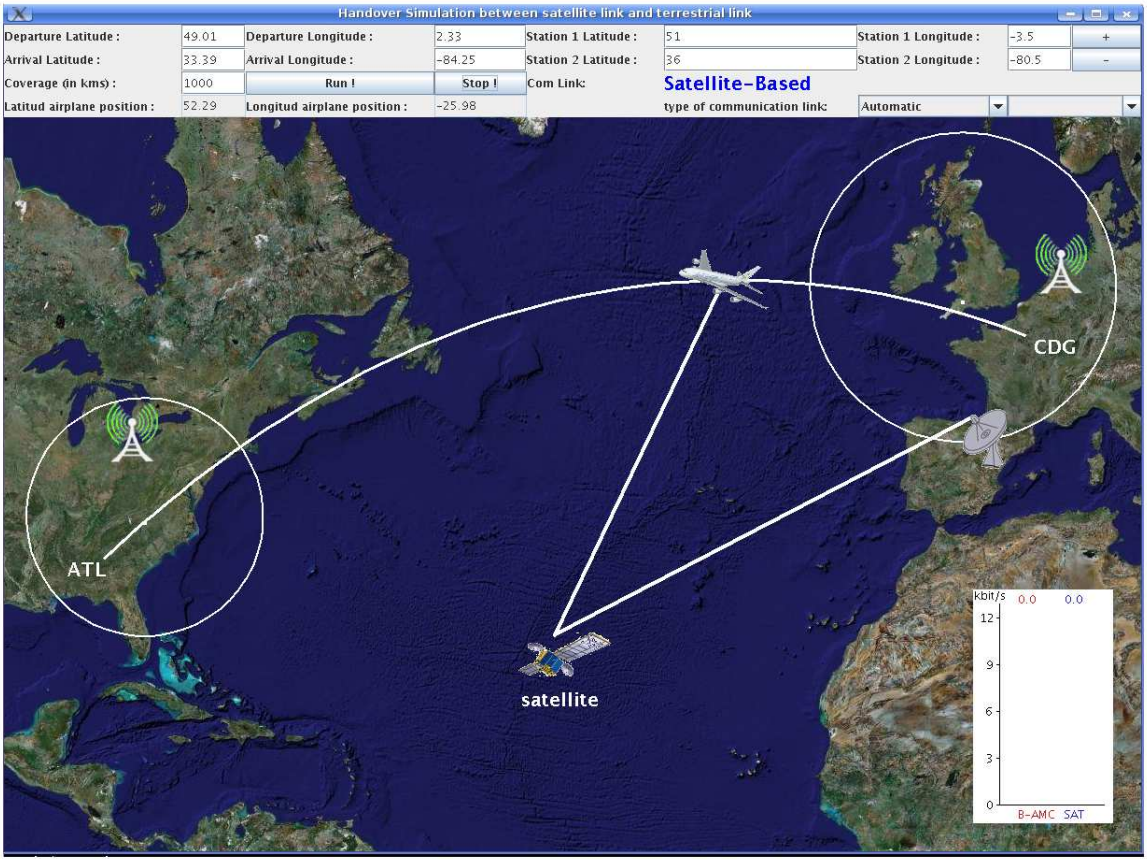

Figure 5.4: GUI map preview

introduced. Second, I will present a study of RTP traffic during the handover. Finally, I will also expose some results of the handover delays in TCP traffic.

5.5.1 Handover process

In a real handover situation, the decision taken to decide when to change from one point of attachment to another depends on networks conditions, signal strength, network coverage, link quality or on quality of service. When the receiver detects that one of those characteristics is under a certain level, the handover process begins. In Picture 5.5, the process is illustrated.

To study what exactly happens when Mobile Router makes a handover, some packet traces during the process has been analyzed, using the GUI. The links are simulated with the Linux traffic control tool, as explained before.

The Figure 5.6 shows a graphic of the handover process, with 4 entities of the network mobility system involved: the Mobile Router, the Access Router 1 (terrestrial link), the Access Router 2 (satellite link) and the Home Agent.

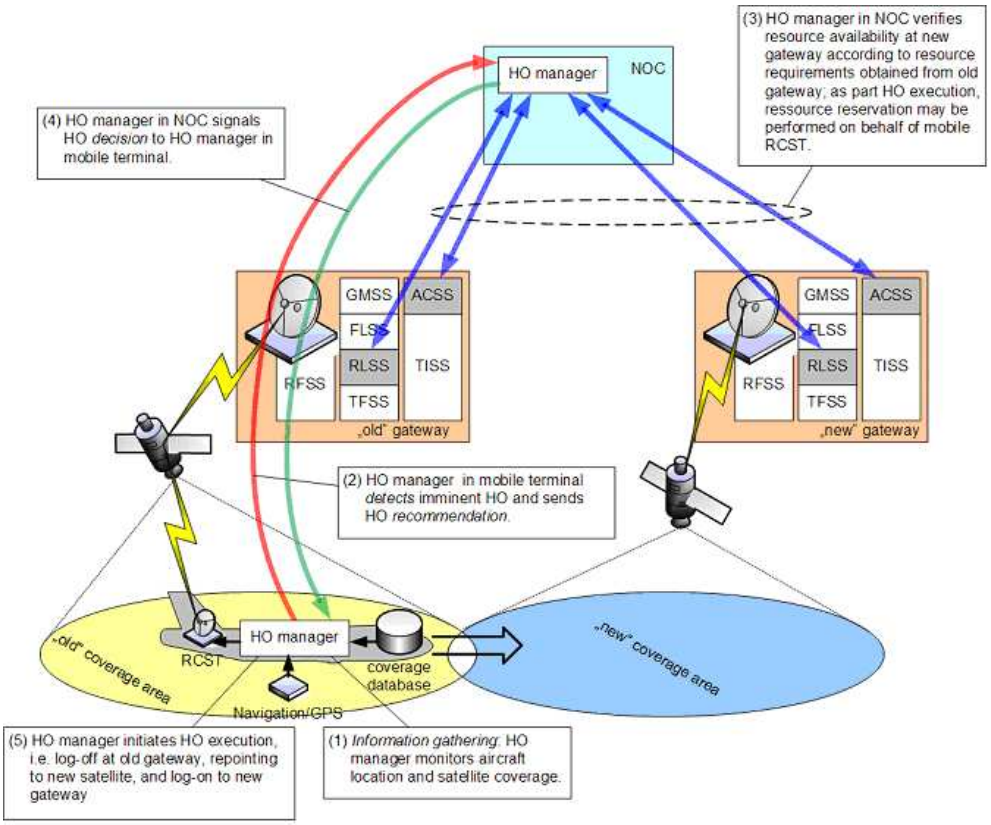

Figure 5.5: Handover process

At first, the Mobile Router sends packets through Access Router 1 (green line) as it uses the terrestrial link. When the GUI modifies the *ip6tables* rules in the Mobile Router for changing from terrestrial link to satellite link, it doesn't receive any Router Advertisement messages from the Access Router 1 anymore, and starts to receive Router Advertisement messages from the Access Router 2. It is translated in a handover detection by the Mobile Router, so then, it stops sending any packet to the Home Agent. Afterwards, a Duplicate Address Detection (DAD) process is achieved, in order to make sure of the unicity of the new Care-Of Address that the Mobile Router is going to have in the new foreign network.

In the new network, the Mobile Router sends a Binding Update to the Home Agent, to notify its new Care-Of Address. The Home Agent answers with a Binding Acknowledgement and changes immediately the destination Care-Of Address of the packets destined to the Mobile Router (green –> red). After that, the handover is done.

In the Access Router 2, a Neighbor Discovery/Advertisement process has to be achieved before sending any packet to the Mobile Router. Because of the 270 ms delay, this process lasts about 540 ms. During this time, we assume the packets (including the Binding Acknowledgement) are dropped in the Access Router 2, because of the queuing limit. However, the Mobile Router sends a second Binding Update, one second after the first one. And it receives the Binding Acknowledgement about 540 ms later. At this moment, the Mobile Router starts again to send

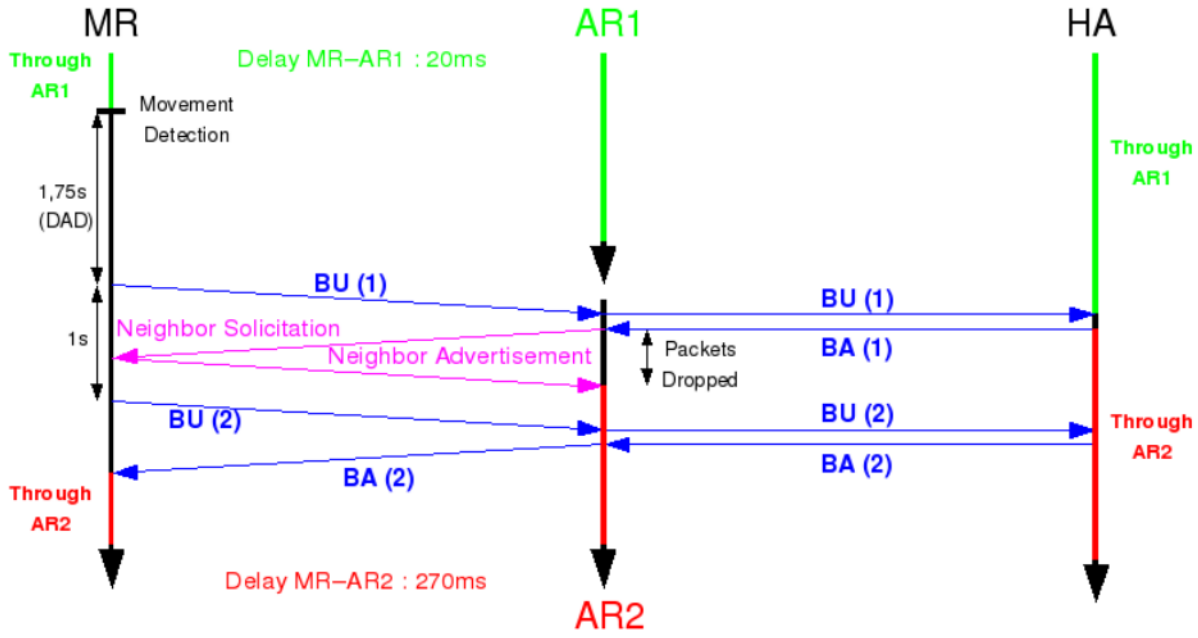

Figure 5.6: test-bed network architecture

packets to the Home Agent, through the Access Router 2.

But the whole process, between the movement detection and the receiving of the Binding Update, will last about 3,5 seconds, which is too much, and leads to a lot of packet losses.

5.5.2 VoIP call simulation

To see how are the delays through handovers during a phone call, I used the VoIP application. For this purpose, I studied the RTP packet traffic of a conversation between the Mobile Network Node and the Correspondent Node during a simulated flight.

To capture the packets sent between both machines, I used the wireshark program. To trigger the handovers I used the GUI while simulating the application.

To emulate the satellite link, I used the following commands using the *tc* tool in the Access Router 2:

```
tc qdisc add dev eth1 parent 1:1 handle 10: tbf rate 64kbit buffer 1600 limit 3000
```

The *tc qdisc* means Traffic Control Queuing Discipline. This command is used to limit the emission rate to 64 kbit/s on the eth1 interface.

```
tc qdisc add dev eth1 root handle 1:0 netem delay 270ms 10ms 25% loss 0.1% 20%
```

This command is used to add a delay of 270 ms on all packets going through the eth1 interface, with the following characteristics:

- A 10 ms fluctuation around the 270 ms delay (thus the delay can fluctuate between 260 ms and 280 ms)
- The packet delay depends on the 25% of the previous packet delay
- A packet loss probability of 0.1% with a correlation of 20% of the previous packet

The terrestrial link has a bandwidth limited to 512 kbit/s and a delay of 30ms.

Before starting the simulation, several daemons in the Mobile Router, Home Agent and Access Routers must be running in order to have Router Advertisement sent all the time, and the Mobile IPv6 daemon in the Mobile Router to be able to get the Care-of Address and create the tunnels.

At the beginning of the simulation, the airplane is supposed to be connected to the terrestrial link, as it has not yet took off. So, when the terrestrial Access Router starts advertising its network prefixes through Router Advertisements, the Mobile Router obtains a Care-of Address belonging to its network and hence the Home Agent will be informed by means of a Binding Update message. We can find out what Care-of Address the Mobile Router have by consulting its Binding Update list. In that case, and as it is depicted in Figure A.1 of appendix A, the Care-of Address is 2001:3:0:0:216:3eff:fe03:2ccf.

Entering the next command on the Mobile Router machine, we can see it:

```
# telnet localhost 7777
Trying 127.0.0.1...
Connected to localhost.localdomain (127.0.0.1).
Escape character is '^]'.
mip6d> bul
== BUL_ENTRY ==
Home address    2001:5c0:1104:7400:0:0:0:2
Care-of address 2001:3:0:0:216:3eff:fe03:2ccf
CN address      2001:5c0:1104:7400:0:0:0:1
 lifetime = 96,  delay = 91000
 flags: IP6_MH_BU_HOME IP6_MH_BU_ACK IP6_MH_BU_MR
 ack ready
 lifetime 92 / 96 seq 25357 resend 0 delay 91(after 88s)
```

As we can see, the home address is binded to the Care-of Address obtained. To check if the Binding Update and Binding Acknowledge packets were well routed through the terrestrial link and before getting out of the ground cover area, I just sniffed the interface eth0 of the Access Router 1 with *tcdump* tool. The result was:

```
# tcpdump ip6 -i eth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
2001:3::216:3eff:fe03:2ccf > 2001:5c0:1104:7400::1: DSTOPT mobility:
   BU seq#=18597 AH lifetime=96
2001:5c0:1104:7400::1 > 2001:3::216:3eff:fe03:2ccf: srcrt (len=2, type=2,
   segleft=1, [0]2001:5c0:1104:7400::2) mobility: BA status=0 seq#=18597
   lifetime=96
fe80::216:3eff:fe01:3579 > home_agent: ICMP6, neighbor solicitation,
   who has home_agent, length 32
home_agent > fe80::216:3eff:fe01:3579: ICMP6, neighbor advertisement,
   tgt is home_agent, length 24
fe80::230:5ff:fed3:51c8 > 2001:a:1::28: ICMP6, neighbor solicitation,
   who has 2001:a:1::28, length 32
2001:a:1::28 > fe80::230:5ff:fed3:51c8: ICMP6, neighbor advertisement,
   tgt is 2001:a:1::28, length 24
fe80::230:5ff:fed3:51c8 > fe80::216:3eff:fe01:3579: ICMP6, neighbor
   solicitation, who has fe80::216:3eff:fe01:3579, length 32
fe80::216:3eff:fe01:3579 > fe80::230:5ff:fed3:51c8: ICMP6, neighbor
   advertisement, tgt is fe80::216:3eff:fe01:3579, length 24
fe80::216:3eff:fe01:3579 > fe80::230:5ff:fed3:51c8: ICMP6, neighbor
   solicitation, who has fe80::230:5ff:fed3:51c8, length 32
fe80::230:5ff:fed3:51c8 > fe80::216:3eff:fe01:3579: ICMP6, neighbor
   advertisement, tgt is fe80::230:5ff:fed3:51c8, length 24
2001:3::216:3eff:fe03:2ccf > 2001:5c0:1104:7400::1: DSTOPT mobility:
   BU seq#=18598 AH lifetime=96
2001:5c0:1104:7400::1 > 2001:3::216:3eff:fe03:2ccf: srcrt (len=2, type=2,
   segleft=1, [0] 2001:5c0:1104:7400::2) mobility: BA status=0 seq#=18598
   lifetime=96
2001:3::216:3eff:fe03:2ccf > 2001:5c0:1104:7400::1: DSTOPT mobility:
   BU seq#=18599 AH lifetime=96
2001:5c0:1104:7400::1 > 2001:3::216:3eff:fe03:2ccf: srcrt (len=2, type=2,
   segleft=1, [0] 2001:5c0:1104:7400::2) mobility: BA status=0 seq#=18599
   lifetime=96
```

