Universidade de Aveiro Departamento de Electrónica, Telecomunicações e Informática, 20162016

Francisco Manuel Malheiro de Castro

Different technologies in Vehicular Networks: Multihoming and Network Coding

Universidade de Aveiro Departamento de
2016 Electrónica, Telecomunicações e Informática,

Francisco Manuel Malheiro de Castro

Utilização de Diferentes Tecnologias em Redes de Veículos: Multihoming e Network Coding

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Electrónica e de Telecomunicações, realizada sob a orientação científica da Doutora Susana Sargento, Professora do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro, e do Doutor Nuno Coutinho, Engenheiro de Sistemas da Veniam.

agradecimentos

Em primeiro lugar gostaria de agradecer aos meus pais, Noé e Maria Castro,por todo o amor, carinho de dedicação, e por estarem sempre lá nos momentos de maior necessidade. De seguida gostaria de agracer a todos os colegas e amigos de curso, especialmente ao Cristiano Gonçalves e Luis Almeida, Felisberto Pereira, Sergio Dias, Daniel Lopes por toda a ajuda e paciência que sempre tiveram durante estes 5 anos de curso.

Gostaria de agradecer tambem ao nosso grupo de investigação por toda a entreajuda que existiu, todo o companheirismo e bom ambiente que sempre existiu. Aqui gostaria de realçar tres pessoas que me marcaram muito pela sua forma de ser, Ruben Oliveira, Andre Martins e Pedro Cirne, que se tornaram tanto em tão pouco tempo. Para além destes, não poderia tambem esquecer de agraceder ao Nelson Capela, que foi um mentor, amigo e uma especie de segundo orientador. Sem ele, este trabalho não teria a qualidade que tem.

Por último, mas não menos importante, gostaria de adradecer á professora Susana Sargento, por me ter demonstrado o potencial e por ter despertado o meu interessa para uma area, por toda a ajuda prestada no desemvolvimento da tese de mestrado, e por me ter sempre incentivado a ir mais além. Gostaria de agradecer tambem ao meu co-orientador Nuno Coutinho pela ajuda e desponibilidade demonstrada.

Resumo

Com o surgimento de notícias relacionadas com carros autónomos, torna-se óbvio que as Redes Veiculares vão ter um papel chave num futuro muito próximo. Para além disso, estas redes podem ser utilizadas para fornecer serviços de entretenimento para os passageiros dos veículos (Internet sem recurso a tecnologia celular). Os maiores desafios relacionados com este tipo de redes estão associados com a extrema mobilidade que os nós têm, as constantes quebras de ligação e as perdas de tráfego devido à degradação do sinal das redes wireless, que num ambiente repleto de obstáctulos como é uma cidade, são uma constate.

Outro desafio/oportunidade é possibilitar que este tipo de redes tirem partido de todos os recursos disponíveis, isto é, como hoje em dia as cidades estão repletas de redes wireless, os nós têm de ter inteligência de selecionar a/as rede/redes que fazem sentido, e encaminhar o tráfego através dessas mesmas redes, tendo em conta a carga de cada rede. O objetivo desta dissertação vai ser resolver/minorar os problemas acima descritos. Em primeira instância, e com o objetivo de aumentar a eficiência de uma VANET já desenvolvida no nosso grupo, foi criado um gestor de conectividade do tráfego de uplink, que é capaz de diferenciar o tráfego, e depois dividido através das redes de acesso, tendo em conta a carga de cada uma. Isto é suportado quer quando o carro tem acesso direto à infraestrutura ou quando tem acesso indireto (quando existe outro carro a agir com midle-man). Para melhorar a performance da VANET em momentos de quebra de ligação, foi criada uma mensagem de perda de ligação que, quando o sinal da ligação se aproxima para níveis considerados maus informa a unidade responsável por dividir o tráfego pelas redes de acesso, que determinada ligação é má, e não é para ser usada.

Por fim, para resolver o problema das perdas relacionadas com as ligações sem fios, optou-se por utilizar o network coding. O maior desafio foi tornar a utilização do network coding transparente para o protocolo de mobilidade. Por outras palavras, criar um programa que, de forma independente, trata de todos os aspetos relacionado com a codificação/descodificação e deixar para o protocolo de mobilidade os aspetos relacionados com a gestão da mobilidade.

Para validar todo o trabalho feito, foram realizados testes de laboratório (gestor de ligações de uplink e network coding) e testes de reais (mensagem de perda de ligação). Os testes de laboratório mostraram que o tráfego de uplink é dividido com sucesso, e que o gestor de conetividade envia o tráfego para as redes de acesso tendo em conta a carga de cada uma, quer em single, quer em multi-hop.

Relativamente aos testes reais, foi demonstrado que a mensagem de perda de ligação permite diminuir perdas associadas ao processe de handovers.

Por fim, relativamente ao network coding foi possível concluir que este permite recuperar de perda de pacotes. Além disto, foi demostrado que esta implementação suporta mobilidade e multihoming quer em single quer em multi-hop.

Abstract

With the increasing interest in self-driving cars, it becomes obvious that the vehicular networks will have a key role in a near future. Furthermore, these networks can be utilized to provide entertainment services (Internet) to the car passengers. The greater challenges related with this kind of networks are associated with the high mobility that the nodes have, the constant drop of connectivity and to the traffic losses due the signal degradation of the wireless networks, due to the huge amount of obstacles presented in the city. Another challenge/opportunity is the possibility of these networks to take advantage of all the available resources available. Nowadays the cities have networks available almost everywhere, the nodes must have intelligence to select the network/networks that make sense, and manage the routing through those networks, taking into account the load of each access network.

The objective of this dissertation will be to solve/reduce the problems described above. At the first instance, the objective is to improve the vehicular network already developed in our group, that already has a multihoming framework that allows the downlink traffic to be divided through the available networks, in a way that optimizes the network performance.

In order to also provide multihoming in uplink, in this dissertation it was developed an uplink connection manager that can differentiate the traffic and route that traffic through different access networks at the same time, taking into account the load of each network. This can be done in single and multi-hop.

In order to improve the multihoming framework, it was developed a message that informs the entity responsible for dividing the traffic that the connections with bad signal quality should not be used to route traffic. This will allow that entity to route the traffic through the other available networks with a good signal quality, avoiding packet losses that would occur due bad signal quality and connection losses.

Finally, in order to recover from packet losses due to bad network signal quality, it was used network coding. The greatest challenge was to create a network coding approach, that was transparent to the mobility protocol, that, in an independent way manages all the aspects related to the encoding/decoding and leave to the mobility protocol the management of all the mobility related aspects. It were also developed two algorithms that find the configurations to the encoding process. One of the algorithms will try to ensure the maximum packet loss recovery, and the other will try to assure a packet loss lower than a threshold, with the minimum overhead possible.

In order to evaluate all the work done in this dissertation, it were performed laboratory tests (uplink manager and network coding) and real world tests (disconnect message). These tests show that the uplink manager is able to differentiate traffic, and route through different access networks at the same time, taking into account the load of each network (in single and multi-hop). The tests related with the disconnect message show that this message removes the packet loss that would normally occur in the handover mechanisms. Finally, the network coding tests show that the network coding can be used to recover from packet loss, even in a vehicular network with multihoming and in single/multi-hop. Moreover, it was possible to conclude that the two developed algorithms accomplish all the proposed objectives.

Contents

List of Figures

List of Tables

Acronyms

AP	Access Point
AR	Access Router
AT	Achieved Throughput
BA	Binding Acknowledgement
BCE	Binding Cache Entry
BS	Base Stations
BU	Binding Update
CoA	Care-of Address
CN	Correspondent Node
DSRC	Dedicated Short-Range Communications
D-ITG	Distributed Internet Traffic Generator
DOT	Department of Transports
EUI-64	Unique Identifier 64-bit
FCE	Flow Cache Entry
FN	Foreign Network
GPS	Global Position System
GSM	Global system for mobile
HA	Home Agent
HoA	Home Address
Hostapd	Host access point daemon
HN	Home Network
HIP	Host Identity Protocol
HI	Host Identity

HIT	Host Identity Tag
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
LFN	Local Fixed Node
LMA	Local Mobility Anchor
LNC	Linear Network Coding
LTE	Long-term evolution
MAC	Media Access Control
MAG	Mobile Access Gateway
mMAG	mobile MAG
MIPv4	Mobility in IPv4
MIP	Mobile Internet Protocol
MIPv6	Mobile Internet Protocol version 6
MH	Multihoming
MN	Mobile Node
MNN	Mobile Network Node
MN-HNP	Mobile Node's Home Network Prefix
MN-ID	Mobile Node IDentifier
MNN	Mobile Network Node
MNP	Mobile Network Prefix
MR	Mobile Router
NAPT	Network Address and Port Translation Networks Architectures and Protocols
NEMO	NEtwork MObility
N-PMIPv6	Network-Proxy Mobile Internet Protocol version 6
OBU	On-Board Unit
P2P	Peer-to-Peer
PBA	Proxy Binding Acknowledgement
PBU	Proxy Binding Update
PMIPv6	Proxy Mobile Internet Protocol version 6

PoA	Point-of-Attachment
Proxy-CoA	Proxy Care-of Address
QoS	Quality of Service
RA	Router Advertisement
RS	Router Solicitation
RSSI	Radio Signal Strength Intensity
RSU	Road Side Unit
SNC	Sistematic Network Coding
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
ULID	Upper-Layer IDentifier
UCE	User Cache Entry
V2I	Vehicle-to-Infrastructure
V2V	Vehicle-to-Vehicle
VANET	Vehicular Ad-Hoc NETwork
VMN	Visited Mobile Node
WAVE	Wireless Acess in Vehicular Envoriments
Wi-Fi	IEEE 802.11 a/g/n
RLNC	Random Linear Network Coding
SNC	Systematic Network Coding

Chapter 1

Introduction

1.1 Motivation

Today, almost every person connects to the Internet, at least once a day. In order to support this every growing demand of wireless networks, it is usual to find numerous Wi-Fi hot spots spread all over the city. Although the cellular technology has improved in the recent years, it is still expensive and with a high battery consumption to the users terminals, when compared with the Wi-Fi technology.

Vehicular networks are a hot topic today, and are turning into reality, nowadays and in the near future. The vehicular to vehicular communications will be essential to provide the vehicle information about what is happening in the surroundings. For example, if a car crashes in the middle of a highway, an emergency message can be sent in broadcast informing all the cars that are near the crash zone, reducing the probability of a new accident. Due to the high potential that vehicular to vehicular communications have in terms of automobile security, the major automobile manufacturers are currently expending a huge amount of effort to create systems that exploit this kind of communications. Moreover, and because all the information related with automobile security is too sensitive, some academic works have proposed solutions to assure reliable data transfer in high dynamic wireless environments. Vehicular to infrastructure communications has a huge potential to revolutionize the way travels are done. Vehicular to infrastructure communications can be used both for entertainment and traffic/city management. In terms of entertainment, it can be used to provide Internet access to the cars passengers, which opens a totally new world of business opportunities, such as pay-per-view content, publicity and applications. Moreover, if a car can access to the infrastructure, it will obtain informations not only about surroundings, but of the whole space, roads and city. This particular aspect can be used to improve traffic management, because now the car can know which routes are more jammed, and decide which routes will be the best. Moreover, it will turn the cars into mobile data collectors. In other words, it will be possible to use a car to collect informations related to sensors spread all over the city, and when a car moves close to an RSU, it will send the sensors information to the cloud. This would avoid the constant use of the cellular technology, and it would allow to deploy sensors in zones that are not covered by any RSU.

Having in mind all that was said, it was developed in our research group an architecture that merges all the previous topics into a single network. In this architecture each car has an On-Board Unit (OBU), that will allow the car to connect to the infrastructure through Wi-Fi, WAVE Road Side Units (RSU) or cellular Base Stations (BS), being even possible to a car be connected both to the Wi-Fi, WAVE RSU and cellular at the same time (multihoming) [13]. In order to take advantage of the multihoming feature, it was also developed an approach that divides the downlink traffic through the available networks and technologies, in an optimal way, increasing the overall networks performance. All the mobility and multihoming processes are transparent to the final user (car passengers), that only have to connect to a Wi-Fi network shared by the OBU, in order to have access to the Internet. All this is done both in single-hop (when the car is connected directly to the infrastructure) and in multi-hop (when a car needs another car to reach the infrastructure.

Although the vehicular network developed by our research group can indeed perform the functionalities that were described, there are still missing functionalities. Firstly, although the downlink traffic takes advantage of the multihoming features, the uplink traffic was still being routed through only one technology. Moreover, it was noticed that the vehicular network multihoming framework was vulnerable to drops of connectivity. Finally, it was found that the network did not have any mechanism to recover from packet losses, which can be dangerous when the information that is being sent is too sensitive to be lost (emergency messages).

The objective/motivation of this dissertation is to improve the vehicular network developed by our research group, to take the most advantage of the vehicular networks potential, with the best performance possible. This will require the design and implementation of a connection manager responsible for all the uplink traffic, that takes advantage of the multihoming capabilities of the network. Moreover, it is of the utmost importance to find a mechanism to ensure the successfully delivery of critical information, taking into account the special features of the vehicular networks. This thesis will study how network coding can be integrated in the mobility and multihoming, and how it can be optimized to improve vehicular network communications.

1.2 Objectives and Contributions

The following items are the objectives and contributions of this thesis:

- **Study mobility protocols:** Understand the current mobility protocols and conclude about the pros and cons of each one.

- **Study multihoming protocols:** Study some multihoming protocols, in order to understand which one is the best to use in a vehicular network.

- **Study network coding implementations:** Conclude about the current work done in the network coding field and how it can be integrated in vehicular networks.

- **Implement an uplink connection manager for multihoming and multi-hop:** Design and implement an uplink connection manager that takes into account all the available resources and is able to optimally choose them to differentiate the uplink traffic.

- **Improve the current multihoming process:** Provide the multihoming framework information about each connection signal quality, and the amount of traffic that each node in the network is processing.

- **Integration of Network Coding:** Integrate the Network Coding approach in the vehicular network, for both multihoming and multi-hop.

- **Evaluate the improvements in the multihoming process:** Evaluate, in a real vehicular network, the new features added to the multihoming process.
- **Evaluate the impact of the network coding in the vehicular network** In order to know the impact of the network coding in the vehicular network, its integration is evaluated in real vehicular networks.

During the execution of this master thesis, it were made two scientific papers about the main conclusions related to the uplink management [14] and the network coding [15] in vehicular networks, which are presented in the respective links provided in the references. The work developed in this dissertation was also used in another paper [13], that describes the overall multihoming approach for vehicular networks.

1.3 Document Organization

This Dissertation is organized as follows:

- **Chapter 2 :** presents the current state-of-the-art in vehicular networks, mobility protocols, multihoming and network coding.
- **Chapter 3 :** describes all the base work that supports the work done in this dissertation.
- **Chapter 4 :** describes all the improvements performed in the mobility and multihoming protocol, and the implementation of the connection manager.
- **Chapter 5 :** describes all the technical challenges found in the integration of the Network Coding in the vehicular network.
- **Chapter 6 :** describes all the testbeds used to evaluate the work done in this dissertation, as well as results and conclusions.
- **Chapter 7 :** Summarizes all the work performed in this Dissertation, and sets new goals to future work.

Chapter 2

State-of-the-Art

2.1 Introduction

In order to understand the work done in this dissertation, it is important to understand all the research already done in vehicular networks, mobility, multihoming and network coding. Throughout this chapter, it will be presented the important aspects related with these areas.

In section 2.2 it will be described the main VANET features, challenges and applications.

In section 2.3 it will be described access technologies that can be used in a vehicular environment, and conclude about which technology should be used in different scenarios.

In section 2.4 it will be described some mobility protocols and some features that a mobility protocol must fulfil in order to be used in a vehicular environment.

In section 2.5 it will be described some multihoming architectures.

In section 2.6 it will be described how network coding can be used for packet recovery, and it will be also presented some other practical examples of network coding applications in vehicular networks.

Finally, in section 2.7 it will be presented an overview of this chapter.

2.2 Vehicular Networks

A Vehicular Ad-Hoc NETwork (VANET) is an ad-hoc network between vehicles that has the potential to revolutionize the automobile industry. To understand how these networks have gathered the world attention, it is important to understand how all begun. The first clear endeavor in this area, according to [16], was made by the United States Federal Communications Commission in 1999, when it was reserved 75 MHz at 5.9 Ghz band to Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) communications. Three years latter, one of the most important project in this area was also created in the USA, where seven of the most prestigious automotive manufactures (BMW, VW, Ford, GM, Nissan, Toyota and Daimler-Chrysler) worked together with the USA Department of Transports (DOT) in a project called Vehicle Safety Communications Consortium, developing the Dedicated Short-Range Communications (DSRC) technology. This allowed, at the time, wireless communications within 1000m with nodes with high speeds [17] [18].

Then, in 2004 the IEEE 802.11 standard group was designated to standardize the DSRC radio technology, once the DSRC operations were identical to the 802.11a, but adjusted to low overhead operations. In 2010 the IEEE 802.11p, also known as Wireless Acess in Vehicular

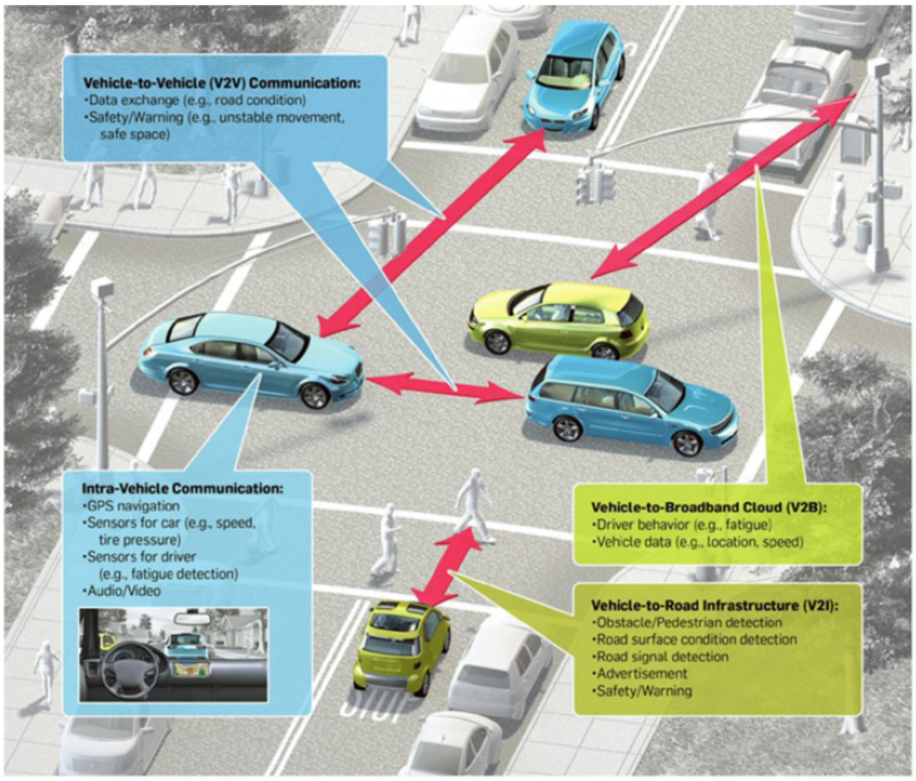

Vehicle-to-Vehicle (V2V) Communication:
•Data exchange (e.g., road condition)
•Safety/Warning (e.g., unstable movement, safe space)
Intra-Vehicle Communication:
•GPS navigation
•Sensors for car (e.g., speed, tire pressure)
•Sensors for driver (e.g., fatigue detection)
•Audio/Video
Vehicle-to-Broadband Cloud (V2B):
•Driver behavior (e.g., fatigue)
•Vehicle data (e.g., location, speed)
Vehicle-to-Road Infrastructure (V2I):
•Obstacle/Pedestrian detection
•Road surface condition detection
•Road signal detection
•Advertisement
•Safety/Warning

Figure 2.1: VANET applications [1]

Envoriments (WAVE), was born. It introduces to the general public a low association time, low delay and a high range technology [2]. Figure 2.1 illustrates a VANET network and some practical functionalities.

In the next sub-sections it will be described the main VANETs features, application and challenges:

2.2.1 VANET Features

In the next items, it is presented some of the VANET main features [1] [19] [18]:

- **Predicted mobility:** Since all the cars are moving through roads, it is possible to predict the cars next moves using its Global Position System (GPS) informations.

- **Huge scale networks:** Nowadays, in the big urban centers, it is normal to have a huge amount of cars in a relatively small area.

- **(Almost) Unlimited processing power:** Since all the OBUs are inside cars, this means that it is possible to have a processing power that matches a normal personal computer (or several).

- **Dynamic network topology:** Nowadays the cars are quite fast, so it is expected that the vehicular network topology changes several times in a small time interval.

2.2.2 VANET Applications

In the next items it will be presented some VANET potential applications:

- **Safety applications:** V2V communication can be used to send emergency messages, for example, to inform all the nearby cars of an accident, avoiding a mass collision. This particular topic has a huge amount of possible applications, that can represent a direct impact on the passengers lives.
- **Provide Internet to passengers:** Nowadays everyone has a smart phone or a small laptop and, with a VANET, it is possible to provide Internet access to passengers, avoiding the expensive use of cellular networks.
- **Optimized city management:** If there is available information about all the cars that travel through the city, it is possible to route them through the less jammed routes. Another example of an optimized city management is implemented nowadays in Porto, where the buses that travel through the city collect information about the thrash level of the garbage bins, avoiding unnecessary trips by the garbage collector.
- **Publicity:** Trading Internet access for publicity is another possible application. Moreover, it is possible to load each RSU with specific publicity from a nearby shop.

2.2.3 VANET Challenges

In the next items it will be presented some challenges related with VANETs:

- **Security:** This is one of the major challenges of VANETs. Due to the WAVE broadcast nature and the fact that this technology does not use any authentication, all the information that is being sent to a specific node is captured by the other nodes within range.
- **Signal degradation:** As can be easily concluded, in a city environment there are several obstacles between the sender and the receiver, reducing the quality of any wireless communication.
- **Network fragmentation:** Due to the high mobility characteristics of the VANETs, the network topology is constantly changing leading to frequent network fragmentation.

2.3 Access Technologies

Nowadays, there are a large number of wireless technologies available to provide connectivity between two nodes. Taking into account the VANET features stated in the previous subsection, and according to [1], there are technologies that stand out. The next items will present some examples of technologies that can be used in vehicular environments.

- **WAVE :** As said in the beginning of this chapter, in 1999, the United States Federal Communications allocated 75 Mhz in 5.9 Ghz range to be used exclusively in V2V and V2I communications. This DSRC spectrum is divided into 8 channels, with 10 Mhz of bandwidth, with the exception of the reserved channel that has 5 Mhz. Besides that, there is one control channel reserved for safety application communications (178) and the remaining 6 channels are Service Channels that can be used freely. As figure 2.2 shows, it is also possible to merge two channels in one, in order to achieve a 20 Mhz channel.

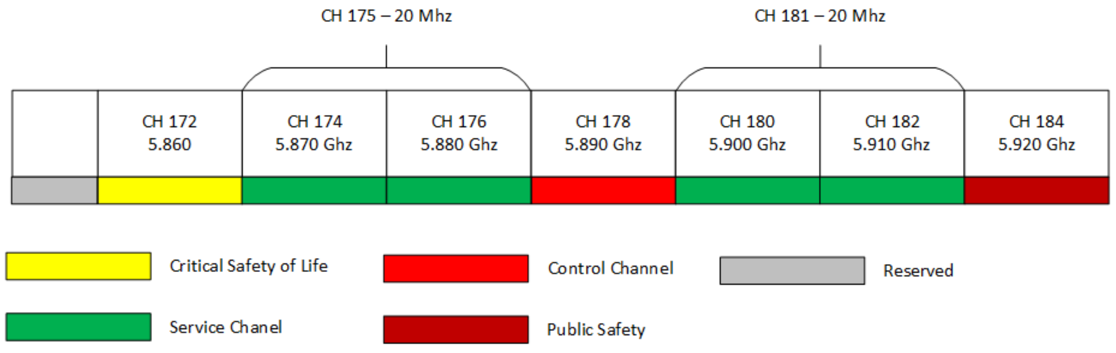


Figure 2.2: DSRC channel allocation (US) based on [2]

This spectrum allocation allows a node speed of 200 Km/h, with a coverage of 300m and a data rate of 27 Mbps. Moreover, WAVE does not use any association process, allowing a faster communication process. This characteristic can be seen as a major advantage, and is truly needed in a vehicular involvement, although it can also present some disadvantages. Due to its broadcast nature, WAVE is vulnerable to bandwidth trashing: a WAVE node can send trash in broadcast, and negatively affect all the other nodes connections that are in the coverage area.

- **Wi-Fi:** In order to standardize wireless communication, the IEEE 802.11 working group developed several standards. IEEE 802.11a works at 5 GHz and allows a data rate of 54 Mbps with a range of 38 m indoor and 140 m outdoor. The 802.11g standard retains the previous data rate and coverage features, but now at 2.5 GHz frequency [1].

  Wi-Fi is probably the most widely deployed wireless technology. Nowadays, it is present in almost every computer/cellphone, due to its relatively good coverage and high data rate. Moreover, it is possible to use a Wi-Fi node configured as an AP, or in a ad-hoc way.

  In a vehicular networks perspective, Wi-Fi presents some disadvantages. First, it has a low transmission range, does not behave well when the nodes are in high-speed; then, before start exchanging data between nodes, these need to handle the association. This is illustrated in figure 2.3, where it is possible to see the normal association process that the nodes must perform, before start exchanging data between themselves. As can be seen, multiple control messages must be traded, increasing the time of the association. Although Wi-Fi could be used in theory for V2V communication, in practice it is a different story. It must be kept in mind that the vehicles are highly mobile, meaning that in the most of the time, the vehicles only are in range of each other for a few seconds, so a fast association process is required [2].

- **Cellular thecnology** 2 G or Global system for mobile (GSM) is one of the cellular standards that enables a user to achieve data rates of 9.6 Kbps. This standard uses two frequency bands, one for uplink (890-915 Mhz) and an other for downlink (935-960 Mhz), and each band is divided in channels with a bandwidth of 200kHz.

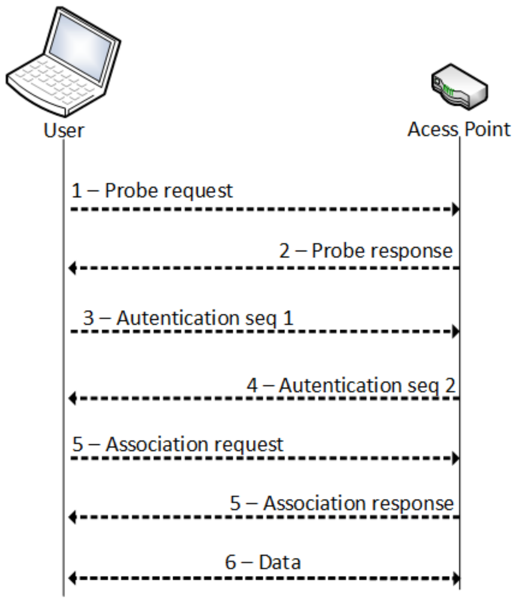


Figure 2.3: Wi-Fi association process

4G or Long-term evolution (LTE) advanced is the standard present in all cellphones nowadays, and allows a data rate of 300Mbps in downlink and 75 Mbps in uplink.

After a quick overview over the cellular standards, it is time to evaluate this technology pros and cons, starting with the pros. Since the cellular technology is widely spread, so it is natural to have a high coverage. Moreover, 4G enable enough data rate to enable a good real time and multimedia quality services.

The only setback to this technology is the price. The telecommunication operators still charge significantly for this kind of services. Moreover, this technology requires access to the infrastructure, for other words, it cannot be used to V2V communications, only for V2I, and V2V will require the access through the base stations, which provides delays higher than 100 milliseconds, that are not acceptable for emergency communications.

2.3.1 Considerations

From the previous explanations about the access technologies, it is now easier to understand in which situation makes more sense to use each technology. Starting from the beginning, WAVE technology should be used, both to V2V and V2I connections due to its special features described previously.

Wi-Fi can be used in V2I communications, but some concerns must be kept in mind. As it was illustrated, the association time can not be forgotten. Due to this limitation, Wi-Fi should only be used to share access to the infrastructure with the car passengers, and in some particular conditions can be used to V2I communications, for example, in a traffic light when the car must stay still for several seconds or when it moves with low speed.

Lastly, it is clear that cellular technology should not be used for V2V communications, for 2 reasons. First, it is still an expensive technology to use, and on the other side, whenever a node wants to communicate with another, it must go through the infrastructure, even if the other node is at 1 m of distance. Having this in mind, cellular technology should only be used in last resort, only when there are not any other technologies available.

2.4 Mobility Protocols

The first good solution in terms of mobility was proposed by IETF, with the MIPv4 [20]. This host-based protocol successfully provides a seamless access to the Internet to an end user capable of moving across different networks. Later, it was developed the MIPv6 protocol, that not only kept the MIPv4 main features but also added new ones, that exploit the fact that this protocol was made in IPv6. It was noticed that this protocol had some problems, namely high handover latency, and new solutions were needed to provide the end user a truly seamless access to the Internet while moving. To fulfill this gap, it was developed a network-based protocol that uses proxies to manage all the mobility related aspects, the PMIPv6 [21]. This particular aspect both reduces the Mobile Node (MN) resources, signaling overhead and handover latency. Another mobility protocol that was studied was the NEMO [22]. This protocol provides mobility to an entire network and, naturally, to its users. Finally, it was studied the N-PMIPv6 [21] protocol that tries to join the main advantages of the NEMO and PMIPv6 protocols.

Latter in this section, it will be given a general overview of each mobility protocol that was stated in the previous paragraph, and it will be concluded if the protocols are suitable to support all the VANETs demands. In the next items, those requirements are presented [23][3]:

- **Seamless mobility :** From the vehicle point of view, mobility should be transparent, i.e, the vehicle should have connectivity regardless of its location.
- **Efficient handover performance :** In a vehicular environment, it is expected that many handovers occur, and the vehicle is constantly changing its location. These handovers must be performed as fast as possible and with a reduced complexity (small amount of control traffic).
- **IPv6 support :** IPv6 has some clear advantages over IPv4 protocol. First, IPv6 has a larger address space, meaning that the lack of IP addresses will not be an issue. Also, IPv6 has a better security and QoS support.
- **Multi-hop support :** In order to extend the RSUs range, a vehicle that is in range of a RSU should be able to provide access to vehicles that are not in range of that RSU.
- **Multiohming support :** Nowadays there are wireless networks available throughout the roads. The mobility protocol used must support multihoming, in order to increase the network performance.
- **Reduced complexity at end devices :** Ideally, the end users should not have to perform the mobility protocol into theirs devices. Moreover, it is expected that the mobility protocol does not affect the end users device performance significantly.
- **Reduced bandwidth consumption :** Due to the high stability potential of the VANETs, it is expected that the mobility protocol uses the minimum control messages possible.

In the next subsections it will be detailed some mobility protocols and it will be concluded about its suitability, taking into account the VANETs and the mobility requirements that were described in this document. The selected protocols aim to tell the history of the evolution of the mobility protocol which was selected as basis to implement the VANET that was used as basis work in this dissertation.

2.4.1 MIPv6

This subsection will give a general overview of the MIPv6 protocol, based on [24]. Firstly, it will be presented the names and roles of the specific entities of this protocol (terminology); then, it will be given a general picture of the protocol behavior, and in the end, it will be discussed if the protocol suits the VANETs demands.

2.4.1.1 Terminology

- **MN :** Node that moves through multiple access networks.
- **Home Agent (HA) :** Node where the mobile node registers its current care-of-address. While the mobile node is away from home, the home agent intercepts packets, and tunnels them to the mobile node registered care-of-address.
- **Correspondent Node (CN) :** Peer node with which a mobile node is communicating.
- **Care-of Address (CoA) :** Address associated with the mobile node while visiting a foreign network.
- **Home Network :** First network that the MN connects.
- **Home Address (HoA) :** Permanent address for the mobile node.
- **Binding Update (BU) :** Message sent to the mobile node home agent, in order to register its primary care-of-address.
- **Binding Acknowledgement (BA) :** Message sent after a BU message, in order to confirm that the message was successfully delivered.
- **Home Network (HN) :** Network where is announced by the home agent (first network that the mobile node connects).
- **Foreign Network (FN) :** Network where the mobile node connects after moving away from the home network.

2.4.1.2 Behavior

In order for the MN to obtain the CoA, the node uses the conventional IPv6 mechanisms, such as stateless or state full auto-configuration. Also, if the mobile node is in the HN, it can receive packets without any special MIP procedure. What is new is the mobility part, i.e, when a node moves away from the HN, and connects to the FN, as shown in the figure 2.4. One interesting aspect of this protocol is that it supports mobility across homogeneous or heterogeneous media, i.e, it can move from an Ethernet to a wireless segment for example, and the MN IP address will remain the same. Continuing with the explanation, when the

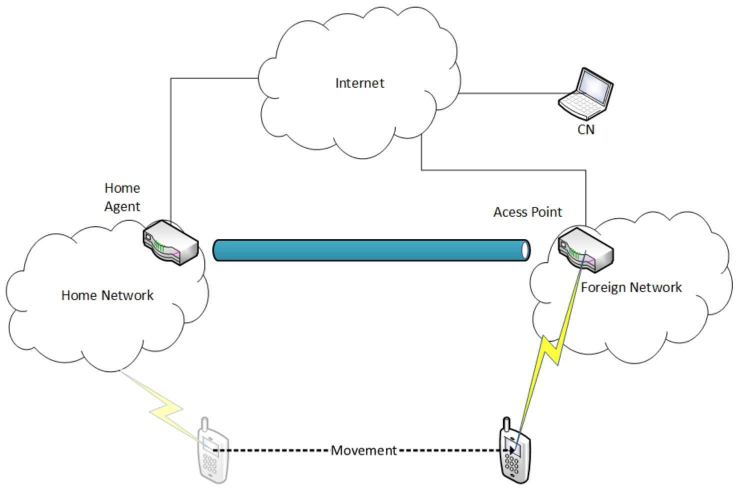


Figure 2.4: MIPv6 protocol overview

node moves to the FN, a BU message is sent, through the Internet to its HA, informing it that its primary CoA has changed. Now, all packets sent by the CN to the MN and vice-versa will be tunneled through a tunnel created between the HA and the MN CoA. This routing process was considered inefficient because it is vulnerable to the HN performance. To solve this problem, it was developed an optimized route mechanism that requires that the MN registers its current biding at the CN, allowing the traffic from the CN to be forwarded directly to the MN, without being routed through the HN. This registration process should being initiated right after the first encapsulated packet arrives at the MN.

2.4.1.3 Considerations

After this explanation it is possible to obtain several conclusions. Despite the fact that the protocol provides mobility to a terminal, it fails to accomplish some critical VANET requirements. Firstly, and most important, this protocol provides user mobility, but not network mobility. Another important aspect that is noticed is the fact that this protocol has high values of handover latency, packet loss and signal overhead [25] that are unacceptable to VANET application. For all these reasons, this protocol was not selected.

2.4.2 PMIPv6

In order to bypass some MIPv6 setbacks, PMIPv6 protocol was developed. In this subsection it will be given a general overview, starting by explaining the specific terminology of this protocol, then, it will be explained the mobility behavior, and lastly, will be discussed about the suitability of this protocol in a VANET scenario. The explanation will be based on RFC 5213 [21].