A Binding Update packet captured sent from the Mobile Router to Home Agent is shown next, as an example:

```
Internet Protocol Version 6
    Version: 6
    Traffic class: 0x00
    Flowlabel: 0x00000
    Payload length: 88
```

```
    Next header: IPv6 destination option (0x3c)
    Hop limit: 64
    Source address: 2001:2::216:3eff:fe53:847b (2001:2::216:3eff:fe53:847b)
    Destination address: 2001:5c0:1104:7400::1 (2001:5c0:1104:7400::1)

Destination Option Header
    Next header: Mobile IPv6 (0x87)
    Length: 2 (24 bytes)
    PadN: 4 bytes
    Option Type: 201 (0xc9) - Home Address Option
    Option Length: 16
    Home Address: 2001:5c0:1104:7400::2 (2001:5c0:1104:7400::2)

Mobile IPv6 / Network Mobility
    Payload protocol: IPv6 no next header (0x3b)
    Header length: 7 (64 bytes)
    Mobility Header Type: Binding Update (5)
    Reserved: 0x00
    Checksum: 0xe03f

    Binding Update
       Sequence number: 6525
       1... ....= Acknowledge (A) flag: Binding Acknowledgement
                  requested
       .1.. ....= Home Registration (H) flag: Home Registration
       ..0. ....= Link-Local Compatibility (L) flag: No Link-Local
                  Address Compatibility
       ...0 ....= Key Management Compatibility (K) flag: No Key
                  Management Mobility Compatibility
       .... 0...= Multiple Care of Address (M) flag: No Multiple
                  Care of Address Compatibility
       .... .1..= Mobile Router (R) flag: Mobile Router Compatibility
       Lifetime: 5 (20 seconds)

    Mobility Options
       PadN: 2 bytes
       Alternate care-of address: 2001:2::216:3eff:fe53:847b
                                  (2001:2::216:3eff:fe53:846b)
```

An example of a Binding Acknowledge packet:

```
Frame 5 (102 bytes on wire, 102 bytes captured)
Ethernet II, Src: 4b:ed:7d (00:16:3e:4b:ed:7d), Dst: 53:84:7b (00:16:3e:53:84:7b)
Internet Protocol Version 6
    Version: 6
```

```
    Traffic class: 0x00
    Flowlabel: 0x00000
    Payload length: 48
    Next header: IPv6 routing (0x2b)
    Hop limit: 63
    Source address: 2001:5c0:1104:7400::1 (2001:5c0:1104:7400::1)
    Destination address: 2001:2::216:3eff:fe53:847b (2001:2::216:3eff:fe53:847b)

Routing Header, Type 2
    Next header: Mobile IPv6 (0x87)
    Length: 2 (24 bytes)
    Type: 2
    Segments left: 1
    Home Address: 2001:5c0:1104:7400::2 (2001:5c0:1104:7400::2)
Mobile IPv6 / Network Mobility
    Payload protocol: IPv6 no next header (0x3b)
    Header length: 2 (24 bytes)
    Mobility Header Type: Binding Acknowledgement (6)
    Reserved: 0x00
    Checksum: 0x3c3e

    Binding Acknowledgement
       Satuts: Binding Update accepted (0)
       0... ....= Key Management Compatibility (K) flag: No Key
                  Management Mobility Compatibility
       .1.. ....= Mobile Router (R) flag: Mobile Router Compatibility
       Sequence number: 6525
       Lifetime: 5 (20 seconds)

    Mobility Options
       Link-Layer Address
          Option-Code: Unknown
          Link-layer address: 00
       PadN: 6 bytes
```

After proving the good behaviour of the system, I run the GUI to check that all Binding Updates and Binding Acknowledges packets were also well transmitted through the corresponding link during the flight simulation. Then, I made several VoIP calls to see how handovers affects to the transmission. For that, I captured the RTP packets in the Mobile Network Node and I traced the packets' delay of the simulated conversation. Next, I present some of the results obtained.

The Trace 5.7 shows the delay of almost 800 RTP packets captured during a simple one-way flight simulation. As I used the GUI to trigger the handovers, we can clearly identify in the graph

the 3 phases of the flight. At the beginning, the plane is over the European continent, so it is connected to the ground base through the terrestrial link. During this connection, the packets' delay is small, moving around 35 ms. Secondly, the plane changes its point of attachment (it flies over the ocean), so a handover is made to the satellite link. While being connected through the satellite, the packets' delay is around 280 ms. And thirdly, the plane flies over the American continent, and when it enters again to terrestrial cover area, another handover to the terrestrial link take place. The ordinate axis represents the delay in ms of every packet, represented by its sequence number, in the abscissa axis.

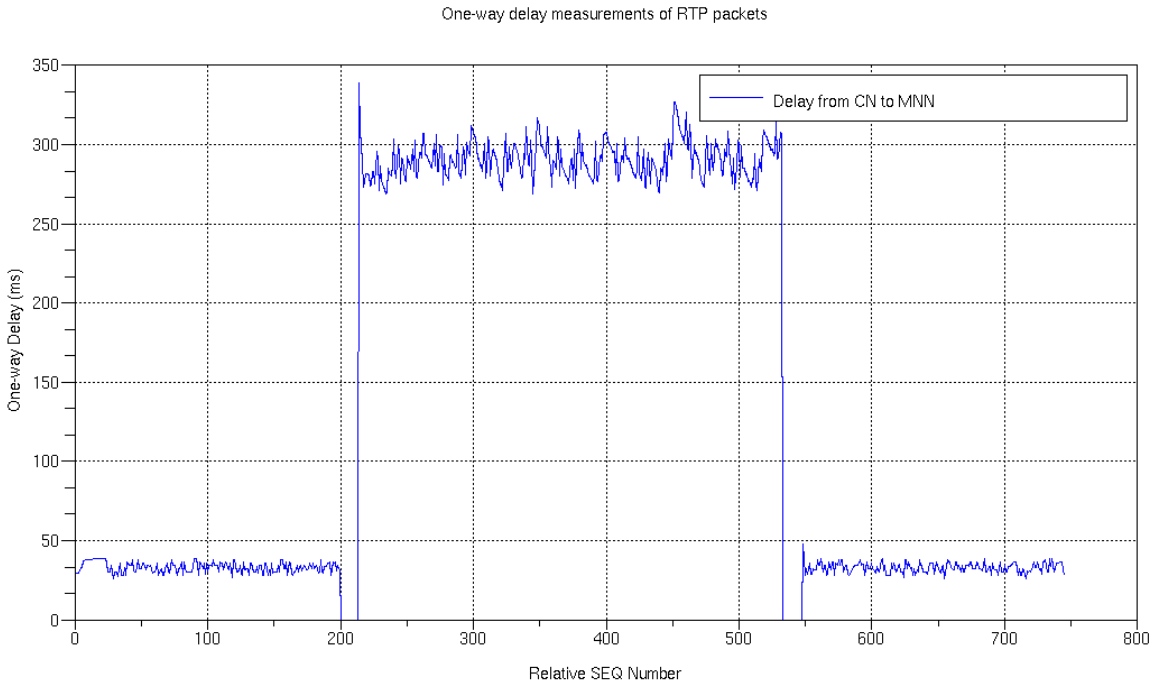

Figure 5.7: RTP packet delay during handovers

Looking at the graph we can observe that there are two gaps while changing the point of attachment. It is caused by a lost of packets during the handover. To represent the packets lost in the graph, I located them in the origin of the ordinate axis (0 ms in one-way delay). It doesn't mean that they have a delay of 0 seconds, but as it is impossible to draw the trace to infinite (what would be the correct representation) and because there is no packet with real 0 seconds delay, I opted to represent them in the bottom of the trace as it makes clearer the whole graph to understand. So, all packets located in 0ms delay line means that they are lost.

As we can see in detail in the Figure 5.8 and Figure 5.9, there are some packets lost when the plane changes the link connection. The 5.8 trace shows the packets' delays during the handover from the terrestrial link to the satellite link. As said before, the packets situated in the origin of the ordinate axis are the packets lost. The Table 5.1 lists the packets captured just before, during and after this handover with its correspondent delay. We can see that 14 packets are lost during the handover process. The 5.9 trace shows the packets' delays while changing from the

satellite link to the terrestrial one. In that case, 15 packets are lost, as we can see in the 5.2 Table.

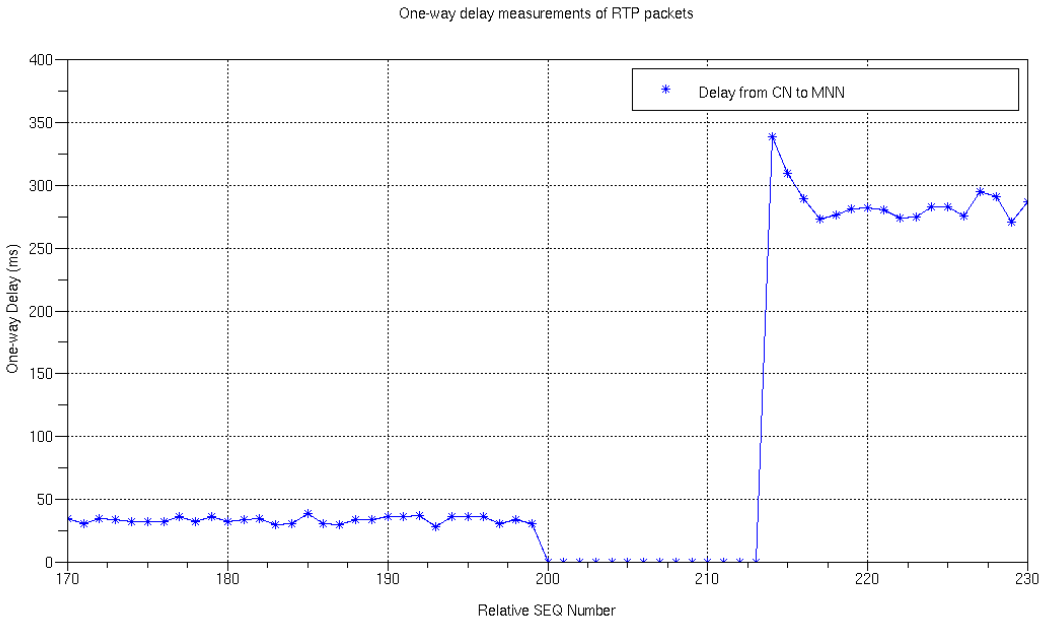

Figure 5.8: RTP packet delay detail during terrestrial to satellite link handover

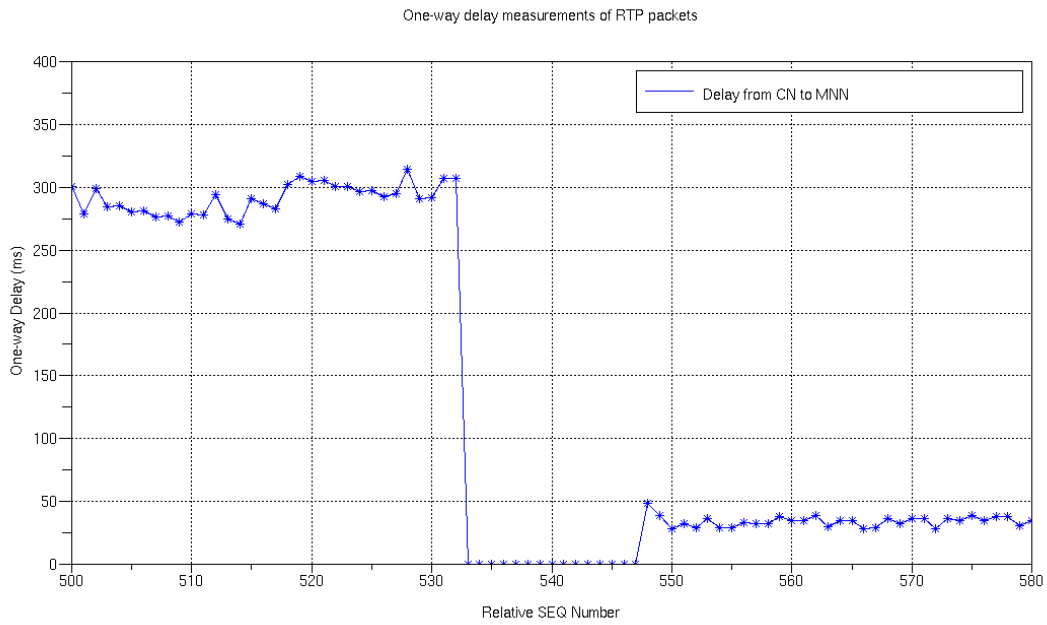

Figure 5.9: RTP packet delay detail during satellite to terrestrial handover

The Graphic 5.10 shows the throughput of the RTP packets that go from Correspondent Node to the Mobile Network Node, of the same call simulation. In that case, the handovers can be identified as the throughput decreases. In the second 25, a handover from terrestrial link to satellite one is done, and then, around second 65, another handover takes place, when the airplane enters to the terrestrial cover area again. As seen before, some packets are lost during the handovers, what is translated on a fall in the trace.

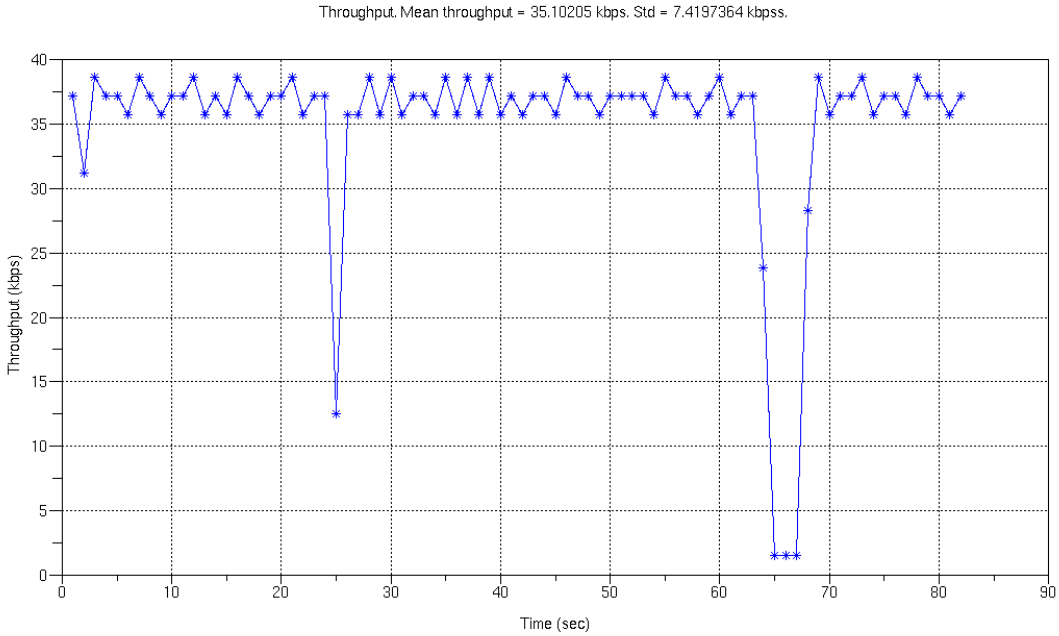

Figure 5.10: RTP packet throughput during handovers

In Appendix C, I present several RTP packet delay and thoughput measurements with different total number of packets captured during VoIP calls.

5.6 Simulation and study of the system using MCoA registration extension

To improve the results obtained before, I looked for a solution in order to reduce the delays and packet loss during the handovers. Creating more than one tunnel at the same time between the Mobile Router and the Home Agent with the aim to attach the mobile network over more than one foreign network, has been the solution studied.

Thus, I tried to integrate the MCoA registration extension protocol to the test-bed and see the new behaviour of the system's handovers. So this section has the objective to implement this new protocol and to see if it is worth to use this new protocol to improve the NEWSKY project benefits.

As I said before, I start up from using the NEMO Platform for Linux (NEPL) developed by the Nautilus6 group. For this platform, it exists the new MCoA extension version that is still under development and thus still contains some bugs and limitations.

Now, the objective is that moments before doing the handover between the two links, the mobile network will be already attached to the two foreign networks, since it is able to have 2 CoA at the same time. The process to bind the new CoA of the new foreign link with the Mobile Route's Home Address takes place before doing the handover. So, when finally the mobile network changes its point of attachment, there is no delay caused by the binding process, what means deleting the old CoA of the Home Agent's binding cache and establish the new bind.

The new issue that MCoA registration mechanism brings to the test-bed is that now we can attach the mobile network to the terrestrial and satellite link at the same time. Regarding to the handovers, it will permit the mobile network to have seamless moves from one link to the other, where the ongoing connections will almost not notice the change.

For that objective, I had to implement several things. First of all, I have modified the java code of the GUI. In that case, I programmed a new code which triggers the handovers in another way. Now, the airplane is connected just through one link while being inside its coverage area, but few meters before reaching the coverage bounds, the Mobile Router detects the new foreign network, so it starts to obtain the new CoA while still being connected and transmitting through the old link. Thus, just before reaching the limits of the first link area, the Mobile Router has already created the new tunnel through the new Access Router, binding its new Care-of Address with its home address in the Home Agent's binding cache. Then, when the airplane crosses the bounds, the Mobile Router just starts to transmit packets through the new link directly. In the Appendix B, the hole handover process is described as well as the GUI code used for that case.

In that case, I have just needed to implement the new extension in both Mobile Router and Home Agent. Now they will have to run another daemond while executing the simulations, with the configuration detailed below:

The *mip6d.conf* file for the Home Agent:

```
NodeConfig HA;
DebugLevel 10;
Interface "eth0";
HaAcceptMobRtr enabled;
HaAcceptMCoAReg enabled;

# Disable MPS/MPA
SendMobPfxAdvs enabled;
SendUnsolMobPfxAdvs enabled;

# MNP configuration
HaServedPrefix 2001:a:1::/64;
BindingAclPolicy 2001:a:1::1000 (2001:a:1::/64) MCoAReg allow;
DefaultBindingAclPolicy deny;
```

```
# IPsec configuration - NO IPSEC AT THE MOMENT
UseMnHaIPsec disabled;
KeyMngMobCapability disabled;
```

Mobile Router *mip6d.conf* file configuration:

```
NodeConfig MN;
DebugLevel 10;
DoRouteOptimizationCN disabled;
DoRouteOptimizationMN disabled;
SendMobPfxSols enabled;
UseCnBuAck disabled;

# It is used Explicit Mode
MobRtrUseExplicitMode enabled;
OptimisticHandoff enabled;

# The Binding Lifetime
MnMaxHaBindingLife 20;

Interface "eth1"
     Bid 200;
     BidPriority 20;
     Reliable true;


 Interface "eth2"
     Bid 100;
     BidPriority 10;
     Reliable true;


 #egress interface
 MnHomeLink "eth0"
      IsMobRtr enabled;
      HomeAgentAddress 2001:a:1::1000;
      HomeAddress 2001:5c0:1104:7400::2/64 (2001:5c0:1104:7400::/64);
      RegMultipleCoA enabled;
      IfMultipleCoA "eth1", "eth2";
```

So, before doing the simulation tests, I have just run the GUI to see if the Binding Updates and Binding Acknowledges were well sent, and the tunnels were created as was planed, setting more than one bind in the Home Agent's binding cache and Mobile Router's binding list. For

this, I used the wireshark in the Mobile Router to capture the packets during the new handover process.

Next, I expose the Home Agent's binding cache when the airplane is attached at both links, where the two Care-of Addresses can be identified.

```
# telnet localhost 7777
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
mip6d> bc
hoa 2001:5c0:1104:7400:0:0:0:2 status registered
 coa 2001:3:0:0:216:3eff:fe67:2a78 BID 200 BidPriority 20 flags AH--
 local 2001:5c0:1104:7400:0:0:0:1
 lifetime 12 / 96 seq 9779 unreach 0 mpa - / 621 retry 0
 MNP: 2001:5c0:1104:7401:0:0:0:0/64
hoa 2001:5c0:1104:7400:0:0:0:2 status registered
 coa 2001:2:0:0:216:3eff:fe53:847b BID 100 BidPriority 10 flags AH--
 local 2001:5c0:1104:7400:0:0:0:1
 lifetime 549 / 600 seq 12148 unreach 0 mpa - / 621 retry 0
 MNP: 2001:5c0:1104:7401:0:0:0:0/64
mip6d> exit
Connection closed by foreign host.
```

A curious thing I could see through the wireshark, is that when the airplanes detects a second foreign network and the new binding process starts, the Mobile Router obtains the new CoA of this foreign link and sends it to its Home Agent through the new tunnel, but the corresponding Binding Acknowledge is sent through the first tunnel. The Figure 5.11 tries to illustrate this fact. However, when the first tunnel is closed, the Binding Acknowledges start to be sent through the correct tunnel. That issue didn't affect our purposes but for further objectives of the protocol, it could be a problem which must be solved.

Next, I proceeded to test the new handover behaviour transmitting RTP packets between the Mobile Network Node and the Correspondent Node like done before, using the VoIP application.

5.6.1 VoIP call simulation with MCoA registration protocol

After capturing some RTP packets using wireshark program, I made some traces of the results. The next graphics show the packets delays during handovers in a simple flight simulation. The handovers have been triggered with the GUI again.

What is observed now in the traces is that during handovers there are almost no packets lost. The 5.12 Graphic shows the RTP packet delay where, as can be confirmed in the Table 5.3 and Table 5.4, only there are two packets lost during the handover from the terrestrial link to the satellite one during the call simulation. The handover from the satellite link to the ground one is totally smooth with no packet lost.

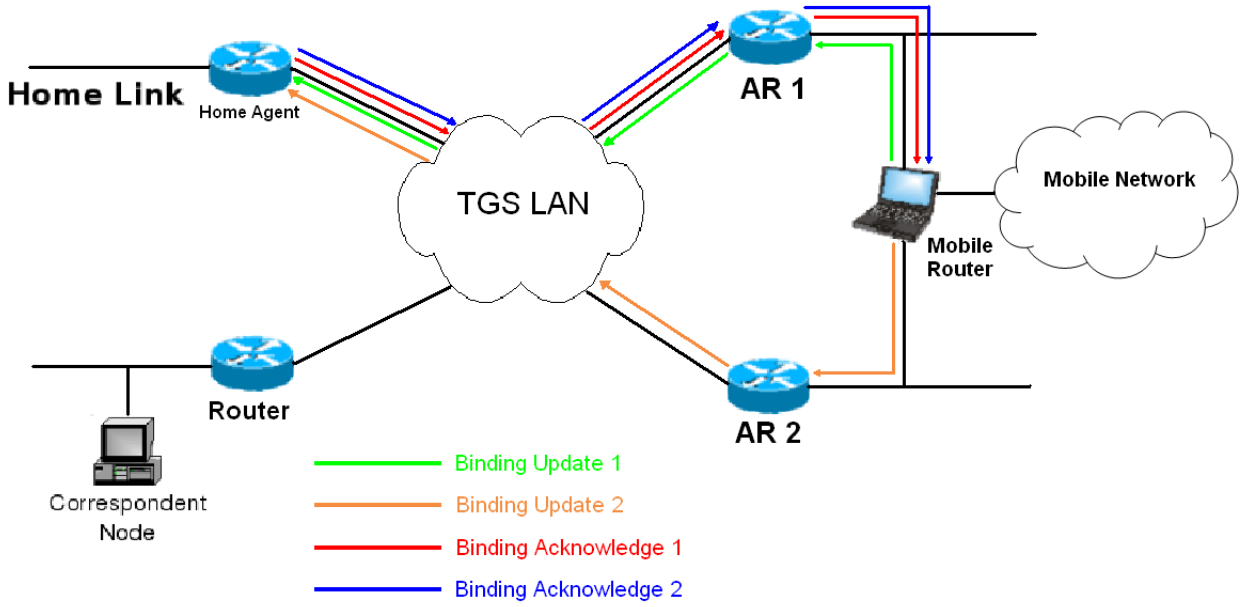

Figure 5.11: Binding messages routes bug illustration

The Trace 5.13 details the RTP packet delay during the terrestrial-satellite link handover. The 5.14 graph shows the packet delay during the other handover.

The Trace 5.15 shows the throughput of the RTP packets transmission. As we can observe, there is a fall around second 25 and second 65, but not as pronounced as before.

Another problem encountered is that the traffic generated locally on the Mobile Router doesn't get benefit from the Multiple Care-of Addresses registration.

However, with this new protocol, not only it is supposed to improve the handovers delay and packet loss, but also brings more advantages, like load balancing and policy routing.

If we could decide through which tunnel has to be sent every type of packet while two or more tunnels are established, we could share the traffic from the mobile network to the correspondent node. With the MCoA registration protocol, this policy routing can be managed with the ip6tables tool. The mechanism uses the BID that is assigned to each interface to mark the packets. Packets marked with BID X will be routed through the interface whose BID is X. If this interface is not available (i.e. it is down), then the packet will be routed through the most preferred interface (the one with the highest BID priority).

To assign the BID priority and to mark the packets with the BID, we have to use the ip6tables tool and the MARK target. Ip6tables rules must be done in the PREROUTING chain, in the mangle table.

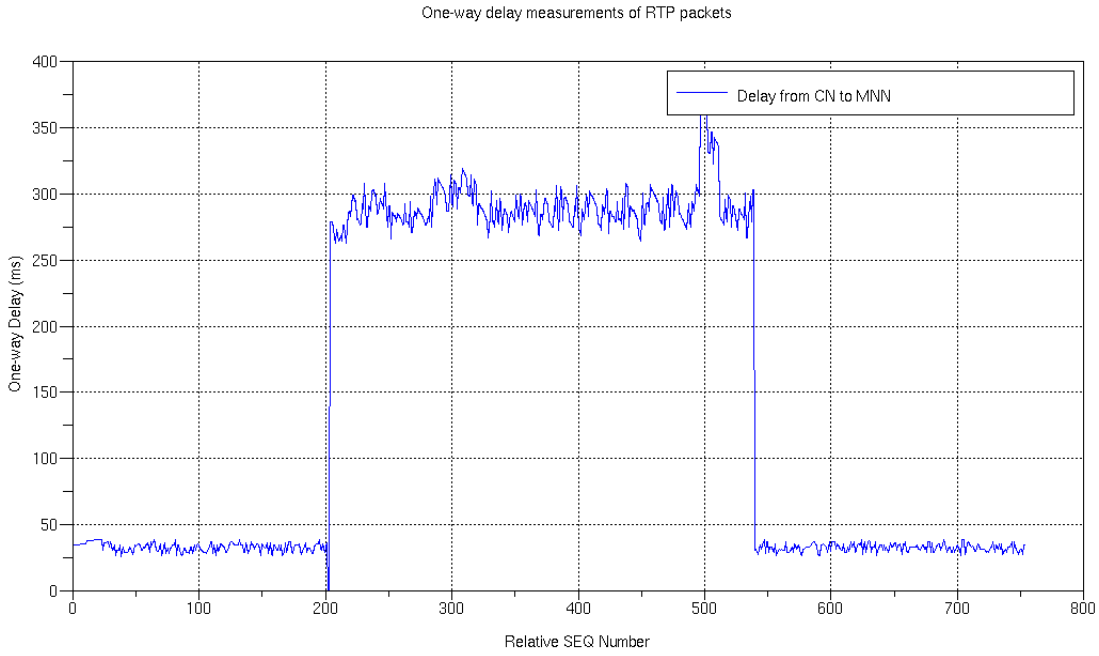

Figure 5.12: RTP packet delay during handover with multiple CoA registration

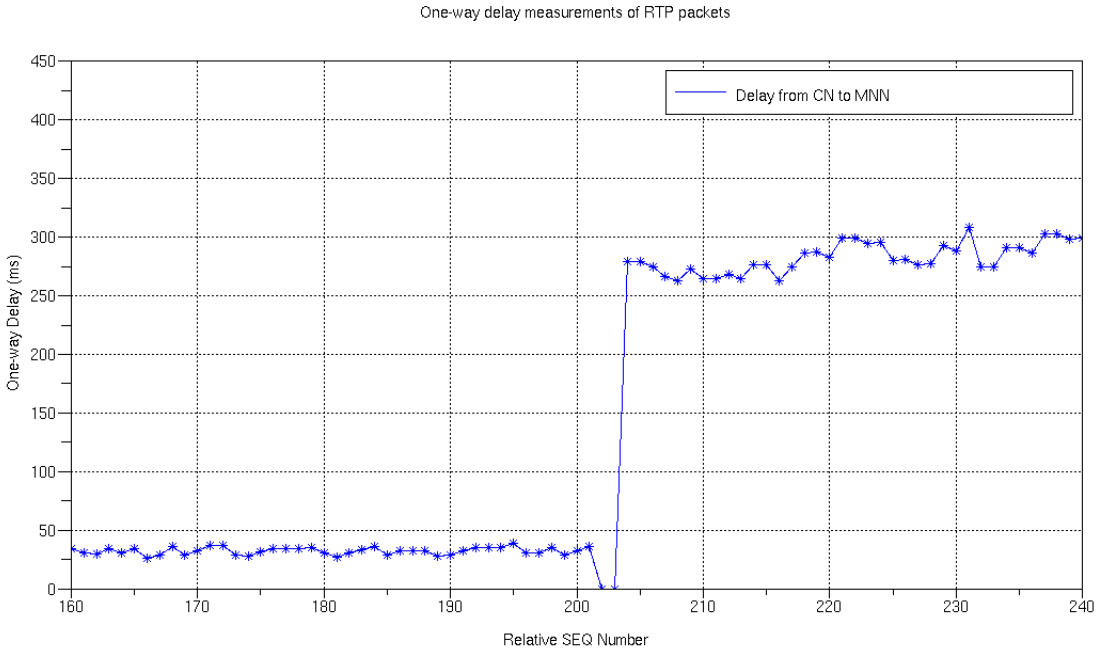

Figure 5.13: RTP packet delay in handover from terrestrial to satellite link with multiple CoA