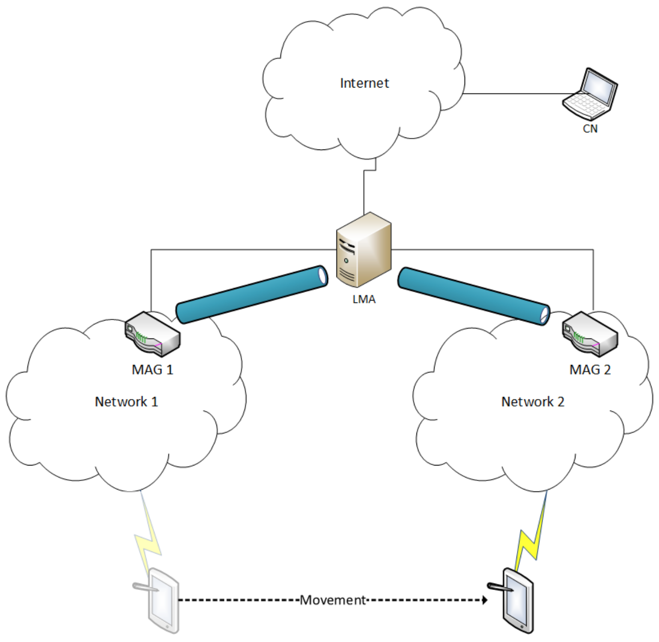


Figure 2.5: PMIPv6 protocol overview

2.4.2.1 Terminology

- **Local Mobility Anchor (LMA) :** This new entity has the same role as the home agent defined in the Mobile IPv6, but with some extra capabilities. This entity is an anchor point for mobile node prefix, manages the mobile nodes binding states and the routing process.

- **Mobile Access Gateway (MAG) :** It is described as an access router to the mobile node. This entity is responsible for tracking the mobile node movement and signal the LMA about all the mobility-related aspects.

- **Proxy Care-of Address (Proxy-CoA) :** Address of the egress interface of the MAG, that will be, for the LMA point of view, the endpoint of the IPv6 tunnel that will be use between the MAG and the LMA.

- **Mobile Node's Home Network Prefix (MN-HNP) :** Is a prefix assigned to a link between the MN and the MAG by the LMA. Moreover, this prefix will be used to configure interfaces, as will be explained latter.

- **Mobile Node IDentifier (MN-ID) :** Number that identifies the mobile node.

- **Binding Cache Entry (BCE) :** Cache responsible for keeping important information about every mobile node (MN-ID, MAG Proxy-CoA and MN-HNP).

- **Proxy Binding Update (PBU) :** Message sent by the MAG to the LMA, with the goal of informing the LMA about the MN binding intentions.

- **Proxy Binding Acknowledgement (PBA) :** This message is sent in response to the PBU. It also contains the prefix given to the Mobile Node.

2.4.2.2 Behavior

Firstly, when the mobile node enters the MAG2 range, a Router Solicitation message is sent to that MAG2. MAG2 acquires the Mobile Node ID and its profile, and sends a PBU to the LMA. Upon accepting the PBU, it creates the BCE that contains all kind of information about the mobile node (Proxy-CoA, MN-HNP,and MN-ID) and creates the IPV6 bi-directional tunnel from the LMA to the MAG 2. Then, it sends a PBA to the MAG accepting the MN registration with the prefix required. When the PBA arrives at the MAG, it sets up its part of the bi-directional tunnel and performs all the necessary configurations to give the mobile node connectivity to the network. Finally, the MAG sends the Router Advertisement (RA) to the mobile node with the MN-HNP given by the LMA, that will be used by the mobile node to configure the egress interface.

In the figure 2.5, it is possible to see the handover process. Here, the MAG1 will detect that the mobile node leaves the network, and then, it will remove the binding and routing state to that node, and will inform the LMA that the mobile node left. Now the LMA will identify the node and wait a specific amount of time until deleting the BCE. This action has two implications. First, if the mobile node tries to join the network, it must perform all the association process previously explained, and secondly, if the tunnel that serves the mobile node that left the network does not serve any mobile node, the tunnel will be deleted [25].

2.4.2.3 Considerations

This protocol indeed fulfill some of the mobility requirements previously described in this document, but has a major gap. As can be concluded, the only entity that has mobility is the MN, that walks through available networks provided by the MAGs that are static. In a vehicular network, it is required that the network moves alongside with the mobile node [21].

2.4.3 NEMO

NEMO was born of the necessity to provide truly network mobility. Next, it will be given a general overview, maintaining the same structure as the previous explanations. NEMO terminology and behavior description will be based on RFC 3963 [22].

2.4.3.1 Terminology

- **Mobile Network Prefix (MNP) :** IPV6 address delegated to a Mobile Router and advertised in the Mobile network.
- **Prefix Table :** List of MNPs indexed by Home Addresses of a Mobile router.
- **Mobile Router (MR) :** This is a router capable of both changing its point of attachment and announce a network to the Mobile Network Node (MNN).
- **Access Router (AR) :** Router responsible for serving a MR.
- **MNN :** Node that is connected to a mobile network, that is announced by a Mobile Router.
- **Visited Mobile Node (VMN) :** Node that can connect to different mobile networks.
- **Local Fixed Node (LFN) :** Node that is always connected to the same mobile network.

2.4.3.2 Behavior

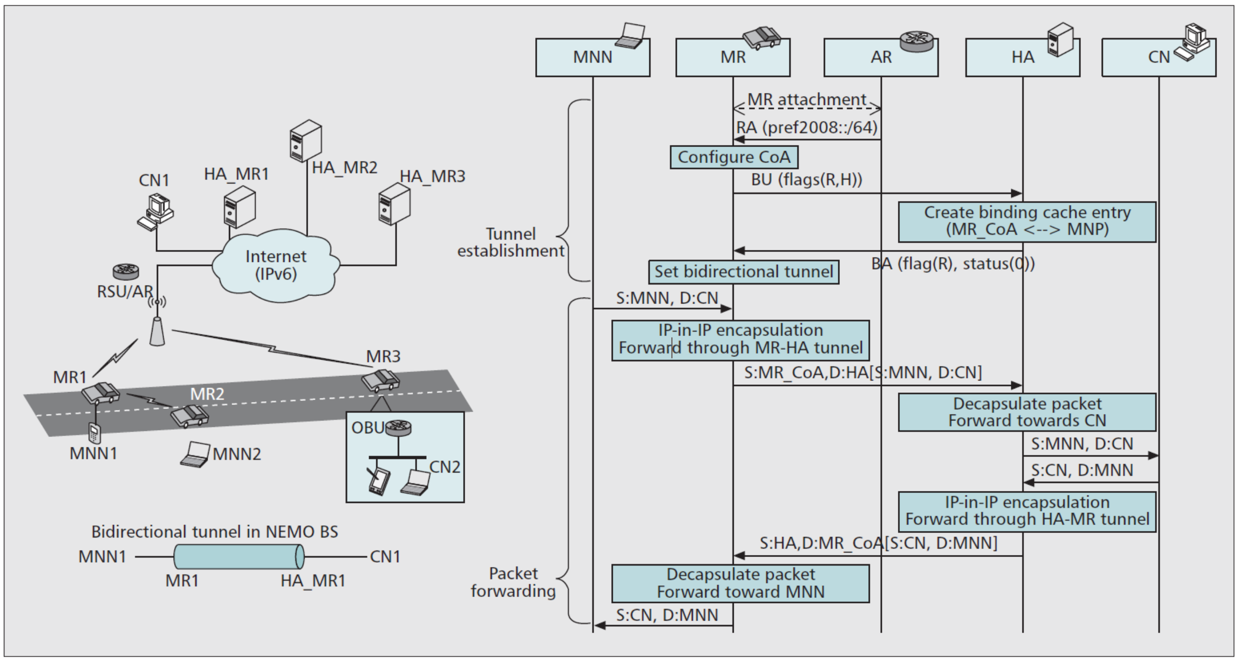

MNN
MR
AR
HA
CN
MR attachment
RA (pref2008::/64)
Configure CoA
BU (flags(R,H))
Create binding cache entry (MR_CoA <--> MNP)
BA (flag(R), status(0))
Set bidirectional tunnel
Tunnel establishment
S:MNN, D:CN
IP-in-IP encapsulation Forward through MR-HA tunnel
S:MR_CoA,D:HA[S:MNN, D:CN]
Decapsulate packet Forward towards CN
S:MNN, D:CN
S:CN, D:MNN
IP-in-IP encapsulation Forward through HA-MR tunnel
S:HA,D:MR_CoA[S:CN, D:MNN]
Decapsulate packet Forward toward MNN
S:CN, D:MNN
Packet forwarding
CN1
HA_MR1
HA_MR2
HA_MR3
Internet (IPv6)
RSU/AR
MR1
MR2
MR3
MNN1
MNN2
OBU
CN2
Bidirectional tunnel in NEMO BS
MNN1
CN1
MR1
HA_MR1

Figure 2.6: NEMO overview and association process [3]

In the figure 2.6, it is given a general overview about the NEMO protocol. This protocol discriminates the nodes that are trying to connect to the mobile networks by its capability of moving through the mobile networks.

First, the MR connects to an AR as can be seen in figure 2.6. The AR is now responsible for sending a RA, that provides the MR with a CoA. Next, the MR sends a BU to the HA in order to register the CoA in the HA. The HA, by its turn, creates an entry in the BCE such as *MNP:PoA* that redirects the traffic destined to the mobile network through CoA. Before sending the *BA*, the *HA* creates a tunnel that will be used whenever a MNN tries to send information to any CN and vice-versa.

In order for the VMN to join the network, the association process of MR is exactly the same, but now, contrary to the LFN, it will register the MNP in its HA. In the HA it will be created a BCE equal to VMN:MNP, and another tunnel is created. Like in the previous case, all traffic is routed through tunnels, but with the difference that now the traffic destined to the VMN is doubly encapsulated, as the figure shows.

2.4.3.3 Considerations

Concluding, from the analysis of this protocol some red lights appear. As it was noticed in [3], although this architecture seams to fulfill all the VANET mobility requirements, it has some gaps. First, it was not designed to support the dynamics and high mobility of a VANET. Another important aspect is that this protocol has limitation in highly dynamic scenarios [3], and it was noticed also in another investigation work [26], that the handover latency is high if the signaling messages are lost due to instability of the wireless link. Having all this into account, this protocol was not selected.

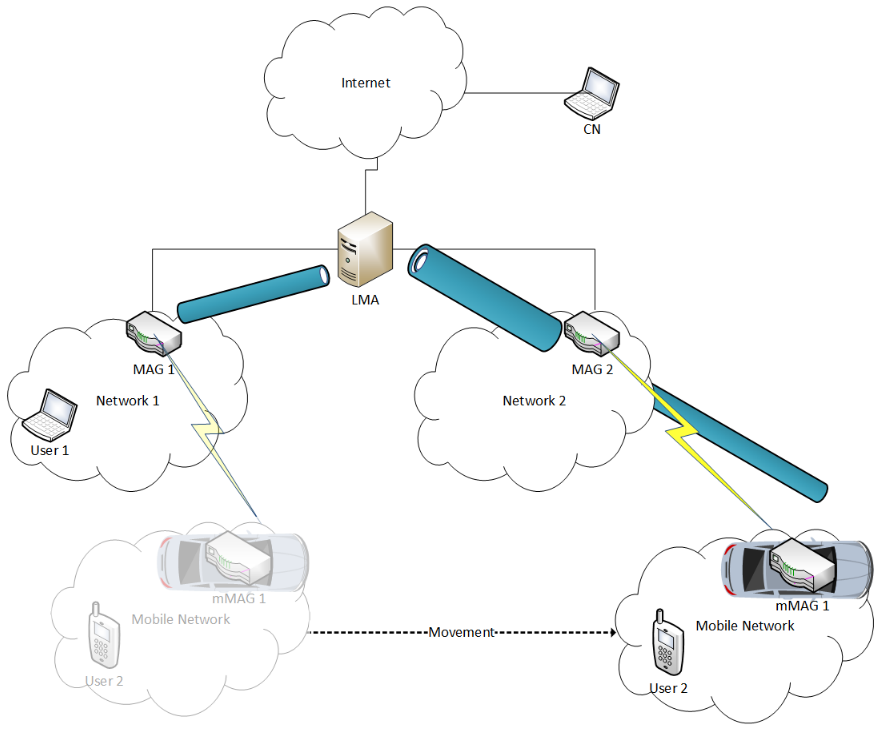


Figure 2.7: N-PMIPv6 protocol overview

2.4.4 N-PMIPv6

Finally, it will be presented a protocol that merges the PMIPv6 and the NEMO features, the N-PMIPv6. The explanation will be based on [27].

2.4.4.1 Terminology

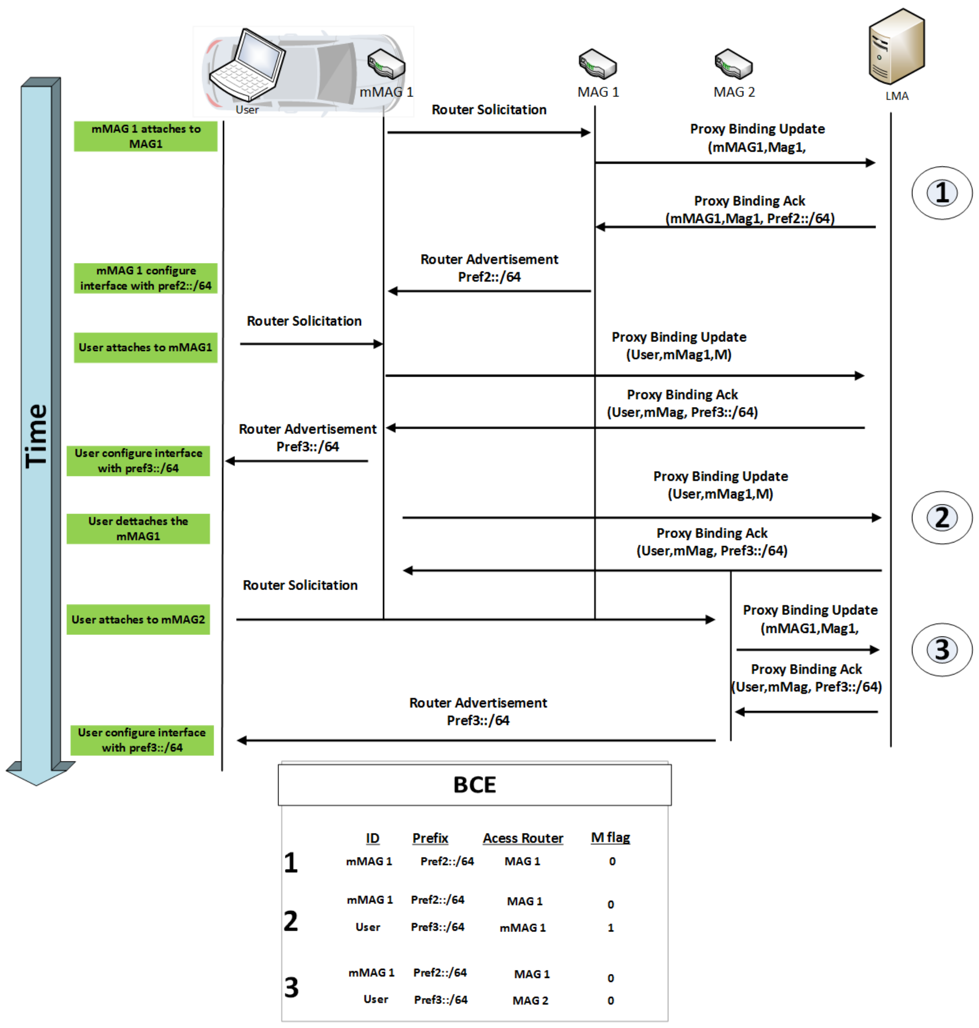
Since this protocol is a natural extension of the PMIPv6 and the NEMO protocols, the terminology is almost identical. It was only added a new entity, as will be described next.

- **mobile MAG (mMAG) :** Is a MR, which work very similarly to a MAG in the PMIPv6.

2.4.4.2 Behavior

In the figure 2.7, it is given a general overview about the NEMO protocol. First, when a mMAG wants to join the network, it must send an Router Solicitation (RS) to the serving MAG. When the MAG detects this event, it sends a PBU (with the MN-ID) to the LMA. Now, the LMA for its turn, checks if there is a BCE that contains useful information, such as MN-ID, MN-HNP, AR, and the M flag (that indicates if the PBU message was sent by a mMAG ($M = 1$) or by a MAG $M = 0$). After the BCE is filled, the LMA creates the IPv6 tunnel that has the MAG as destination, and forwards the PBA to the MAG, that contains the MN-HNP assigned to the mMAG. Then, the MN-HNP is forwarded to the final destination (mMAG) through a RA, that configures the mMAG IPv6 address on the egress interface. Now, when the user inside the vehicle wants to join the network, a similar process occurs. The user sends a RS message to the mMAG, that by its turn sends a PBU to the LMA, that again fills the BCE like in the previous case, but now with the M flag equals to 1 (because the PBU was sent by a MR). Then, it will create the IPv6 tunnel with the mMAG as endpoint and will send the PBA with the MN-HNP that was attributed to the user. When the PBA arrives, the mMAG creates its endpoint of the IPv6 tunnel and sends the RA to the user.

In the figure 2.8, it is shown an example of how the control messages are traded in this protocol. Moreover, it shows an interesting feature of this protocol. In this image, it can be noticed that the user 3 (originally connected to the mMAG 1), moves to the MAG 2. The mMAG detects that action and sends a PBU informing the LMA. Latter, the user 3 connects to the MAG2 and its IPv6 address will be the same as when it was connected to the mMAG. This particular aspect means that the mobile node can now roam freely through the available MAGs, and the IPv6 address will remain the same.


User
mMAG 1
MAG 1
MAG 2
LMA
Time
mMAG 1 attaches to MAG1
Router Solicitation
Proxy Binding Update (mMAG1,Mag1,
Proxy Binding Ack (mMAG1,Mag1, Pref2::/64)
1
mMAG 1 configure interface with pref2::/64
Router Advertisement Pref2::/64
User attaches to mMAG1
Router Solicitation
Proxy Binding Update (User,mMag1,M)
Proxy Binding Ack (User,mMag, Pref3::/64)
Router Advertisement Pref3::/64
User configure interface with pref3::/64
User dettaches the mMAG1
Proxy Binding Update (User,mMag1,M)
Proxy Binding Ack (User,mMag, Pref3::/64)
2
User attaches to mMAG2
Router Solicitation
Proxy Binding Update (mMAG1,Mag1,
Proxy Binding Ack (User,mMag, Pref3::/64)
3
Router Advertisement Pref3::/64
User configure interface with pref3::/64
BCE

	ID	Prefix	Acess Router	M flag
1	mMAG 1	Pref2::/64	MAG 1	0
2	mMAG 1	Pref2::/64	MAG 1	0
	User	Pref3::/64	mMAG 1	1
3	mMAG 1	Pref2::/64	MAG 1	0
	User	Pref3::/64	MAG 2	0

Figure 2.8: N-PMIPv6 association/disassociation process

2.4.4.3 Considerations

Concluding, this protocol has indeed some interesting features. Firstly, it allows mobility of an entire network through the several available networks, with an efficient handover mechanism. Secondly, this solution is perfect to be implemented in a real involvement, since the final users only have to connect to the network shared by the mMAGs, and do not have to install any extra software or do any extra configuration in their devices. Last but not least, the mMAG concentrates a certain number of terminals (users), and in a handover scenario, the MAGs only has to inform of the LMA of the mMAG movements, instead of each users movements [27].

For all these reasons, this was the selected mobility protocol by our research group. Due to that, in the next chapter it will be explained how each entity works in detail.

2.5 Multihoming

Nowadays, with all the development in wireless technologies, a user can have many networks of different technologies available at the same time. When that happens, some question may arise. Which technology should be used? Should we use all available networks? If yes, how shall the traffic be divided through each available network?

To answer to all these questions many research works studied the advantages of using several technologies at the same time, concluding that Multihoming (MH) can indeed increase reliability, performance, and help to manage efficiently all available resources. One of the most interesting feature of multihoming is the load balancing possibility. Because in multihoming it is normal to have multiple connections available, it turns possible to manage the amount of traffic that goes/comes in which connection based on its load or connection quality. But nothing comes for free, and normally this is done by introducing control traffic between the user and Point-of-Attachment (PoA)s, and adding extra complexity to the entity that is responsible to manage the multihoming operations. The first clear endeavor to use the MH features was done by the IETF, that created SCTP [28]: a connection-orientated protocol placed between the user application and the IP layer. Although this protocol can indeed use MH, it is in a backup logic, i.e, the device will have a primary address and a secondary one, but the secondary will be only used when the primary fails. Although this protocol does not take advantage of all the MH potential, [29] [30] proved that SCTP can provide better throughput and more robustness when used in wireless access networks. In order to exploit better the MH potential, extensions to this protocol have been created. In [31], it was proposed WiSE, a sender-side, transport layer protocol, that selects in real-time the best available network. Simulations shown that, when compared with STCP, WiSE extends the MH use, achieving gains both in throughput and network robustness. Moreover, [32] developed another SCTP extension, that aims to use the MH feature to facilitate seamless vertical handover in cellular data networks. The performed tests demonstrate delay and throughput improvements. Recently, in [33], it was proposed a multi-interface bandwidth aggregate architecture, that estimates the characteristics of the applications and dynamically manages how the traffic will be routed through the available interfaces. Through computer simulations, the authors concluded that the developed architecture improved the response time, that enhances the system's performance and consequently the users experience.

In [23], it is assured that MH will have a key role in vehicular networks. It refers that MH could be used for session continuity, vertical handovers and load balancing mechanisms. In [34], the authors propose CoMoRoHo, where the MH feature was used, in vehicular network, to reduce both the latency and the amount of packet losses. In [35], the authors proposed MH framework, that works as a N-PMIPv6 [27] extension. In this framework, it is proposed a

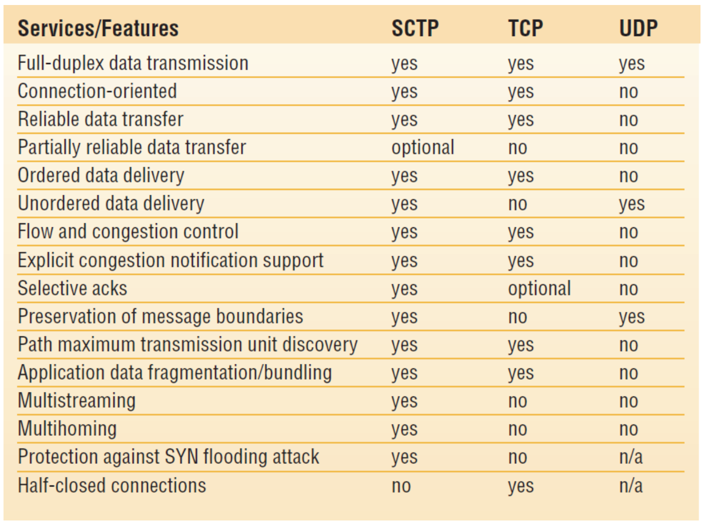
MH system that is able to divide a flow/flows through the available resources, in an optimized way, improving the network performance in terms of throughput, delay and packet loss. In [36], the authors propose a mobility manager based on MIPv4/6 [20][24], focused in network selection and improved vertical and horizontal handovers, using a metric that combines the delay and the jitter. Moreover, the frequency of the control messages (Biding Updates) are changed in real time, depending on the speed of the vehicle, in order to react quicker to network changes. The authors successfully decrease the amount of packet losses. Latter, in [37], it was developed QUVoD, a Vehicular Ad-Hoc NETwork (VANET) that uses the MH features in a Peer-to-Peer (P2P) network. The authors concluded that the use of both 4G and VANET networks at the same time increase both the data rate and reliability.

As we can observe, multihoming is mainly used to improve the handover mechanism and networks robustness.

In the next subsections will be detail some multihoming architectures, with more detail.

2.5.1 SCTP - Stream Control Transmission Protocol

According to [28], the SCTP protocol born to fulfill some gaps in the Transmission Control Protocol (TCP). TCP assures both reliable and ordered data transfer. This last aspect is accomplished with a delay increase due to SYN/ACK mechanism, meaning a handicap to applications that need a reliable and timely delivery. Moreover, the fact that TCP is stream oriented instead of message oriented increases the complexity of sending reliable control messages in some applications. Moreover, the limited scope of TCP sockets and the vulnerability to denial-of-service attacks mean that something needed to be done. Nevertheless, SCTP was created to fulfill those gaps and it was discovered latter that it had the potential to be much more. Besides satisfying all the previous handicaps, it also supports multihoming and multi-stream.



Services/Features	SCTP	TCP	UDP
Full-duplex data transmission	yes	yes	yes
Connection-oriented	yes	yes	no
Reliable data transfer	yes	yes	no
Partially reliable data transfer	optional	no	no
Ordered data delivery	yes	yes	no
Unordered data delivery	yes	no	yes
Flow and congestion control	yes	yes	no
Explicit congestion notification support	yes	yes	no
Selective acks	yes	optional	no
Preservation of message boundaries	yes	no	yes
Path maximum transmission unit discovery	yes	yes	no
Application data fragmentation/bundling	yes	yes	no
Multistreaming	yes	no	no
Multihoming	yes	no	no
Protection against SYN flooding attack	yes	no	n/a
Half-closed connections	no	yes	n/a

Figure 2.9: Comparison between UDP, TCP and SCTP main features [4]

SCTP, like the TCP protocol, is a connection-oriented protocol placed between the user application and the IP layer. To support multihoming, in the initial association process between the two SCTP endpoints, the endpoints exchange a list with multiple IP addresses in combination with a SCTP port, through which they can communicate. One interesting fact is that multihoming is used as a backup mechanism, i.e, there is a primary address that is used to route all traffic, and there are secondary addresses that are used when the primary address fails, or when the upper-layer application explicitly requests the use of those addresses. In resume, figure 2.9 shows the SCTP, TCP and UDP main features.

Later, in order to add some mobility to this protocol, [38] proposed a SCTP extension that allows that the SCTP stack to dynamical add/delete an IP address that was originally configured in the association process and change the primary address. Concluding, this solution assures multihoming but not satisfies the requirements of this and the previous dissertations, because as was said before, the multihoming in STCP is used in a backup logic, and is not used to divide traffic through all available connections in order to increase overall network performance.

2.5.2 HIP - Host Identification Protocol

The Host Identity Protocol (HIP) was designed to add extra security and mobility within the IP architecture, but as in the SCTP protocol, it was concluded that it had the potential to be much more. As turns out, HIP provides a name space for interconnecting distinct networks, reducing the arm that NATs translation mechanism implies. In order to support multihoming, this protocol split the identifier/locater from the same entity. In the IP protocol, the IP address has two purposes. Firstly to act as locator, i.e, describe the current topological location of the host network graph, and secondly to act as host identifier, i.e, describe the identity of the host to the upper layers. This particular aspect turns the mobility, and particularly the multihoming, almost impossible [39].

To solve this problem, HIP separates the locator and the identity role of IP addresses by introducing a new name space, the Host Identity (HI), that is a public cryptographic key from a public-private key. A host processes the corresponding private key, and prove the ownership of the public key, in other words, its identity. In this implementation the locator will be the IP address, but the identifier, or as the authors called, the Host Identity Tag (HIT), is an IPv6 address with 28-bits prefix (2001:0010::/28) called *Orchid*, that is followed by 100 bits taken from a cryptographic hash of the public key. Moreover, the mobility is also assured, because when a node moves, it will naturally gain a new IP address, and both cryptographic keys and HIT will remain the same, meaning that the ongoing session is the same. Finally, in order to ensure absolute mobility, a Rendezvous node was implemented, responsible for mapping the nodes identification with their location. This particular aspect means that a HIP node can change its location in the network without any setback, having only to update the binding between its location and its identity in the rendezvous node [40][41].

2.5.3 Shim6

Shim6 architecture [42] was designed to ensure multihoming and overall network mobility, but without compromising the scalability of the overall routing system. To do so, and like the HIP protocol, it decouples the identification and location role of the IP address, but now this is done by an extra sublayer inside the IP layer instead of a new name space. This sublayer is also responsible to translate the addresses used for exchanging packet on the wire (the locator) to the constant addresses of the upper layers (Upper-Layer IDentifier (ULID)), and vice-versa. This ULID address is used for session identification and is selected among the set of available addresses and remains static until the end of the session. Moreover, the other addresses that were not selected to be the ULID address will be considered as locators [43]. Another similarity to the HIP protocol is the use of cryptographically generated addresses to bind a set of IP addresses to a single node, and to check to which node an IP address belongs. Next, it will be presented some bullet that resume the key features of this protocol [41]:

- **Location-independent identifier:** In order to be possible to match a node with multiple connections, as said before, the identifier and the locators must be independent.

- **Compatibility with IP routing :** Unlike the HIP protocol, that needs to add an extra name space that implies that all network stack has to be changed. The shim6 protocol, due to a more simplistic approach in this field is both compatible with the IP routing and at the same time capable of communicating with the node that is not running the Shim6. These two aspects, nowadays, are of the utmost importance.

- **Transparency :** Specific characteristic of all mobility protocols. This means that the mobility should be transparent to the upper-layer protocols.

- **Security :** Multiohming has many benefits but, due to the number of available connections at the same time, it opens some security concerns. In Shim6 particular case, it could be vulnerable to *hijacking attacks* and *flooding attacks*. This kind of threats are mitigated in this protocol by the use of the cryptographically generated addresses and hashed based addresses.

Concluding, this protocol is closer to the multihoming requirement defined earlier, but still not good enough. The load balancing and flow division feature is missing in this implementation.

2.5.4 Proxy multihoming as a PMIPv6 extension

Finally, it is presented a multihoming architecture [35] [8] developed in our group, that aims to take advantage of all the access networks available in order to increase the overall network performance. The interesting part is, as the name indicates, this multihoming architecture works as a PMIPv6 protocol extension, where the mobility is assured by the PMIPv6, and the multihoming management will be assured by a set of new entities that will be added to the LMA and to the MAGs. To achieve multihoming, the authors used IP replication in the user side in a controlled manner in order to have multiple paths to the same terminal. This last aspect along side with the fact that this solution is implemented over the network layer means that, for the end user, no changes or extra software is required. This architecture besides satisfying the items stated in the Shim6 architecture (except the security one) has some more interesting features:

- **Effycient traffic division :** This architecture manages the traffic division in order to effectively improve the Quality of Service (QoS) of the end user and add advantages in the access network layer.

- **Real time multihoming optimization :** This architecture uses entities that, in real time, communicate with each other to trade control information that allow the multihoming manager to adapt, without impairing network performance.

Concluding, taking into account what was stated in this chapter, this architecture was the obvious choice to exploit the multihoming benefices by several reasons. Firstly, and most important, the vehicular network was built upon the N-PMIPv6 mobility protocol, that is an extension to the original PMIPv6. So it is more practical to add the multihoming feature to the mobility protocol since both were built upon the PMIPv6. It was also noticed that the previous protocols had its own mobility mechanism.

Moreover, this architecture truly exploits the multihoming benefits, i.e, it takes into account all the entities presented in the network, achieve the optimal traffic division that, in the end, effectively improve the final user QoS and add benefits to the access network.

Since this was the multihoming architecture selected, in the next chapter it will be explained with detail how it was implemented and integrated in the vehicular network developed in our group.

2.6 Network Coding

Network coding is a concept that changes the way that the information is transmitted. In typical networks, when a set of data is sent from a source node to a destination node through a chain of intermediate nodes, it is used the well-known store-and-forward method. In this way, whenever an intermediate node receives data packets from the input link, these are stored, and then, a copy of each data packet is forwarded to the next node through the output link. With network coding, data packets are not simply routed by network nodes; the network nodes also have the capability to combine packets. Typically, when data packets (from the same or different sources) arrive to a given intermediate node, these are placed in a buffer to be posteriorly processed. Then, the intermediate node takes a specific number of these data packets and mixes them, which is designated by the encoding process. To use the encoded information, the destination node needs to perform the decoding process. In [44], the authors concluded that network coding can be used to reduce packet losses, increase throughput and even add more security and resilience to a network.

Network coding was firstly introduced in a theoretical work done by [45], where the author successfully demonstrated that by using network coding in the middle nodes it is possible to save bandwidth. Then, some research works related to network coding utilities appeared, like in [46], where it was demonstrated that network coding could outperform normal routing in certain conditions. In [47], it was built a large scale content distribution network based on network coding, demonstrating through simulations that the use of network coding improves the download time by 2-3 times when compared with no network coding scenario. A year latter, Katti [48], developed COPE, an architecture for wireless networks that intelligently mixes packets before actually send them (applies network coding) and successfully improves the overall throughput of the network. Recently, in 2014, [49] developed an architecture that exploits the network coding in P2P vehicular networks, and concluded that the use of network coding increases significantly the performance. In this section, it will be explained what is Network Coding, its main features and how it can be used in vehicular environments.

2.6.1 Linear Network Coding

One of the first practical implementation of the network coding was the Linear Network Coding (LNC) [50]. In this implementation, a number of normal packets are linearly combined in order to form one coded packet. In this way, one coded packet contains relevant information about a certain amount of normal packets. The process to create this packet is called "Encoding".

To obtain the original information, it is executed the "Decoding process". For the decoder successfully decode the information, two things must happen. First, the decoder must know the coefficients used by the encoder to linearly combine the normal packet, and must also receive enough coded packets, that enables the decoder to extract the specific packets that are missing. In this case, the encoder must receive the same amount of coded packets as the number of normal packets used to create the coded packet.

In the next subsection, it will be explained several network coding approaches.

2.6.2 Random Linear Network Coding

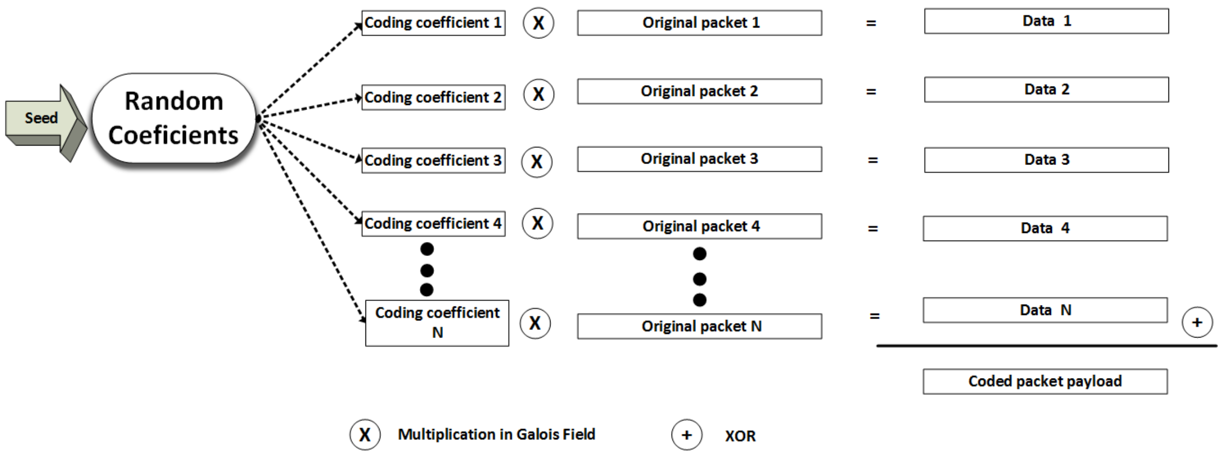


Figure 2.10: RLNC encoding procedure

The main difference between the Random Linear Network Coding (RLNC) and the LNC is in the coding coefficients. In the RLNC [50], these coding coefficients are obtained randomly from a Galois field. This particular aspect improves the performance of the RLNC, if compared with the LNC, in terms encoding/decoding complexity, throughput, and network overhead.