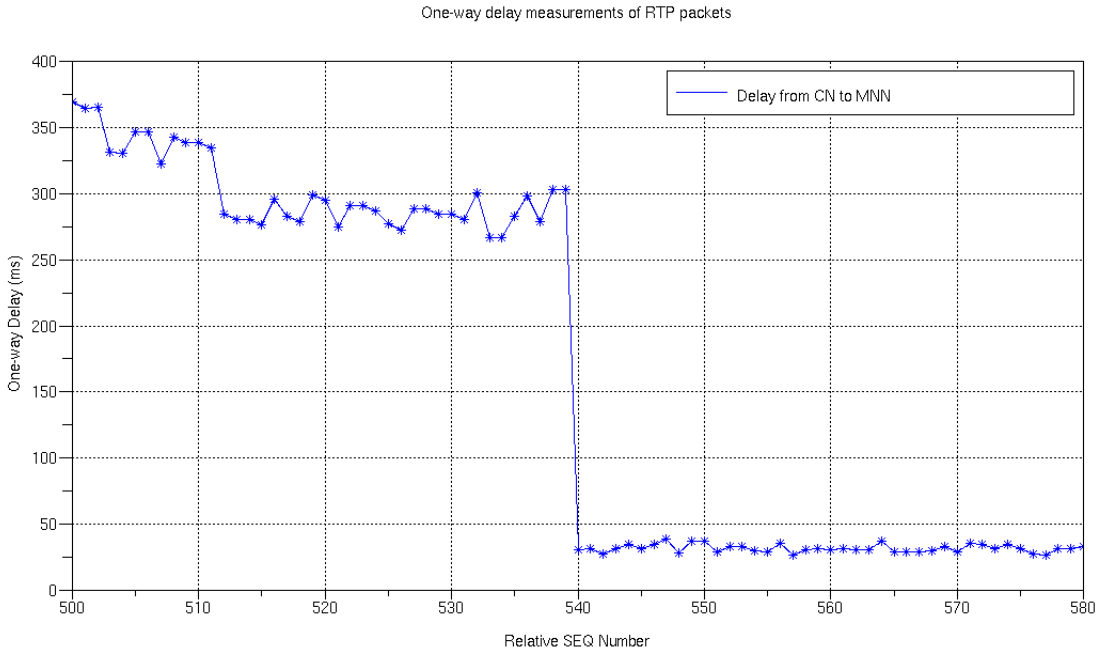

Figure 5.14: RTP packet delay in handover from satellite to terrestrial link with multiple CoA

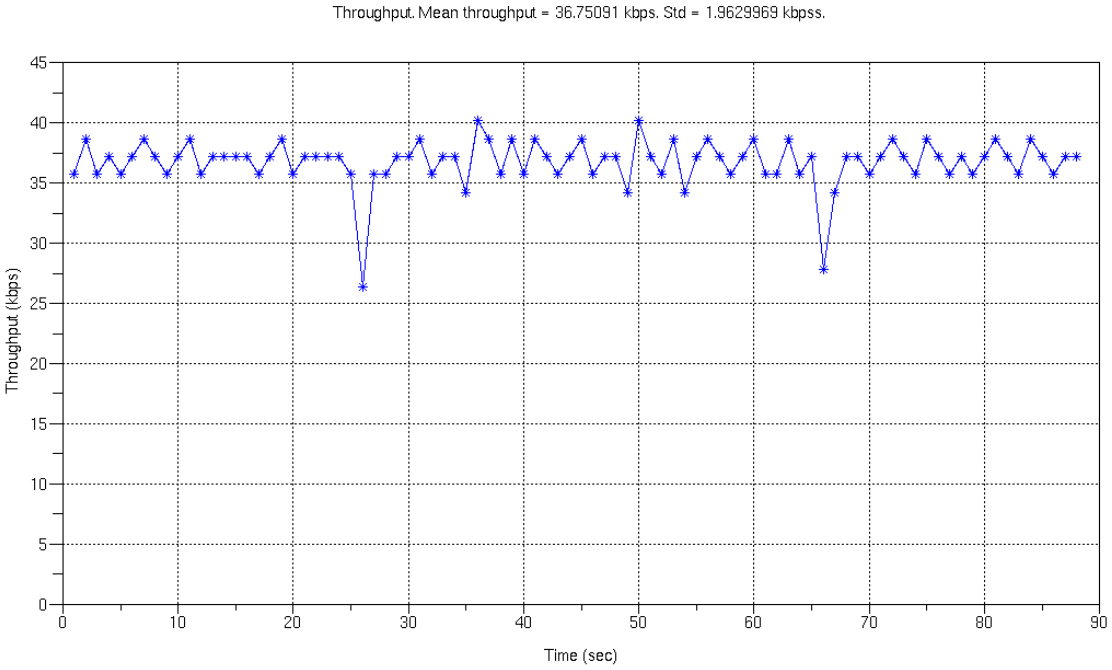

Figure 5.15: RTP packet thoughput with multiple CoA

For example, on the Mobile Router, to mark as 100 all icmpv6 packets whose destination is 2001:a:1::1, we can do:

```
ip6tables -A PREROUTING -t mangle
          -p icmpv6 --destination 2001:a:1::1
          -j MARK --set-mark 100
```

Those packets will be sent through the interface whose BID is 100. the same rule has to be created on the Home Agent:

```
ip6tables -A PREROUTING -t mangle
          -p icmpv6 --source 2001:a:d:1::1
          -j MARK --set-mark 100
```

The idea of using this policy routing in the test-bed would be sharing the traffic to have load balancing and send traffic through the 2 links. Also we could use the two tunnels to sent critical packets through the terrestrial link and the other via satellite link, as the first one is more reliable and faster.

The current limitation is that each rules created on the Mobile Router must be also created on the Home Agent. We plan to support in the future some policy exchange mechanism between the Mobile Router and the Home Agent in order to configure automatically the Home Agent.

Another limitation with actual versions is that only packets forwarded by the Mobile Router (for example packets sent by Mobile Network Node via the Mobile Router) can benefit from the policy routing. Packets generated by the Mobile Router itself will not be routed according to the rules. It is been currently working to improve the current situation to also allow the MR to benefit from the policy routing.

Table 5.1: RTP packets delay measurement during the handover from terrestrial to satellite link

Packets sequence number	**delay (ms)**	**Packets sequence number**	**delay (ms)**
2961	33.825	2993	lost
2962	34.310	2994	338.804
2963	29.799	2995	309.467
2964	30.212	2996	289.37
2965	38.155	2997	273.221
2966	30.271	2998	276.699
2967	29.558	2999	281.603
2968	33.497	3000	281.8
2969	33.820	3001	280.887
2970	35.893	3002	273.657
2971	35.848	3003	274.749
2972	36.555	3004	282.951
2973	28.033	3005	283.057
2974	36.503	3006	275.475
2975	35.993	3007	295.41
2976	36.476	3008	290.917
2977	30.581	3009	270.929
2978	34.090	3010	286.869
2979	30.600	3011	277.262
2980	lost	3012	272.78
2981	lost	3013	271.53
2982	lost	3014	268.784
2983	lost	3015	269.304
2984	lost	3016	284.777
2985	lost	3017	281.294
2986	lost	3018	279.948
2987	lost	3019	287.314
2988	lost	3020	303.49
2989	lost	3021	283.604
2990	lost	3022	278.773
2991	lost	3023	298.957
2992	lost	3024	288.765

Table 5.2: RTP packets delay measurement during handover from satellite to terrestrial link

Packets sequence number	**delay (ms)**	**Packets sequence number**	**delay (ms)**
3302	271.172	3329	lost
3303	291.109	3330	lost
3304	286.653	3331	lost
3305	282.681	3332	lost
3306	302.619	3333	lost
3307	309.081	3334	lost
3308	304.777	3335	lost
3309	305.247	3336	47.851
3310	300.742	3337	38.633
3311	300.739	3338	28.322
3312	296.704	3339	31.997
3313	297.19	3340	28.698
3314	292.69	3341	36.050
3315	294.788	3342	28.481
3316	314.722	3343	28.680
3317	290.89	3344	32.565
3318	291.579	3345	31.692
3319	307.517	3346	31.873
3320	307.066	3347	37.951
3321	lost	3348	34.203
3322	lost	3349	34.142
3323	lost	3350	38.411
3324	lost	3351	29.898
3325	lost	3352	34.623
3326	lost	3353	34.110
3327	lost	3354	28.186
3328	lost	3355	28.71

Table 5.3: RTP packet delay measurement during the handover from terrestrial to satellite link with MCoA registration

Packets sequence number	**delay (ms)**	**Packets sequence number**	**delay (ms)**
49630	30.514	49656	262.832
49631	32.952	49657	273.251
49632	36.421	49658	264.708
49633	28.924	49659	264.962
49634	32.241	49660	268.466
49635	32.234	49661	264.221
49636	32.375	49662	276.872
49637	28.037	49663	276.829
49638	28.421	49664	262.907
49639	32.611	49665	274.529
49640	34.789	49666	286.468
49641	35.009	49667	287.204
49642	34.968	49668	282.932
49643	38.559	49669	298.851
49644	31.023	49670	299.168
49645	30.481	49671	294.422
49646	34.945	49672	295.192
49647	29.117	49673	280.349
49648	32.609	49674	281.082
49649	36.109	49675	276.819
49650	lost	49676	277.528
49651	lost	49677	292.496
49652	278.978	49678	288.478
49653	279.174	49679	308.406
49654	274.327	49680	274.763
49655	266.843	49681	274.908

Table 5.4: RTP packet delay measurement during the handover from satellite to terrestrial link with MCoA registration

Packets sequence number	**delay (ms)**	**Packets sequence number**	**delay (ms)**
49958	338.957	49983	282.787
49959	334.446	49984	298.712
49960	284.862	49985	279.089
49961	280.339	49986	303.07
49962	280.552	49987	302.834
49963	276.202	49988	30.429
49964	296.135	49989	31.184
49965	282.72	49990	26.941
49966	279.269	49991	30.931
49967	299.203	49992	34.414
49968	294.746	49993	31.181
49969	274.763	49994	34.854
49970	290.693	49995	38.802
49971	291.222	49996	28.375
49972	286.713	49997	37.038
49973	277.032	49998	36.983
49974	272.504	49999	28.427
49975	288.964	50000	33.064
49976	288.408	50001	33.006
49977	284.869	50002	29.375
49978	284.339	50003	28.609
49979	280.742	50004	34.997
49980	300.415	50005	26.49
49981	266.654	50006	30.827
49982	266.405	50007	31.258

Chapter 6

Proposed future work and conclusions

In this chapter the remaining and future work is explained. Next, there is a final conclusion of the project.

6.1 Remaining and future work

This document has explained how to design a system which could permit mobile networks to change its point of attachment to the Internet without interrupting its ongoing transmissions. Besides the final system worked well, there are still some work to do. Some tasks have to be performed in the direction of continuing the development and integration of the test-bed, which will be used not only for NEWSKY, but also for potential future projects. Issues like integration of Wimax system as one of the access networks or the study and implementation IPv6 header compression algorithms for efficient transmission of IPv6 packets in bandwith-limited satellie link, are some of the tasks that the NEWSKY test-bed will try to implement soon as it can improve the whole aeronautic communications system.

The work remained to be done within the NEWSKY Laboratory test-bed project can be summarized in:

Quality of Service: concerning the application part in the project, there is at the moment no quality of service implemented. This remains to be done, in order to set priorities between the different kinds of traffic (Voice over IP, messaging, data streaming).

Codecs: voice over IP can be achieved using a lot of different kinds of codecs, with different compression rates and specific characteristics. That's why some studies remain to be carried out, in order to choose the best suited codecs to our system.

Overhead in the satellite link: the overhead of RTP packets in the satellite link is extremely high (3 or 4 times the size of the encapsulated data). That's why we need to find solutions in order to reduce the overhead, because it puts down the efficiency of the link in a dramatic way. The solution could be first to use another codec that puts more data in each RTP packet. Also header compression can be a good possibility.

BGAN Tests: for the moment, the satellite link was only simulated thanks to the use of the Linux traffic control tool. But for the final demonstration, a real satellite link will have to be used. Therefore, some tests with the real BGAN modem have to be carried out.

Weather Streaming Application: the weather streaming application has not been developed yet. Up until now, we have only made FTP transfers in order to test the data transfer application.

6.2 Conclusions

The objective of this project is to improve aeronautical traffic management, designing a system which, employing the latest technologies and mechanisms, could permit pilots and passangers in airplanes connect to Internet with minimum delay and packet loss as well as have access to new applications and services.

In this document, it has been discussed the configuration of a test-bed in order to implement a network inside airplanes and also the protocols that have been employed. First of all, it has exposed the reasons why we have used the IPv6 protocol for the system configuration instead of the IPv4. Then, it has dealt with different methods to get mobile networks. Finally, through the test-bed, the different kind of mobile protocols have been tested.

Considering the Internet Protocol issue, we have compared the two most used in recent days: IPv4 vs IPv6. We have seen that the second one not only brings many more advantages and improvements but also IPv6 supports the newest mobile protocols that IPv4 can't handles. But as the satellite link does not support IPv6 yet, and the project was focused in this IP protocol version, we had to implement different mechanisms to traverse IPv6 packets through IPv4 link. The mechanisms chosen has been the L2TP and NAPT-PT.

About the network mobility, firstly we thought of implementing the mobile IPv6 protocol in every node was the solution. The MIPv6 is a protocol which permits a node to move through different networks keeping its connections alive. However, thousands of passangers can occupy the same airplane so, if every node uses this protocol, the Mobile Router would collapse with many Binding Updates and Binding Acknowledges. Also, it means that every node should support the protocol.

The solution studied in this document is NEMO which stands for network mobility. It permits entire mobile networks to change its point-of-attachment through different links while maintaining its ongoing connections. In this mechanism, only the router supports the mobile protocol, so the different nodes are not aware of the mobility and doesn't know about the handovers.

Another important issue of the project is the handover process. With NEMO protocol, when a mobile network changes its connection link, many packets are lost. To solve that problem,

different solutions has been exposed, but the MCoA registration protocol has been the one studied and implemented. This mobile extension allows to create more than one tunnel at the same time between the Mobile Router and the Home Agent, so the information can travel through several links.

Some simulations with the test-bed has been carried out to compare the different handover's behaviours using some applications implemented in Linux. First, a VoIP call simulation was done using only NEMO protocol. We have seen that during handovers between ground and satellite links, some packets were lost. In the other hand, during VoIP call simulations using MCoA registration protocol we could see that the number of packets lost was reduced considerably. In conclusion, the Multiple Care-of Address registration permits a seamless handover, reducing delays packets lost. Also, thanks to the policy of routes, this mechanism permits to have load balancing and share the traffic through several links at the same time.

Finally, I would like to remark that while doing this intership I obtained many benefits. I have learned a lot of interesting things concerning aeronautical communications. Moreover, working in an enterprise enriched myself and above all the fact of realizing my thesis in Germany. And mention that I have enjoyed taking part of an excellent teamwork, very nice and helpful.

Bibliography

[1] www.newsky-fp6.eu

[2] A. Jahn and M. Holzbock and others, "Evolution of aeronautical communications for personal and multimedia services",*Communications Magazine, IEEE.*

[3] G. Maral and M. Bousquet, *Satellite Communications Systems.* Wiley, third ed., 1998.