In the figure 2.10, it is illustrated the creation of an encoded packet. Although the coefficients used to encode the packet are randomly generated, again, the decoder must use the exact same coefficients in order to obtain the original packet from the coded packets. To solve this limitation, this approach uses pseudo-random generated values. In other words, there is a seed responsible to generate the values, and if the seed is the same, the generated values will also be equal. Now, to create an encoded packet, the pseudo-random value is multiplied (Galois field) by the normal packets. This action is repeated N times, and N is the number of normal packets used in the creation of each packet. The results from the previous operations are then XORed, and the payload of a coded packet is obtained.

Finally, in the decoder side, in order for this entity to obtain the original packet from the coded packets, the decoder must know the coding coefficients using in the encoding process. The coded packet is equal to $C = c * O$, the $O = C *(1/c)$, where C is the coded packet, c is the coding coefficient and O the original packet. Although this seams easy, this encoding/decoding operations are based in some heavy weight mathematics, (that can be found in [51]).

In [51] it is shown that, when used in multi-cast, this network coding approach can improve network robustness, throughput (see butterfly mesh in figure 2.12 in the section 2.6.4.1) and even security (decoder needs to know the seed).

2.6.3 Systematic and Non Systematic Network Coding

There are two major network coding approaches, the Systematic and the Non-Systematic. As shown in the figure 2.11, in the Systematic approach a generation (N+M) has N original packets and M redundant coded packets. However, in the Non Systematic case a generation has N plus M redundant coded packets. In both cases, in order to decode the original information, the decoder must receive N from the N plus M sent packets by the encoder. Based on this, the systematic network coding approach is typically seen as a best option to use in environments with losses: even if all M encoded packets are lost, the destination node can use the received original packets.

Taking into account these aspects, we selected as the base approach to be extended to our scenario, the Systematic Network Coding approach proposed in [52]. In the next subsections it will be explained with detail how the encoding and the decoding are done.

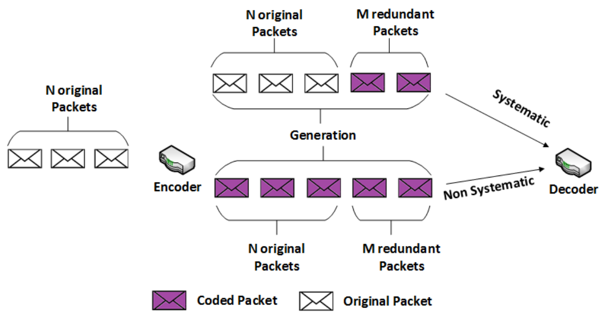


Figure 2.11: Systematic vs Non systematic NC

2.6.4 Network Coding practical applications

2.6.4.1 COPE

To take advantage of the network coding potential, in [5] the authors developed a new forwarding architecture that aims to increase throughput. In the figure 2.12 it is illustrated how the network coding can increase the throughput. First, the two senders broadcast the packets to the next hop nodes. The router will build an encoded packet, that is the result from the two received packets XORed and then the packet is sent to the next hop router. This router receives the coded packet and broadcasts it to the receivers. The receivers that have already revived one normal packet, with the coded packet can now obtain the packet that was missing. For example if packet A content is [1001] and packet B is [1111], the result of the XOR operation will be [0110]. Now taking into example the left receiver, in order to obtain the other packet that is missing, it takes the payload from the packet A, XORs it with the payload of the received coded packet, resulting in a packet equal to [1111], that is equal to packet B. This way, the receivers obtain two messages in every time unit instead of the normal one message. Taking into advantage from the IEEE802.11n broadcast nature, [5] created a system that disposes the point-to-point abstraction and adopts the broadcast nature of the wireless channel. Moreover, this architecture is based in two pillars that will be described in the next two items:

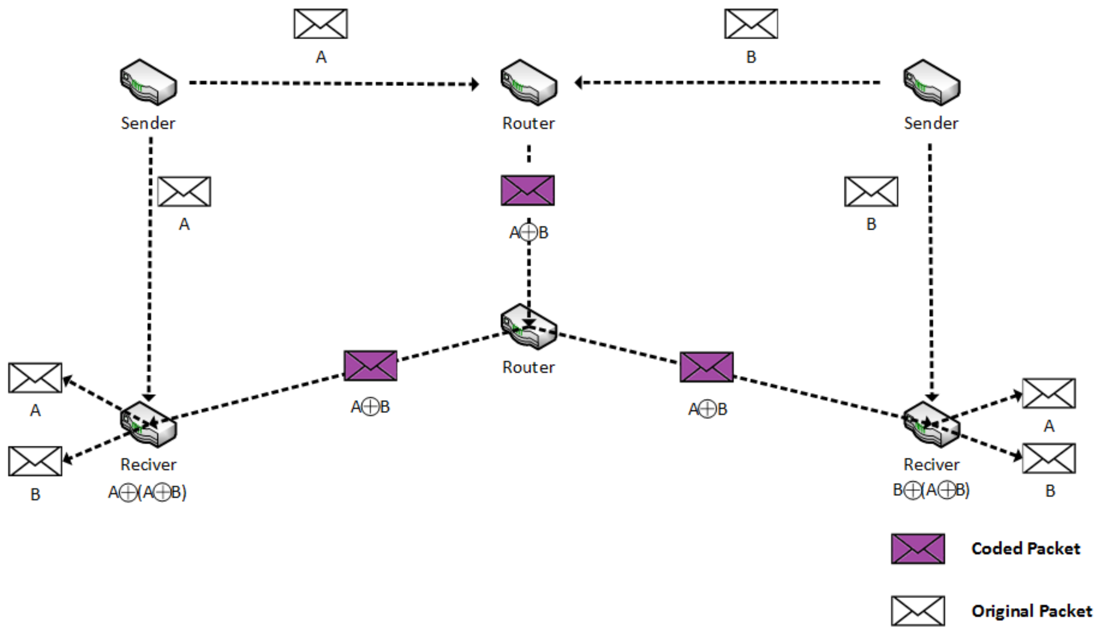


Figure 2.12: Butterfly mesh example

- **Opportunistic Listening :** Since this approach uses the broadcast as base, some interesting opportunities will arise. In COPE, all nodes are in promiscuous mode (allow the node to listen all communication over the wireless medium), and store the overhead packets for a limited period, in order to decide which packets will be combined and sent. Moreover, each node is responsible for sending a *reception report*, informing the neighbor nodes which packets are stored.

- **Opportunistic Coding:** : To effectively increase the network performance, and taking into account the base architecture explained in the beginning of this subsection, one major concerns arises. How does a node know what packets to XOR? Will the node be able to decode the packet? Taking as example the figure 2.13, it will be explained how the authors in [5] ensure a smart use of the network coding properties. In the figure 2.13 there are 4 nodes, each one with different packets stored. Since node 4 is the one with the larger number of packets, it will be responsible to broadcast the coded packet to the remaining nodes. Now a dilemma appears, since node 4 has 4 packets, there are many combinations for creating the coded packet. For example, if the middle node chooses to XOR the packet number 1 and 2, the node 1 and 3 can decode the information, but the node 2 is unable to do so. If, on the other side, XOR the packets 1 and 3, the nodes 2 and 3 can decode the information without problems, but now the node 1 is unable to do decoding. If the middle node encodes the packets 1,2 and 3, and broadcasts these packets, all the nodes will be able to decode the packets.

This work is of the utmost importance, because it showed that the network coding has practical utility. From the tests performed in [48], the authors conclude that, when the wireless medium

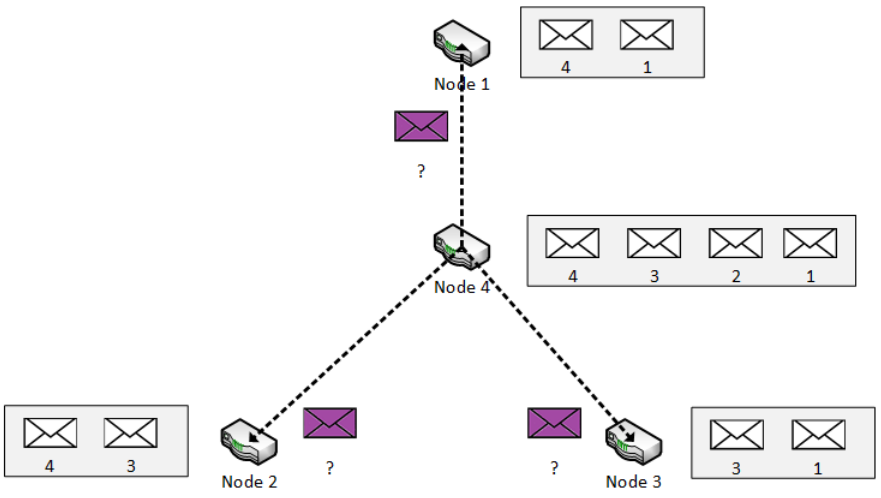


Figure 2.13: Opportunistic Codding dilemma, based on [5]

is congested, and the traffic consists in UDP flows, it is possible to have 3 or 4 times more throughput than without network coding. Another conclusion obtained was that, when it is used the mesh testbed, it is possible to obtain throughput gains between 5 and 70 %, depending on the ratio between uplink and downlink traffic.

2.6.4.2 VANETCODE

To implement a mechanism that effectively enables rapid sharing of emergency messages and multimedia files in vehicular networks, in [6] it was developed VANETCODE, that joins VANETs and Network coding.

In this approach, the content is divided in small blocks, that are linearly encoded and sent to the neighbor nodes taking advantage of the wireless broadcast characteristic. Another interesting fact is that, in this approach, it is eliminated the need of peer selection, content selection and neighbor discovery, which in authors words, increase the delay and steal resources from the VANET.

In the figure 2.14, it is shown an example on how the VANETCODE works. In this figure it is possible to notice three vehicles driving through a highway, in the moment that the three vehicles are within range of the RSU. In the RSU, there is a File X that first is divided in two small blocks (B1 and B2), and then, resulting blocks are also divided in smaller blocks (B11 B12 B21 B22).

Then, in order to form the encoded packets, the RSU randomly selects coefficients (C11 and C12 in case of the node 1) and uses linear encoding. To do so, first is combined the first element of each block. In other words, B11 is multiplied with C11 and B21 is multiplied with C12, and finally the results are added together, creating the first element of the resultant encoded block. To create the second, B12 is multiplied with C11, and B22 with C12 and again, the result of these operations is added, and so on. Another important aspect is that, in order for the vehicles to successfully decode an encoded packet, two things must happen. First, the node must know the coding coefficients used, and the node must receive enough number of blocks with linearly independent coefficients, in order to solve a set of linear equations. This aspect was solved sending the coefficients inside the coded packet. Second, since the coding

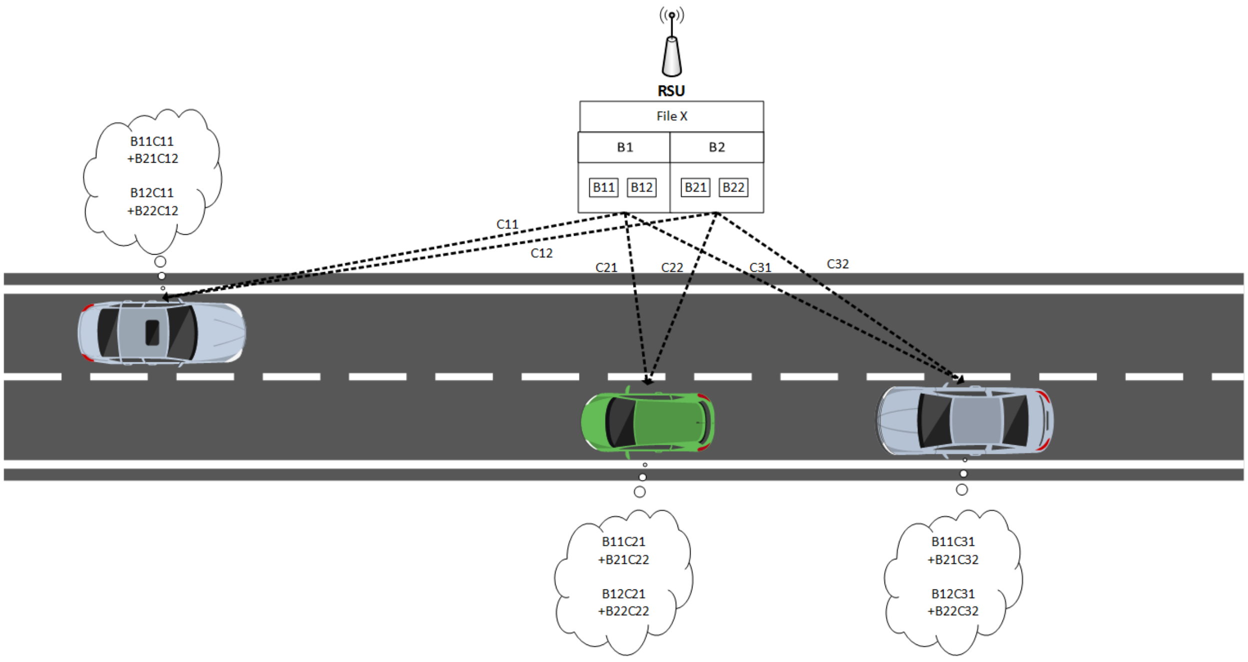


Figure 2.14: VANETCODE example, base on [6]

---

coefficients are chosen from a two raised to sixteen range, the authors considered that is almost impossible to have linear dependent blocks.

Another interesting choice in the architecture was to divide twice the File X. The reason for this is simple, since in the vehicular network it is normal to have low connection times, it is logical that if the nodes or RSUs sent larger data chunks, the amount of information that can be lost due to connection suddenly drops will be higher. This hierarchical division also allowed the authors to reduce the coding coefficient overhead. As shown in the figure, since B11 and B12 come from the division of B1, B11 and B12 can be coded using the same coding coefficient.

Now, lets imagine that the cars drive away from the RSU, and all the vehicles are out of range of the RSU. Now the vehicles will enter in a *share* state, which means that they will broadcast all their data between them. This forwarding process is similar to the procedure done by the RSU. The node selects random coefficients and linearly combines all the blocks and sends them in broadcast with the information about which linear coefficients were used. Last, in order do avoid broadcast collision, the nodes will wait for a random time interval before sending any encoded packet.

Finally, it will be presented some conclusions about of VANETCODE:

- **Archicheture highly depended on multicast :** From the previous explanation it is possible to conclude that, if only one vehicle requests the X file, this architecture will not work.

- **Conclusions base on simulation and assumptions :** The authors concluded that their VANETCODE achieved quick content distribution, and even outperforms other architectures that had the same goals. The problem is that this conclusion is based on assumptions (RSU in each 2-10 miles for example) and the tests performed are not in a real testbed scenario, but through simulations.

- **Overhead introduced in the network is not mentioned :** Although it is stated throughout the document that this solution does not increases the network overhead, there are no overhead results. Moreover, if the nodes do not communicate with each other about the packets that they have and only broadcast them blindly constantly, it would be interesting to quantify overhead values that this architecture requires to work properly.

- **A routing protocol is not required :** One of the major advantages of this architecture is the fact that, using all the broadcast capabilities, it is removed the need of peer selection, neighbor discovery and content selection, making it lighter and faster than other available architectures.

- **Computacional overhead :** Since all information must be encoded and then decoded, this will increase the computational load.

2.6.4.3 Distributed Storage Codes to Reduce Latency

Finally, it is presented another possible use for network coding, also in a VANET environment. In this work, [49] developed an approach that aims to create a P2P file sharing network without resorting to any RSU, because for the authors, RSUs are hard to deploy in high densities.

In the developed architecture there are nodes called seeds (cabs and buses), that are responsible for storing the available data (movies, music, etc) and sharing it with all vehicles that want to access to that data. In this work, the data is stored in two ways, and later on which one will be compared. The first one is to have many copies of each file, and the second is to have the stored files using erasure code, more precisely a digital fountain code, famous for its speed in encoding/decoding and to have a good performance (high probability) rebuilding the coded data successfully.

In this architecture it is considered that there are N seed nodes, with a storage capacity of C bits reserved for file sharing applications. Each seed has the capability to have m different files with size M. Having this in mind, a set of logical assumptions were made in order to build a robust model. First, it is assumed that $C >= M$. In other words, the seed must have space to store the file. Secondly, the seeds try to distribute the m files by as many nodes as possible, and it is the responsibility of the nodes to select the desired file randomly (each file has the same probability of being selected) in each encounter between the seed and the vehicles. It was also considered a limit of bits transfered from the seed to the vehicles (bandwidth constrains). Another interesting fact, that can be considered a feature of P2P architectures, is that each file will not be stored in all the seeds, and a file will not be stored multiple times in the same seed.

It is possible to conclude from this explanation that this is a push-based mechanism, that on other words, means that the vehicles need to request data to a node, before starting to receive any data.

Finally, it will be presented some conclusions about this architecture:

- **Push archicheture :** Since all the information that is sent needs to be requested, there is not (almost) any overhead introduced, due to the limited control traffic that this solution requires.

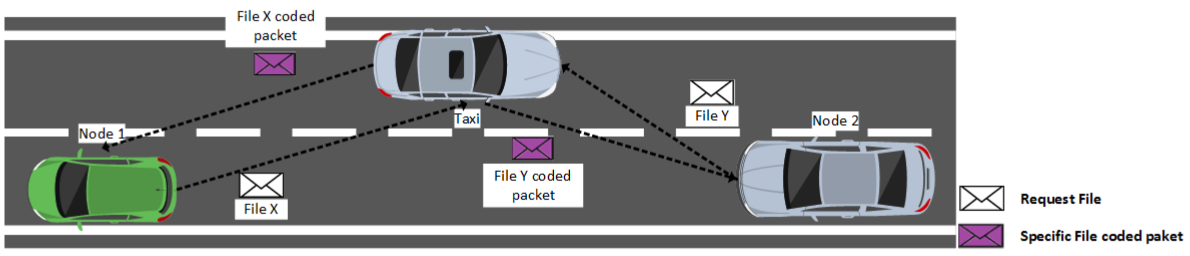


Figure 2.15: Example of network coding used in a P2P network

- **Realistic simulation in order to validate purposed architecture :** In order to validate the architecture, it used the GPS traces of 1000 Beijing taxis and 1608 buses in Chicago, where it was used a realistic model of IEEE 802.11p WAVE, in order to simulate the communication between the vehicles and the taxis/buses.
- **Coded content outperforms the uncoded content :** From the analysis of the results, the authors concluded that, in the worst case scenario, the coded content has an equal performance as the uncoded one, but in the majority of the cases, the coded content outperforms in a clear way the uncoded approach.
- **File size has a major impact on the results :** One of the most interesting conclusion was that, if the file size is relatively small (100MB) there were not any differences between the coded or the uncoded approach. However, when the size of the file increases, the difference between the coded and the uncoded approach becomes clear, and the coded approach clearly presents better results.
- **(Still) High latency :** Downloading a 1GB file takes 6 to 10 hours using the coded approach. In this matter, the authors defend that the 1000 nodes used, still leads to a quite sparse network, leading to a relatively low number of encounters between the buses / taxis and the vehicles. Moreover, the authors also advocated that in a vehicular network with 100 000+ vehicles this result would be, naturally, much better.

2.7 Chapter Considerations

In this chapter it was given an overview about vehicular networks, mobility protocols, multihoming and network coding.

It was possible to observe that the vehicular network main challenge is related to the high mobility of the nodes and constant handover between network. It was interesting to see also the huge potential that vehicular networks have. First, for the automobile manufactures, that can use these networks to send all kind of security messages and reduce road casualties. Secondly, there are, and will be, a huge market that can be exploited when all vehicles are connected to each other. To overcome all the mobility related difficulties, it was studied mobility protocols that suit the VANET special features, and it was concluded that, among the studied protocols, the N-PMIPv6 offered the best solution because it offers mobility to a whole network in a seamless way to the final user. Moreover, it was also studied some multihoming approaches. It was found an architecture (that was built as an extension to the PMIPv6 protocol) that uses all the available resources and performs an efficient traffic division in order to increase the overall network performance in Wi-Fi and cellular one-hop networks.

As it was said in the beginning of this document, one of the aims of this dissertation is to integrate network coding in the vehicular network to decrease the amount of packet losses, but since network coding is a relatively new area, it was decided not only to give the reeder an idea of what can be used for packet recovery, but also other possible network coding applications. It was concluded that network coding can be used to packet recovery, increase throughput, and even to optimize a P2P network.

Chapter 3

Base Work

3.1 Introduction

To understand the work done in this dissertation, it is important first to understand the status of the work that was already done when this dissertation begun. As referred before, the first objective of this dissertation was to developed an approach for uplink multihoming. This will require an uplink manager that will be implemented over a mobility protocol and a connection manager already implemented in our research group. The second major objective was to integrate a Systematic Network Coding (SNC) in the vehicular network with multihoming, and again there was already a base work done in this field.

Section 3.2 will describe the mobility protocol used as the basis for this dissertation.

Section 3.3 will describe the downlink multihoming and the original connection manger.

Section 3.4 will describe the base approach that was used to integrate the network coding in the VANET.

Finally, Section 3.5 will resume this chapter.

3.2 Mobility Protocol evolution

The first mobility approach for vehicular networks developed in our group is the one in [7]. In this work it was adapted the Open Air Interface Proxy Mobile IPv6 (PMIPv6 OAI) [53] (an implementation of the PMIPv6), that is based on a UMIP [54], in order to make possible to implement a multi-technology seamless handover mechanism for vehicular networks with a mobility manager that also selects the best technology available, in order to maintain the vehicle connected to the infrastructure without any session breaks [55].

The work in [7], using the modified PMIPv6 protocol as base, implemented the following features:

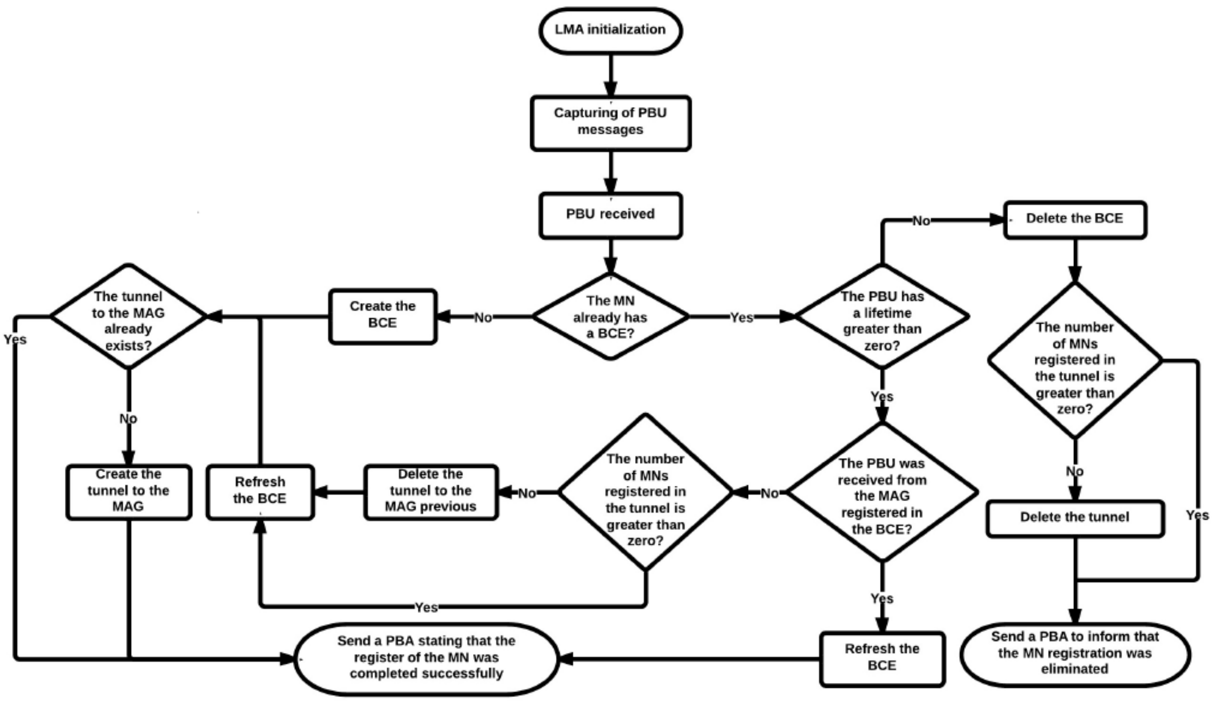
- A mobility mechanism based on N-PMIPv6;
- A connection manager that selects and connects to the best networks;
- An approach to allow the users inside the car to acess Internet via IPv4;
- An extension to mobility messages to provide IEEE802.11p technology.

The next section explains with detail how the LMA and MAG/mMAG behave in N-PMIPv6.

3.2.1 N-PMIPv6

As referred in chapter 2, N-PMIPv6 has three entities, the LMA, the MAGs and the mMAGs, and all together guarantee the global connectivity between themselves. This subsection will explain the flow diagram of the LMA, and of the MAG/mMAG.

LMA flow diagram



LMA initialization
Capturing of PBU messages
PBU received
The MN already has a BCE?
No
Yes
Create the BCE
The tunnel to the MAG already exists?
Yes
No
Create the tunnel to the MAG
Refresh the BCE
Delete the tunnel to the MAG previous
The number of MNs registered in the tunnel is greater than zero?
The PBU has a lifetime greater than zero?
The PBU was received from the MAG registered in the BCE?
Delete the BCE
The number of MNs registered in the tunnel is greater than zero?
Delete the tunnel
Refresh the BCE
Send a PBA stating that the register of the MN was completed successfully
Send a PBA to inform that the MN registration was eliminated

Figure 3.1: LMA flow diagram [7]

When the LMA receives a PBU sent by a MAG informing that a MN is trying to join the VANET, the LMA awakes, and checks if the node is already registered in the LMA or not. If the node is not registered, it is created a BCE and, if the tunnel between the MAG that send the PBU and the LMA is not created, the IPv6 tunnel will be created and a PBA is sent to the LMA signaling that the registration process is completed.

On the other hand, if the MN is already registered, the PBU lifetime field is checked, and if the value is equal to zero, the node is deleted from the BCE. If there are not any other nodes that use the tunnel used by the node that was deleted, the tunnel is deleted and a PBA is sent to the MAG informing that the node was deleted. Moreover, if the lifetime of the PBU is larger than zero two things can happen:

- **PBU is received from an already resgistred MAG :** This means that it is only needed to refresh the BCE and send a PBA to the MAG to inform that the MN session was refreshed.
- **Handover occured but the tunnel still has MNs associated :** If the mobile node is already registered in the BCE but the MAG is not, this can only mean that a handover

occurred, and now, if the tunnel to the MAG that was serving previously the node (before the handover) did not serve any other node, it is naturally deleted, and the BCE is refreshed. Finally, if the tunnel that links the LMA and the MAG was not created, it is created and a PBA is sent to that MAG informing that the registration process was successful.

MAG/mMAG flow diagram

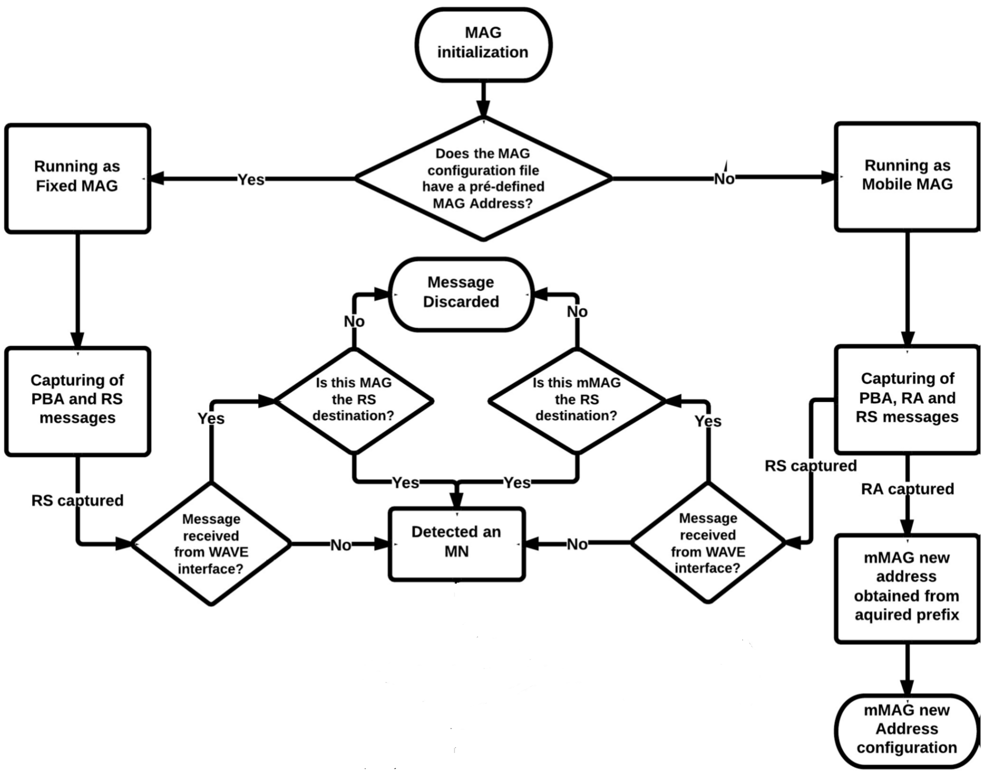

MAG initialization
Does the MAG configuration file have a pré-defined MAG Address?
Yes
No
Running as Fixed MAG
Running as Mobile MAG
Message Discarded
No
No
Capturing of PBA and RS messages
Is this MAG the RS destination?
Is this mMAG the RS destination?
Capturing of PBA, RA and RS messages
Yes
Yes
RS captured
Yes
Yes
RA captured
RS captured
Message received from WAVE interface?
No
Detected an MN
No
Message received from WAVE interface?
mMAG new address obtained from aquired prefix
mMAG new Address configuration

Figure 3.2: MAG/mMAG flow diagram [7]

As shown in figure 3.2, it was introduced the possibility of a MAG to behave like a mMAG depending on the mobility protocol configuration file. The behavior of these two entities is very similar and it will be explained next.

First, in order for the MAG to work as a mMAG, in the N-PMIPv6 configuration file, the egress address must not be configured. The mMAG will be capturing PBA, RS and RA. If a RA is captured, it means that the mMAG joined successfully the network, and with that RA the mMAG is able to obtain the network prefix in order configure the interface address. If, on the other hand, a RS is received, it means that another node is trying to joint the network and, in case this message is received in the WAVE interface, due to its broadcast nature, it is required to check if the mMAG was the real destination of that RS (flow diagram presented in the figure 3.2), and if it was, the MN will be processed.

The first action when a MN is detected is to check if it is allowed in the network, and if it is (flow diagram presented in the figure 3.3), two things can happen:

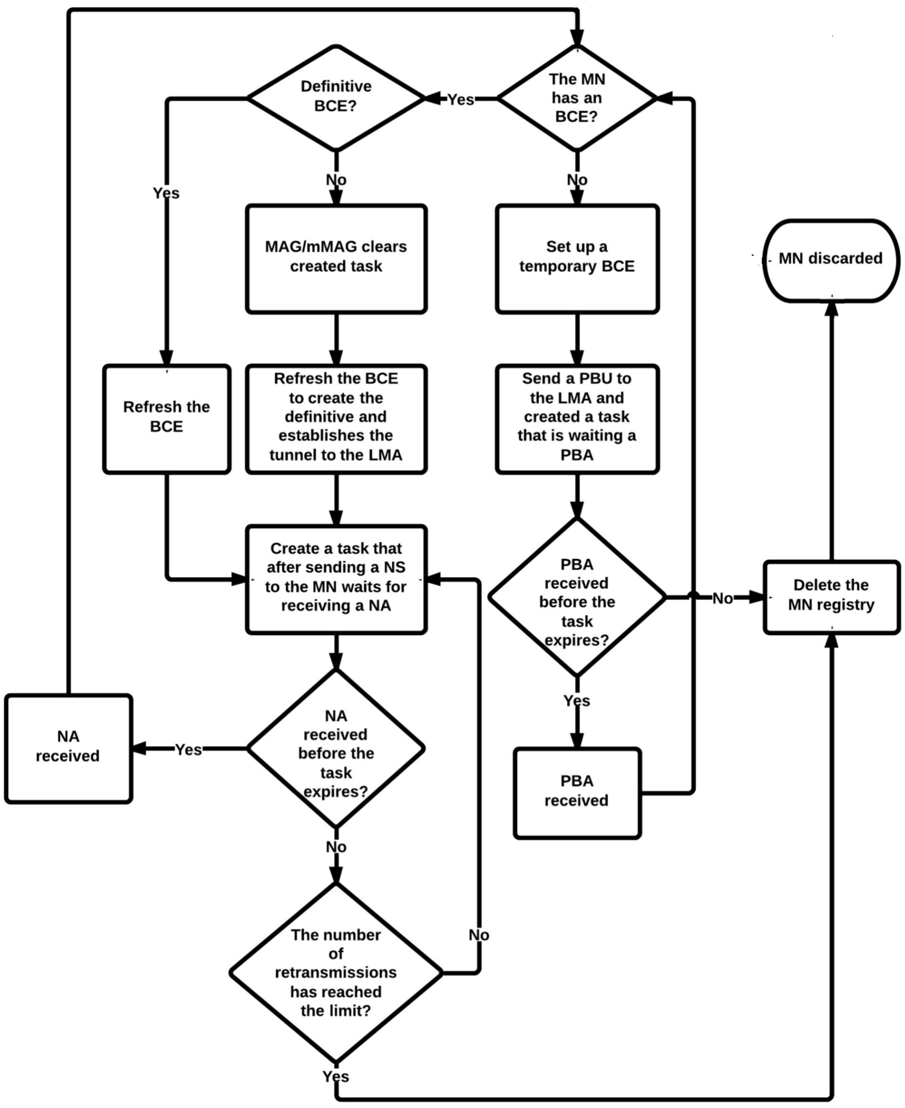

The MN has an BCE?
Yes
Definitive BCE?
No
Yes
MAG/mMAG clears created task
Set up a temporary BCE
MN discarded
Refresh the BCE
Refresh the BCE to create the definitive and establishes the tunnel to the LMA
Send a PBU to the LMA and created a task that is waiting a PBA
Create a task that after sending a NS to the MN waits for receiving a NA
PBA received before the task expires?
No
Delete the MN registry
Yes
NA received
Yes
NA received before the task expires?
PBA received
No
The number of retransmissions has reached the limit?
No
Yes

Figure 3.3: MAG/mMAG mobile node registration flow diagram [7]

- **The MN already has a BCE** If the MN already has a BCE associated to it, this means that the registration process already started before. This means that, if the BCE was temporary, it now becomes definite and an IPv6 tunnel to the LMA is established. If, on the other hand, the BCE is definitive, the goal of the RS is to maintain the session and the BCE is only refreshed.