[4] Y.-W. Chen and J.-M. Shih, "Binding updates for mobile networks by using multicast mechanism in IPv6 environment", *Advanced Information Networking and Applications, 2005. AINA 2005. 19th International Conference on*, vol. 2, pp. 790 – 795, March 2005.

[5] www.triagnosys.com

[6] M. A. Miller, *Implementing IPv6.* M&T Books, 1998.

[7] P. Loshin, *IPv6 Clearly Explained.* Morgan Kaufmann, 1999.

[8] S. Hagen, *IPv6 Essentials.* O'Reilly, 2006.

[9] www.isc.org

[10] S. Deering and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", *RFC 2460*, 1998.

[11] R. Hinden and S. Deering, "IP Version 6 Addressing Architecture", *RFC 4291*, 2006.

[12] A. Conta and S. Deering, "Internet Control Message Protocol (ICMPv6) for the IP Version 6 Specification", *RFC 1885*, 1995.

[13] http://ipv6.com/articles/general/ICMPv6.htm

[14] R. Droms, J. Bound, B. Volz, T. Lemon, C. Perkins, M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", *RFC 3315*, 2003.

[15] W. Townsley, A. Valencia, A. Rubens, G. Pall, G. Zorn, B. Palter, "Layer Two Tunneling Protocol "L2TP"", *RFC 2661*, 1999.

[16] G. Tsirtsis, P. Srisures, "Network Address Translation - Protocol Translation (NAT-PT) ", *RFC 2766*, 2000.

[17] D. Johnson, C. Perkins, and J. Arkko, "Mobility Support in IPv6", *RFC 3775*, 2004.

[18] Charles M. Kozierok, *The TCP/IP Guide: A Comprehensive, Illustrated Internet Protocols Reference*, No Starch Press, 2004.

[19] James D. Solomon, *Mobile IP: The Internet*. PTR Prentice Hall, 1997.

[20] V. Devarapalli *et al.*, "Network Mobility (NEMO) basic support protocol", *RFC 3963*, 2005.

[21] Reuters, *United to offer Web in the air*, 2005.

[22] T., K. Uehara, *Connecting Automobiles to the Internet*, 2002.

[23] C. Perkins, and J. Arkko, "IP Mobility Support for IPv4", *RFC 3344*, 2004.

[24] www.nautilus6.org/doc/nepl-howto/

[25] R. Wakikawa, V. Devarapalli, G. Tsirtsis, T. Ernst, K. Nagami, "Multiple Care-of Addresses Registration", *RFC 5648*, 2009.

[26] www.nautilus6.org

[27] www.inmarsat.com/bgan

[28] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, E. Schooler, "SIP: Session Initiation Protocol", *RFC 3261*, 2002.

Appendix A

Test-bed architecture and address configuration

A.1 Test-bed architecture

The figure A.1 depicts the scenario used to simulate the applications and to take the handover tests. It is the final architecture of the test-bed with all machines and networks involved with its respective IPv6 addresses.

The ground and satellite links are the ones used for the handover experiment. The TGS LAN simulates the Internet or Air Traffic Control, where the Access Routers, Home Agent and the Correspondent Node are attached.

The IPv6 addresses are assigned as follows:

TGS LAN is emulating Air Traffic Control (ATC) ground network, and has 2001:a:1::/64 prefix. The Home Network has 2001:5c0:1104:7400::/64 prefix, with HA at 2001:5c0:1104:7400::1/64 as the default router. Air Traffic Service (ATS) Mobile Network has 2001:5c0:1104:7401::/64 prefix, with MR at 2001:5c0:1104:7401::1/64 as the default router. Access Network 1 has 2001:3::/64 prefix, with AR1 at 2001:3::2/64 as the default router. Access Network 2 has 2001:2::/64 prefix, with AR2 at 2001:2::2/64 as the default router.

A.2 Mobile Router Address configuration

In the case that we use only the NEMO protocol, the Mobile Router and the Home Agent only create a unique tunnel for every Care-of Address. As we can see in the address configuration of both machines, the tunnel created is the virtual interface ip6tnl1 and tap1. They are created when the mipv6 daemon is running.

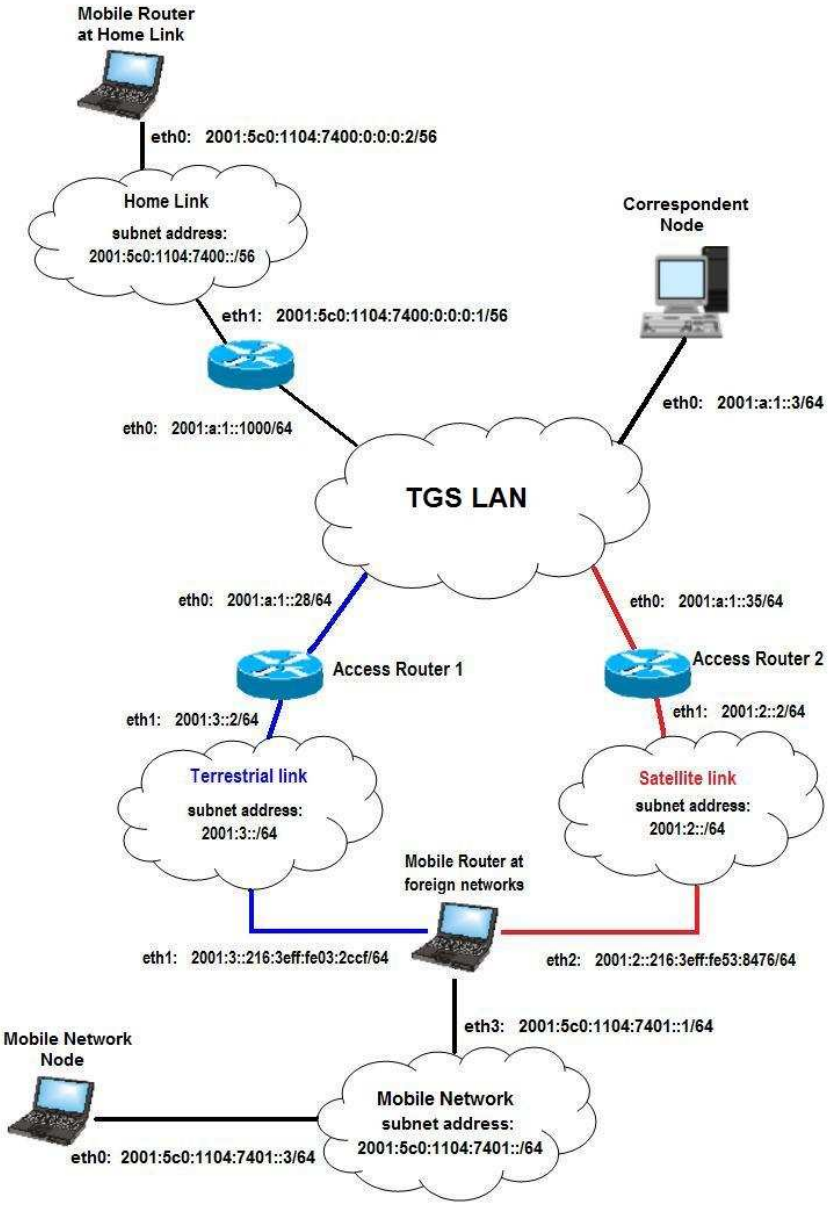

Figure A.1: Final test-bed architecture

```
eth0      Link encap:Ethernet  HWaddr 00:16:3E:1B:98:6E
          inet addr:172.21.0.6  Bcast:172.21.255.255  Mask:255.255.0.0
          inet6 addr: fe80::216:3eff:fe1b:986e/64 Scope:Link
```

```
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:148 errors:0 dropped:0 overruns:0 frame:0
          TX packets:123 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:17032 (16.6 KiB)  TX bytes:14785 (14.4 KiB)
          Interrupt:5

eth1      Link encap:Ethernet  HWaddr 00:16:3E:7A:70:0E
          inet6 addr: fe80::216:3eff:fe7a:700e/64 Scope:Link
          inet6 addr: 2001:3::216:3eff:fe03:2ccf/64 Scope:Global
  UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:91 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:9464 (9.2 KiB)  TX bytes:688 (688.0 b)
          Interrupt:7 Base address:0x2100

eth2      Link encap:Ethernet  HWaddr 00:16:3E:53:84:7B
          inet6 addr: fe80::216:3eff:fe53:847b/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:32 errors:0 dropped:0 overruns:0 frame:0
          TX packets:13 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3328 (3.2 KiB)  TX bytes:2398 (2.3 KiB)
          Interrupt:10 Base address:0x4200

eth3      Link encap:Ethernet  HWaddr 00:16:3E:54:8C:D3
          inet addr:172.21.0.9  Bcast:172.21.255.255  Mask:255.255.0.0
          inet6 addr: 2001:5c0:1104:7401::1/64 Scope:Global
          inet6 addr: fe80::216:3eff:fe54:8cd3/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:29 errors:0 dropped:0 overruns:0 frame:0
          TX packets:34 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:5642 (5.5 KiB)  TX bytes:3796 (3.7 KiB)
          Interrupt:11 Base address:0x6300

eth4      Link encap:Ethernet  HWaddr 00:16:3E:57:C6:EF
          inet6 addr: fe80::216:3eff:fe57:c6ef/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:688 (688.0 b)
          Interrupt:5 Base address:0x8400
```

```
eth5      Link encap:Ethernet  HWaddr 00:16:3E:57:C6:FF
          inet6 addr: fe80::216:3eff:fe57:c6ff/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:688 (688.0 b)
          Interrupt:7 Base address:0xa500

ip6tnl1   Link encap:UNSPEC  HWaddr 20-01-00-03-00-00-00-00-00-00-00-00-00-00-00-00
          inet6 addr: 2001:5c0:1104:7400::2/128 Scope:Global
          inet6 addr: fe80::216:3eff:fe1b:986e/64 Scope:Link
          UP POINTOPOINT RUNNING NOARP  MTU:1460  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:11 errors:0 dropped:0 overruns:0 frame:0
          TX packets:11 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:732 (732.0 b)  TX bytes:732 (732.0 b)
```

The eth0 is the one used to address to the Home Link. In the simulation we don't use it at all as the plane is always away attached at foreign links but it is necessary for the Home Agent to bind the Care-of address to the Home Address as explained in chapter 4.

A.3 Home Agent Address configuration

```
eth0      Link encap:Ethernet  HWaddr 00:30:05:D3:51:C8
          inet addr:172.21.0.3  Bcast:172.21.0.255  Mask:255.255.255.0
          inet6 addr: 2001:a:1::1000/64 Scope:Global
          inet6 addr: fe80::230:5ff:fed3:51c8/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:382 errors:0 dropped:0 overruns:0 frame:0
          TX packets:255 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:49772 (48.6 KB)  TX bytes:36657 (35.7 KB)
          Interrupt:19 Base address:0xc000
```

```
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)

tap1      Link encap:Ethernet  HWaddr 00:FF:B6:DB:DD:52
          inet6 addr: fe80::2ff:b6ff:fedb:dd52/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:140 overruns:0 carrier:0
          collisions:0 txqueuelen:500
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

Appendix B

GUI source code and flow chart

B.1 GUI source code

To design the graphical user interface for the test-bed demonstration, I worked on java language. Next, the source code of the GUI program, with the MCoA protocol implemented in the system, is exposed.

```
import java.awt.* ;
import java.awt.event.*;
import java.io.*;
import java.lang.*;
import java.util.*;
import javax.swing.*;

/* The DemoGUI class is the main class of the application. It's a JFrame which contains all the
elements of the GUI. Moreover, this class implements Runnable, which means it is able to be run
as a Thread, calling the run() method. The static methods GreatCircle, Coverage, isReachable and
GeoToPixel belong to this class too.*/

public class DemoGUI extends JFrame implements Runnable
{
private JTextField lati_d, longi_d, lati_sd, longi_sd;
private JTextField lati_a, longi_a, lati_sa, longi_sa;

private JTextField cov_dist;
private JLabel current_time, com_link;

private Container contPane;
private Panell pan;
```

```
private Thread displayRates, anim;

private boolean planeReachable, planeReachableOld;
private int sleep_time;



public DemoGUI()
{
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
         setTitle("Handover Simulation between satellite link and terrestrial link");
contPane = getContentPane();

JLabel lati_d_lab = new JLabel("Departure Latitude :");
lati_d = new JTextField("49.01");
JLabel longi_d_lab = new JLabel("Departure Longitude :");
longi_d = new JTextField("2.33");
JLabel lati_sd_lab = new JLabel("Station 1 Latitude :");
lati_sd = new JTextField("51");
JLabel longi_sd_lab = new JLabel("Station 1 Longitude :");
longi_sd = new JTextField("-3.5");
JButton plus_button = new JButton("+");
plus_button.addActionListener(new ActionListener()
{
public void actionPerformed(ActionEvent e)
{
if(sleep_time > 1)
sleep_time-=2;
}
});

JLabel lati_a_lab = new JLabel("Arrival Latitude :");
lati_a = new JTextField("33.39");
JLabel longi_a_lab = new JLabel("Arrival Longitude :");
longi_a = new JTextField("-84.25");
JLabel lati_sa_lab = new JLabel("Station 2 Latitude :");
lati_sa = new JTextField("36");
JLabel longi_sa_lab = new JLabel("Station 2 Longitude :");
longi_sa = new JTextField("-80.5");
JButton minus_button = new JButton("-");
minus_button.addActionListener(new ActionListener()
{
public void actionPerformed(ActionEvent e)
{
sleep_time+=2;
}
});

JLabel cov_dist_lab = new JLabel("Coverage (in kms) :");
cov_dist = new JTextField("1000");

JButton run_button = new JButton("Run !");
run_button.addActionListener(new ActionListener()
{
public void actionPerformed(ActionEvent e)
{
```

```
if(!anim.isAlive())
runAnimation();
}
});

JButton stop_button = new JButton("Stop !");
stop_button.addActionListener(new ActionListener()
{
public void actionPerformed(ActionEvent e)
{
if(anim.isAlive())
stopAnimation();
}
});

JLabel com_link_lab = new JLabel("Com Link: ");
com_link = new JLabel();


contPane.setLayout(new GridBagLayout());
GridBagConstraints c = new GridBagConstraints();

c.fill = GridBagConstraints.BOTH;
c.weightx = 1;
contPane.add(lati_d_lab, c);
contPane.add(lati_d, c);
contPane.add(longi_d_lab, c);
contPane.add(longi_d, c);
contPane.add(lati_sd_lab, c);
contPane.add(lati_sd, c);
contPane.add(longi_sd_lab, c);
contPane.add(longi_sd, c);
c.gridwidth = GridBagConstraints.REMAINDER;
contPane.add(plus_button, c);

c.gridwidth = 1;
contPane.add(lati_a_lab, c);
contPane.add(lati_a, c);
contPane.add(longi_a_lab, c);
contPane.add(longi_a, c);
contPane.add(lati_sa_lab, c);
contPane.add(lati_sa, c);
contPane.add(longi_sa_lab, c);
contPane.add(longi_sa, c);
c.gridwidth = GridBagConstraints.REMAINDER;
contPane.add(minus_button, c);

c.gridwidth = 1;
contPane.add(cov_dist_lab, c);
contPane.add(cov_dist, c);
contPane.add(run_button, c);
contPane.add(stop_button, c);
contPane.add(com_link_lab, c);
c.gridwidth = GridBagConstraints.REMAINDER;
contPane.add(com_link, c);
```

```
pan = new Panneau();
pan.setPreferredSize(new Dimension(1218, 785));
// c.ipady = 800;
contPane.add(pan, c);
calculateLines();
pan.setPlane(GeoToPixel(Double.parseDouble(lati_d.getText()),
                  Double.parseDouble(longi_d.getText())));

displayRates = new DisplayRates(pan);
displayRates.start();

sleep_time = 25;
anim = new Thread(this);

pack();
setVisible(true);
}


public void runAnimation()
{
calculateLines();
lati_d.setEditable(false);
lati_a.setEditable(false);
longi_d.setEditable(false);
longi_a.setEditable(false);
lati_sd.setEditable(false);
lati_sa.setEditable(false);
longi_sd.setEditable(false);
longi_sa.setEditable(false);
cov_dist.setEditable(false);
anim = new Thread(this);
anim.start();
}


public void stopAnimation()
{
anim.stop();
lati_d.setEditable(true);
lati_a.setEditable(true);
longi_d.setEditable(true);
longi_a.setEditable(true);
lati_sd.setEditable(true);
lati_sa.setEditable(true);
longi_sd.setEditable(true);
longi_sa.setEditable(true);
cov_dist.setEditable(true);
}


public void run()
{
int N=500;
int j;
```

```
int direction = 1;
pan.setDirection(direction);

while(true)
{
for(int i=0;i<=N;i++)
{
j = i;
if(direction == -1)
j = N-i;
double[] pos_avio = GreatCircle(Double.parseDouble(lati_d.getText()),
Double.parseDouble(longi_d.getText()), Double.parseDouble(lati_a.getText()), Double.parseDouble
(longi_a.getText()), (double)j/N); pan.setPlane (GeoToPixel(pos_avio[0], pos_avio[1]));

planeReachableOld = planeReachable;
if(isReachable(Double.parseDouble(lati_sd.getText()),
Double.parseDouble(longi_sd.getText()), pos_avio[0], pos_avio[1], Double.parseDouble
(cov_dist.getText())) || isReachable(Double.parseDouble(lati_sa.getText()),
Double.parseDouble(longi_sa.getText()), pos_avio[0], pos_avio[1],Double.parseDouble(cov_dist.getText())))
planeReachable = true;
else
planeReachable = false;

if((j == 0) || (planeReachable != planeReachableOld))
{
if(planeReachable)
{
com_link.setForeground(Color.red);
com_link.setText("Ground-Based");
try
{
Runtime.getRuntime().exec("ssh root@192.168.10.74 ./entering_ground.sh");
}
catch(IOException e){}
}
else
{
com_link.setForeground(Color.blue);
com_link.setText("Satellite-Based");
try
{
Runtime.getRuntime().exec("ssh root@192.168.10.74 ./entering_satellite.sh");
}
catch(IOException e){}
}
}

pan.repaint();
try{Thread.sleep(sleep_time);}
catch (InterruptedException e){}
}
direction *= -1;
pan.setDirection(direction);
}
}
```

```
public void calculateLines()
{
int N=50;
double[] geo;
Point[] points = new Point[N+1];
Point[] departure_cov = new Point[N+1];
Point[] arrival_cov = new Point[N+1];

for(int i=0; i<=N; i++)
{
geo = GreatCircle(Double.parseDouble(lati_d.getText()), Double.parseDouble
(longi_d.getText()), Double.parseDouble(lati_a.getText()),
Double.parseDouble(longi_a.getText()), (double)i/N);
points[i] = GeoToPixel(geo[0], geo[1]);
departure_cov[i] = Coverage(Double.parseDouble(lati_sd.getText()),
Double.parseDouble(longi_sd.getText()), Double.parseDouble(cov_dist.getText()),
 (double)i/N*2*Math.PI);
arrival_cov[i] = Coverage(Double.parseDouble(lati_sa.getText()),
Double.parseDouble(longi_sa.getText()), Double.parseDouble(cov_dist.getText()), (double)i/N*2*Math.PI);
}

Point pos_sd = GeoToPixel(Double.parseDouble(lati_sd.getText()),
                Double.parseDouble(longi_sd.getText()));
Point pos_sa = GeoToPixel(Double.parseDouble(lati_sa.getText()),
                Double.parseDouble(longi_sa.getText()));

pan.setLines(points, departure_cov, arrival_cov, pos_sd, pos_sa);
}


public static double[] GreatCircle(double lat1, double lon1, double lat2,
double lon2, double alpha)
{
double x1 = Math.cos(lon1*Math.PI/180)*Math.cos(lat1*Math.PI/180); // cartesian
                coordinates on a radius 1 sphere (u1 vector)
double y1 = Math.sin(lon1*Math.PI/180)*Math.cos(lat1*Math.PI/180);
double z1 = Math.sin(lat1*Math.PI/180);

double x2 = Math.cos(lon2*Math.PI/180)*Math.cos(lat2*Math.PI/180); // cartesian
                coordinates on a radius 1 sphere (u2 vector)
double y2 = Math.sin(lon2*Math.PI/180)*Math.cos(lat2*Math.PI/180);
double z2 = Math.sin(lat2*Math.PI/180);

double Psi = Math.acos(x1*x2 + y1*y2 + z1*z2); // earth center angle in radians
                between (lon1, lat1) and (lon2, lat2)

double x3 = (x2 - x1*Math.cos(Psi))/Math.sin(Psi); // orthogonal angle to u1, in
                the (u1, u2) plane, on the u2 side (u3 vector)
double y3 = (y2 - y1*Math.cos(Psi))/Math.sin(Psi);
double z3 = (z2 - z1*Math.cos(Psi))/Math.sin(Psi);

double phi = alpha*Psi;

double x = x1*Math.cos(phi) + x3*Math.sin(phi);
double y = y1*Math.cos(phi) + y3*Math.sin(phi);
```

```
double z = z1*Math.cos(phi) + z3*Math.sin(phi);

double lat = Math.asin(z)*180/Math.PI;
double lon = Math.atan2(y,x)*180/Math.PI;

return new double[]{lat,lon}; // latitude and longitude in degrees
}


public static Point Coverage(double latc, double lonc, double dist, double alpha)
{
double xc = Math.cos(lonc*Math.PI/180)*Math.cos(latc*Math.PI/180);
double yc = Math.sin(lonc*Math.PI/180)*Math.cos(latc*Math.PI/180);
double zc = Math.sin(latc*Math.PI/180);

double x1 = -xc*Math.sin(latc*Math.PI/180)/Math.cos(latc*Math.PI/180);
double y1 = -yc*Math.sin(latc*Math.PI/180)/Math.cos(latc*Math.PI/180);
double z1 = (1 - zc*Math.sin(latc*Math.PI/180))/Math.cos(latc*Math.PI/180);

double x2 = yc*z1 - zc*y1;
double y2 = zc*x1 - xc*z1;
double z2 = xc*y1 - yc*x1;

double x3 = x1*Math.cos(alpha) + x2*Math.sin(alpha);
double y3 = y1*Math.cos(alpha) + y2*Math.sin(alpha);
double z3 = z1*Math.cos(alpha) + z2*Math.sin(alpha);

double angle = (double)dist/20000*Math.PI;

double x = xc*Math.cos(angle) + x3*Math.sin(angle);
double y = yc*Math.cos(angle) + y3*Math.sin(angle);
double z = zc*Math.cos(angle) + z3*Math.sin(angle);

double lat = Math.asin(z)*180/Math.PI;
double lon = Math.atan2(y,x)*180/Math.PI;

return GeoToPixel(lat,lon);
}


public static boolean isReachable(double lat_station, double lon_station, double lat_plane,
        double lon_plane, double dist_max)
{
double xs = Math.cos(lon_station*Math.PI/180)*Math.cos(lat_station*Math.PI/180);
double ys = Math.sin(lon_station*Math.PI/180)*Math.cos(lat_station*Math.PI/180);
double zs = Math.sin(lat_station*Math.PI/180);

double xp = Math.cos(lon_plane*Math.PI/180)*Math.cos(lat_plane*Math.PI/180);
double yp = Math.sin(lon_plane*Math.PI/180)*Math.cos(lat_plane*Math.PI/180);
double zp = Math.sin(lat_plane*Math.PI/180);

double Psi = Math.acos(xs*xp + ys*yp + zs*zp);
double dist = Psi/Math.PI*20000;

if(dist<dist_max)
return true;
```

```
else
return false;
}


public static Point GeoToPixel(double lat, double lon) // latitude and longitude in degrees
{
return new Point((int)(47 + 11.3821 * (lon + 89.6539)),
(int)(612-650.4559*(Math.log((1+Math.sin(lat*Math.PI/180))/
(1-Math.sin(lat*Math.PI/180)))/2-0.4021)));
}


public static void main(String[] args)
{
        //Schedule a job for the event-dispatching thread:
        //creating and showing this application's GUI.
javax.swing.SwingUtilities.invokeLater(new Runnable(){
public void run()
{
new DemoGUI();
}
         });
}
}


/* The Panell class is a JPanel, which contains all the graphical information of the GUI.
It corresponds to the area of the GUI, where the map is displayed. */
class Panell extends JPanel
{
private Image map;
private Image avio;
private Image avioni;

private Point[] route;
private Point[] departure_cov;
private Point[] arrival_cov;
private Point pos_sd;
private Point pos_sa;
private Point pos_avio;

private int direction;
private int bottom_left_x, bottom_left_y, axe_x;
private int rate1_px, rate2_px;
private String rate1_kbit, rate2_kbit;
private boolean bigScale;


public Panell()
{
map = getToolkit().getImage("atlantic.png");
avio = getToolkit().getImage("avio.png");
avioni = getToolkit().getImage("avioni.png");
direction = 1;
bottom_left_x = 1038;
```

```
bottom_left_y = 755;
axe_x = 28;
rate1_px = 0;
rate2_px = 0;
rate1_kbit = "";
rate2_kbit = "";
}

public void setLines(Point[] route, Point[] departure_cov, Point[] arrival_cov,
        Point pos_sd, Point pos_sa)
{
this.route = route;
this.departure_cov = departure_cov;
this.arrival_cov = arrival_cov;
this.pos_sd = pos_sd;
this.pos_sa = pos_sa;
}

public void setPlane(Point pos_avio)
{
this.pos_avion=pos_avio;
}

public void paint(Graphics g)
{
g.drawImage (map, 0, 0, this); // Draws the satellite image (background
                image of the Panell class)


g.setColor(Color.WHITE); // Draws the white rectangle where the rates are displayed
g.fillRect(bottom_left_x,bottom_left_y-250,150,250);

g.setColor(new Color(192,32,32)); // draws the rate information of AR1 (in red)
g.drawString(rate1_kbit, bottom_left_x+axe_x+20, bottom_left_y-235);
g.fillRect(bottom_left_x+axe_x+20,bottom_left_y-20-rate1_px,20,rate1_px);
g.drawString("B-AMC", bottom_left_x+axe_x+20,bottom_left_y-5);

g.setColor(new Color(32,32,192)); // draws the rate information of AR2 (in blue)
g.drawString(rate2_kbit, bottom_left_x+axe_x+20+50, bottom_left_y-235);
g.fillRect(bottom_left_x+axe_x+20+50,bottom_left_y-20-rate2_px,20,rate2_px);
g.drawString("SAT", bottom_left_x+axe_x+20+50,bottom_left_y-5);

g.setColor(Color.BLACK); // draws the horizontal and vertical axes, and the "kbit/s"
                caption (in black)
g.drawLine(bottom_left_x+axe_x,bottom_left_y-20,bottom_left_x+140,bottom_left_y-20);
g.drawLine(bottom_left_x+axe_x,bottom_left_y-20,bottom_left_x+axe_x,bottom_left_y-235);
g.drawString("kbit/s", bottom_left_x+1,bottom_left_y-239);

if(bigScale) // draws the scale in case of a high rate (up to 600 kbit/s) (in black)
{
g.drawLine(bottom_left_x+axe_x-3,bottom_left_y-20,bottom_left_x+axe_x,
                        bottom_left_y-20);
g.drawString("0", bottom_left_x+14,bottom_left_y-15);
g.drawLine(bottom_left_x+axe_x-3,bottom_left_y-70,bottom_left_x+axe_x,
                        bottom_left_y-70);
g.drawString("150", bottom_left_x,bottom_left_y-65);
```

```
g.drawLine(bottom_left_x+axe_x-3,bottom_left_y-120,bottom_left_x+axe_x,
                        bottom_left_y-120);
g.drawString("300", bottom_left_x,bottom_left_y-115);
g.drawLine(bottom_left_x+axe_x-3,bottom_left_y-170,bottom_left_x+axe_x,
                        bottom_left_y-170);
g.drawString("450", bottom_left_x,bottom_left_y-165);
g.drawLine(bottom_left_x+axe_x-3,bottom_left_y-220,bottom_left_x+axe_x,
                        bottom_left_y-220);
g.drawString("600", bottom_left_x,bottom_left_y-215);
}
else // draws the scale in case of a low rate (up to 12 kbit/s) (in black)
{
g.drawLine(bottom_left_x+axe_x-3,bottom_left_y-20,bottom_left_x+axe_x,
                        bottom_left_y-20);
g.drawString("0", bottom_left_x+14,bottom_left_y-15);
g.drawLine(bottom_left_x+axe_x-3,bottom_left_y-70,bottom_left_x+axe_x,
                        bottom_left_y-70);
g.drawString("3", bottom_left_x+14,bottom_left_y-65);
g.drawLine(bottom_left_x+axe_x-3,bottom_left_y-120,bottom_left_x+axe_x,
                        bottom_left_y-120);
g.drawString("6", bottom_left_x+13,bottom_left_y-115);
g.drawLine(bottom_left_x+axe_x-3,bottom_left_y-170,bottom_left_x+axe_x,
                        bottom_left_y-170);
g.drawString("9", bottom_left_x+14,bottom_left_y-165);
g.drawLine(bottom_left_x+axe_x-3,bottom_left_y-220,bottom_left_x+axe_x,
                        bottom_left_y-220);
g.drawString("12", bottom_left_x+7,bottom_left_y-215);
}


Graphics2D g2d = (Graphics2D)g;
g2d.setColor(Color.RED); // draws the "CDG" and "ATL" airport captions, and the
                Great Circle Route (in red)
g2d.setFont(new Font("SansSerif",Font.BOLD,20));
g2d.drawString("CDG", (int)route[0].getX()+10, (int)route[0].getY()+20);
g2d.drawString("ATL", (int)route[route.length-1].getX()-40, (int)route[route.
                length-1].getY()+20);
g2d.setStroke(new BasicStroke( 3.0f ));
//for(int i=0; i<route.length-1; i++)
// g2d.drawLine((int)route[i].getX(), (int)route[i].getY(), (int)route[i+1].getX(),
(int)route[i+1].getY());

g2d.setColor(new Color(192,32,32)); // draws the 2 base stations, and the coverage
                circles (in red)
g2d.setStroke(new BasicStroke( 4.0f ));
g2d.drawLine((int)pos_sd.getX(), (int)pos_sd.getY(), (int)pos_sd.getX(), (int)pos_sd.getY());
g2d.drawLine((int)pos_sa.getX(), (int)pos_sa.getY(), (int)pos_sa.getX(), (int)pos_sa.getY());
g2d.setStroke(new BasicStroke( 2.0f ));
for(int i=0; i<route.length-1; i++)
{
g2d.drawLine((int)departure_cov[i].getX(), (int)departure_cov[i].getY(),
(int)departure_cov[i+1].getX(), (int)departure_cov[i+1].getY());
g2d.drawLine((int)arrival_cov[i].getX(), (int)arrival_cov[i].getY(),
                        (int)arrival_cov[i+1].getX(), (int)arrival_cov[i+1].getY());
}
```