- **The MN does not have a BCE** If the MN does not have a BCE associated to it, a temporary BCE is created, and the MAG/mMAG will wait a predefined timeout for a PBA sent by the LMA, confirming that the node is suitable to join the network. If in the predefined timeout no PBA arrives, the node will be discarded.

Now, in case the PMIPv6 configuration file has an egress address, it will behave like a MAG. The only difference between the mMAG and a MAG is in one small detail. Since the

MAG is designed to work as a RSU, it does not make sense to process RAs, only RSs and PBAs.

N-PMIPv6 network abstration and the multihop feature

Now, it is time to explain one of the major features of this protocol, that allows mobile nodes in multi-hop to connect the VANET in the same manner than the single-hop ones. As can be seen in the figure 3.4, when the mMAG1, using the connection manager and the N-PMIPv6 protocol, connects to the MAG1, an IPv6 tunnel is created between the LMA and the MAG1, and all the traffic from the mMAG1 to the LMA will be routed through there. Next, a new mobile node connects to the mMAG1, and since the N-PMIPv6 protocol is running, the normal procedure will be executed. There will be sent a PBU to the LMA and created an IPv6 tunnel from the LMA to the mMAG1 to serve the mMAG2. As the figure shows, from the mMAG 2 point of view, between it and the LMA there is only an IPv6 network, and the packets are sent through that network.

Concluding, it is clear to see that this protocol enables chains of mMAG without any relevant change.

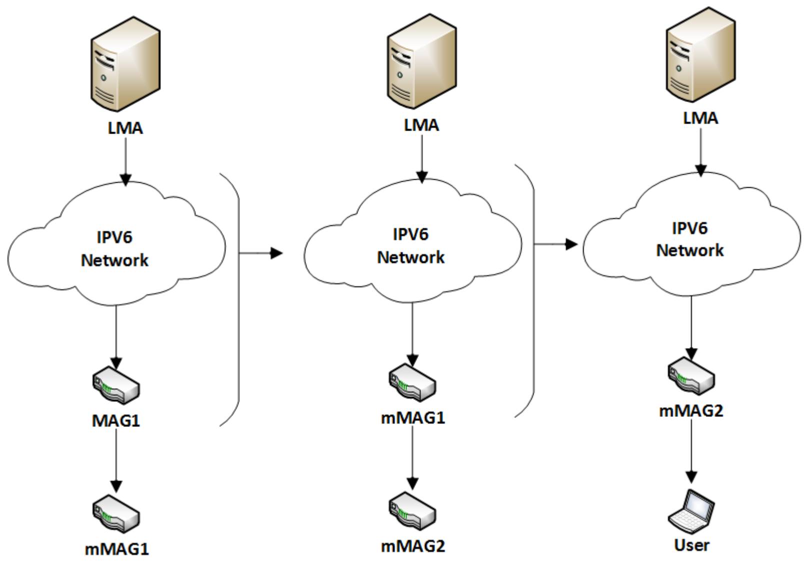


Figure 3.4: Network abstraction based on [7]

3.2.2 N-PMIPv6 Extra Features

In the previous subsections it was described the work done by Diogo Lopes [7], that was used as basis for the dissertations of Marco Tenrinho [10] and Andre Martins [9]. Both these dissertations were the base work of these dissertation, so it is important to understand their contribution:

- Implement multihoming in a single IEEE 802.11p.
- Integrate a multihoming framework in the mobility protocol;
- Give to the users access to Internet, either in single or multi-hop;

- Ability to differentiate traffic to be sent through different networks.
- Vehicular network with simultaneous access to WAVE, Wi-Fi and cellular, through downlink multihoming.

3.2.2.1 Multihoming architecture

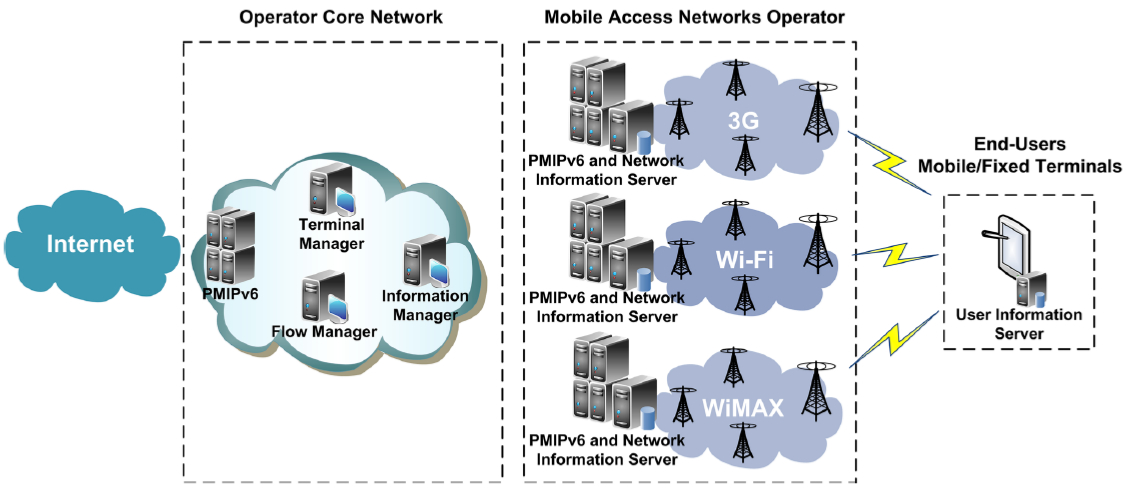

Operator Core Network
Mobile Access Networks Operator
Internet
PMIPv6
Terminal Manager
Information Manager
Flow Manager
3G
PMIPv6 and Network Information Server
Wi-Fi
PMIPv6 and Network Information Server
WiMAX
PMIPv6 and Network Information Server
End-Users Mobile/Fixed Terminals
User Information Server

Figure 3.5: Multiohming architecture [8]

In order to integrate a rule that optimizes the division of traffic through the available technologies, it was used an architecture developed by Nelson Capela [8], a Phd student also working in our research group. The proposed architecture is shown in figure 3.5. On the top level, the architecture is divided in three major entities, the Core Network, Mobile Access Networks and the End-Users, each one with its own sub-entities responsible for a set of specific tasks [8]:

Operator Core Network

- **PMIPv6** Mobility protocol running as LMA.
- **Terminal Manager** Responsible for managing the users interfaces.
- **Information Manager** Responsible for collecting all the useful information for the multihoming process.
- **Flow Manager** Manage all the aspects related with the terminals traffic.

Mobile Acess Networks Operator

- **PMIPv6** Mobility protocol running as a MAG.
- **3G/Wi-Fi/WiMAX** Access networks in which the user can connect in order to access the Internet.

- **Network information server** This entity is responsible to store and provide information about the state of the access network.

End-Users Mobile/Fixed Terminals

- **User Information Server** Responsible to store and provide usefull data about the users terminal.

Having this global architecture in mind, the main goal of [10] was to integrate this multihoming approach in the previous developed N-PMIPv6 protocol, but trading the WiMAX technology for the WAVE, naturally.

3.2.2.2 Multiohming Framework

To bypass the limitations in terms of multihoming, [8] proposed several changes from the original PMIPv6 protocol. The image 3.6 illustrate this framework.

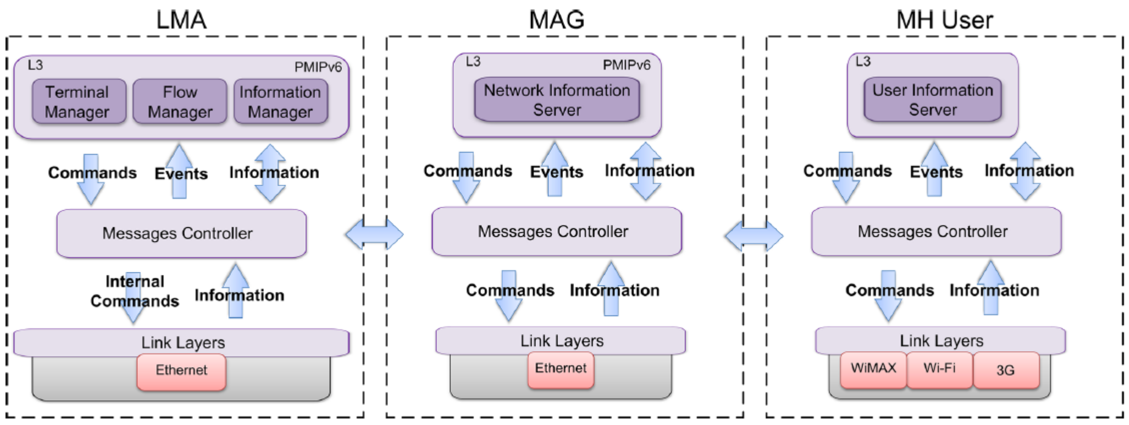


Figure 3.6: Multiohming framework [8]

LMA

- **Terminal Manager** Because the PMIPv6 uses the MAC address of the interface as connection identifier, in a multihoming scenario the same user would have two identifiers. To solve this issue, it was created the terminal manager, responsible for the association of the different connections to an end-user terminal. To do so, it was created an User Cache Entry (UCE) containing relevant information such as the terminal identifier, the number of connected interfaces and a set of information about each connected interface.

- **Flow Manager** To introduce intelligence and dynamism to the routing process of the mobility protocol, it was created the flow manager entity. This manager is responsible for detecting all traffic that arrives at the LMA, and according with the available resources at each moment, it divides the traffic through the access networks in order to maximize the network performance. In this manager, it was introduced a Flow Cache Entry (FCE), that interconnects a received flow to the end-user terminal. Each UCE contains useful information that makes possible to divide a flow and send it through multiples routes.

- **Information Manager** The information server is responsible for managing all the information from the surroundings, in order to calculate a multihoming rule that assures that all network resources are used in the best way.

MAG

- **Network Information Server** This entity is responsible to provide the necessary information to the Information Manager and check the current state of the access networks.

MH User

- **User Informantion Server** This entity communicates with the Network Information Server and enables the measurement approach. Moreover, it stores useful data such as the interface name, the RSSI, interface quality level, noise level, packets losses, achieved throughput and the PoA load.

In figure 3.7 it is presented a flow chart where it is described how the UCE and the FCE are introduced in the N-PMIPv6 protocol.

3.2.2.3 IPv4 over IPv6 Internet Support

To give the users inside the car access to the Internet over IPv4, since all the mobility protocol is based in IPv6 protocol, it was used IPv4 over IPv6 tunnels between the LMA and the mMAGs. In this subsection, it will be explained based on [9], how these tunnels are managed, and also how the tunnels could be used to provide multihoming to the uplink traffic.

In figure 3.8, it is shown the registration process of a new node. When the mMAG detects the two available RSUs, it will send a RS through each available interface. When the RS arrives at the MAG, this entity checks if the mobile node has permissions to join the network, and if it has, the MAG will create a BCE and then sends a PBU to the LMA. Now the LMA will create a BCE, UCE and a FCE and fills them with the information previously described throughout this chapter. Then, before sending the PBA confirming the registration, the LMA creates the IPv4 over IPv6 tunnels with source in the LMA address and destination address equals to *2001:<USER_ID>::<TECH>:<BOARD_ID*, where the USER ID is the IPv6 prefix given by the LMA to the user, the TECH represents the technology in which the RS was sent, and BOARD ID is a unique number that identifies each mMAG. Next the PBA is sent to the MAG, that creates the IPv6 tunnel to the LMA and configures all the routes and rules needed to ensure connectivity between the mMAG and the LMA. Finally, when the mMAG receives the RA, the interface is configured with the IP address according with the RA (will be better explained later in this chapter), and the IPv4 over IPv6 tunnels will be created. To route the traffic to the Internet, it is used a Network Address and Port Translation (NAPT) server, that allows that all requests inside the VANET be treated as LMA requests. This protocol translates all the IP addresses inside the VANET in a way that for the Internet, the VANET is considered only one user. When the Internet sends back the request data, the LMA re-translates the addresses and route the traffic through the tunnels as if it was its own traffic.

Finally, the multi-hop Internet support for the regular users is also guaranteed by the same IPv4 over IPv6 tunnels and by the network abstraction explained in this chapter, in section 3.2.1.

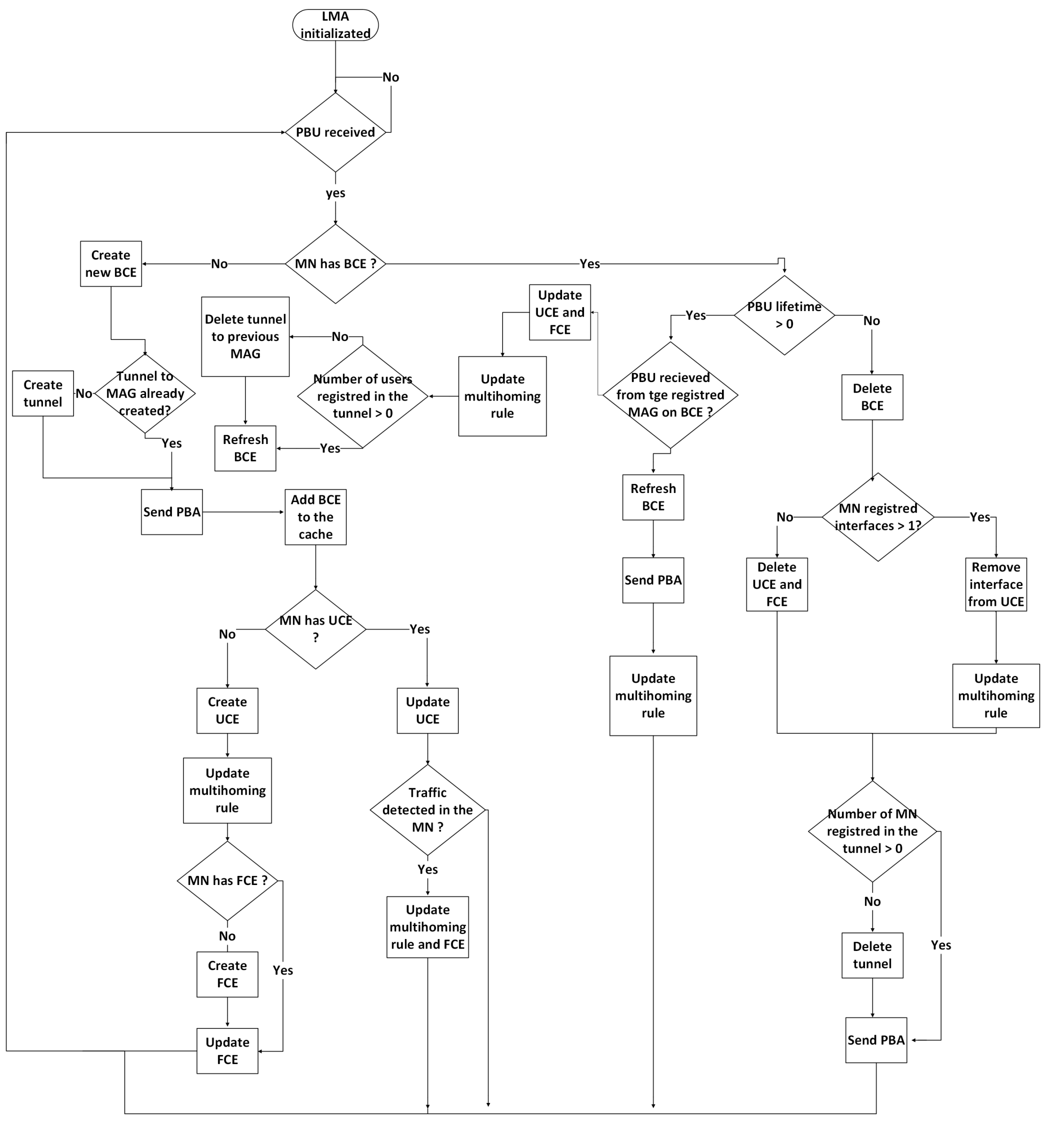


Figure 3.7: LMA multihoming extra logic based on [9]

This process ensures multihoming in the downlink however, in uplink this is not supported. If there were multiple technologies available to the mMAG, it was only used one IPv4 over IPv6 tunnel to route all the uplink traffic through a default route, that only had into account the type of technology to apply the default route (WAVE has priority over the Wi-Fi connections). Another limitation found was in the LMA IPv4 over IPv6 tunnel creation. In order to create these tunnels, the LMA needed to know the ID of the mMAG that is trying to join the VANET. This information was only available in the mMAG, so it was created a small "data base" that matched the IP source address with the addresses presented in that "data base" and retrieves the ID of the mMAG. This represents a major gap in the uplink management, because if a new node tries to join the network, and the IP addresses of the WAVE and Wi-Fi interfaces were not in the "data base", the tunnels would not be created correctly.

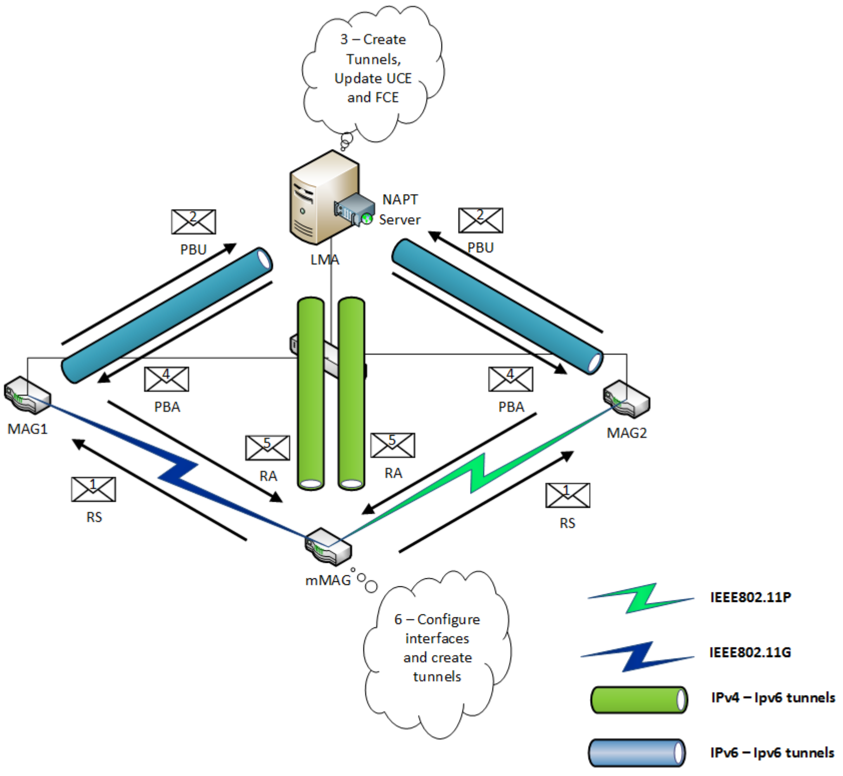


Figure 3.8: IPv4 over IPv6 tunnel creation

3.3 Connection Manager Operation

In this section it will be explained the behavior of the mobility connection manager implemented in the previous dissertations [10][9], that was used as starting point for the uplink manager. The next explanations will be based on those dissertations.

This connection manager allows the mMAG to:

- **Connect to MAGs with diferrent technlogies at the same time :** The goal is take advantage of the Wi-Fi, WAVE and cellular interfaces presented in the mMAG in order to increase network performance and availability.
- **Connect to one Wi-Fi MAG with the highest RSSI :** Because the OBU only has one Wi-Fi interface, it is only possible to connect to one Wi-Fi MAG without the use of a virtual interface. As criterion to choose the best Wi-Fi network, it was used the RSSI of each network.
- **Connect to several WAVE MAGs :** Because the WAVE technology does not require an association process, the connection manager will connect to all WAVE networks that satisfy a minimum RSSI threshold.
- **Perform the interface configurations :** As stated in chapter 3.2.1 , upon the registration of a new node in the network, RS and RA messages are exchanged. Now, with the multihoming feature, the same interface may receive multiple RS/RAs and act depending on each situation. The interface will be configured with an IPv6 address that is built

based on the RA prefix and on the Unique Identifier 64-bit (EUI-64). This last address will be built using the destination MAC address of the RA message.

- **Manages the uplink/downlink routes :** If there are many connections available, it is also needed to manage all routes in order to maximize the network performance.

The mobility connection manager is a program formed by three distinct thread, the WAVE, and Wi-Fi threads, responsible for managing all aspect related to each technology, and a third to manage the RA reception. Next it will be explained the flow diagram of each thread. Notice that a cellular thread exits, but will not be used in this thesis.

3.3.1 WAVE thread

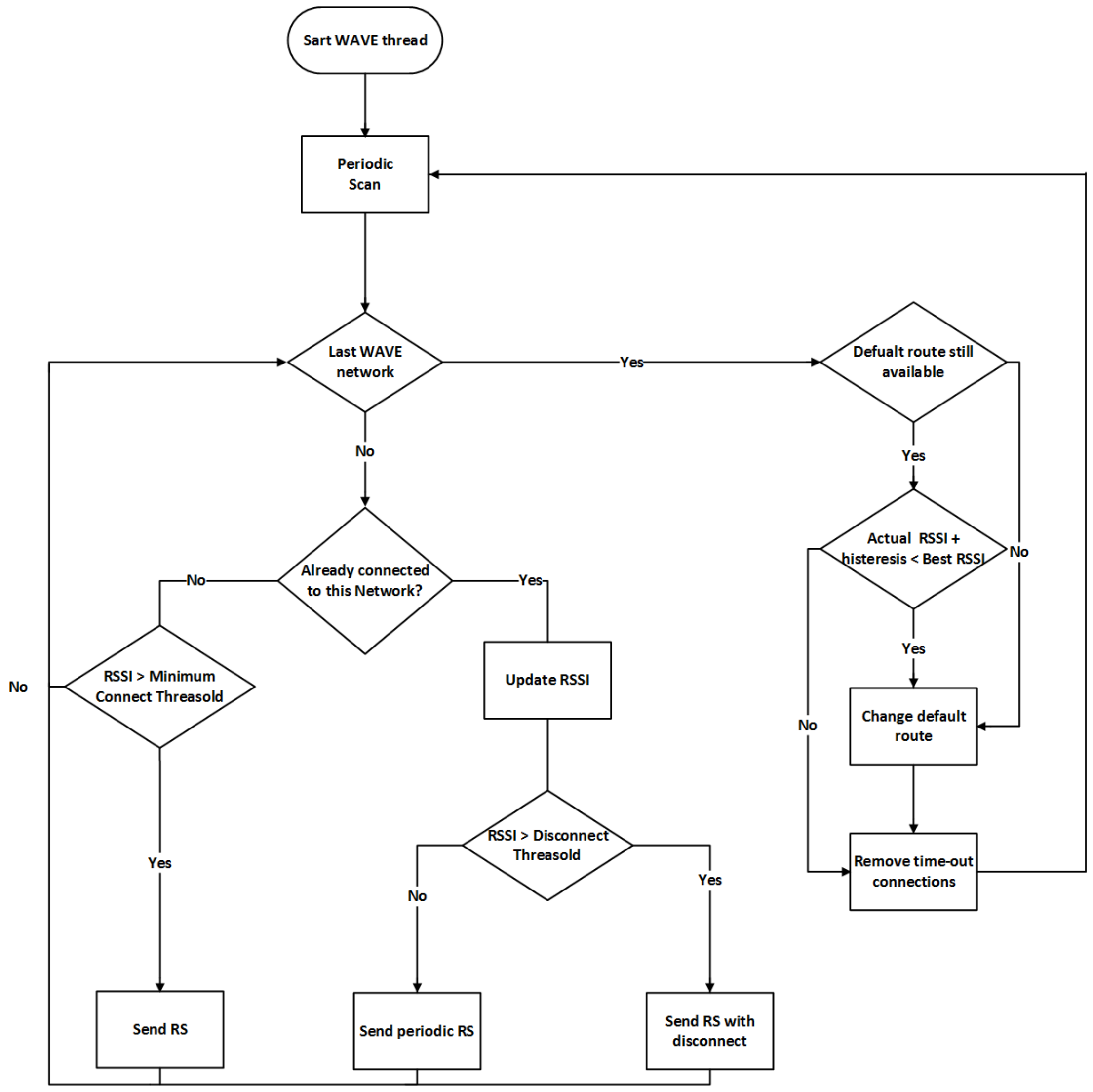

Sart WAVE thread
Periodic Scan
Last WAVE network
Yes
Defualt route still available
No
Yes
Actual RSSI + histeresis < Best RSSI
No
Already connected to this Network?
No
Yes
RSSI > Minimum Connect Threasold
No
Update RSSI
Yes
Change default route
No
RSSI > Disconnect Threasold
Remove time-out connections
Yes
No
Yes
Send RS
Send periodic RS
Send RS with disconnect

Figure 3.9: WAVE thread flow diagram based on [10]

Before explaining the WAVE thread flow diagram, some key aspects must be discussed.

First, the amount of scans performed is configurable, but due to the broadcast nature of the WAVE technology, by default, the value is very high. This means that the connection manager has a low response time to changes in WAVE networks. Another important fact is that, to define the uplink default route, the WAVE networks will always have priority compared with the Wi-Fi networks. This happens because the WAVE technology was designed for vehicular networks, and it will, in normal conditions, outperform the Wi-Fi technology in this type of environments.

To have a better view of the WAVE thread behavior, a flow diagram is presented in the figure 3.9. As can be seen, first it is performed a periodic scan in order to obtain all available WAVE networks, and then process them. First, it is checked by the connection manager if the mMAG is already connected to the network that is being processed. If the answer is yes, the RSSI value is compared with a defined threshold, and if the value is not good enough, a RS with a low RSSI value is sent to the MAG (disconnect message). Otherwise, it is sent a periodic RS in order to keep the connection with that network alive.

On the other hand, if the mMAG is not connected to the network, it is checked again the value of RSSI, and if the value is good enough, an RS is sent to the PoA in oder to start the registration process in the mobility protocol. Otherwise, the connection manager simply discards that network and moves on to the next on the list.

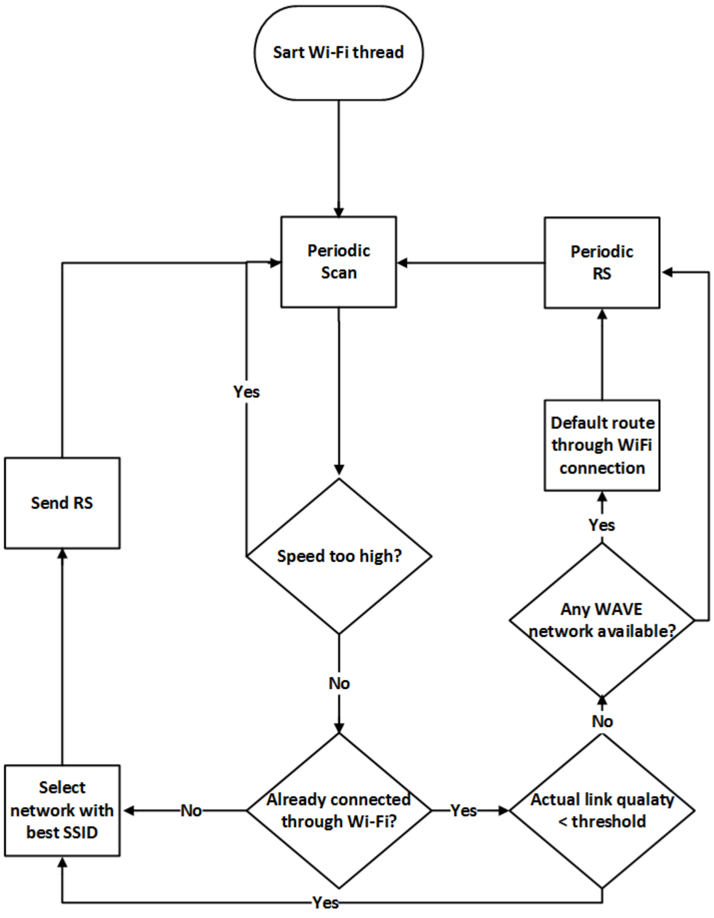
Lastly, when the connection manager processes the last WAVE network, it tries, if needed, to update the default route. First, it checks if the current default route is still available, and if the network used is not available anymore, the default route is changed to the network with the best RSSI. If the default route is still available, it is checked if it is worthy changing the default route to the best WAVE available network. This is done by comparing the RSSI of the actual default route plus an hysteresis with the RSSI of the best WAVE network. The hysteresis is used to avoid constant jumps between networks, because the RSSI value is in constant change in vehicular networks.

Finally, it is measured the last time that the WAVE PoAs sent a RA, and if the time exceeds the expiration time, the networks will be deleted from the current connections database.

3.3.2 Wi-Fi thread

Now it is time to understand the logic behind the Wi-Fi thread, but first it is important to explain some points. First, the value of periodic scan is, like in the WAVE thread, configurable, but now this value will be much lower. The reason for that is, as it will be concluded later in this document, the scans have a major impact in the Wi-Fi performance. Another factor that must be taken into account when using Wi-Fi connections in vehicular networks is the speed of the vehicle. Due to Wi-Fi limited range and long association time, when a vehicle travels with a high speed, it does not make sense to try to connect to this type of network.

Looking into the flow diagram (figure 3.10), the first action performed is a scan to find all available networks, then it is checked if the car is going too fast, and if it is the case, all networks are discarded. If the car is going with a relatively low speed, it is checked if the mMAG already has one Wi-Fi connection, and if the answer is negative, the connection manager selects the Wi-Fi network with better link quality and sends a RS in order to initiate the association process. If on the other hand, the mMAG is already connected to one Wi-Fi network, and because only one Wi-Fi connection is possible without the use of virtual interface, it is checked if the value of the link quality is good enough, and if it is, the connection is kept. Otherwise, if the connection has a bad signal quality, it is selected the network with best signal



Sart Wi-Fi thread
Periodic Scan
Periodic RS
Yes
Default route through WiFi connection
Send RS
Yes
Speed too high?
Any WAVE network available?
No
No
Select network with best SSID
No
Already connected through Wi-Fi?
Yes
Actual link qualaty < threshold
Yes

Figure 3.10: Wi-Fi thread flow diagram based on [10]

and a RS is sent.

If the Wi-Fi connection is a stable one, the connection manager will now manage the uplink default route. In resume, if there is any WAVE connection available, the uplink route should be through that network. However, if there is only the current Wi-Fi connection, the default route should be trough the Wi-Fi network, naturally. Finally, a periodic RS is sent to keep the connection alive.

3.3.2.1 RA handler thread

Finally, it is time to explain the last piece of the puzzle, the RS handler thread. This thread has a vital importance, because it is responsible to treat the RAs sent by the MAG and perform all interface configurations to allow the connectivity to the PoAs. Next, it will be explained the working flow presented in figure 3.11.

When the OBU receives an RA, before processing the packet, first it checks if it is the destination, and if it is not, the packet is discarded. If the destination is correct, the connection manager will parse the RA. It first retrieves the prefix in order to build an IPv6 address, and then uses the MAC of the interface to build the EUI-64. These addresses are then used to configure the interfaces in order to turn possible the connectivity between the PoAs and the mMAG. Now that the connection is established it is time to check if it is all good with the

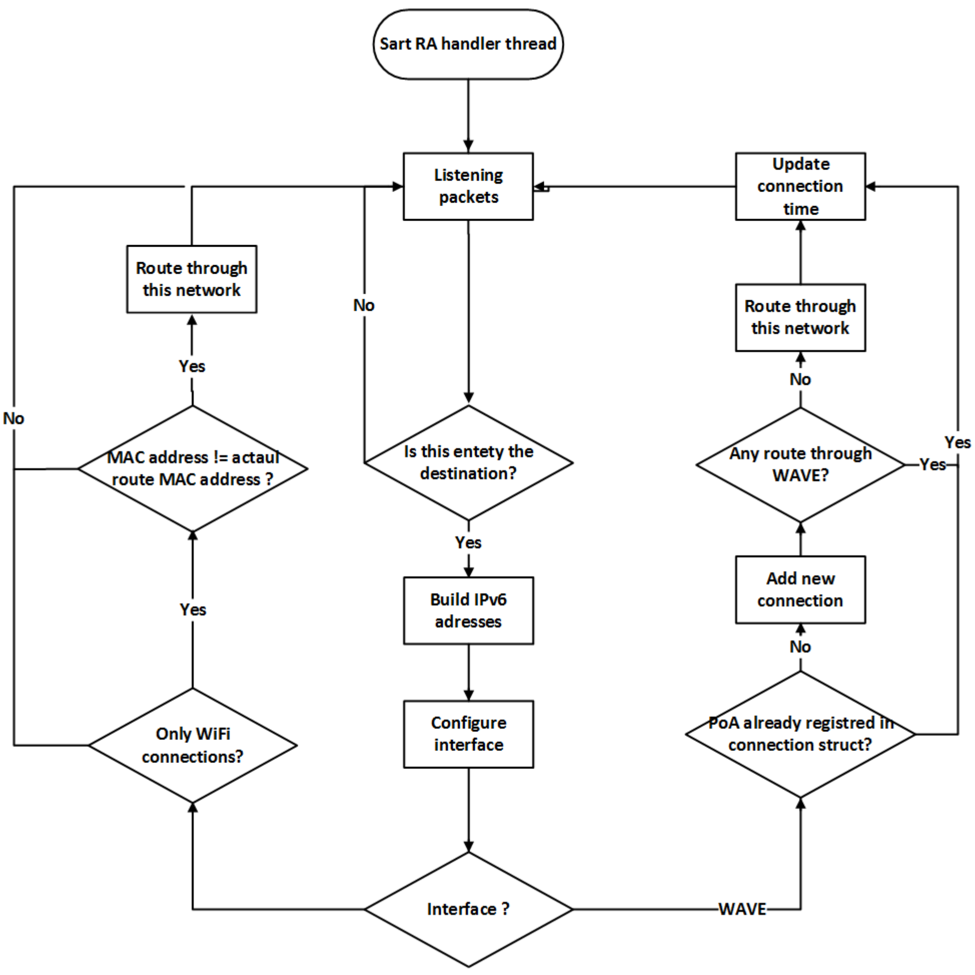


Figure 3.11: RA handler thread flow diagram based on [10]

uplink default route. First, it is checked in which interface the packet was received (WAVE or Wi-Fi), matching the destination MAC address of the packet with the MAC address of the interfaces. If it was received in the WAVE interface, and if it is a new WAVE connection, it is added in the connection manager database, and is checked if there is any uplink default route. If there is not, it is added through the WAVE interface that received the RA. On the other hand, if the connection is already in the database, it is updated the time since the connection received a RS, avoiding being deleted due to time-out.

In case that the RA is received in the Wi-Fi interface, the only concern now is to update a possible uplink default route due to a Wi-Fi handover. To do so, first it is checked if there are only Wi-Fi connections, and if the answer is yes, next the current MAC address is compared with the MAC address of the uplink default route. If they are different, it means that an handover between Wi-Fi interfaces has been performed and the current uplink default route is outdated, so the next logical step is to update the route to the new Wi-Fi connection.

3.4 Network Coding Operation

One of the goals of this master thesis was to integrate the network coding in the multihomed vehicular network. To do so, it was used as base a previous developed program called *Gekko*, developed by Nuno Coutinho in his Phd thesis [56]. The Gekko uses a SNC approach that was proposed by Andre Rodrigues [52] during his master thesis dissertation, in the University of Porto.