```
if(direction == 1) // draws the plane in the direction Paris -> Atlanta
g.drawImage (avioni, (int)pos_avio.getX()-50, (int)pos_avio.getY()-34, this);
else // draws the plane in the direction Atlanta -> Paris
g.drawImage (avio, (int)pos_avio.getX()-50, (int)pos_avio.getY()-34, this);
}

public void setRate1_px(int rate1_px)
{
this.rate1_px = rate1_px;
}

public void setRate2_px(int rate2_px)
{
this.rate2_px = rate2_px;
}

public void setRate1_kbit(String rate1_kbit)
{
this.rate1_kbit = rate1_kbit;
}

public void setRate2_kbit(String rate2_kbit)
{
this.rate2_kbit = rate2_kbit;
}

public void setBigScale(boolean bigScale)
{
this.bigScale = bigScale;
}

public void setDirection(int direction)
{
this.direction = direction;
}



}


/* The DisplayRates class is a Thread which simply reads the files rate1.txt and rate2.txt
every 0.5 second. Then, it analyses the rate to determine the scale and sends all the information
to the Panneau object for display.*/
class DisplayRates extends Thread
{
Panell pan;
int rate1_int, rate2_int;
int count;

public DisplayRates(Panell pan)
{
this.pan = pan;
count=0;
}
```

```
public void run()
{
while(true)
{
try
{
FileReader fichier1 = new FileReader("rate1.txt");
FileReader fichier2 = new FileReader("rate2.txt");
StreamTokenizer entree1 = new StreamTokenizer(fichier1);
StreamTokenizer entree2 = new StreamTokenizer(fichier2);
entree1.nextToken();
entree2.nextToken();
rate1_int = (int)entree1.nval;
rate2_int = (int)entree2.nval;
fichier1.close();
fichier2.close();
}
catch(IOException e) {}

if((rate1_int > 12500) || (rate2_int > 12500))
count = 4;
if(rate1_int > 625000)
rate1_int = 625000;
if(rate2_int > 625000)
rate2_int = 625000;

if(count == 0)
{
pan.setBigScale(false);
pan.setRate1_px(rate1_int/60);
pan.setRate2_px(rate2_int/60);
}
else
{
count--;
pan.setBigScale(true);
pan.setRate1_px(rate1_int/3000);
pan.setRate2_px(rate2_int/3000);
}
pan.setRate1_kbit(String.valueOf((float)((rate1_int+50)/100)/10));
pan.setRate2_kbit(String.valueOf((float)((rate2_int+50)/100)/10));
pan.repaint();
try{sleep(500);}
catch(InterruptedException e){}
}
}
}
```

The *entering_ ground.sh* and *entering_ satellite.sh* are two scripts executed in the Mobile Router in order to change the **ip6tables** when the airplane changes its point of attachment between the terrestrial and satellite link and vice versa. This GUI program must be run in the Mobile Router, or at least, to access the Mobile Router through *ssh* command with the aim to be able to run the scripts.

B.2 GUI Flow Chart

To better understand the handovers process through the GUI program, I just created a flow chart of the different positions the airplane goes throw. The process it done with MCoA registration protocol implemented.

MCoA application in the GUI program

Observations:

- 2 tunnels without load balancing.
- This application is only available for AUTOMATIC mode in the GUI.
- The connection/disconnection of the satellite-based tunnel is done without depending on any power level coverage parameter, but just taking in mind a desired distance from the airplane to the ground-based zone bounds.
- The connection with the satellite is done when the airplane reaches a certain distance from the departing position and before reaching the ground coverage area bounds.

The process has 5 positions according to the connections of the airplane. The flow chart B.1 shows these different positions.

Information:

Position 0 (p=0): the airplane has not taken off yet. The connections with ground and satellite are closed. In the GUI demonstration code, eth1 (terrestrial link) and eth2 (satellite link) are down.

Position 1 (p=1): the airplane has connection with ground base and it is flying inside the ground-based area and has no connection with the satellite. Eth1 is up and eth2 is down (Figure B.2).

Position 2 (p=2): The connection with satellite is opened. From that moment, we are using MCoA application with 2 tunnels. Eth1 and eth2 are up. But data packets should be only transmitted through the ground-base connection, while through the satellite tunnel should only be Binding Update packets and Binding Acknowledge packets. This means that the most privileged tunnel is the ground-based one (eth1).

Position 3 (p=3): The airplane has reached the ground-based connection area boundary and from then, the ground connection (eth1 down) is no longer operative. So, data packets should be now transmitted through tunnel 2 (thought satellite connection). That's the moment where we should observe the improved seamless handover (Figure B.3).

Position 4 (p=4): The airplane has reached the destination ground-based connection boundary. From now, the 2 tunnels are opened again. If no changes are done, now the data packets are transmitted though eth1 again, but the 2 tunnels send binding updates (Figure B.4).

Then, it comes again the position 1.

The parameter p is the one used in the java code to define the positions.

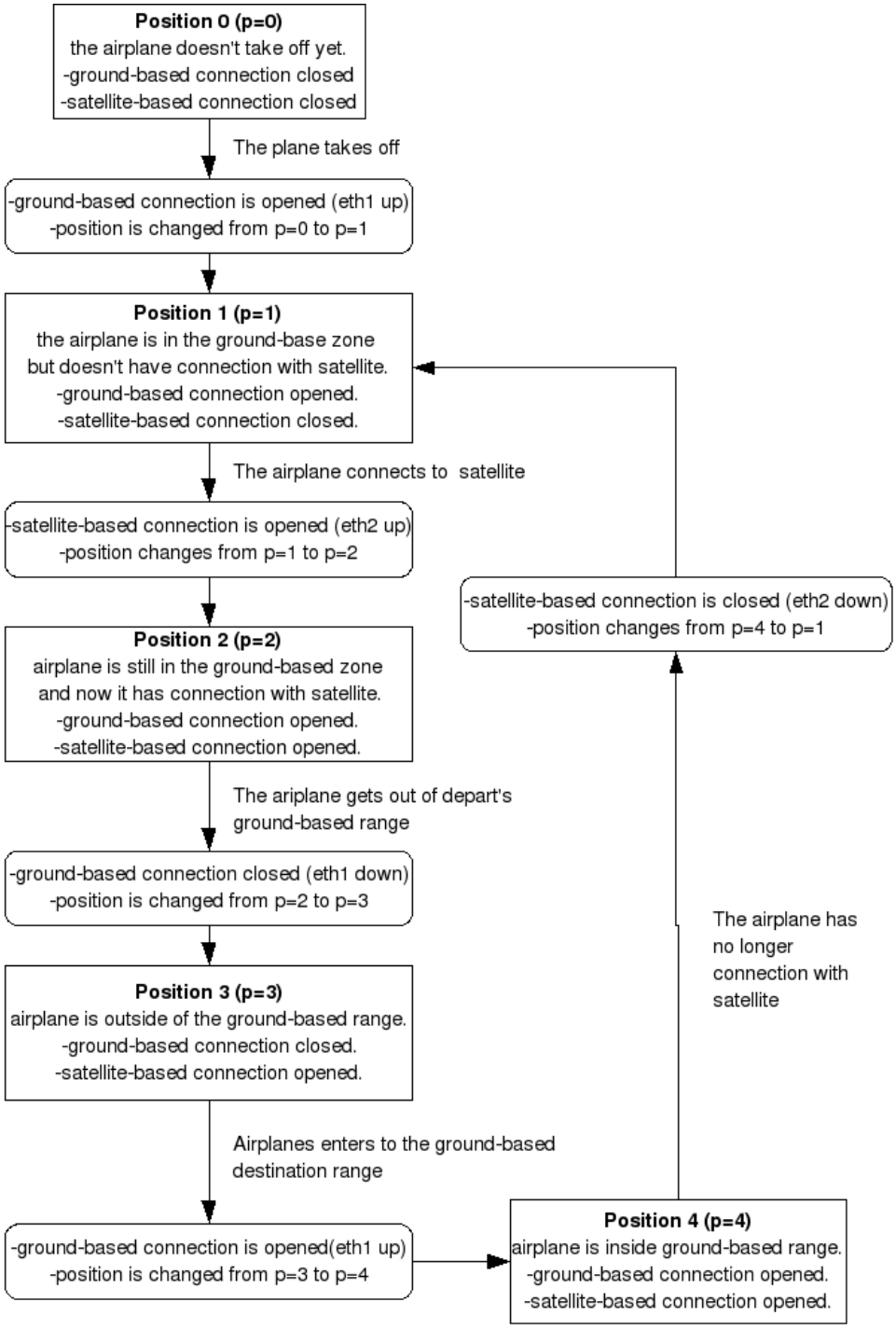

Figure B.1: GUI flow chart

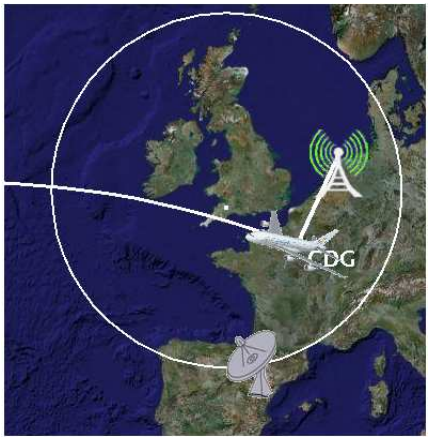

Figure B.2: Position 1

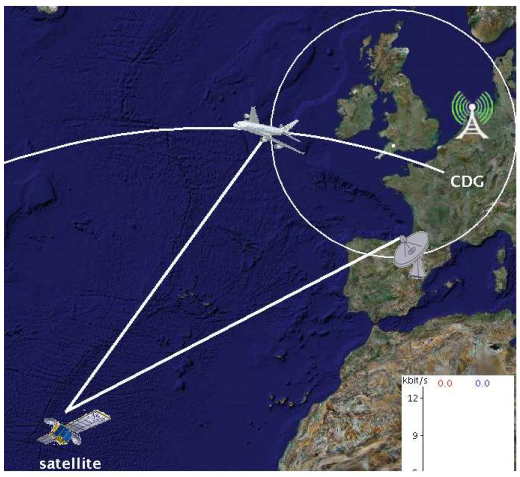

Figure B.3: Position 3

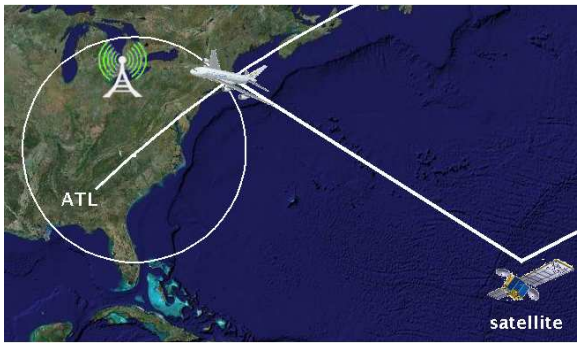

Figure B.4: Position 4

Appendix C

RTP packet delay and throughput measurements

Here I expose some of the results obtained in the measurements to study the handover behaviour and how affects them to the system. These results are a complementation of the tests presented in chapter 5, using the same procedure to obtain the results.

C.1 Measurements using only NEMO protocol

The first graph C.1 shows the RTP packet delay obtained during a VoIP call in a simple GUI flight simulation. Almost 800 packets have been captured with wireshark to trace the graph. The C.2 trace depicts the RTP packet throughput.

In that call simulation 14 packets have been lost while changing the point of attachment from the terrestrial link to the satellite one. During the handover from the satellite to the ground link, 16 packets have been lost.

The C.3 trace shows the delay of 5000 RTP packets captured during a VoIP simulation call. In that case, 32 packets have been lost during the handover from terrestrial to satellite link and 24 packets while changing from satellite to terrestrial link. The C.4 graph represents the throughput.

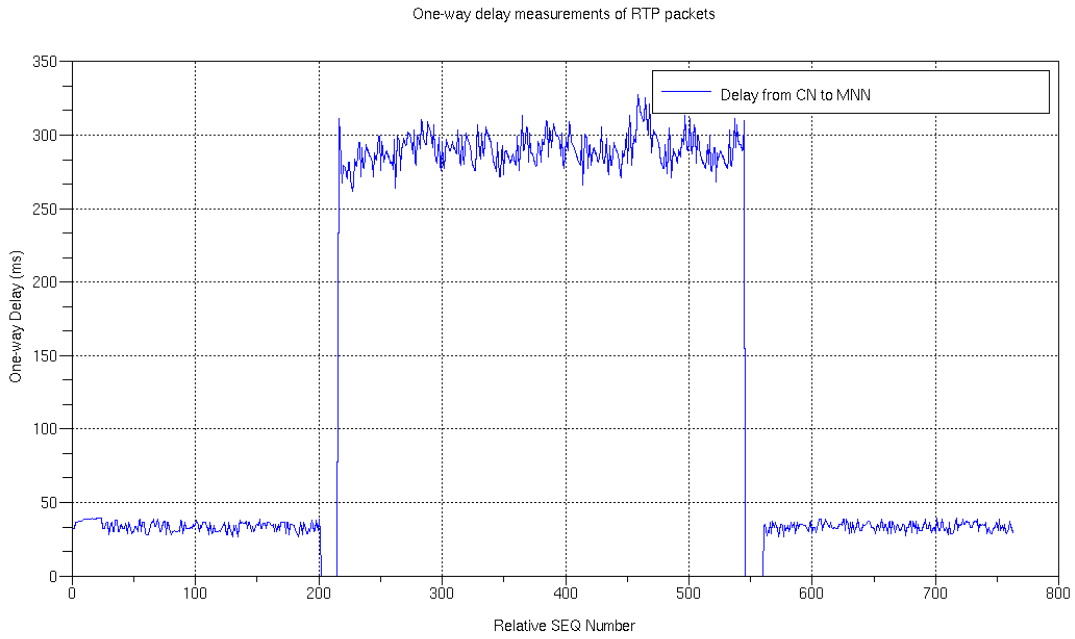

Figure C.1: RTP packet delay during handovers

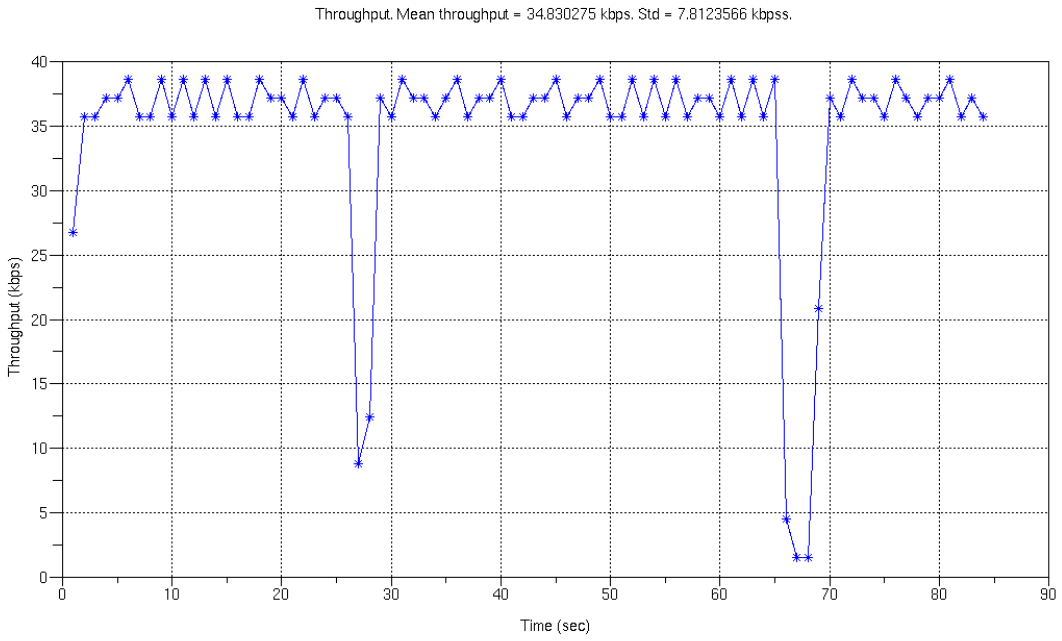

Figure C.2: RTP packet throughput during handovers

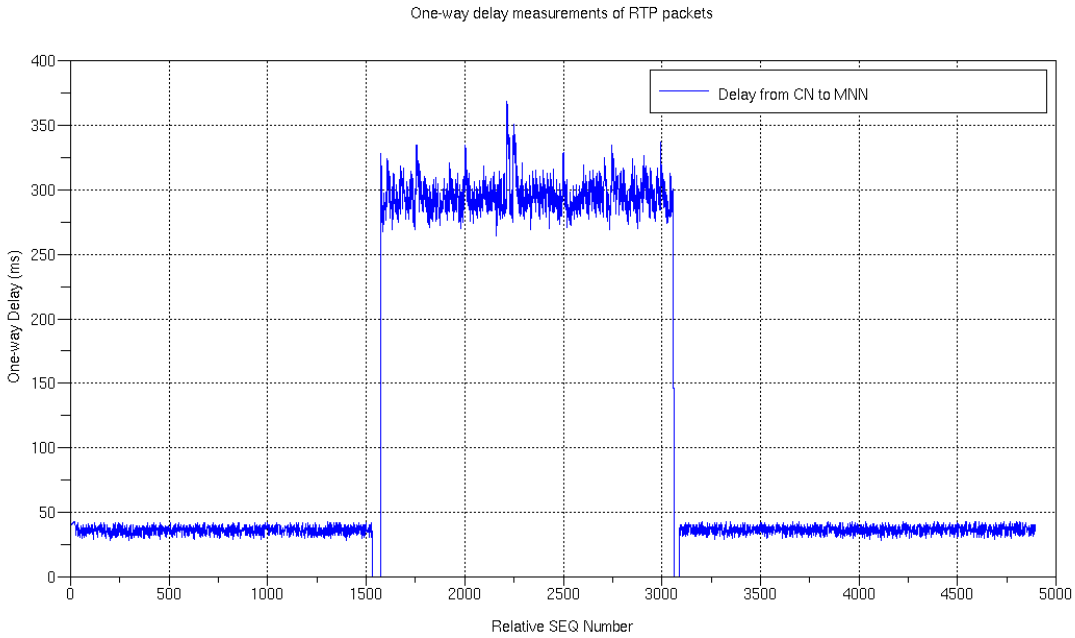

Figure C.3: RTP packet delay during handovers

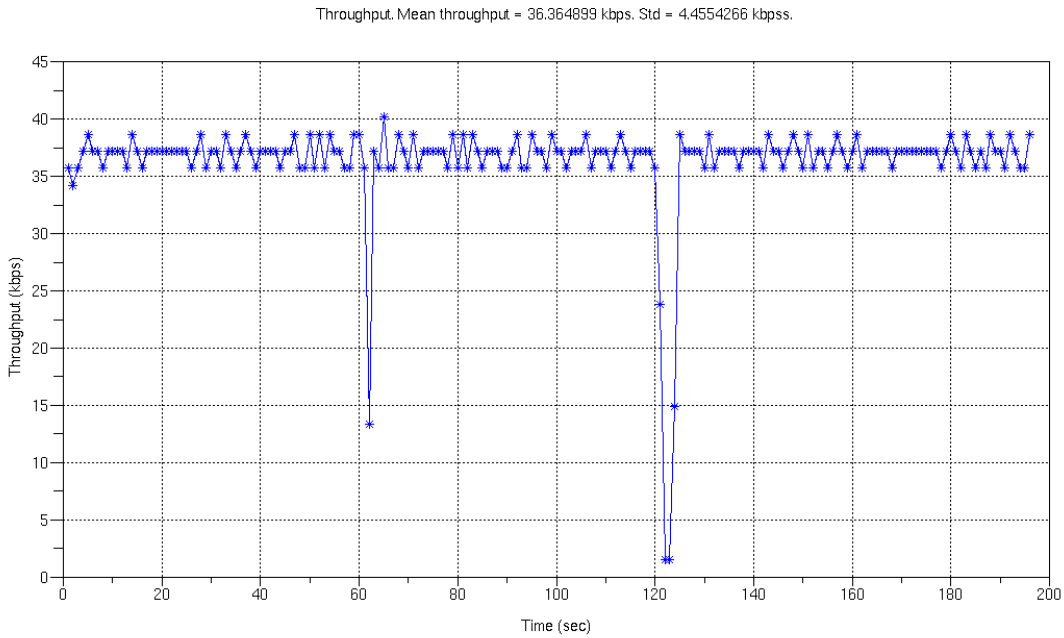

Figure C.4: RTP packet throughput during handovers

C.2 Measurements with MCoA registration protocol

The C.5 trace shows the delay of 800 RTP packet captured during a VoIP call with MCoA registration mechanism implemented in the test-bed. The procedure followed to obtain the results are the same used in section 5.6.1. The C.6 trace depicts the RTP packet throughput.

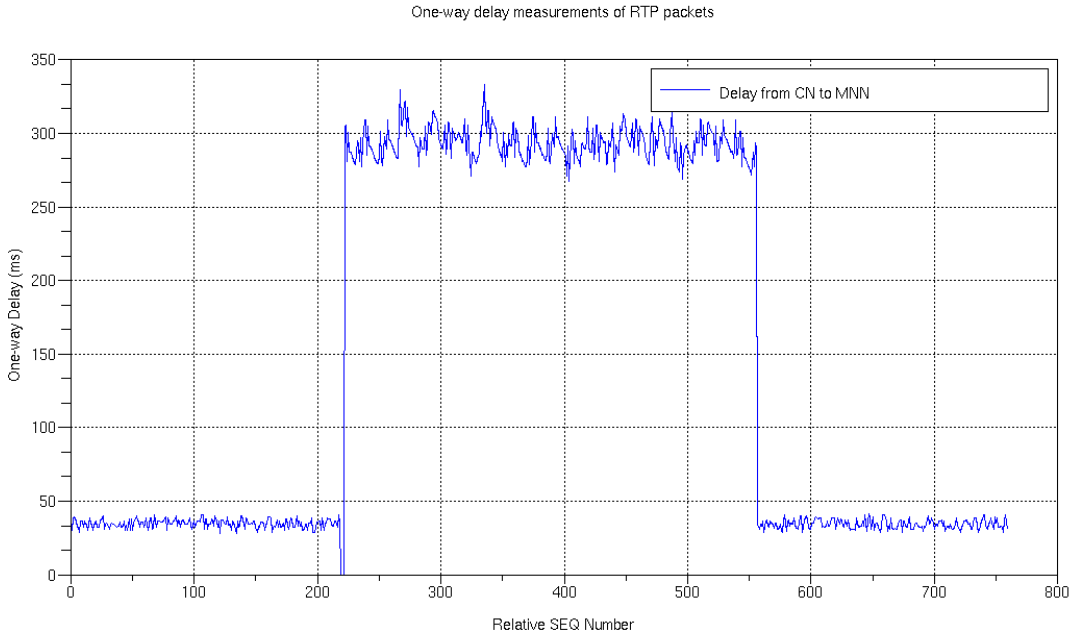

Figure C.5: RTP packet delay during handover with multiple CoA registration protocol

In that call simulation 2 packets have been lost while changing the point of attachment from the terrestrial link to the satellite one. During the handover from the satellite to the ground link, 0 packets have been lost.

The C.7 trace shows the delay of 4500 RTP packets captured during a VoIP simulation call. In that case, 5 packets have been lost during the handover from terrestrial to satellite link and 1 packets while changing from satellite to terrestrial link. The C.8 graph represents the throughput.

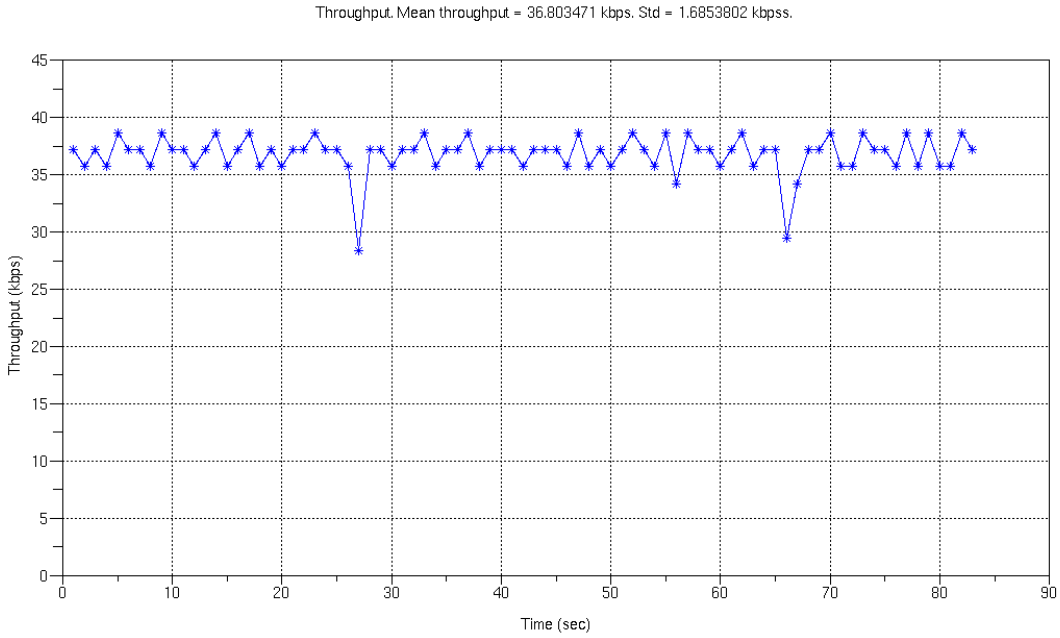

Figure C.6: RTP packet throughput during handover using multiple CoA registration protocol

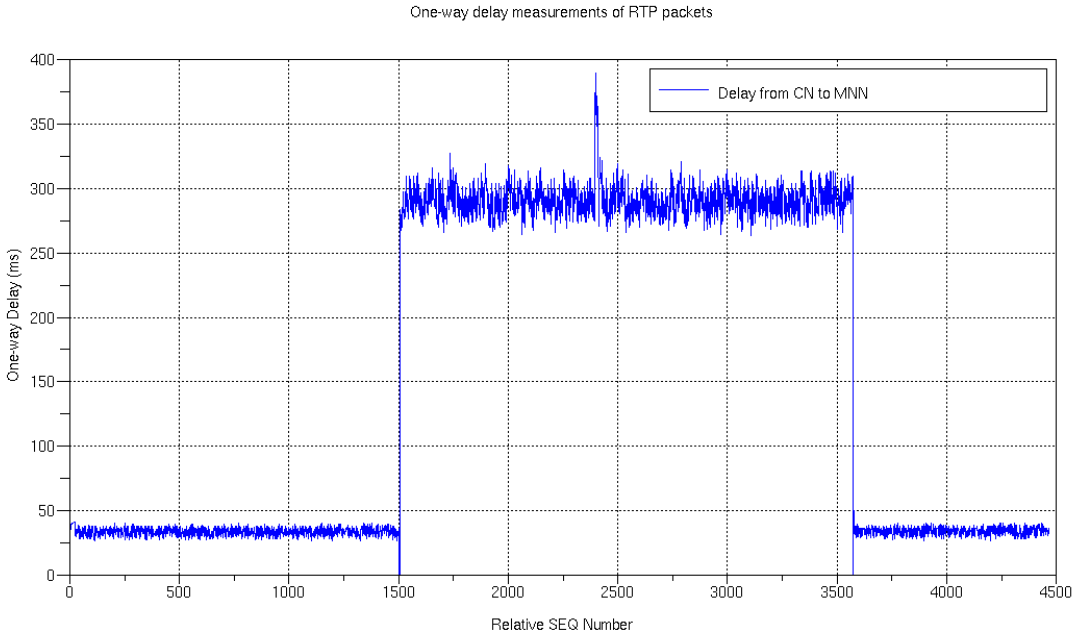

Figure C.7: RTP packet delay during handover using multiple CoA

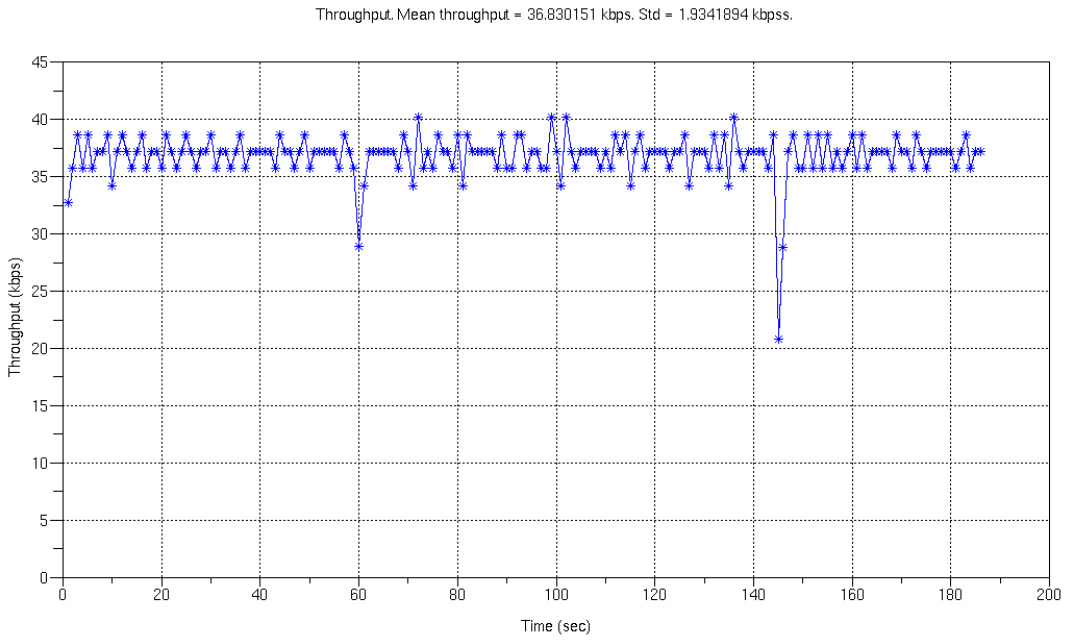

Figure C.8: RTP packet delay during handover using multiple CoA