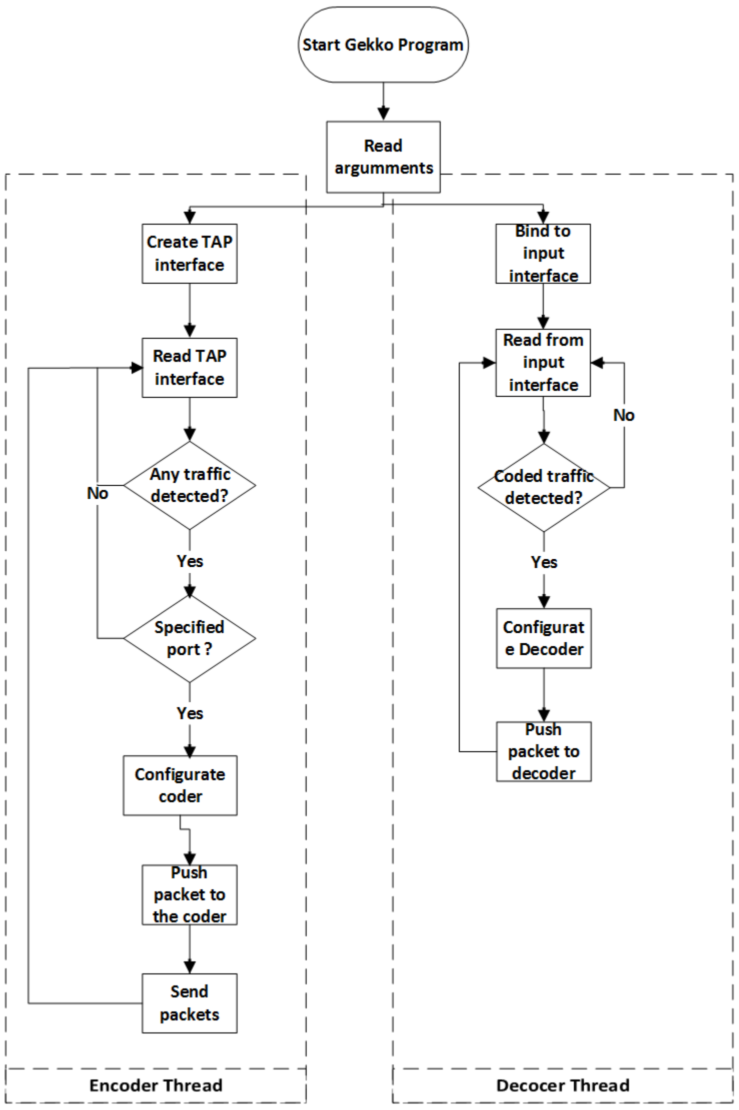


Figure 3.12: Gekko flow diagram

This section will explain in detail how the original Gekko program works. First, Gekko is the name of the top level program that controls both the encoding and decoding processes. In resume, this program launches two threads, the encoder and decoder, meaning that a node can act as encoder and decoder, at the same time. In oder for a node to knows its role, some configurations must be done, that will be explained later.

In the next subsections, it will be explained the behavior of the encoder and decoder threads.

3.4.1 Encoding thread

In the figure 3.12 it is shown the Gekko global flow diagram. As can be seen, the input program arguments are parsed and then stored to be used in the encoder and decoder threads.

In the encoder thread, the first step is to configure the TAP interface, in which all the incoming packets will be read from. Now, in order for the program go further, the TAP interface must receive UDP traffic with a specific destination port (3005 by default). If this happens, the packet will be encoded and then sent with a static MAC address through an interface passed as argument to the program, in a layer two socket. Before the packet is sent, it is added an extra header between the Ethernet header and the coded payload, with useful information, such as the generation number, the packet ID, the seed used, the number of M coded and N original packets in each generation, and finally the l size of the original packets without the Ethernet header (size of the encoded payload). This information is vital to the decoding process, as it will be explained later in this section. As can be deduced from what was said, all the original packet layer 3 is encoded.

3.4.2 Decoding thread

In the decoder thread, the first action is to perform a bind to the output interface passed as argument to the program, and then the thread will wait for the coded packets to arrive. A coded packet can be detected by analyzing the Ethernet header type. All the packets that went through the encoding process, even the N normal packets (packets that are not coded) have a Ethernet type equal to *0XCODE*. If indeed, the traffic that is being sent is from the coder, first the decoder will be configured, using the information inside the network coding header (N, M, Seed, payload length), the program will initialize arrays, allocate the necessary memory and initialize multiples variables. Next the packet will be decoded, and the thread will process the next incoming packet.

3.4.3 Encoding process

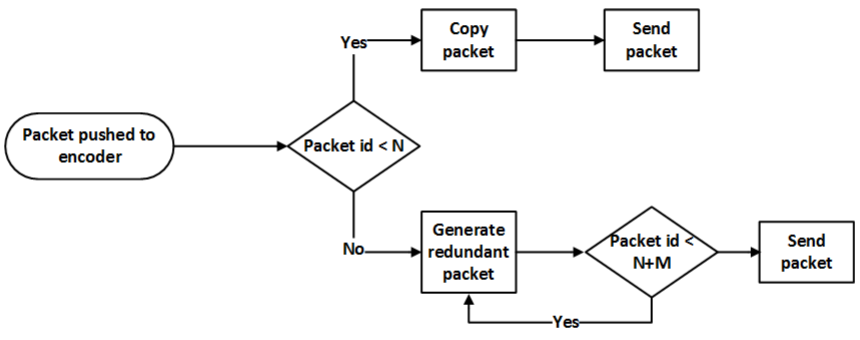


Figure 3.13: Encoding process

When, in the previous subsection it is referred that the packet is pushed to the encoder, some actions will be performed. In the algorithm 3.13 it is described how the encoding process is performed. All packets that are pushed to the encoder include an identification

(ID) number, that identifies the packets inside the generation. While that ID is lower than the number of N normal packets, the encoder saves a copy of the packet and forwards the uncoded packet. On the other hand, if the ID is equal to N, it means that all the normal packets were processed and forwarded to the next node, and it is time to generate the encoded packets and send these packets to the decoder. In order to generate a redundant packet, it is used pseudo random values and the original packets. The pseudo random values will be generated through a Pseudo Random Number Generator (PRNG) that generates random values depending on a seed (values generated with the same seed will be equal). To generate a redundant packet, it will be generated N pseudo random values, called coding coefficients, that will be used to multiply by the N normal packets previously stored. Finally, the results obtained in the previous multiplication are added (XOR), and it is obtained the coded packet payload, illustrated in the figure 3.14. To enable the decoder to successfully decode the coded payload, an extra header is introduced between the layer 2 and 3, that carries vital information about that packet that is being sent, such as the generation size (N+M), the number of coded packets (M), the packet ID inside the generation and the seed used to generate the coding coefficients.

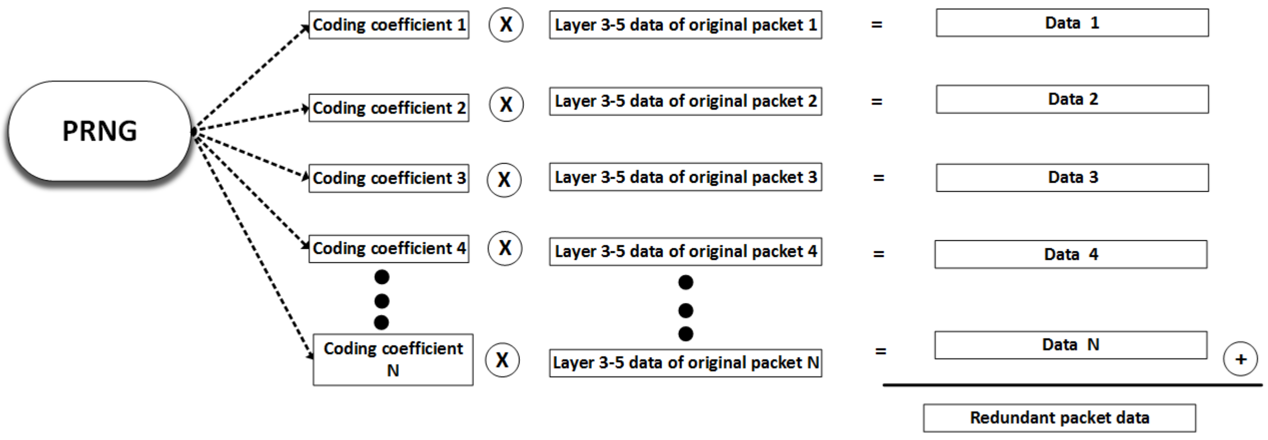

PRNG
Coding coefficient 1
Coding coefficient 2
Coding coefficient 3
Coding coefficient 4
Coding coefficient N
X
Layer 3-5 data of original packet 1
Layer 3-5 data of original packet 2
Layer 3-5 data of original packet 3
Layer 3-5 data of original packet 4
Layer 3-5 data of original packet N
=
Data 1
Data 2
Data 3
Data 4
Data N
+
Redundant packet data

Figure 3.14: Generation of redundant packets

3.4.4 Decoding process

When a coded packet is pushed to the decoder, first, it is checked if it belongs to the generation that is being processed, and if the answer is yes, the packet is saved and the matrix is updated. Otherwise, if the packet received is from the next generation, it means that all packets from the previous generation have been sent, and it is time to decode and send the decoded packets to the final destination.

In order to decode the coded information, it was created a decoder matrix, responsible for saving crucial information, shown in figure 3.16b, and its updating process differs depending on the type of traffic that is being processed. If it is a normal packet that is being processed, the update consists in accessing the line and column equals to the packet ID number and put the value 1 (identity matrix) in the coding and decoding coefficients sub matrix.

If the packet, on the other side, is redundant, the update is totally different. Now the matrix will be filled with the coding coefficient values used in the encoding process. The best way to understand this procedure is with an example. Let us suppose that the redundant

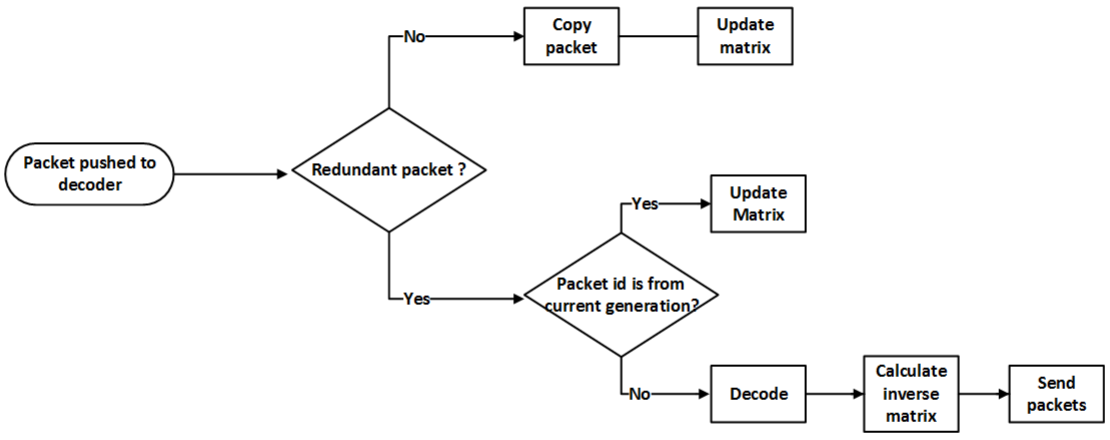
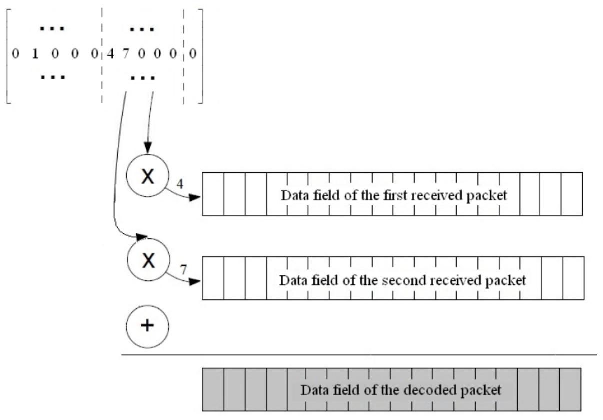


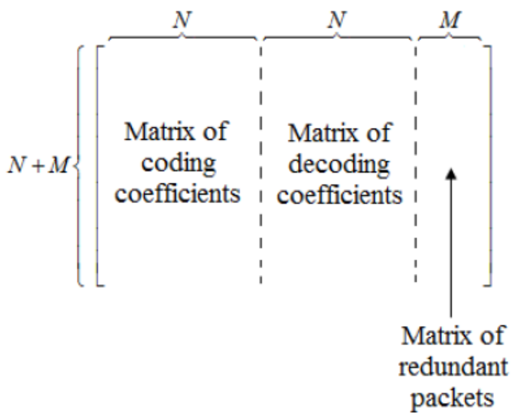
Figure 3.15: Decoding process

packet with the ID equal to 12 arrives at the decoder. In line 12, in the column 0 it is set the coding coefficient used, in the coding process, to multiply the packet with ID equal to 0 (normal packet), in line 12 and in the column 1 is set the coding coefficient used to multiply the packet with the ID equal to 1, and so on. The last sub matrix, the "Matrix of redundant packets", serves to identify the number of the redundant packets. Using the previous example, if the packet with ID 12 arrives at the decoder, and supposing a number of normal packets (N) of 10, in line 12 and in the second column, it will be inserted the value 1.

Then, to obtain the values of the "Matrix of decoding coefficients" it is only needed to calculate the inverse matrix of the "Matrix of encoding coefficients". Finally, the decoder has all the information needed to decode a coded packet and obtain a original packet. This procedure is illustrated in the figure 3.16a. To obtain the packet from the line that has numbers, the decoder will multiply (in Galois Field) the decoder coefficients by the data of the original packets, and then will add (also in Galois Field) the previous results. Following the example of the image 3.16a, the packet is obtained, first, by multiplying the data of the first original packet by 4, then multiplying the data of the second original packet by 7, and finally adding the previous results together. It is clear to see that the aim of those operations are, as it would be expected, to reverse all the operation performed in the encoder side.


0 1 0 0 0 4 7 0 0 0 0
X
4
Data field of the first received packet
X
7
Data field of the second received packet
+
Data field of the decoded packet

(a) Decoding codded packet [52]


N
N
M
N+M
Matrix of coding coefficients
Matrix of decoding coefficients
Matrix of redundant packets

(b) Decoder matrix [52]

3.4.5 Delay recovering mechanism

After analyzing of the decoding process behavior, it was noticed that the solution developed was vulnerable to out-of-order packets. This a problem when a packet from the next generation arrives at the decoder before all the packets from the previous generation (out of order packets). Taking into account the decoding process described in the figure 3.15, it can be concluded that in this type of situations, the information from the previous generation is lost, increasing the packet losses in the network, unnecessarily.

This vulnerability is a major setback, in vehicular networks with multihoming capabilities. For example, in downlink scenario, the flow is divided in the LMA and sent to the MAG, that uses different technologies and has different link qualities. This aspect increases significantly the amount of out-of-orders packets, reducing the efficiency of this approach.

In order to solve this limitation, Andre Rodrigues [52] found a solution to minimize this problem. Instead of only using one matrix, it was designed a sliding window, that is an array

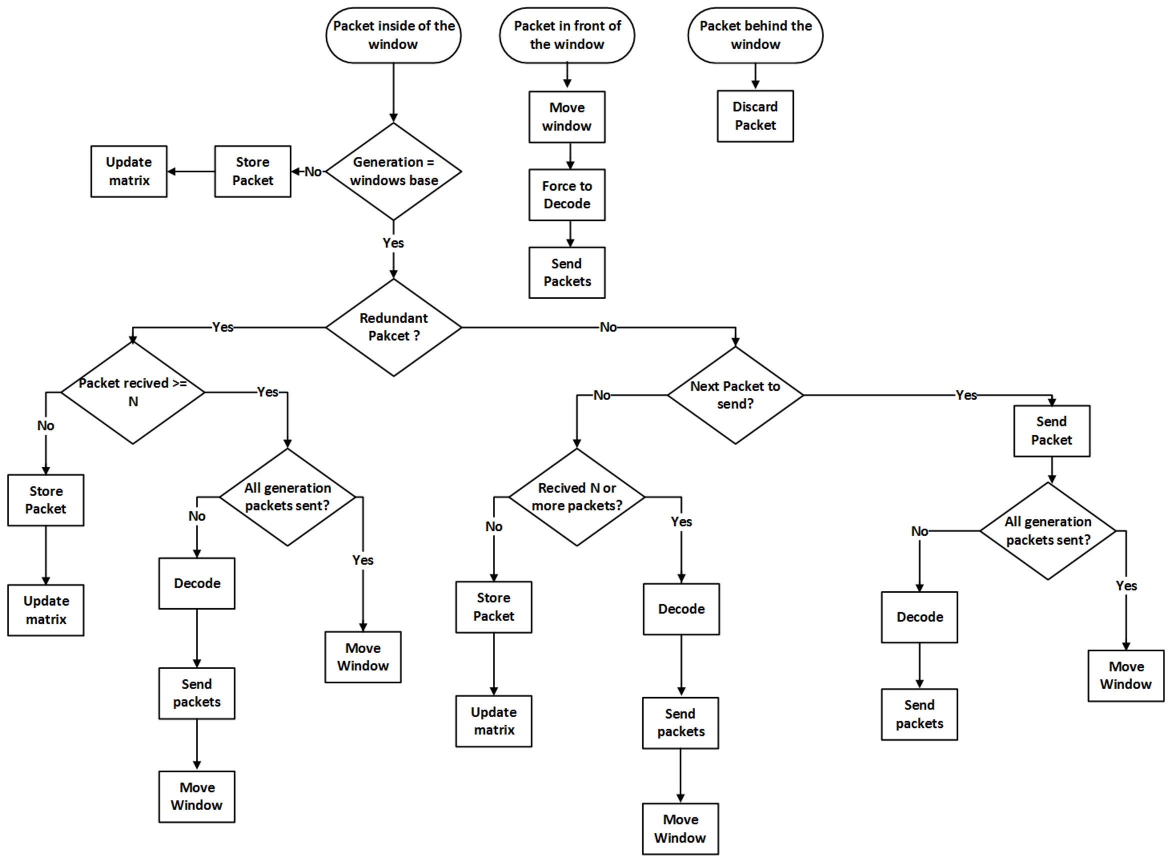


Figure 3.17: Out of order recover mechanism

where each position contains one decoder matrix belonging to a certain generation. This way, if a packet arrives out of order and the decoder matrix is saved in the "sliding window", the packet that would be previously discarded, is now processed, decoded and forwarded. For example, if the decoder is configured with a window size of five, this means that a packet can arrive with a delay of five generations that will still be processed.

To understand better the logic behind this mechanism, it is presented the flow diagram

3.17. When a packet arrives at the decoder, the generation ID is checked and, according to its position relatively to the window, some actions are taken. If the packet is behind the window it will be simply discarded. On the other side, if the packet generation is in front of the window, the packet will be stored, the matrix updated and the window will be moved forward, sending all packets from the generation that would stay behind the window and consequently discarded.

Last, if the packet is inside the window and if the base of the window is different from the generation ID, the packet is stored and the matrix is updated. If, on the other hand, the generation ID is equal to the window base, depending on which type of packet (normal or redundant) is being processed, the logic will be different. If it is a redundant packet (packet ID > N) and the number of packets received is lower than N, the packet is stored and the matrix updated, since to recover from missed packets, it is needed at least N packets from the M+N total generation size. On the other hand, if the packet is redundant but the number of received packets is larger than N, it is checked if all normal packets (N) have already been sent. This is important in order to avoid decoding unnecessary packets, because if all N packets have already been sent, there is no need to decode an encoded packet, it is only needed to forward the window, in order to handle the next generation. If on the other side, the number of packets sent is lower than N, this means that it is needed to decode redundant packets in order to obtain the missing packets, and then send to the decoder and finally move the window. If the packet received is a normal one, and if it is the next to be sent, it will be forwarded. If it is the last of its generation, the window will move forward. On the other side, if the packet arrived out-of-order, if the decoder already received N or more packets (coded and uncoded), it will decode the needed packets and send, ordered, to the next hop. If there are not yet enough packets to enforce the decoding process, the packet is stored and the matrix updated.

3.5 Chapter Considerations

This section will present a resume of the main functionalities of the N-PMIPv6, the connection manager and the network coding that were already available in the start of this dissertation. With respect to the mobility protocol, these are the available functionalities:

- Support full network mobility for the vehicles and its users;
- Integrate multihoming in the mobility protocol;
- Support the connection to multiple PoAs using the same (WAVE) interface;
- Support connection to a single Wi-Fi network and to a cellular BS;
- Provide IPv4 Internet to the users inside the car;
- Route downlink traffic through all available resources;
- Multihoming rule that maximizes the network performance.

Regarding the connection manager, these were the key features:

- Connect to multiple networks with different technologies at the same time;
- Connect to a network based on its RSSI;

- Configuration of the interfaces and routes;
- Uplink only supports single technology.

Finally, regarding the network coding base used, these are the available features:

- Encode and Decode using the SNC approach;
- Routing in layer 2 based on static MAC addresses.

In resume, regarding the mobility and the connection manager, it was found a stable mobility protocol that takes the most advantage possible from the multihoming functionality in downlink. It is possible to conclude also that the main scope of the previous work was in downlink management. On the other side, the network coding available contains the base implementation of the encoder and decoder, that will be extended for the multihomed vehicular network.

Chapter 4

Uplink manager and N-PMIPv6 improvements

4.1 Introduction

If in the past the majority of the traffic was sent in downlink, nowadays the balance between downlink and uplink is almost even. Today, almost everyone carries a smart phone capable of posting photos or movies on the Internet or upload important files to the cloud. Having that in mind, it is important to manage properly the available resources in order to provide the best service according to the user needs.

Today there are available wireless networks everywhere, and this number is increasing every day, so it is important to create a good multihoming manager capable of selecting the best available networks, and route the traffic through them.

The previous network mobility protocol, as referred in the previous chapter, only performs multihoming in downlink traffic. In uplink, the traffic is sent through a single network, only taking into account the RSSI of the available networks. Therefore, to achieve a good uplink manager capable of dividing the traffic taking into account its type, available networks and the achieved throughput of the PoAs, it is required to add the following features to N-PMIPv6 protocol and to the connection manager :

- **Differentiate Traffic :** To give priority to a certain type of traffic, first it must be differentiated.
- **Perform uplink with multihoming :** Give the vehicular network the capability to route uplink traffic through Wi-Fi and WAVE technologies simultaneously(or other existent technology).
- **Route traffic according to PoA achieved throughput :** Take into account the achieved throughput of all PoAs available and manage the routes in order to increase the network performance.
- **Multi-hop support :** Perform all the previous features, but in a multi-hoop scenario.
- **Uplink connection manager :** Implement an uplink connection manager responsible for manage all the uplink routes.

The work performed also improves several aspects and adds extra functionalities to the previous N-PMIPv6 implementation:

- **Disconnect message :** To increase the network performance, when a connection has a bad signal, it is better for the multihoming framework to discard this connection and use the others available. To do so, the LMA must be informed which connections are having low quality, and send this information to the multihoming entities responsible for dividing the traffic through the remaining MAGs. This is also performed in the multi-hop case, where if the single-hop mMAG that is serving a multi-hop mMAG sends a disconnect message, both connections must be removed from the multihoming decision process.

- **Reconnect message :** It allows the mMAG that sends a disconnect message to the LMA to regain the connectivity as soon as the signal becomes good again.

- **Multi-hop flag :** Flag that informs the LMA which nodes, that are trying to join the network, are in multi-hop.

- **Optimize multihoming rule :** Analyze and send to the LMA the amount of throughput available in the WAVE interface of the single-hop mMAG, in order for the LMA to take into account this load when performing the calculation of the percentage of traffic allocated to each available MAG.

This chapter presents the challenges and technical problems during the implementation of all the functionalities, and the solutions found to solve those same problems.

Section 4.2 will explain the stages required to design the modules that build the uplink connection manager.

Section 4.3 will describe how the connection manager behaves.

Section 4.4 will detail the improvements done in the N-PMIPv6 and in the global vehicular network.

Finally, section 4.5 will resume the work discussed in this chapter, and conclude about the main challenges presented in those implementations.

4.2 Uplink Manager

Figure 4.1 illustrates one of the scenarios for the uplink multihoming: a mMAG (OBU in the vehicle) disseminates an IPv4 network that is used by a user that connects to that network to access the Internet, and by a data collection unit that aggregates information from sensors that aim to send the collected data to the cloud. Having this scenario as an example, the aim of this work is to be able to send this traffic through different networks simultaneously, e.g. the Internet access through a low-delay network and the sensors information through a lower-quality network in vehicular environments.

Knowing that the WAVE technology has been developed specifically for vehicular environments, and the fact that, when the mMAG provides the IPv4 network to the user, it must use a virtual Wi-Fi interface that reduces drastically the available bandwidth in Wi-Fi networks, the WAVE technology will be used to route the users traffic (most priority), and Wi-Fi will be used to route the sensors traffic (least priority). This differentiation rule is just an example to show the connection manager behavior.

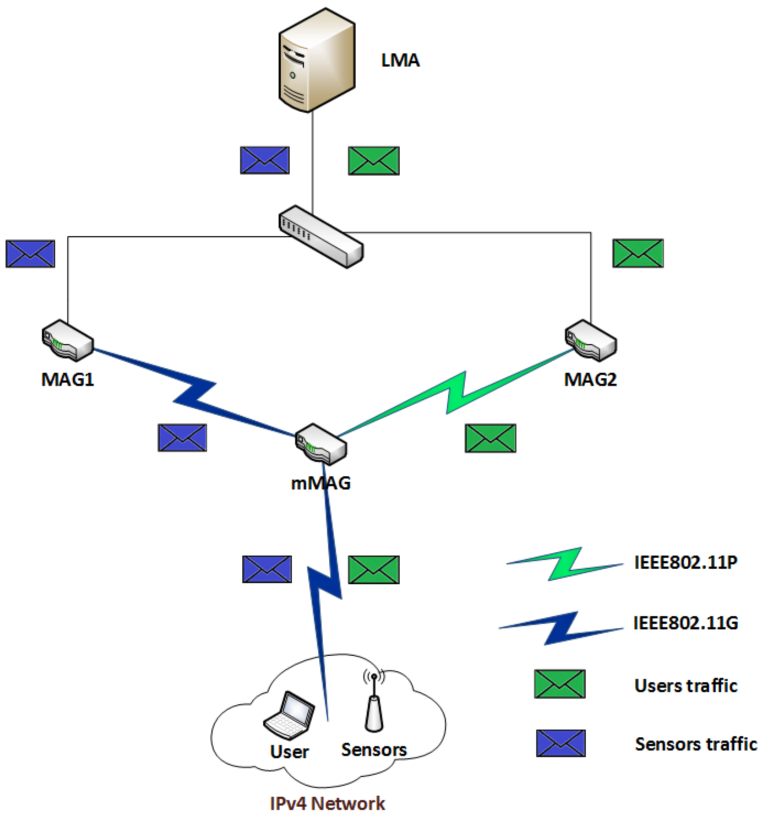


Figure 4.1: Uplink Management Challenge

4.2.1 Differentiate traffic

The first step required in the support of multihoming capabilities is to differentiate the traffic, and then be able to route it through the available technologies at the same time. The traffic is then differentiated by the destination port of the packets that arrive at the mMAG. Using the above example, the sensors traffic will be marked using *iptables* (based on the destination port) and then routed to the IPv4-over-IPv6 Wi-Fi tunnel using *iprules*. On the other hand, to route the users' traffic to the IPv4-over-IPv6 WAVE tunnel, it will be used a default route. Then, it is used an *iprule* that forces all the traffic that traverses the tunnels to be forwarded to the IPv6 addresses of the next-hop networks. One important aspect to keep in mind is the fact that the sensors are special nodes spread through the city, that build their own packets and that can adopt a pre-defined port value.

The required *iptables* configurations:

- *iptables -A FORWARD -i <VirtualInterfaceName> -o ip4tnl<USERID> <TECH> –dport <PORT> -n conntrack –ctstate NEW -j ACCEPT*
- *iptables -A FORWARD -i <VirtualInterfaceName> -o ip4tnl<USERID> <TECH> -m conntrack –cstate ESTABLISHED,RELATED -j ACCEPT*
- *iptables -A FORWARD -i ip4tnl<USERID> <TECH> -o <VirtualInterfaceName> -m conntrack –cstate ESTABLISHED,RELAETD -j ACCEPT*

- *iptables -t mangle -I PREROUTING –dport <PORT> -j MARK –set-mark <MARK_-NUMBER>*

These four commands allow the traffic that enters in the virtual interface to be forwarded to the ip4tnl and vice-versa. Moreover, all traffic that is received in the mMAG virtual interface with destination port of *<PORT>* is marked to be forwarded through a specific route, that will be detailed in the subsection 4.2.2.

4.2.2 Routing

- **From users to mMAG :** In order for the users to have a connection to the vehicular network, the mMAG must share a Wi-Fi network. In order to do so, it was used the Hostapd daemon [57]. Next it is configured a DHCP server, that is responsible for providing valid IP addresses and a default route to all users connected to the Wi-Fi network.

  Another key aspect of this implementation is that, since the goal here is to perform multihoming, the mMAG is connected to one Wi-Fi MAG and WAVE MAG. This means that the mMAG already has one Wi-Fi connection when it tries to announce the Wi-Fi IPv4 network in which users will connect. This aspect implies the creation of a virtual interface to multiplex the use of the Wi-Fi interface. To do so, the following commands were used:

  - *iw phy phy0 interface add <VirtualInterfaceName> type managed*
  - *ip link set <VirtualInterfaceName> address 00:00:00:00:0X:X up*
  - *ip a a <IpAddress> dev <VirtualInterfaceName>*
  - *hostapd /etc/hostapd.conf_g*
  - *udhcpd /etc/udhcpd.conf_g*

  These commands create a virtual interface called *<VirtualInterfaceName>* with a MAC address equal to *00:00:00:00:0:X*, where X is equal to the last two digits of the mMAG ID and with an IP address that is in the same network of the one that was given by the Hostapd and UDHCPD to the users that connect to the IPv4 network. Lastly, the final two commands only run the Hostapd and UDHCPD daemons and their configuration files.

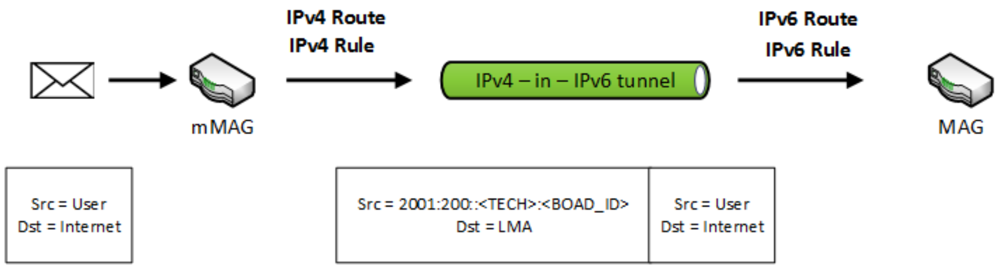


Figure 4.2: mMAG to MAG packets

- **From mMAG to MAG :** After the traffic is marked, it was needed to add an IP rule and a IP route in order to force all the traffic marked to be sent to the *ip4tnl<USERID>G*, and it will be routed through the Wi-Fi network. All the other traffic will be considered user's traffic and will be forced to *ip4tnl<USERID>P* through a default route.

  As figure 4.2 shows, when the traffic is routed to the IPv4-over-IPv6 tunnel, an extra header is added to packets, with the source IP *2001:<USER_ID>::<TECH>:<BOARD_-ID* address and with LMA IP address as destination address. Now, it is added an IPv6 rule that forces all the traffic with the source address mentioned above to be routed to the link local of the next hop MAG interfaces. Because both mMAG ID and user ID are the same, it will be the *TECH* field that indicates if the traffic will be routed to the next hop MAG WAVE or Wi-Fi interface link local.

  Please note that the IPv4 over IPv6 tunnel is built between the mMAG and the LMA, and not between the mMAG and the MAG. Figure 4.2 shows that, first it is needed an IP rule and a IP route, to force the IPv4 traffic to be routed through the tunnel, and then, when the extra headers are added, it is required IPv6 rules and routes to forward the packet to the next hop MAG.

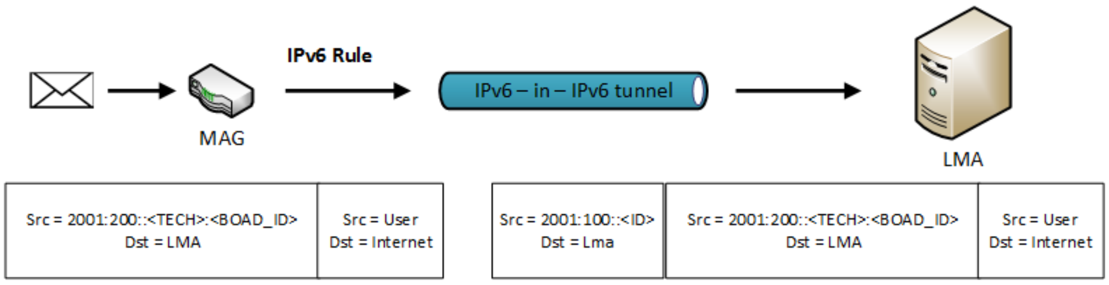


Figure 4.3: MAG to LMA packets

- **From MAG to LMA :** For the traffic to arrive in the LMA, it must first be routed through the MAGs. As explained in the chapter 2, the traffic is encapsulated in IPv6 tunnel and forwarded to the LMA. This process is illustrated in the figure 4.3.

- **LMA IPv4-in-IPv6 tunnel creation :** Last, it is noticed that the process to build the IPv4-in-IPv6 tunnels in the LMA needs improvements, because the LMA does not know the ID of the nodes that are trying to join the network (problem described in section 3.2.2.3). To make this process faster and less "hard coded", it was decided to use the RS and the PBU message to carry the ID of the node that wants to connect to the network.

  After analyzing the RS message format (4.12), it was decided to use the reserved field, more specifically the last 16 bits, since the first ones are used for another purpose, that will be explained later. Finally, it was analyzed the PBU message format and it was decided to use the last 8 bits from the reserved field, as shown in figure 4.13. Now that the LMA knows the ID of the node that wants to connect to the network, when the first PBU is received, in the registration process, the value of mMAG ID is stored in the BCE and UCE and then used to create the tunnel. To do so, a python program was created that uses the information stored in the BCE, to create the IPv4 over IPv6 tunnels dynamically.

4.2.3 Uplink traffic management based on networks achieved throughput

To inform the mMAG of the load of each available connection, the first question that emerged was which interface to analyze to retrieve the value of the MAG actual load, the ingress or egress interface.

- **MAG egress interface :** This is the interface *eth0* that connects the MAG to the LMA. The main advantage of considering this interface is that it captures all the traffic being processed by the MAG since all the packets meant to be forwarded to the LMA, through the *eth0* interface. The only disadvantage is that this approach means a change in the multihoming architecture, because in the past work it was analyzed the ingress interface.

- **MAG ingress interface :** This is the Wi-Fi or WAVE interface depending if it is a WAVE MAG or a Wi-Fi MAG. In theory, this approach also captures all the traffic because it can be assumed that all traffic traverses these interfaces, either the downlink or the uplink traffic. The major disadvantage of this approach is that, when using WAVE, as was noticed in [10], when there are multiple WAVE networks that are in the range of each other, and because WAVE is a broadcast technology, the output of one MAG becomes the input of the other MAG. If all the MAGs are in the range of each other, the bandwidth is limited to one MAG rate, and the objective is to obtain a value of the MAG load.

The decision taken has been to change the previous multihoming approach and analyze the egress interface. To create a smart uplink management mechanism, the OBU (in the mMAG) needs to know the achieved throughput of each RSU. The only way for the OBU in the mMAG to know that value is by receiving this information from the RSUs (in the MAGs). We developed an algorithm that analyses the incoming interface in real-time, and calculates the average of the input and the output traffic that is going through the RSU. Then, the equation presented in expression (4.1) is used to determine the respective achieved throughput (AT). Here, C is the maximum achieved throughput that a RSU in the MAG can achieve without any traffic, the Ps is the amount of packets per second (input and output), and the S is the mean packet size.

$$AT = C - (Ps_{in} * S_{in} + Ps_{out} * S_{out}) \tag{4.1}$$

Now, the information must be sent to the OBU and, in order to keep a low overhead, it is used the Router Advertisement (RA) message already present in the N-PMIPv6 protocol. However, there is no available field with enough space to carry the AT value. Therefore, the AT value is converted in a value between 0 and 31 and is sent in the 5 reserved bits of the RA message. To obtain this value, it is used expression (4.2), where the Max is the maximum values that can be achieved with 5 bits (31) and the $MaxAT$ is the maximum value of the AT of all RSUs.

$$RSU_{AT} = \frac{AT * Max}{MaxAT} \tag{4.2}$$

The metric that is sent in the RA message contains a value that establishes a relationship between all the RSUs ATs, enabling the connection manager to route the uplink traffic according to that information.

4.2.3.1 Uplink traffic impact on the achieved throughput

The OBU has the information about each RSU AT, and it aims to determine the best amount of traffic to send in each technology and network. However, when the OBU sends traffic to the RSU, the AT value of the respective RSU will be affected, i.e, the AT value that the OBU receives is affected by the amount of traffic that it sends in the uplink. This makes the AT comparison unfair: analysis of values that already include the impact of the traffic that will be moved, and values that do not include this impact. To design a truly efficient and fair load-balancing mechanism, the AT value used in the decision process must be processed at the connection manager, in order for the OBU to use a value that is not affected by its own uplink traffic.

The solution proposed is to calculate the amount of traffic that the OBU is sending in the uplink (OBU^{load}), and add this value to the AT value that corresponds to the connection that is being used to route the uplink traffic, expression 4.3. In this way, the value used in the decision process will not be affected by the uplink traffic introduced by the OBU under analysis. Finally, the connection manager will search for the highest AT' value associated to each available WAVE connection, and will configure an *iprule* to route all the WAVE traffic through that network, as explained in subsection (4.2.1). For the connections that are not being used by the uplink traffic, AT' becomes equal to AT.

$$AT' = AT + OBU^{load} \tag{4.3}$$

4.2.4 Single-hop thread

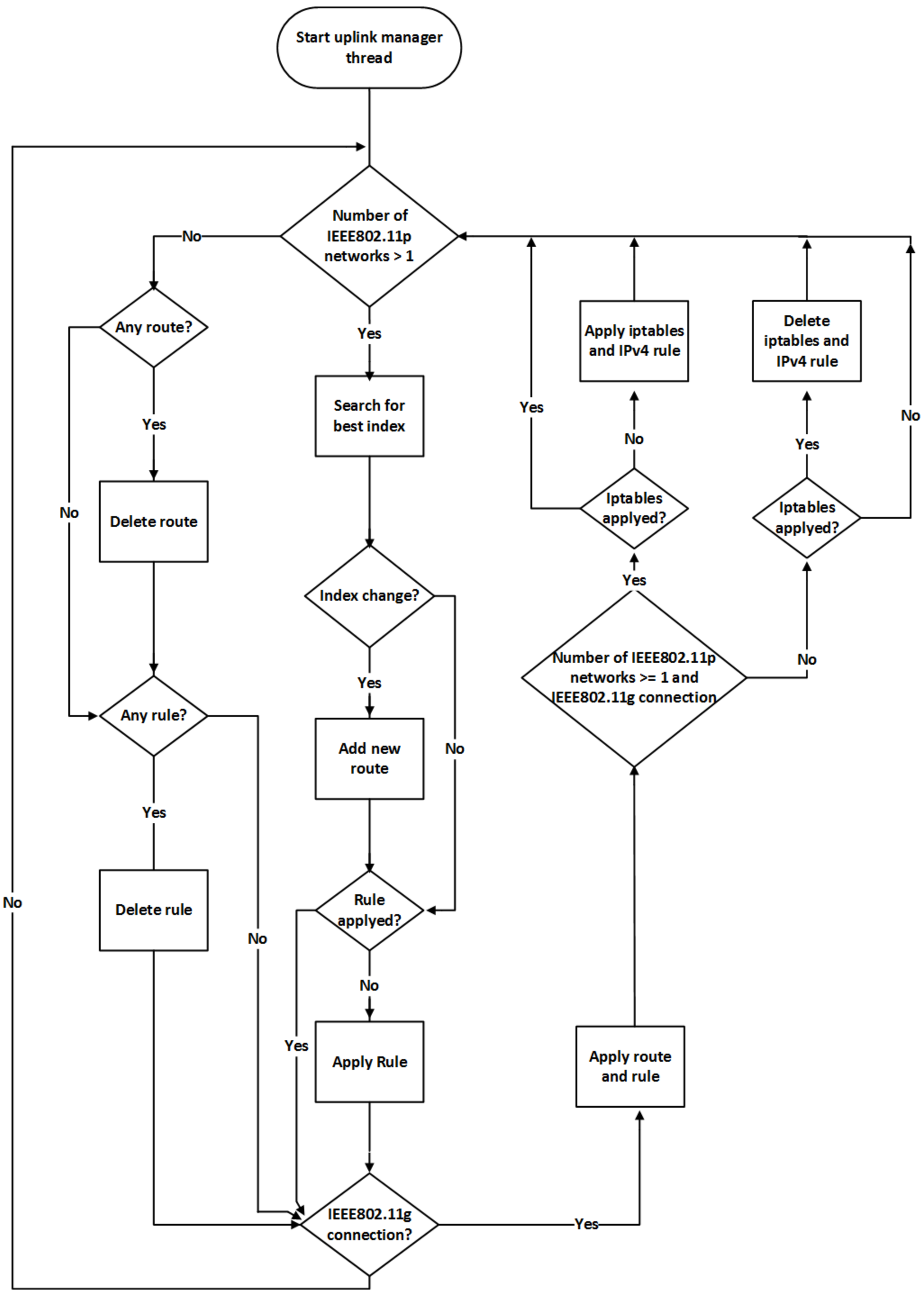


Figure 4.4: Single-hop thread

Now that all the multihoming objectives were fulfil, it was created a thread that manages all uplink routing based on the information analyzed before. To make the process cleaner, it was created a set of functions:

- **void addIp6Route(unsigned char Mac, char tech) :** This function, through the use of system calls, creates an IPv6 route to the next hop interface based on the *MAC* input via interface WAVE or Wi-Fi depending on the *tech* value. *ip -6 route add default dev wlan<tech> via <linkLocal> table <tableNumber>*.

- **void delIp6Route(unsigned char Mac, char tech) :** This function does the same as the previous function, but in this case it deletes the route. *ip -6 route del default dev wlan<tech> via <linkLocal> table <tableNumber>*.
- **void addIp6Rule(char tech) :** This function adds a rule that forces all the traffic that goes throuth the ip4tnl<tech> tunnel to be fowarded through the correct interface. *ip -6 rule add from 2001:200::<tech>:<boardId> <tableId>*.
- **void delIp6Rule(char tech) :** This function does the same as the previous function, but in this case it deletes the rule. *ip -6 rule add from 2001:200::<tech>:<boardId> <tableId>*.
- **void apllyIptables(char command) :** Using system calls, it uses the commands stated in subsetion 4.2.1 and also the IPv4 rule that forces all the traffic from a specific port to be routed through the *ip4tnl<USERID>g*. The *command* input is used for adding, or cleaning the *iptables*.
- **void searchHigherAtWAVE(void) :** This function returns the WAVE connection with higher AT value.
- **void cleanData (void) :** This function removes all routes, rules and *iptables* previously added.
- **void readInterfaceTraffic(void) :** This function writes in global variables the value of *mean_packet_size* and *mean_packets_per_second* of the WAVE interface. It is used to calculate the amount of the traffic that the mMAG introduces in the access network.
- **int networkUpdates(void) :** This function is always checking, in the actual connection array, if there are any changes in the MAC addresses, and returns 1 when changes are present. In this way, this function detects if there are changes in the network topology.
- **int checkDefaultRoutes(void) :** This function returns 1 if the mMAG has an IPv4 default route.

Before starting to explain the uplink manager logic, it is important to explain some key aspects. First, as stated in chapter 3, the IPv4overIPv6 tunnels were already created by the previous implementation of the N-PMIPv6 and the creation is performed in the mobility protocol, not in the connection manager. Another important aspect is relative to the connection manager. In Wi-Fi network, it is only possible to have one connection, so there is no need to search for the Wi-Fi network with highest AT value. Finally, the function *searchHigherAtWAVE* takes into account the load that the traffic that goes through the WAVE access networks aims to introduce in these networks. This is an important feature, because it avoids constants jumps between routes.

In the figure 4.4 it is illustrated how a node behaves when in single-hop. First, it is checked if there are one or more WAVE connections, and if there are, it is called the function *searchHigherAtWAVE* to find the best WAVE network. Next, it is checked if the index of the best network is equal to the index of the actual WAVE route, and if the answer is no, it is deleted the previous route and added a route to the best network. Otherwise, if the index of the best WAVE network matches the actual index, it is checked if there is already a rule applied to force all traffic that goes through the ip4tnl<USERID>p to be routed to the best WAVE network. In case the number of WAVE networks reaches 0, the rules and routes will be deleted. Next, it is checked if there is a Wi-Fi connection, and in affirmative case it is applied the IP route and rule to the Wi-Fi network. Last but not least, it is checked if there are one or more WAVE connections and one Wi-Fi, and if that condition is matched, the *iptables* are applied in order to mark all incoming traffic with destination port of 1510 (for example) and it is set a rule to forward this marked traffic through the ip4tnl<USERID>g. If the previous condition is not matched, the *iptables* are removed, and the process described returns to the beginning and repeats itself.

4.2.5 Uplink Multi-hop Support

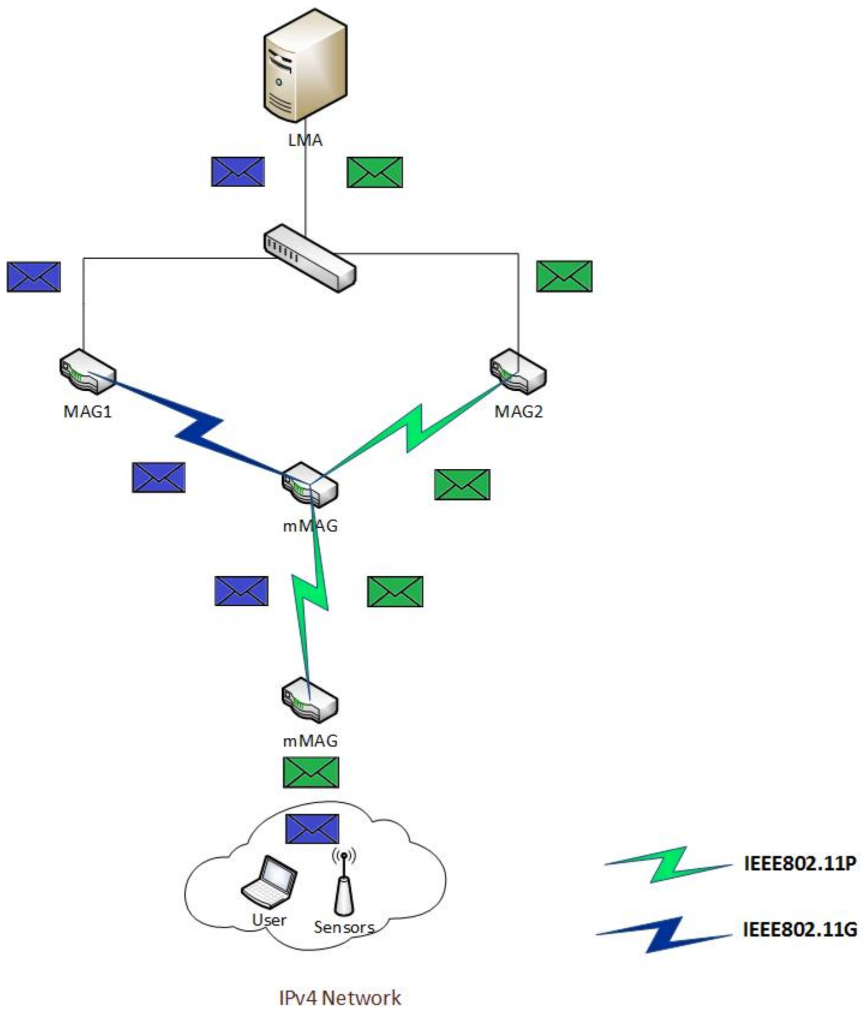


Figure 4.5: Multi-hop Scenario

This section describes the required changes to have the same functionalities previous implemented, but now in a multi-hop scenario. As can be seen from the figure 4.5, the main challenge of this scenario is that all the traffic from mMAG2 is routed to the ip4tnl<USERID>p, and when the same traffic arrives at the mMAG1, this entity is now unable to mark the traffic based on the port, because of the extra header that is added in the tunnel. The solution to this problem is to send all the traffic from the mMAG 2 direct to the next hop mMAG 1 IPv4 interface address. To do that, first, the mMAG needs to know if it is in single or multi-hop, and also needs to know the next hop mMAG ID. As the figure shows, and considering these values just as examples, all the WAVE interfaces IPv4 addresses are equal to 172.16.X.Y, where the X is first and Y the last 2 numbers of the board ID. One more time, in order to keep the network overhead low, the necessary information will be sent inside the RS message.

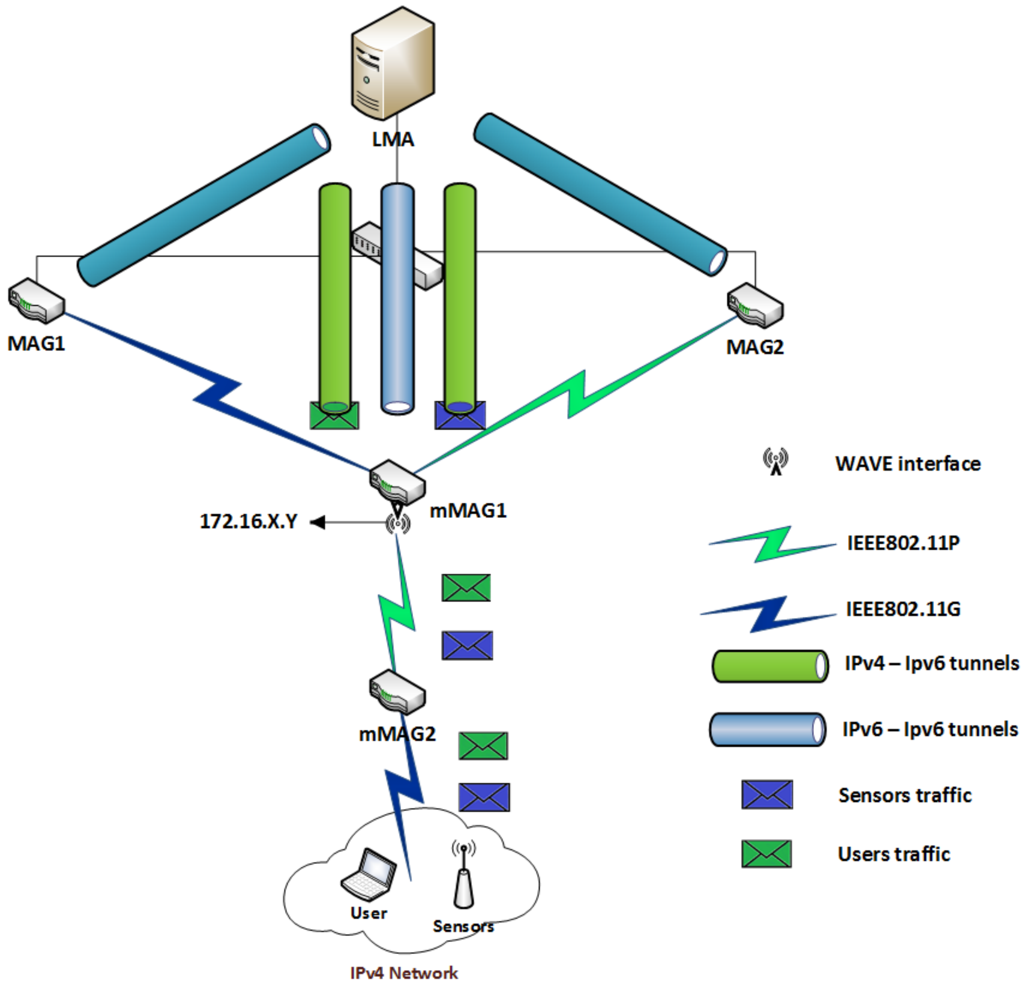


Figure 4.6: Multi-hop Scenario solution

When the mMAG2 sends the RS to the mMAG1 to start the registration process, the mMAG 1 will reply with a RA, that contains a field called "code" that will be used to signal if the mMAG2 is in multi-hop or not. This flag is set to 1 whenever a mMAG sends a RA, because if a mMAG sends a RA, it means that the destination node is in multi-hop. The other remaining bits are used to carry the mMAG ID. Now, if a mMAG is in multi-hop, it will route all the traffic to the next hop mMAG WAVE interface, and this entity performs the multihoming management explained in the section 4.2.4.

For the single-hop mMAG to be able to differentiate traffic based on the port, changes in the *iptables* were needed since, in a multi-hop perspective, the node that announces the IPv4 network will be the multi-hop mMAG. Since the single-hop mMAG will not announce the IPv4 network to the users, it will not be needed to configure a virtual interface, and the *iptables* that mark the traffic arriving from this interface must change to mark the traffic arriving from the WAVE interface, because this is the technology that links the single-hop mMAG to the multi-hop mMAG.

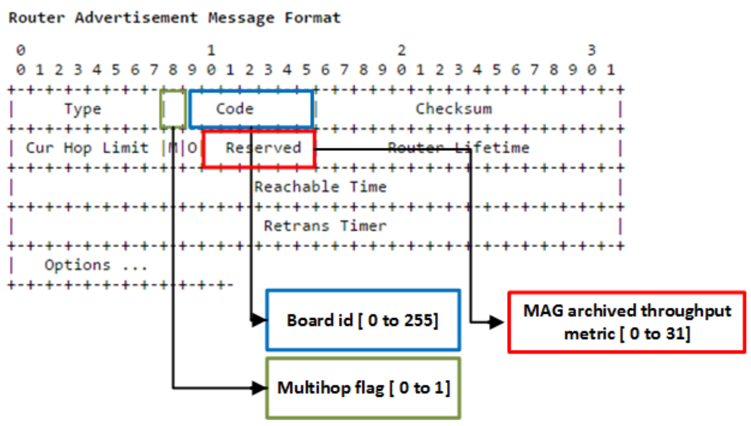


Figure 4.7: Router Advertisement Message based on [11]

4.2.6 Multi-hop behavior

Since it is possible to have both single-hop and multi-hop connections, the connection manager must be prepared to handle these types of scenarios. In these scenarios different decisions can be taken: if the multi-hop OBU has single-hop WAVE connections, the multi-hop connections will be ignored, and the traffic will be routed assuming the single-hop behaviour, as in Algorithm 1; if there are no single-hop connections, it means that only multi-hop connections are available and all the traffic will be routed through the best next-hop WAVE interface. The sensors' traffic will be routed through Wi-Fi, if it is available, considering always single-hop Wi-Fi; if it is not available, it will be routed through WAVE with the same conditions as the users' traffic.

As can be concluded, the single-hop connections have priority over the multi-hop ones. In multi-hop, the communication between OBUs is performed using a WAVE connection. Consequently, the same interface is responsible for receiving and forwarding the packets, which significantly reduces the available bandwidth due to the shared medium.

Algorithm 1 Multi-hop thread

```
procedure MultihopThread()
Case 1: Any single hop wave connection:
    addIPv6Route();
    addIPv6Rule();
    deleteIPv4DefaultRoute();
    addDefautlToIpv4TunnelWAVE();
    return();
Case 2 : Any single hop Wi-Fi connection:
    addIPv6Route();
    addIPv6Rule();
    deleteIPv4DefaultRoute();
    addDefautlToIpv4TunnelWiFi();
    return();
Case 3: Any multi hop wave connection:
    deleteIPv4DefaultRoute();
    addDefautlToInterfaceWAVE();
    return();
```

As it was previously explained, the single-hop connections will have priority over the multi-hop ones. The explanation is easy, since the links between mMAGs are done using WAVE connections, the same interface will be responsible to receive and forward the packets, reducing the available bandwidth due to the fact that the medium is shared.

It can also be noticed that the algorithm does not contemplate the chance of a multi-hop Wi-Fi network. This occurs because, as was explained in section 2.3, in our mobility protocol, the mMAG only connects between each other using WAVE technology.

4.3 Uplink manager thread

In order to accommodate both the single and multi-hop scenarios, is was required to create a top level thread, that is responsible for detecting the node role in the network, and act accordingly. The uplink top level thread will detect if the mMAG is in multi-hop or not. If it is, it will manage the uplink routes according to the algorithm 1. On other side, if the mMAG is in single-hop, it will be managed according to the flow diagram shown in figure 4.4.

As can be seen in the figure 4.8, first it is checked if there are more that one single-hop connections, and in case of affirmative answer it is called the single-hop uplink manager (figure 4.4) and returns to the begin. On the other side, if there are not more than one single hop connections, it is checked if there are any multi-hop connections and, in case of negative answer, the cycle returns to the beginning. In case of an affirmative answer, it is checked if there were network changes since the last time that the multi-hop uplink manager was called, and in case this condition returns true, it is called the multi-hop uplink manager. Note that if there was only one single hop connection available, the implementation of the connection manager ensures uplink connectivity by default routes (it is not needed marks or *iptables* because there is only only route available). Another aspect that must be taken into account is the importance of the function that returns one if it detects changes in the network. Since the multi-hop uplink manager, as will be explained better in the next section, applies an uplink route and returns to the beginning of the thread, without the detection of changes in the network, the multi-hop manager would add multiple times the same route unnecessarily.

In order for the connection manager to detect if there is a connection in multi-hop, when a mMAG receives a RA, it is checked the multi-hop flag value presented in the RA message, and if it is one, it activates the flag *multiHoopDetected*.

Also it is added to the structure *MacsConnected* (structure that has the MAC addresses of the current WAVE connections) the field *multihoop* in order to know which WAVE connections are in multi-hop and which are not. In Wi-Fi case, since it is only possible to have one connection, it was created a global variable that is set to 1 if a RS directed to the Wi-Fi interface has the multi-hop flag set to 1.

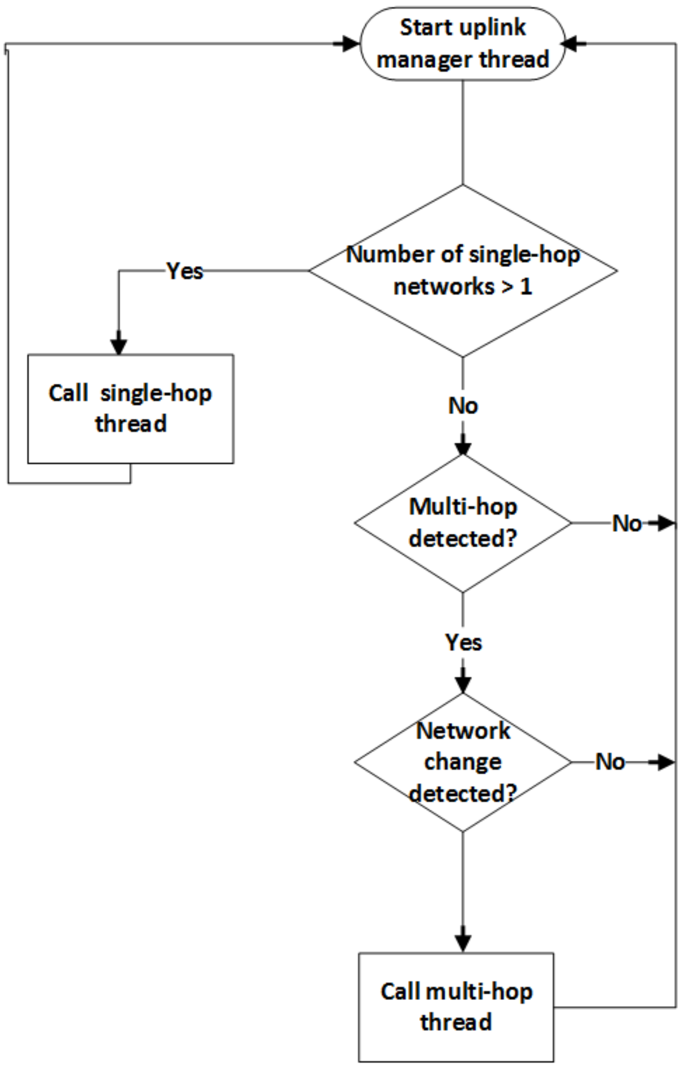


Figure 4.8: Uplink top level thread

4.4 N-PMIPv6 Improvements

In the base of the N-PMIPv6 multihoming approach several optimization aspects were still missing. The goal of this section is to explain the improvements done in the overall network performance and new functionalities added into the mobility protocol and multihoming integration.

In [10] it was implemented a disconnect message to inform the LMA that a connection is with a bad signal level. When the LMA received this message, the connection was discarded and the traffic was forwarded through the other available connections. This functionality was only implemented in WAVE connections, but its performance could still be improved. It was decided to perform major changes in the connection manager and on the LMA logic to fulfill these gaps.

One totally new feature added to the network was the ability of a reconnection. In the previous implementation, once the LMA received a disconnect message, it was impossible to use that route again without waiting the specified time-out, removal of the connection in the LMA's UCE and BCE. Now, if a disconnect message is sent due to the bad signal, and a car moves again to a place with good WAVE or Wi-Fi signal, the connection is instantaneously taken into account in the multihoming framework.

Another new functionality is the multi-hop disconnect message, in WAVE and Wi-Fi connections. The goal is, when the single-hop mMAG, that serves a multi-hop mMAG, sends the disconnect message to the LMA, the multi-hop mMAG connection must be also discarded, otherwise the LMA would continue to send traffic to a connection that did not exist anymore.

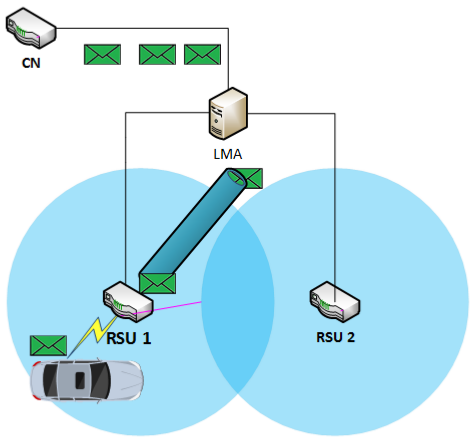
Another limitation found was in the multihoming rule, in multi-hop cases. The problem found was that the LMA did not have proper information about the load of the multi-hop mMAG, in order to divide the downlink traffic properly.

Lastly, it was noticed that the multi-hop flag was missing from the N-PMIPv6 implementation (M flag). The previous implementation of the N-PMIPv6 only compared the traffic source, and if it was from a specified source, it was considered to be a multi-hop connection. To remove that "hard coded" solution, the PBU will carry a flag that indicates if the node that is trying to join the network is in single or multi-hop.

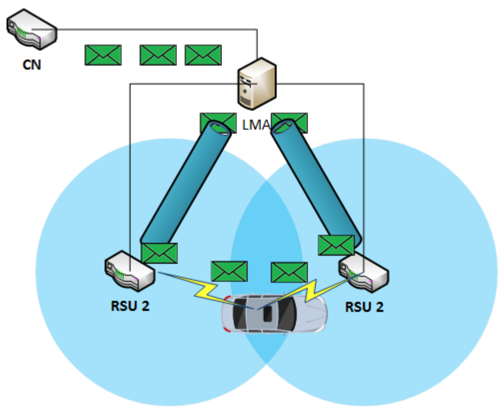
4.4.1 Disconnect message

As can be seen in the figure 4.9, it is sent a disconnect message when the connection signal is poor, to inform the LMA to discard this connection and use the others with better signal. The advantage of this message is, when the signal is bad and while the car is moving away from the RSU, it sends a disconnect message, before actually losing the connection, letting the LMA to use the other connections with good quality signal to route the downlink traffic. In this way, instead of a momentarily drop of packets sent due to the lost of one connection, the traffic is sent by the other RSUs with good signal without losing any packets (packet loss was happening with the previous handover precesses).

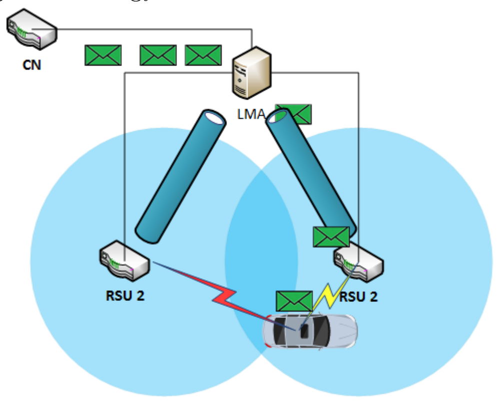
To solve this problem several changes were performed in the connection manager and in the N-PMIPv6. In the connection manager it is needed to evaluate the connection signal quality of each RSU connection, and when the signal is poor, send a disconnect message to the next hop MAG. Now, in the N-PMIPv6 domain, this disconnect message must be processed and forwarded to the LMA, that must discard this connection and use the others available.



(a) Connected in single technology



(b) Connected in multihoming



(c) Disconnect message

Figure 4.9: Disconnect message time-lapse

To achieve this, the following changes are required in the connection manager and in the N-PMIPv6:

- **Connection Manager :** The first challenge is to find a way to retrieve the correct values of signal quality of the actual Wi-Fi connection in order to know when to send to the LMA the disconnect message. To do so, the information about the interface link quality is retrieved from the file *proc/net/wireless/*, that has a field called level. Another important aspect is that the interface level is a negative value, and to be possible to use one threshold, for WAVE and Wi-Fi connections, the level value has to be converted to absolute. Now that the value of connection quality is available in a variable, it will be needed some changes in the Wi-Fi thread. First, now the node will try to find new Wi-Fi networks whenever the connection level is larger than the scan threshold, or if a disconnect is performed.

  Then, it is checked if the level threshold is larger than the disconnect threshold, and if it is, several actions are performed. First, the number of Wi-Fi networks is updated to 0, the IPv6 uplink rule and route are deleted, the disconnect flag is changed to 1, it is sent an RS with the first 16 reserved fields containing the value of the connections RSSI, and lastly, the mMAG performs the disconnect from the Wi-Fi network. Inside the function that controls the RS messages, it is checked if the disconnect flag is 1, and if it is, the mMAG does not send the periodic RS through the disconnected interface anymore. Later, in order to add support to the reconnect feature, when the signal returns to good levels, the RS is sent to the MAG, with the information about the connection RSSI.

- **N-PMIPv6 MAG side :** When the MAG receives the RS message, it will retrieve the connection RSSI and add that information to the BCE, refreshing that value each time a RS is received. Next, the RSSI value is sent to the LMA through a special socket, that has a type to identify that a disconnect is being sent.
- **N-PMIPv6 LMA side :** Now that the LMA receives the message with a type equal to disconnect, the LMA will save the RSSI information in BCE and UCE for further use. Next, it will force an update to the connection information field, that consists in accessing the UCE and update the interface disconnect flag(1), the RSSI value (level) and access technology (WAVE or Wi-Fi), and finally will force an update of the rule, responsible for dividing in the best way the downlink traffic, that now, due to the fact that the UCE interface disconnect flag is equal to 1, it will not consider this interface in the calculation of the percentage of traffic that goes through each available connection.

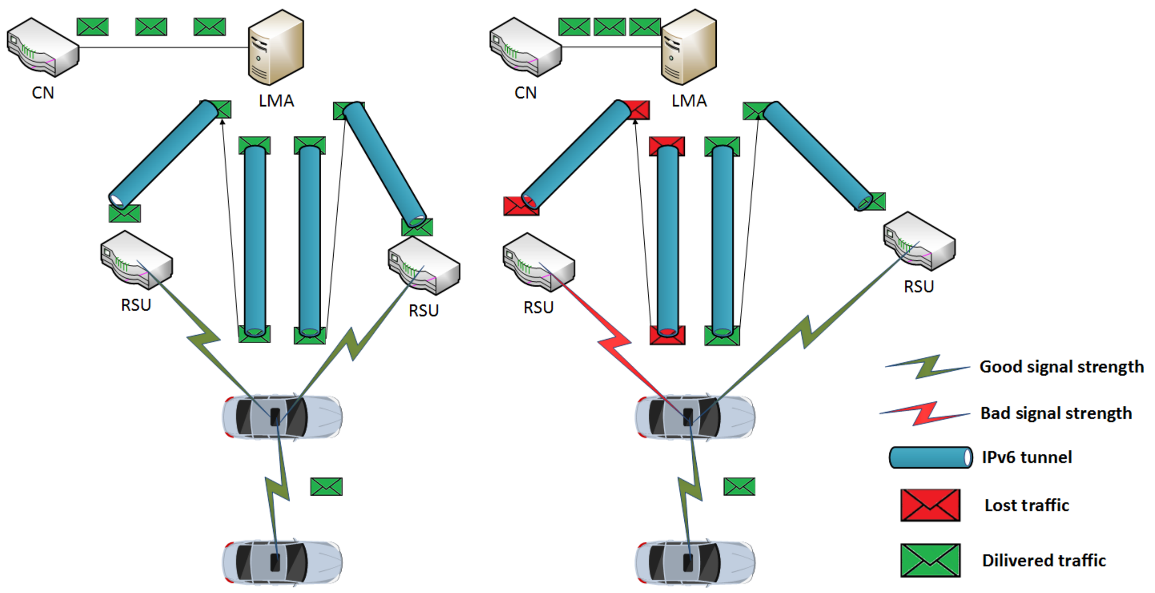
  To give support to the reconnection mechanism, due to previous robust implementation of the disconnect message, it only was needed small changes in the N-PMIPv6. Before the proper calculation of the rule percentage of each interface, it is written in the auxiliary UCE the value of the interface RSSI, and if the interface RSSI is larger than a defined threshold of connection quality and if the interface has the disconnect flag set to 1, that flag will be changed to 0. Next time the algorithm calculates the percentage of traffic that goes through each connection, the reconnected connection will be taken into account again.

Previously, it was mentioned the use of an auxiliary UCE that contains the same information that the normal UCE and was created to avoid concurrence problems due to the use of the normal UCE in another thread. As stated in *n-PMIPv6 LMA* items, if the RSSI returns to good values, the value of the auxiliary UCE disconnect flag is changed (1 to 0) and the assert fails. To solve this setback, when the flag is changed, the index of the UCE is stored in a global variable and later in the code, when the thread that uses the auxiliary UCE is killed, and before the assert, it is checked if the value of the global variable is equal or larger than 0, and if it is, it is changed the disconnect flag to 0 from the UCE with that index.

4.4.2 Disconnect message in multi-hop

After the single hop disconnect message is implemented, it was tested the multi-hop scenario but a problem has been found. In the figure 4.10 it is demonstrated the situation. In the right figure it is shown the normal multi-hop scenario. First, the CN sends traffic to the second car, that is in multi-hop. The traffic goes through the LMA, that calculates the percentage of traffic that goes through each interface and routes the traffic to the tunnel that links the LMA and the multi-hop node. Then, the traffic is encapsulated again in the respective tunnels that link the LMA and the RSUs, and through the normal routing mechanisms it is delivered to the second hop car.

Now, considering the disconnect message scenario, when the single-hop car moves away from the first RSU, the signal quality will decrease and a disconnect message is sent to the



(a) Multi-hop normal situation (b) Multi-hop disconnect message problem

Figure 4.10: Multi-hop scenarios

LMA, that performs the actions stated in the previous section, and stops sending traffic through the tunnel that links the LMA to the RSU, but continues to send traffic through the tunnel that links the LMA to the single-hop car (tunnel that serves the multi-hop car). When the connection between the single-hop car and the first RSU is lost, the LMA will continue to route the traffic through that tunnel. The traffic that arrives to the first RSU, does not have connectivity to the single-hop car and the packets turn out to be lost. Figure 4.10b illustrates this problem.

To solve this problem, changes in the N-PMIPv6 protocol were needed. Now, instead of the mobility protocol to creates a tunnel per interface, it will create a tunnel per mMAG. In this way, all the traffic will be routed through the only IPv6 tunnel that was created between the LMA and the mMAG, and then, the LMA will route the traffic through the tunnels that exist between the LMA and the RSUs.

4.4.3 WAVE limit in case of Multi-hop

During the execution of this dissertation, it was noticed that the multihoming rule was not optimized in the multi-hop scenario. The problem is illustrated in the figure 4.11.

When there is a multi-hop mMAG that is not in range of the MAGs and there is downlink and uplink traffic, the multihoming rule has a gap that will be explained next. This limitation is of particular importance because, in the most of the cases, the multi-hop car is not in range of the RSUs, and it is normal to have uplink and downlink traffic at the same time.

As the figure 4.11 illustrates, when there is available a WAVE and a Wi-Fi MAG, in the point of view of the MAG, the WAVE medium limit is only affected by the 3 Mbps sent between the mMAG 1 and the MAG 1. However, in the point of view of the mMAG 1, the WAVE medium limit is affected by other factor. The mMAG 1, as it is illustrated in the figure 4.11, receives 3 Mbps from the mMAG 2, and this traffic is forwarded to the LMA through the

WAVE technology, and due to its broadcast nature, the WAVE medium limit will be affected not only by the 3, but by 6 Mbps. This means that, in the previous implementation, the LMA is sending more traffic through the WAVE MAG than it should.

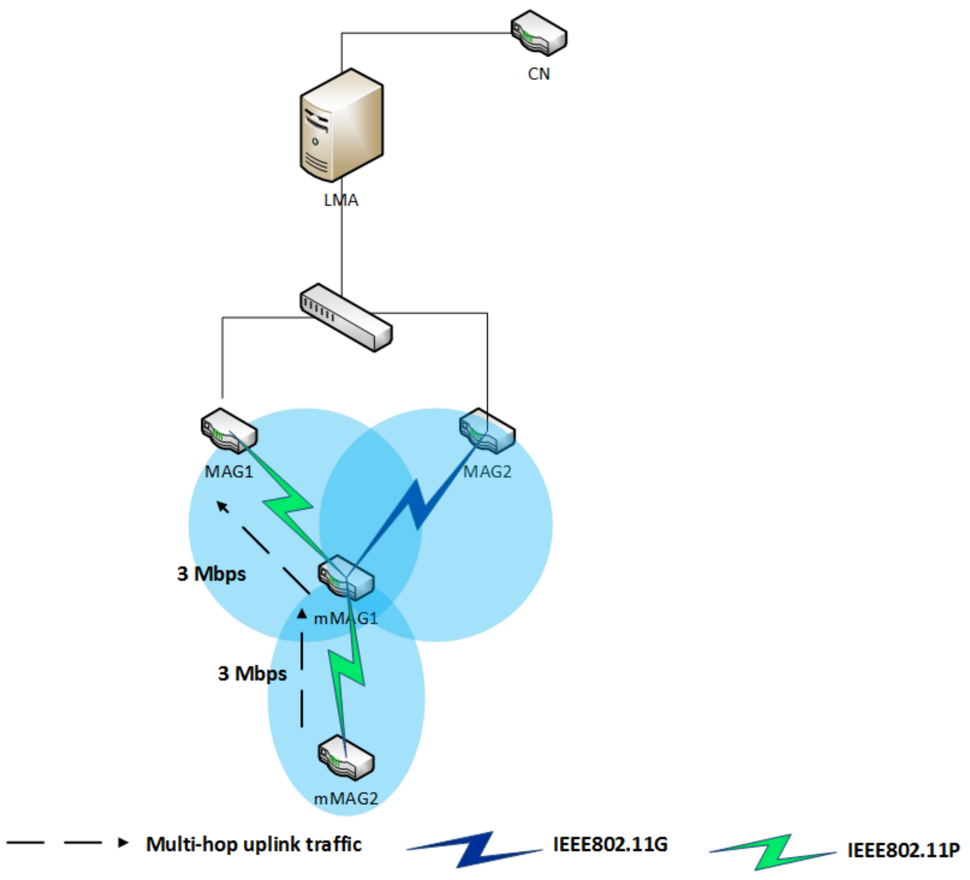


Figure 4.11: WAVE limit in multi-hop

To solve this limitation, in the RS message that the single-hop mMAG sends to the MAG, it will contain the value of the achieved throughput of the WAVE interface (figure 4.12). It was noticed that the code field is not used in the N-PMIPv6 protocol. Like in the case that the MAG sent its achieved throughput to the mMAG, due to the limit amount of bits, it is required to convert the values to a scale that has a maximum value of 255. To calculate the achieved throughput, it was used the expression 4.1 and to convert the achieved throughput to a scale that fits the code field, it was used the expression 4.2.

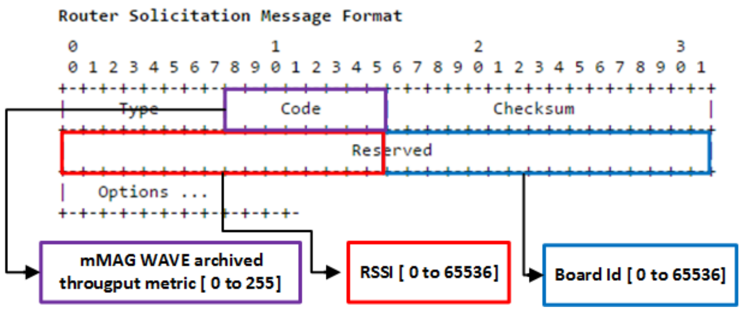

Router Solicitation Message Format
Type
Code
Checksum
Reserved
Options ...
mMAG WAVE archived througput metric [0 to 255]
RSSI [0 to 65536]
Board Id [0 to 65536]

Figure 4.12: RS message format based on [12]

When the RS arrives at the MAG, it is introduced in the BCE and sent to the socket that is responsible to deliver the information to the LMA. Finally, the LMA return the converted value to its original form.

$$\text{AT} = \frac{\text{mMAG}_{AT} * MaxPoACapacity}{MaxScale} \tag{4.4}$$

Using the scenario of the figure 4.11 as example, the LMA has information about the AT (equation 4.4) of the MAG 1 and the mMAG1. In order to calculate the amount of traffic that goes through each connection, the LMA will use the lowest AT value. This happens because the LMA, in order to deliver the traffic to the multi-hop mMAG, the traffic will pass through the MAG 1 and the mMAG 1, so it is normal that the node with the lowest AT will limit the amount of traffic that can be routed through that path.

4.4.4 Multi-hop flag

One key aspect that is missing in the N-PMIPv6 implementation is the multi-hop flag (M flag) that indicates if the node is in single or multi-hop, giving to the LMA a tool to better manage the downlink traffic division.

When the mMAG in multi-hop tries to join the network, it will send an RS message to the mMAG in single-hop, and the mMAG upon the receiving checks if it is a MAG or a mMAG. This is done by comparing the node egress address with the loopback address, and if they are equal it means that the mMAG only has ingress address and that is an mMAG. If the node is a mMAG and receives a RS, it means that the node that is trying to join the network is in multi-hop. Now, the mMAG that received the RS will send a PBU (figure 4.13) to the LMA, due to the network abstraction explained in section 3.2.1. When the LMA receives the PBU, it checks the M flag, and then copies that value to the BCE and next to the UCE.

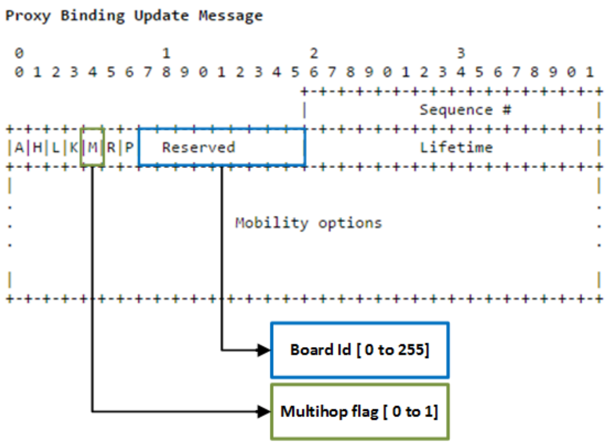


Figure 4.13: PBU message format based on [12]

4.5 Chapter Considerations

This chapter described the approach for uplink multihoming and the changes required to optimize the mobility and multihoming processes.

First, it was possible to differentiate traffic depending on its type (port) using *iptables*, and with the use of *iprules* to route that each traffic using different technologies. To achieve a totally dynamic uplink manager, it was required to send the mMAG ID to the LMA and MAG. After that, it was possible to implement an uplink thread that manages the single and multi-hop uplink. In the single-hop scenario, the uplink manager takes into account the MAG AT and the load that the mMAG will add in the network avoiding constants and unnecessary changes of routes. In the multi-hop scenario, it was defined that the single-hop connections will have priority over the multi-hop ones and the WAVE over the Wi-Fi, in order to keep this management as simple as possible due to the multi-hop complexity.

After the uplink management was concluded, it was created and improved the Wi-Fi and WAVE disconnect message, respectively, in order to improve the overall network performance. It was also added the reconnect feature that allows the usage of a disconnect network if the signal quality returns to good levels, both in single and multi-hop. Then, it was created the multi-hop flag that allows the LMA to know when and which node is in multi or single-hop, without the need of "hard coded" solution to identify that node. Finally, it was optimized the multihoming framework to take into account the amount of load that the WAVE interface of the single hop mMAG has, improving the multihoming the framework efficiency

To support these major improvements in our VANET, changes in the N-PMIPv6 mobility messages were also required, such as:

- **RS :** The RS will carry information about the mMAG WAVE interface load, mMAG ID and the connection level (RSSI).

- **RA :** The RA will carry the MAG AT, the multi-hop flag and the mMAG board ID in order to support the multi-hop uplink manager.

- **PBU :** The PBU will carry the mMAG ID and the multi-hop flag.

Concluding, with all these changes, the N-PMIPv6 becomes more dynamic and it will not depend on "hard coded" solutions to add routes or create tunnels.

Chapter 5

Network Coding Integration

5.1 Introduction

Due to the high mobility of vehicular networks, sometimes this types of networks are vulnerable to packets losses due to connection signal variations, reflections and other events.

In our approach, we aim to use network coding especially to improve losses in wireless environments and make use of the several available paths through multihoming. Apart from several existing strategies, two parameters that significantly affect the performance and the impact of the network coding are: the number of original packets that are used to generate an encoded packet (N); and the number of extra encoded packets sent (M). An inappropriate configuration of these parameters could dramatically increase the packet loss probability and/or introduce an exaggerated overhead, without taking any benefits. To take the most advantage of the network coding potential, we propose two algorithms that, taking into account the percentage of packets losses, they determine the encoder configurations that provide the best QoS: one algorithm is designed to achieve the lowest packet loss, and the other is designed to achieve a packet loss lower than a threshold, with the lowest overhead.

As stated in 3.4, the base network coding approach has several limitations:

- ***TAP* interface did not monitor packets :** Since only traffic that was sent to the *TAP* interface was processed by the network coding, the traffic is not traversing the coding process.
- **Decoded packets were not forwarded :** After the coded packets have been decoded, they were not sent to the next hop or returned to the kernel to be forwarded.
- **Only works with one technology :** The network coding process was designed to work with one technology at a time.
- **Routing based in static MAC addresses :** A vehicular network is known by its mobility, so the resource to static MAC addresses to perform routing is not a good option.
- **Only works in IPv4 :** Its important that the network coding works in both IPv4 and IPv6 protocols.
- **Improve the way packets losses are simulated in the network :** A new framework is designed to simulate the packet loss.

The organization of this chapter will be the following.

Section 5.2 will describe a practical example in which the network coding can be used.

Section 5.3 will describe the technical challenges found in the integration the network coding in our VANET.

Section 5.4 will describe how the network coding in performed in uplink scenarios.

Section 5.5 will describe the changes performed in the uplink base work to be possible to use the network coding in downlink.

Section 5.6 will detail the input parameters required for the network coding framework to work properly.

Section 5.7 will describe the two algorithms that return the optimal encoder configurations. In the end of this subsection there will be a discussion about all the advantages and drawbacks of each algorithm.

Finally, section 5.8 will present an overview about the work performed in this chapter.

5.2 Network Coding in vehicular networks

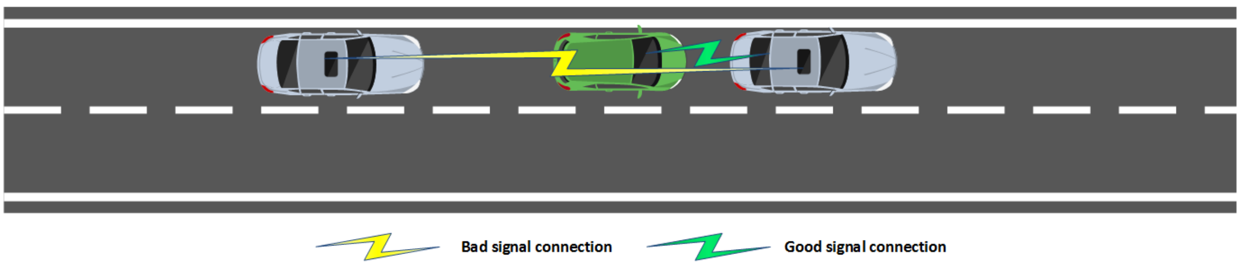


Figure 5.1: Emergency break situation

Figure 5.1 illustrates a highway scnerario, where the front car must brake abruptly. Immediately, the car sends an emergency message in broadcast, to the surrounding cars, in order to avoid a mass collision. The second car, because it is driving closer and has no obstacles between, has a good connection to the first vehicle and receives the emergency message. However, the second car, due to the presence of a car in the middle (obstacle) and a longer distance to the first car, it has a poor signal quality, meaning that the emergency message could be lost, and increasing the probability of the third car crashes into the second car.

To help prevent this type of dangerous situations we want to use network coding, to increase wireless data transmission reliability. Moreover, we want to go even further, and provide reliable data transition in multihoming scenario, in order to take advantage of both multihoming and network coding features, already described in this document in section 2.

As stated in chapter 3, we choose the systematic network coding approach. In this approach, for each N packets, M redundant packets will be sent. Instead of sending random redundant packets, we will send redundant packets that contain information about several packets. This particular aspect will turn possible to recover from high values of packets losses, with a small amount of redundant packets.

Concluding, with the use of network coding, we will be able to:

- **Recover from packets losses:** With network coding, we will be able to recover from packet losses, using cleaver redundancy.

- **Recover from out-of-order packets:** With network coding, all the packets will be delivered to the final destination, in order. This is particularly important in multihoming scnerario, because multihoming increases the out-of-order packets, due to the several paths used.

- **Improve network security:** As said before in this document, WAVE is a broadcast technology. This means that all nodes "hear" the neighbors transmissions, even if they are not the destination. Because in network coding all the packets are sent by an encoder, this will mean that, in order for the neighbors to access to the data, the traffic must be processed by a decoder (even the packets that are not coded, due to the extra header added in the encoder).

5.3 Challenges in the Network Coding Integration

In the next subsections it will be described all the challenges and solutions found, while integrating the network coding framework in our VANET.

5.3.1 Capture incoming traffic

The traffic that is sent to the base encoder was not processed by the network coding because the program only analyzes the packets that go through the *TAP* interface. The original program lacks the *TAP* interface configurations, and a mechanism to force the traffic that goes through the incoming interface to be forwarded through the *TAP* interface. To solve this problem, it is built a bash program that is called before calling the network coding. This bash script contains the following commands:

- *ip tuntap add mode tap dev TAP<ID>*
- *ip link set tap<id> master <INPUT_INTERFACE>*
- *ip link set dev TAP<ID> up*
- *ebtables -t nat -A PREROUTING -i <INPUT_INTERFACE> -j dnat –to-dst ff:ff:ff:ff:ff:ff*

These commands configure the interface *TAP*<ID> in tap mode, allowing to use a socket in this interface to capture all traffic that goes through this interface, including layer 2 header. Next, the tap<ID> interface is set to master of *br-lan* allowing that all traffic that goes through <INPUT_INTERFACE> also goes through *TAP*<id> interface. Finally, the *ebtable* changes all the destination MAC address of the traffic that arrives in the <INPUT_INTERFACE> to the broadcast address, routing the traffic through the *TAP* interface.

5.3.2 Simulate network packet losses

The original way used to simulate packet losses in the Gekko program was to, in each generation, not send a specific packet. Taking as an example a generation with eight normal and one coded packets, if it is supposed to simulate a network packets losses percentage around 12 or 13 percent, this was done by simply not sending a packet in each generation. The results were not as expected, because with only one coded packet in each eight normal packets, it was possible to obtain a final packet loss of zero. After comparing this result with the theoretical base of the SNC, it was concluded that some information could not be recovered. To make the simulation of packets losses more real, it was used the C rand function, that returns a random value between 0 and 1. Then this value is converted to percentage, and if this value is higher than the value of a packet loss threshold, the packet is simply not sent. With this method of inserting the packets losses, the theoretical SNC packets losses match the practical tests performed in our network.

5.3.3 Forward decoded traffic

It was noticed that the traffic was coded correctly and sent to the next hop to be decoded, but it was not sent afterwards. To solve this problem, it was created a function that routes this traffic through a layer 2 socket to the next hop based in the MAC address of the next hop, in this case the LMA MAC address.

5.3.4 Integrate Systematic Network Coding in the N-PMIPv6

The major challenge of this work is to integrate the network coding without losing any key feature of the vehicular network, i.e, mobility, multihoming, and the multi-hop support. In the next subsections it will be described the strategies used to turn the network coding seamless to the mobility protocol.

5.3.4.1 Routing strategy

One of the main challenges of integrating network coding with the current vehicular network is the fact that network coding routing is performed in layer 2 (the routing in our vehicular network is layer 2 routing), and the mobility protocol routing is performed in layer 3.

Another important aspect is the fact that, in the coded packets, the information about IP addresses is coded: in a multi-hop scenario the redundant coded packets could not be routed, because the node does not know the final destination of the packets due to the coded IP addresses.

To overcome these issues, the traffic processed by the network coding will be routed in layer 2. The N-PMIPv6 and the connection manager will configure all the related network aspects, and assure the required mobility, while the network coding will use the data provided by the N-PMIPv6 and the connection manager to perform the management of the network coding packets in layer 2. Taking this into account, it is created a shared memory region, where the connection manager/N-PMIPv6 (depends if traffic is in uplink or downlink) writes the MAC addresses of the next hop node and the network coding will read that value and perform the routing. This shared memory region will be based on a structure that has the following three fields: the control information to access the shared memory, the next-hop WAVE MAC address and the next-hop Wi-Fi MAC addresses.

In our implementation, the connection manager/N-PMIPv6 must be initialized first, in order to avoid attempts by the network coding to access to a memory region that is not created. Therefore, the connection manager/N-PMIPv6 is executed first, the shared memory region is created, and before the writing process begins, it changes the value of the control variable to *W*, indicating that it is being written information, denying access from the network coding program to the MAC addresses fields inside the shared memory region. When the connection manager finishes the writing process, it changes the value of the control variable to C, informing the network coding program that it is clear to read the values of the MAC addresses. When the network coding program gains access to the shared memory region, it will change the control variable to R, informing the PMIPv6/connection manager that the MAC addresses are being read. This mechanism avoids concurrency problems.

5.3.4.2 Multihoming feature

Both uplink and downlink traffic are multihomed in the vehicular network, and therefore, both need to be handled through network coding. In the network coding approach, to successfully decode the encoded payload, packets from a generation cannot be separated: while in multihoming, if packets from a generation end up being split through the available nodes, the decoder will be unable to decode the information. Therefore, the routing of normal traffic and its coded packets from one generation need to reach the same destination to enable a correct decoding.

In an uplink scenario, the OBU is responsible for encoding the packets for the Wi-Fi, WAVE and other technology RSUs. The uplink traffic needs several encoder threads, one dedicated to each technology (for example, one dedicated to the Wi-Fi and another to the WAVE technology). This way, the several technologies can be used, and packets from a generation are not separated. In a downlink scenario, the same problem exists, and the solution is to have, in the OBU, one decoder thread per RSU.

Multiple threads accessing the same *TAP* interface lead to concurrency problems. To solve this problem, it is added an extra *TAP* interface, with the same configurations as in section 5.3.1. Now, despite the fact that all traffic goes through the two *TAP* interfaces, only one thread is reading the *TAP* interface at the same time, solving the concurrency problem.

With these changes, it is now possible to use the network coding in the same way as the mobility protocol without any packet losses inherent to this approach.

5.3.4.3 Multi-hop support

If in a downlink perspective the multi-hop functionality does not introduce any special challenge, taking into account our routing strategy, in the uplink traffic the processes of encoding and forwarding the packets are not enough. As previously explained, the payload used to create a coded packet starts in the layer 3 header, i.e, the information about the destination port is coded: the middle node (second mMAG) is unable to access the destination port information, and consequently, it is unable to route the traffic based on the port (as in our connection manager).

To solve this problem, in the network coding header it is introduced a new field, the packet destination port. Whenever a packet is being processed in the encoder, the information of its destination port is saved, and when that packet arrives to the next hop, that node only needs to read the destination port from the network coding header and route the traffic accordingly.

5.4 Network Coding in uplink scenarios

This section describes the network coding process in uplink scenarios and the required extensions to both encoder and decoder algorithms.

Figure 5.2 depicts the uplink scenario that will be used to explain the uplink network coding extensions.

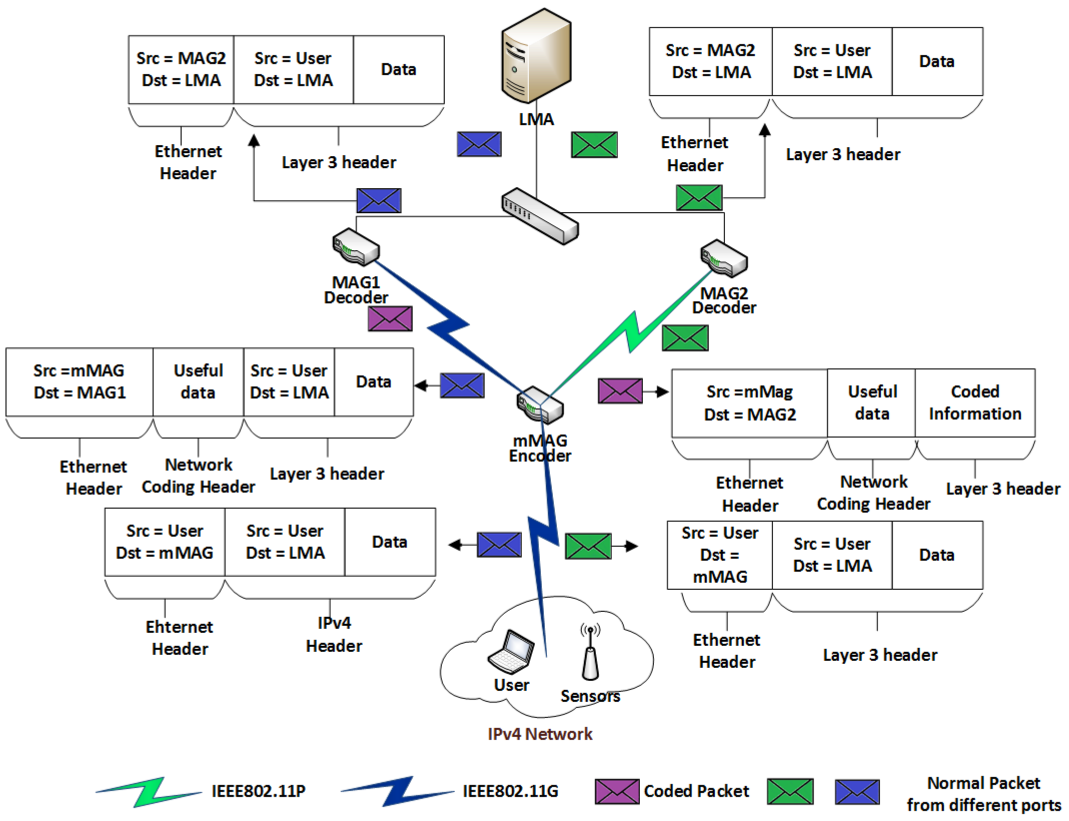


Figure 5.2: Network coding uplink scenario

5.4.1 Encoder process

As previously discussed, to provide multihoming support, the encoder will have an encoder thread per technology in use, 2 encoder threads in our specific case of WAVE and Wi-Fi (our vehicular network also includes LTE). This section describes the structure of each encoder, which is depicted in algorithm 1.

First, each encoder reads the traffic from a different *TAP* interface, and the condition to perform the network coding depends on the traffic and its destination port. As a specific example, all traffic that arrives with destination port equal to 3005 will be processed by the WAVE encoder, and 3004 will be processed by the Wi-Fi encoder.

The *TAP* interface configurations are performed and it is allocated memory to store the packets that will be received. Then, the selected network coding algorithm predicts the required values of N and M to be used in the encoder. Next, the encoder reads from the *TAP* file descriptor the traffic that enters the input interface. If it is defined that the encoder type is "NONE", the packets will not be encoded, only forwarded to the next hop MAG. If on the other side, the type is "SNC", the encoder will be configured with generations with N uncoded and M coded packets. In this case, the packet goes through the encoding process and it is added useful data to the network coding header. Now that the packet is ready to be sent, it is checked if the encoder is in single or multi-hop. If the encoder is in single-hop, the traffic will be split between the available technologies and networks based on the traffic destination port; on the other hand, if the encoder is in multi-hop, all traffic is forwarded based on the next hop WAVE interface MAC address, because in our VANET all multi-hop connections are performed using WAVE technology (vehicle-to-vehicle connections are WAVE-based). Before the traffic is sent, it is added a field to the network coding header with the information of the traffic destination port. This is performed because, in a multi-hop scenario, all layer 3 information is coded, and when those coded packets arrived to the middle node, that node needs to have the information to route the packets to the correct destination.

Algorithm 2 Encoder process

```
 1: procedure Begin
 2:     Create TAP interface
 3:     Calculate coder parameters
 4:     while 1 do
 5:         Read TAP interface
 6:         if Traffic detected and specific port then
 7:             if Coder is not configured then
 8:                 configure coder
 9:             Push packet to encoder
10:             if Multi-hop flag == 0 then
11:                 if Encoder WAVE thread then
12:                     Read shared memory region
13:                     Send packets through WAVE
14:                 if Destination Port == X then
15:                     Read shared memory region
16:                     Send packets through Wi-Fi
17:             if Destination Port == Y then
18:                 Read shared memory region
19:                 Send all packets through WAVE
```

5.4.2 Decoder process

The decoder process is depicted in algorithm 3. First it is performed an interface bind to capture all input traffic. Then, the decoder is configured based on the network coding header, and with the required input arguments of the network coding algorithm to determine the best network coding configurations.

Then, it is checked if the node has the multi-hop flag activated. In order for the mMAG to know if it is supposed to decode or forward the encoded packets to the next node, it was used a multi-hop flag. In a single-hop scenario, the node that receives the encoded traffic will have the multi-hop flag set to 0, meaning that it must decode the information and send it to the LMA. If, on the other hand, the decoder node has the multi-hop flag set to 1 (single-hop mMAG in multi-hop scenarios), it is required to have a shared memory region, as explained in section 5.3.4.1, in order for the decoder to access the next hop MAC addresses information. Next, the encoded information is sent along side with the network coding header. This happens because this node is only responsible for forwarding the traffic, and it is required that the network coding header remains untouched, in order for the next hop to be able to perform the decoding without problems. Last but not least, the traffic is routed based on the network coding header port field.

Algorithm 3 Decoder process

```
 1: procedure BEGIN
 2:     Bind to input interface
 3:     while 1 do
 4:         Read input interface
 5:         if Coded traffic detected then
 6:             if Coder is not configured then
 7:                 configure decoder
 8:             if Multi-hop flag == 0 then
 9:                 Push packet to decoder
10:                 Read shared memory region
11:                 Send traffic LMA
12:             if Multi-hop flag == 1 then
13:                 if Destination Port == X then
14:                     Read shared memory region
15:                     Send packets through Wi-Fi
16:                 if Destination Port == Y then
17:                     Read shared memory region
18:                     Send packets through WAVE
```

5.5 Network Coding in downlink scenarios

As shown in the figure 5.3, now the goal is to use network coding in a downlink scenario. It will be used the same approach implemented in the uplink scenario, but with small changes that will be explained in the following sections.

5.5.1 Encoder process

Taking into account figure 5.3, it is possible to notice some aspects that are specific from the downlink scenario. First, the source of the traffic is the correspondent node, that sends packets to the mMAG through the LMA, that divides the traffic through the IPv6 tunnels. Now, the traffic arrives to the encoder *TAP* interface in IPv6 and with an extra tunnel header.

The first change here is to convert all the process from an IPv4 to an IPv6 logic, and then bypass the extra IPv6 tunnel header, in order to retrieve the destination port information.

The LMA is the entity responsible for dividing the traffic through the available MAGs in order to maximize the network performance. This particular aspect means that each RSU is responsible for coding the incoming packets and forwarding the traffic to the next node, meaning that it is only needed one *TAP* interface and one encoder process for each RSU.

Finally, to assure the overall vehicular mobility, it is also needed a shared memory region, but this time with the N-PMIPv6. This shared memory zone follows the same principles of the downlink shared memory region.

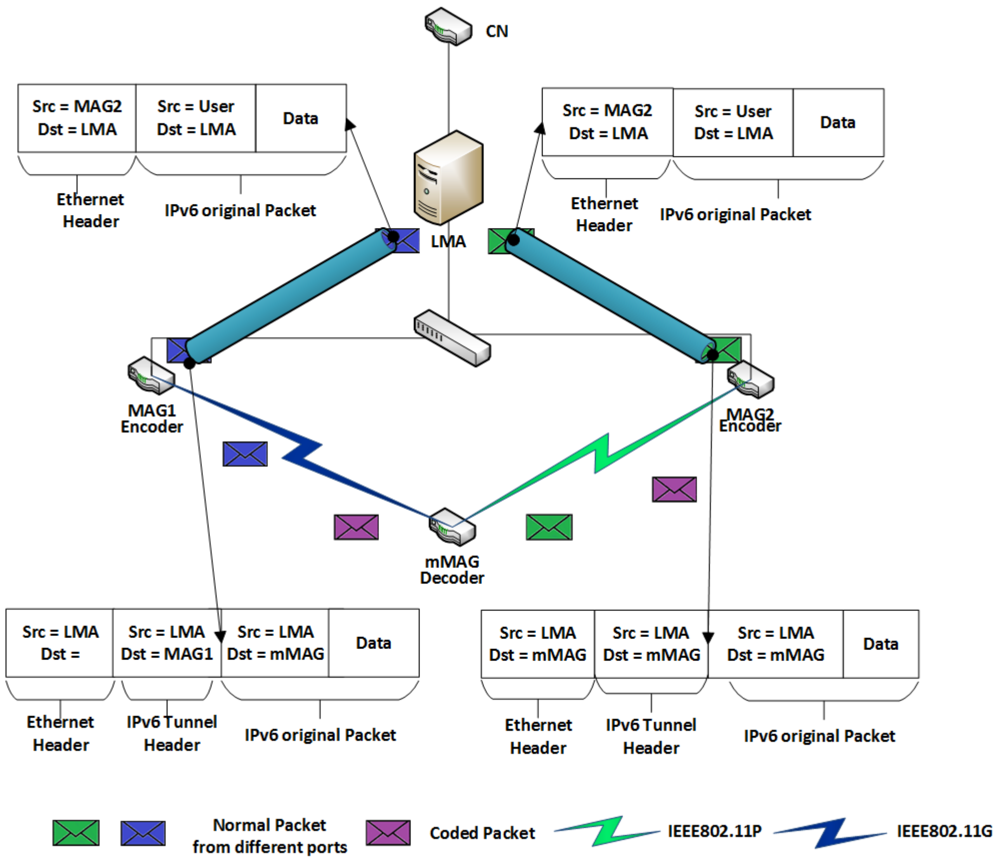


Figure 5.3: Network coding downlink scenario

5.5.2 Decoder process

In the decoder side it is needed only one small extension. The only difference from the uplink scenario is that, in downlink, it is considered that the destination of the packets is the mMAG. Now, instead of sending the decoded information to an other node, the decoder will return the decoded information to the upper layers, to be processed.

Concluding, with these small changes, it is now possible to code all the IPv6 downlink traffic with multihoming, where all the mobility is one more time assured by the N-PMIPv6 and the connection manager.

5.6 Gekko input arguments

Having in mind a future improved connection manager, capable of managing all the network coding usage, it were added some input parameters that make that management easier. In the past when the Gekko was called, these arguments were needed: *<Input Interface> <Output Interface> <Network Coding Type> <N Number of normal packets> <M Number of coded packet>*. Since now there are available two algorithms that find the most suitable values of M and N, those arguments were removed to give space for other more pertinent arguments.

From now on, the gekko program will receive as input argument *<Input Interface> <Output Interface> <Network Coding Type> <% of losses in WAVE> <% of losses in Wi-Fi*

<Desired Algorithm [1 or 2] > <Multi-hop flag[0 or 1]> . As can be concluded from the argument, all the features added to the Gekko program are performed dynamically without "hard coded" aspects. Now it is possible to give the program the values of packet losses percentage of the Wi-Fi and WAVE connection, select the most suitable algorithm and the program will configure the coder/decoder and perform all the routing by itself.

In a multi-hop scenario, the two mMAGs must use the *mMAG* flag set to 1, and the multi-hop mMAG must select the *Network Coding Type* to *NONE*, because the function of this node in this type of scenarios is to forward the traffic based on its port.

Concluding, with these changes it is now possible, in the future, to implement a connection manager that fulfills all these input arguments and manage all the aspects of the network coding, making this process more efficient.

5.7 Network Coding configuration algorithms

In this section we propose two algorithms to determine the best network coding configurations to optimize the network performance in terms of the quality provided to the services and, when possible, to minimize the overhead. These algorithms determine the number of uncoded (N) and coded packets (M) per generation. In our approach, the two algorithms take into account parameters such as the packet losses and the overhead, although each algorithm has a specific goal: the first one minimizes the packet losses without addressing the network overhead, while the second algorithm targets a performance threshold in terms of packet losses with overhead minimization.

5.7.1 Algorithm for packet losses' minimization

The main goal of this algorithm is to obtain the optimal configurations that assure the minimal packet losses.

Using the basis of work in [58], we use a mathematical expression that predicts the amount of packets successfully delivered (in percentage) when applying the systematic network coding with N uncoded packets, M coded packets and P_e packet loss probability, as can be seen in the equation 5.1:

$$Success = 1 - P_e * \sum_{i=N}^{N+M-1} \binom{N+M}{i} (1-P_e)^i * p^{N+M-i-1} \tag{5.1}$$

The percentage of successful delivery packets is now available, and the loss probability can now be determined. With the available expression relating the N and M values with the percentage of packet losses, it is possible to develop an algorithm that runs the possible configurations and predicts their impact in the network performance.

However, several optimizations can be performed as can be shown in Algorithm 4: we consider only combinations where N is larger than M; we include an overhead (M/N) limitation; the algorithm can return the values of M and N when the achieved error is lower than or equal to 0.1 % (this value can be considered zero), decreasing the number of iterations required by the algorithm when the network packet losses are already low; we include an upper bound to the amount of uncoded packets to 64 to achieve a reasonable value of generation size ($M+N$) and low decoding delays.

Algorithm 4 Network Coding Algorithm for packet losses' minimization

```
1: procedure BEGIN
2:     for $N = 2$ to $nLimit$ do
3:         for $M = 1$ to $N < M$ do
4:             $overhead = M/N$
5:             if $overhead < OverheadLimit$ then
6:                 $packetLosses$ = NetworkCoding_PacketLosses (M,N,Input_Packet_losses)
7:                 if $packetLosses < actualPacketLosses$ then
8:                     $actual_packet_losses = packetLosses$
9:                     save_NC_Configurations()
10:                    if $actualPacketLosses < 0.1$ % then
11:                        save_NC_Configurations()
12:                        break
```

5.7.2 Algorithm for packet losses' target with overhead minimization

Algorithm 5 Algorithm to ensure packet losses with minimum overhead

```
1: procedure BEGIN
2:     N = $step$
3:     while $(M/N) < NetworkOverheadThreshold$ do
4:         $packetLosses$ = NetworkCodingPacketLosses (M,N,InputPacketLosses)
5:         if $packetLosses < packetLossesThreshold$ then
6:             break
7:         $M++$
8:     $actualOverhead = (M/\ N\ )$
9:     $N = 8$
10:    for $n = 1$ to $n = 4$ do
11:        $M = actualOverhead * N$
12:        if $M == 0$ then $M = 1$;
13:        $packetLosses$ = NetworkCodingPacketLosses($M, N, InputPacketLosses$)
14:        if $packetLosses < actualPacketLosses$ then
15:            $actualPacketLosses = packetLosses$
16:            saveNetworkCodingConfigurations()
17:        if $N = 64$ then
18:            break
19:        $N = N * 2$
```

Besides the fact that the previous algorithm always provides with the minimum error possible, in order to achieve that, it will return a configuration with the maximum overhead possible in most of the cases.

To solve that, it was designed an algorithm that tries to ensure a network packets losses lower than a threshold as input to the algorithm, using the minimum network overhead (M / N) possible.

First, it is attributed the step value to N variable. This step will dictate the amount of overhead incremented in each test (1/step) until the value of packets losses threshold is achieved or it was reached the defined overhead limit. To do so, it is increased the value of M coded packets, until the overhead is lower than the overhead threshold, and in each iteration it is calculated the value of the network packet losses after it is applied the network coding in the network. If that error is lower or equal to the packet loss threshold, the while cycle breaks and the value of M is stored. After this cycle, since the value of N was fixed and the M was the only to change, it is possible to know the minimum overhead to achieved a network coding packet losses lower than the threshold passed as argument to the function. Now that the value of overhead is known, the algorithm will calculate new network coding packet losses but now only with the values that match the value of the overhead calculated previously. If the value of the packet loss threshold cannot be satisfied, this algorithm will use the maximum value of the network overhead, and again calculate the network coding packet losses that match the maximum overhead. The algorithm structure is presented next, in algorithm 5:

From the analysis of the algorithm 5, it can be seen that N is initialized with the value of *step*. This way, the algorithm tests configurations with an increase of overhead of $1/step$ in each iteration until reaching the overhead threshold. Moreover, the value of N is defined to 8, in the line 9. This is done in order to check if, with the same overhead calculated in the while cycle, it is better to use configurations with small generation sizes. It can also be noticed in line 19, that the N is multiplied by two, in each iteration: it will be calculated network configurations with N equal to 8, 16, 32 and 64, and with a M value that depends on the N and the overhead calculated in the previous cycle.

5.7.3 Algorithms Vantages and Disadvantages

After a good explanation of the behavior of the two algorithms it is important to understand the vantages and disadvantages of each algorithm.

First, lets start by the algorithm 4. The main advantage of this algorithm is that it will produce always the values of N and M that assure the minimal packet loss possible. This major advantage comes with a price. First, in most of the times the algorithm runs through all possible configurations. Another normal drawback is that, most of the times, it will chose a configuration that adds the maximum overhead possible to the network.

The main advantage of algorithm 5 is that it assures the minimum error possible with the minimum overhead. Another advantage is that it is much lighter than the first one, because in the worst case scenario it will increase the overhead from 1/step until reaching the maximum value of overhead, and then only performs more four iterations in the search of the best configuration with the overhead obtained in the previous cycle.

The obvious disadvantage is that this algorithm does not assure the absolute minimum packets losses possible, and the algorithm 4 will have normally a best performance, or in the limit scenario, equal performance than the algorithm 5.

5.8 Chapter Considerations

The main challenge of the work developed in this chapter was to integrate the network coding in the multihomed vehicular network, without losing any of its features. The solution found was to use a shared memory region, where the N-PMIPv6 and the connection manager manage all the vehicular network connectivity and then pass the values of the MAC addresses to the shared memory region. Then, the Gekko program only needs to read those values and perform the routing process based on those MAC addresses through layer 2 sockets.

Another major conclusion is that it is only possible to use network coding in an multihoming scenario if, in the coder side, there are as many technologies as coding parallel processes (downlink). This happens because of the need of the coded packets to belong to a specific generation, meaning that if this coded packet, for some reason, ends up being separated from its generation, the decoder is unable to recover the lost information. In the uplink scenario, this problem was solved by creating an encoder thread and a *TAP* interface for each technology in use, meaning that all coding processes are totally independent of each other.

Finally, it were created two distinct algorithms to determine the best configuration for network coding: one dedicated to reduce packet losses and the other to satisfy a desired packet loss probability adding the minimum overhead to the network.

Concluding, it is now possible to use network coding in single and multi-hop, in uplink and downlink without majors problems. Moreover, the network coding is configured automatically based on the network demands.

Chapter 6

Evaluation

6.1 Introduction

The previous chapters explained the new features added to the VANET. Now, it is important to test and evaluate how those changes affect the vehicular network. This is performed through lab and real world experiments. This chapter will explain the details behind the performed tests, will show and discuss the results, and in the end it will present some considerations.

Section 6.2 will detail the equipment used in the lab and in the road tests.

Section 6.3 will describe the results related with the uplink manager and the disconnect/reconnect messages. This section will have two subsections, one for the lab results and an other for the real world testbeds.

Section 6.4 will describe the performed tests using the integration of SNC in the vehicular network.

Last, section 6.5 will discuss some considerations about this chapter.

6.2 Equipment

To implement a vehicular network, it is needed an OBU, that goes inside the car, and an RSU that is a fixed station placed near the road. In all the testbeds, the OBUs and RSUs can be similar pieces of equipment, but with distinct mobility roles. The OBUs behave like mMAGs and the RSUs behave like MAGs, in a mobility point of view.

These devices must have some key specifications in order to fulfill all the connectivity and processing demands. First, they must have a CPU that processes all the information and must be able to run programs that manage the network mobility (N-PMIPv6 and the connection manager) and several wireless interfaces, in order to be possible to have connectivity between OBUs and RSUs and OBUs with OBUs. It is also needed a Global Position System (GPS), in order to have a reliable information about the position of the vehicles.

The devices that fulfill all these demands are called NetRider (depicted in the figure 6.1) and have the fallowing specifications:

Single-Board Computer (SBC) : Alix3D3 Module, AMD Geode LX800 with 500Mhz of CPU frequency, x86 architecture, 59 MB of memory and 1 Ethernet interface.

Wi-Fi interface : Wi-Fi interface compliant with IEEE 802.11a/b/g/n.

WAVE interface : WAVE interface with Atheros AR5414 chipset that supports the ath5k drivers.

Cellular interface : Cellular interface.

Global Position System (GPS) : GPS GlobalTop (MediaTek MT3329);

Omnidirecional Wi-Fi antenna : Omnidirecional antenna with 5 dBi gain in 2.4 GHz frequency.

Omnidirecional WAVE antenna : Omnidirecional L-Com Antenna that has a gain of 5 dBi in frequency range of 5.850 and 5.925 Ghz.

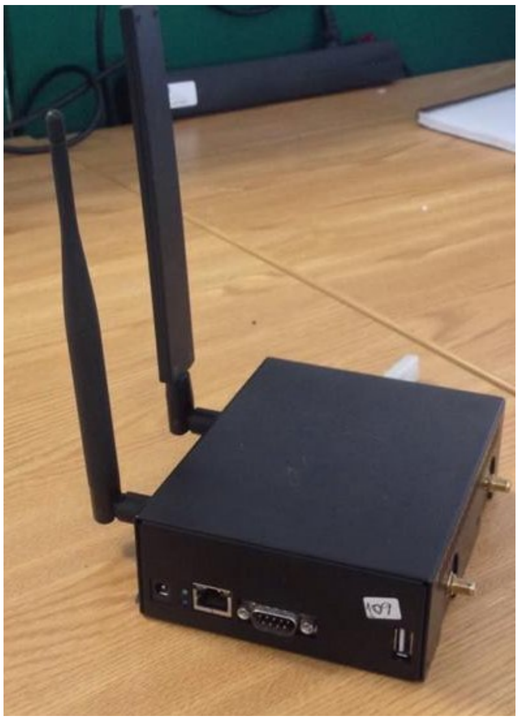


Figure 6.1: Veniam Netrider

Another entity is the LMA, that is responsible to aggregate all the mobility protocol information and find the best way to divide the downlink flows through the available connections. These aspects mean that the LMA must have a high processing power. Having this in mind, it was selected a Samsung laptop, running Ubuntu v14.04 with 196.8 GB of disk space and 2200 MHz octacore processor.

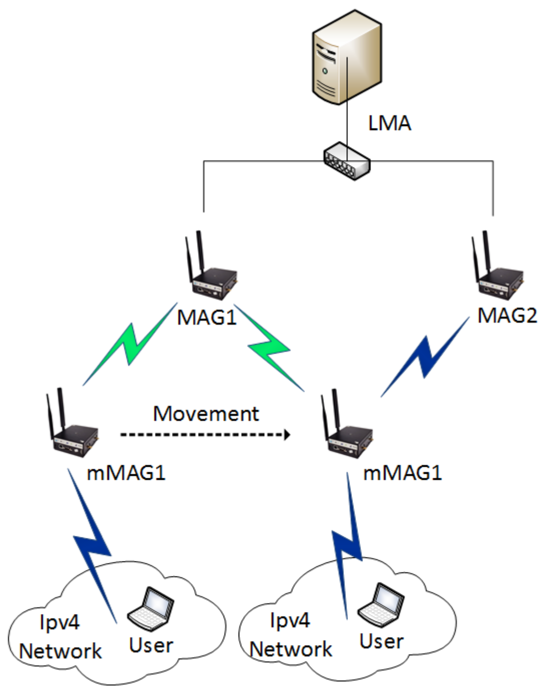
Lastly, to act as a CN it was used another NetRider when the traffic destination is another NetRider. On the other side, if the destination is a PC, it was used a HP laptop running linux Ubuntu. These different CN entities, according to the destination entity, are due to architectural differences between the entities.

6.3 Mobility and Multiohming Results

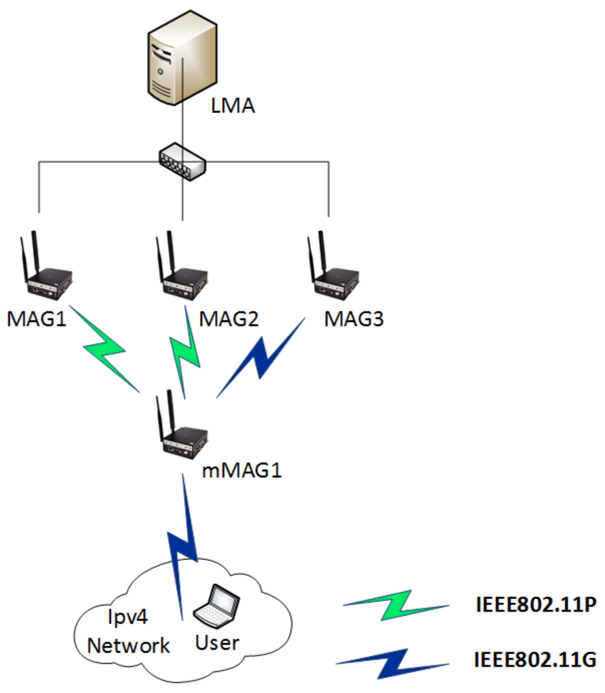
6.3.1 Laboratory Scenarios

This subsection presents the laboratory testbeds. First, it will be described the testbed to evaluate the uplink manager features, and then the testbeds used to validate the disconnect/reconnect feature.

6.3.1.1 Technical specifications



(a) Laboratory Testbed 1



(b) Laboratory Testbed 2

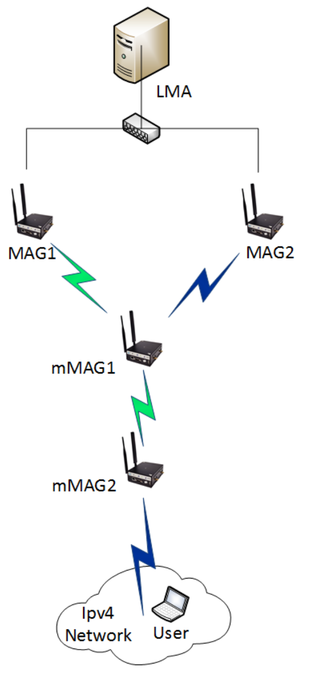
Figure 6.2: Laboratory single-hop Testbeds

Testbed 1, as the image 6.2a illustrates, is a testbed where the mMAG starts with the easiest possible configuration, i.e, a single-hop with only one Wi-Fi network available. Throughout the test, the mMAG 1 will detect a good WAVE network and will connect to that network.

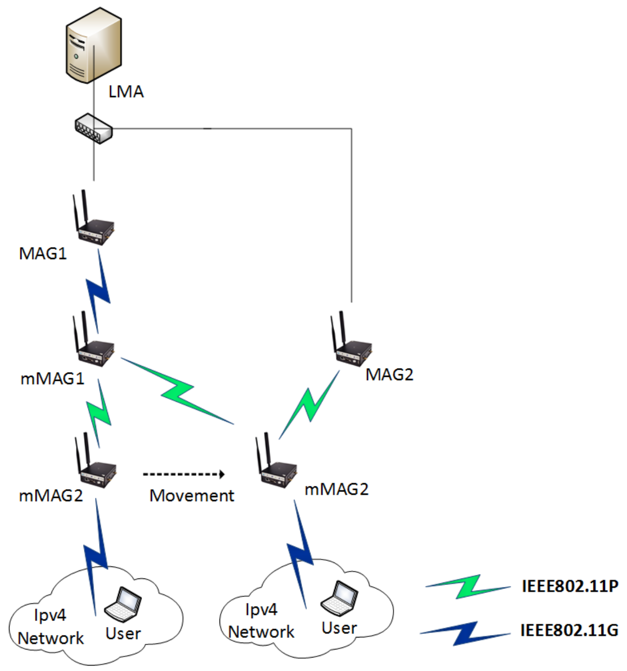
Testbed 2 is shown in figure 6.2b. As can be seen, the mMAG 1 will be connected to one Wi-Fi and two WAVE MAGs.

Testbed 3 (figure 6.3a) is a multi-hop testbed, where the single-hop mMAG will be connected to a Wi-Fi MAG and to a WAVE MAG. The multi-hop mMAG will be connected to the single-hop mMAG through WAVE technology.

Finally, in the figure 6.3b it is possible to observe testbed number 4. In the beginning, the single-hop mMAG 1 is connected to a Wi-Fi MAG and to the multi-hop mMAG 2 through WAVE. Then, the mMAG 2 will move and find a good WAVE MAG, and will connect to that network. The mMAG2 will have, simultaneously, a multi-hop and a single-hop WAVE connection available at the same time.



(a) Laboratory Testbed 3



(b) Laboratory Testbed 4

Figure 6.3: Laboratory multi-hop Testbeds

In these testbeds all MAGs are running the N-PMIPv6 protocol, and the mMAGs are running the mobility protocol and the connection manager. The user laptop connects to the VANET network through a Wi-Fi network shared by a mMAG. Last, the connection between the LMA and the MAGs is performed by Ethernet cables. Each test will last 60 seconds. Moreover, traffic with destination port equal to 3005 will simulate the users traffic, and traffic with destination port equal to 3004 will simulate the sensors traffic. Each test was repeated three times, in order to validate the results.

6.3.1.2 Experiments and results

The overall objective of these tests is to demonstrate the new functionalities introduced by the implemented uplink manager.

Test 1 : The first test aims to validate both the mobility and the traffic differentiation functionalities. The testbed of Figure 6.2a will be used and two UDP flows of 3 and 1 Mbps will be sent from the User to the LMA with a destination port of 3005 (simulating users traffic) and 3004 (simulating sensors traffic), respectively.

The best way to show the amount of traffic that is being routed through each technology is to analyze the traffic traversing each IPv6 tunnel. In Figure 6.4 the plot with the blue line (first chart) presents the throughput in the IPv6 tunnel between the Wi-Fi MAG and the LMA. The green plot (second chart) presents the throughput in the IPv6 tunnel between the WAVE MAG and the LMA. As expected, when the mMAG only has one connection available, all uplink traffic is routed through this connection. When the mMAG connects to the WAVE MAG, the traffic is separated according to the traffic type. The delay is approximately 2 milliseconds in both paths before and during the traffic split, and the average packet loss is 0.037%. These results show that it is possible to route different types of traffic through different routes, and change the routes of the traffic in multihoming, without increasing the packet losses and the delay.

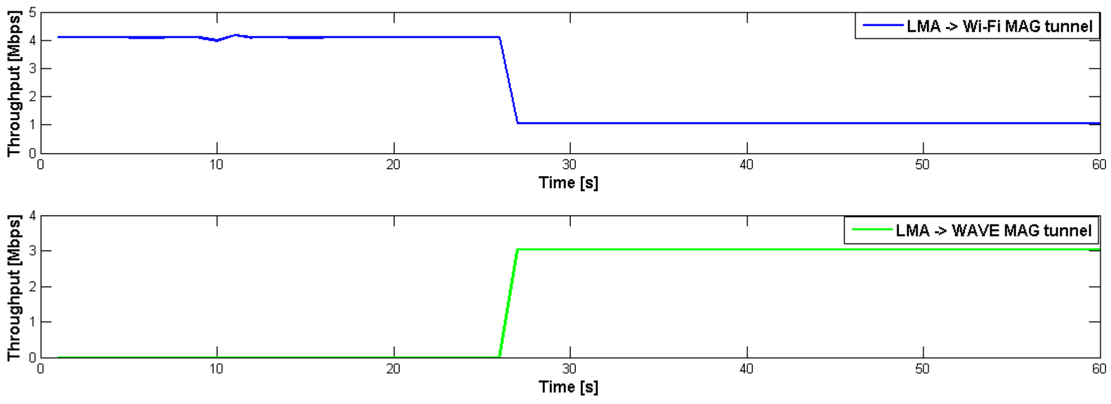


Figure 6.4: Throughput in the IPv6 tunnels - Traffic differentiation

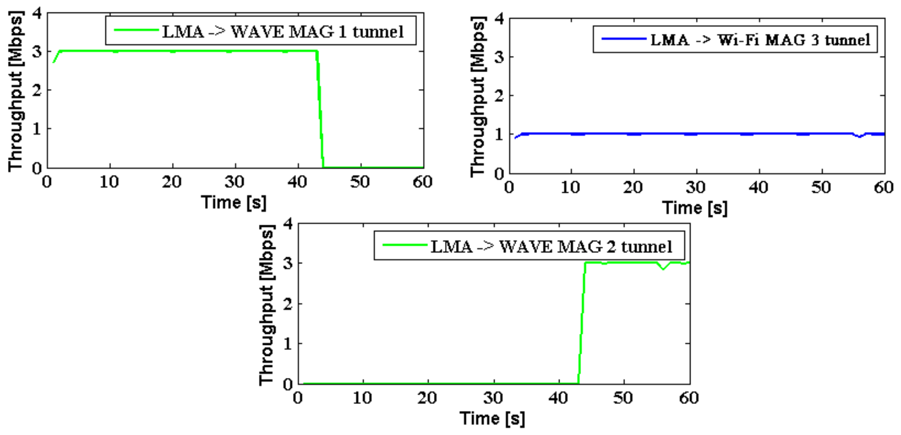


Figure 6.5: Throughput in the IPv6 tunnels - Load-balancing

Test 2 : The testbed in Figure 6.2b is used to evaluate the load-balancing mechanism. Two UDP flows of 3 and 1 Mbps will be sent from the User to the LMA with a destination port of 3005 and 3004, respectively. After approximately 41 seconds, the LMA will send 4 Mbps to the MAG 2 (MAG that was routing the mMAG uplink traffic), until the end of the experiment. Figure 6.5 illustrates the throughput in all tunnels that connect the MAGs to the LMA. The results show that, initially, the users' traffic is being routed through the WAVE connection, and the sensors traffic is routed through the Wi-Fi connection. Then, after approximately 43 seconds, the connection manager detects that the RSU responsible for routing the traffic has a load increase and decides to move the mMAG traffic to the other available WAVE RSU. During this experiment, the average packets' delay is 1.9 milliseconds, and the average packet loss is 0.012%. Again, the traffic movement from one MAG to another in multihoming did not affect the network performance.

Test 3 : The multi-hop multihoming approach is evaluated with the testbed of Figure 6.3a, where the User sends an UDP flow of 2 Mbps with a destination port of 3005 and another UDP flow of 1 Mbps with destination port of 3004. Figure 6.6 depicts the throughput of the single-hop tunnels, which shows that the throughput is maintained stable and with the envisioned traffic values even in multi-hop scenarios. The average packet's delay is 10 milliseconds (it is similar to the multi-hop cases without multihoming) and the average packet loss is 0.012%, which shows the seamless functionality of the proposed multihoming approach in multi-hop scenarios.

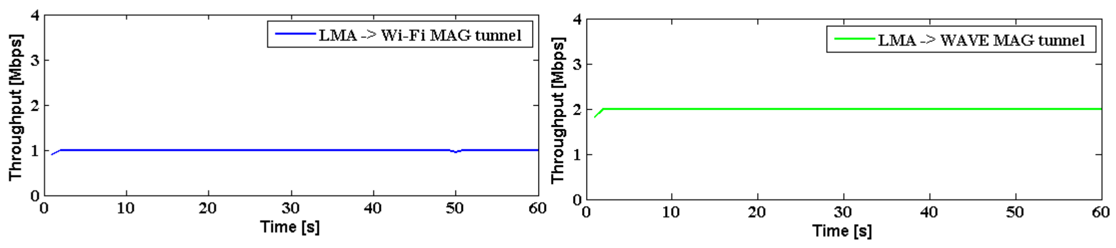


Figure 6.6: Throughput in the IPv6 tunnels - Multi-hop analysis

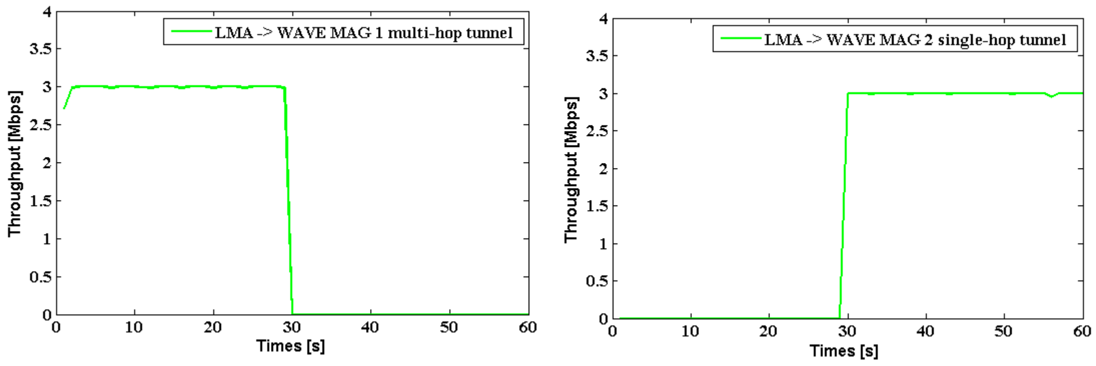
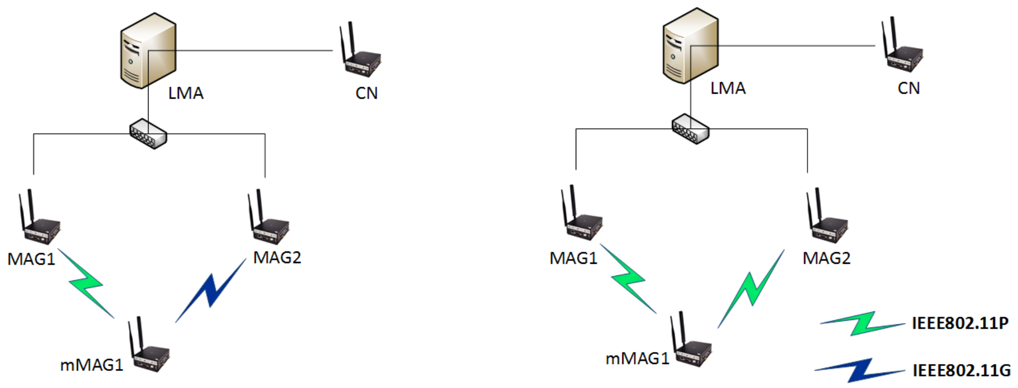


Figure 6.7: Throughput in the IPv6 tunnels - Multi-hop and single-hop connections

Test 4 : Finally, we test how the mMAG behaves when it has available both a single and a multi-hop connection using the testbed of Figure 6.3b. In this test, the User will send an UDP flow of 2 Mbps with a destination port of 3005 and an UDP flow of 1 Mbps with a destination port of 3004. After approximately 29 seconds, the mMAG will connect to a single-hop WAVE MAG.

Figure 6.7 shows the throughput of the tunnel created between the LMA and the WAVE MAG1 (left chart), and the throughput of the tunnel created between the LMA and the WAVE MAG2 (right chart). Before the time at 29 seconds all traffic is reaching the LMA through the IPv6 tunnel between the LMA and the MAG 1: the traffic is being routed by the tunnel that connects the LMA and mMAG1, as expected, because it is the only available connection. Then, the connection manager detects the presence of MAG2 and all traffic changes from LMA-WAVE MAG1 tunnel to LMA-WAVE MAG2 tunnel, since the single-hop connection has priority over multi-hop connections. During this test, the average packet loss is 0.00% and the average delay is 7 milliseconds. Again, there is no impact on the performance of the flow packets.

6.3.2 Disconnect/Reconnect Scenarios



(a) Laboratory Testbed 5

(b) Laboratory Testbed 6

Figure 6.8: Laboratory Downlink Testbeds

The overall aim of these tests is to evaluate the performance of the disconnect/reconnect message. As shown in figure 6.8, it were used two downlink testbeds in order to evaluate the disconnect and the reconnect feature. In testbed 5, the mMAG 1 is connected to a WAVE and to a Wi-Fi MAG. The LMA is connected to the MAGs and to the CN using a normal Ethernet cable. Testbed 6 is similar to testbed 5, but now the mMAG 1 will be connected to two WAVE MAGs.

6.3.2.1 Technical specifications

Both the LMA and the MAGs run the N-PMIPv6 mobility protocol and the mMAG is only running the connection manager.

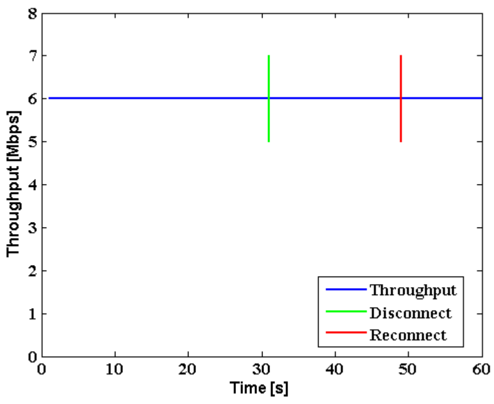
To generate and analyze the traffic, it is used the D-ITG tool [59] to obtain conclusions about the amount of packet loss and delay. Then, to analyze the results, it is used the Matlab [60] program. Moreover, in order to validate the results, all tests were repeated 3 times, but will be presented only a graph because the conclusions are the same.

6.3.2.2 Performed Tests

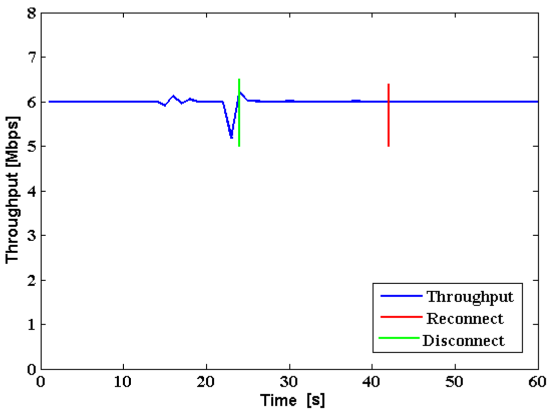
This subsection presents with detail each test.

Test 5 : Using laboratory testbed number 5, the CN sends 6 Mbps to the mMAG during 60 seconds. 30 seconds after the beginning of the test, the Wi-Fi signal quality drops, and 20 seconds later, that same connection regains the good signal quality.

As it would be expected, when the mMAG is connected to two WAVE MAGs and when a connection has poor signal quality, the mMAG sends a disconnect message to the LMA, in order to discard the bad connection, and to route all traffic to the other connection with a good signal quality. From the analysis of figure 6.9a, it is possible to conclude that this transition is done without any cost, in terms of delay or packet loss. Next, in the red line, it is shown when the reconnect is performed, and again, the traffic now goes through the two MAGs, and the transition is performed without increasing the delay or the packet losses. Concluding, these two features are a major improvement in the current network because the disconnect message removes the packet losses due to poor signal quality. Furthermore, before the reconnect message, if an OBU for some reason disconnected from a RSU, it would have to wait 20 seconds (time for the LMA to disassociate the lost connection), and repeat the association process in order for the multihoming framework to consider again that connection in the routing process.



(a) Test 5 - WAVE Disconnect/Reconnect



(b) Test 6 - Wi-Fi Disconnect/Reconnect

Figure 6.9: Reconnect tests

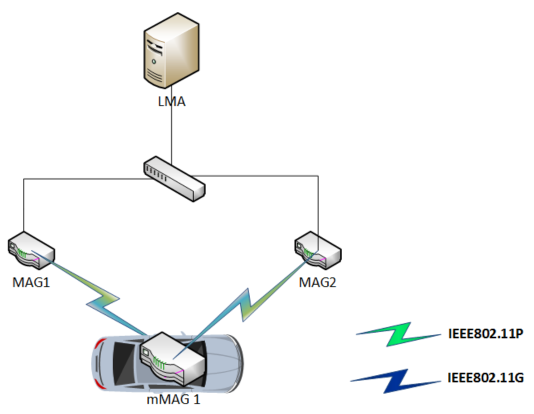
Test 6 : Using laboratory testbed number 6, the CN sends 6 Mbps to the mMAG during 60 seconds. 20 seconds after the beginning of the test, the WAVE signal quality drops, and 20 seconds later that same connection regains the good signal quality.

From the analysis of the figure 6.9b, it is possible to obtain similar conclusions as the previous test, but now with a small particularity. Before the disconnect message has been sent, it is possible to notice a drop in the bit rate. This happens because, before the disconnect is sent, it is done a Wi-Fi scan. This scan implies that the interface runs across the channels to find new Wi-Fi networks, and the throughput drops slightly. Another interesting fact is that a drop is always followed by a small pike. This happens because, while the mMAG is scanning for other networks, the signal quality decreases and the delay increases. This phenomenon will be better detailed in the real-world tests.

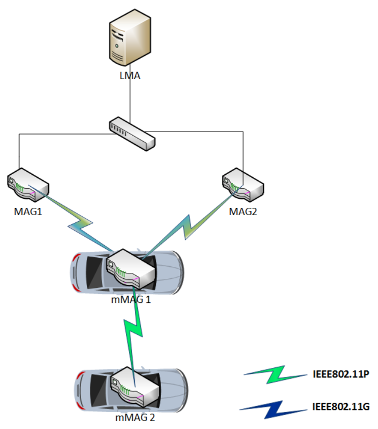
6.3.3 Real World Scenario

In order to evaluate the work done in a real vehicular environment, some real world experiments were done. All the equipment and configurations used in the previous tests remain the same, but in these tests the OBUs are placed inside a real car, and the RSUs are in the road side.

Figure 6.10 shows the testbeds that are used to evaluate the disconnect message in a real scenario. As can be seen, it were performed single and multi-hop tests using different types of technologies at the MAG 1/2.



(a) Real world testbed 1



(b) Real world testbed 2

Figure 6.10: Real world testbeds

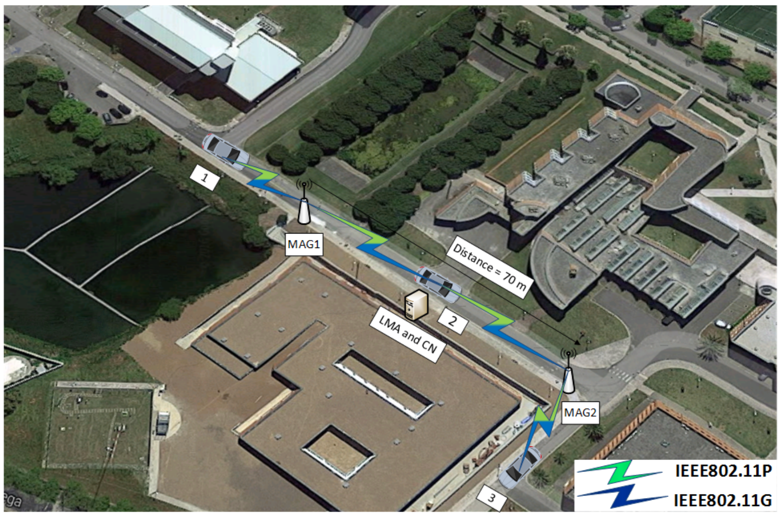


Figure 6.11: Real World testbed

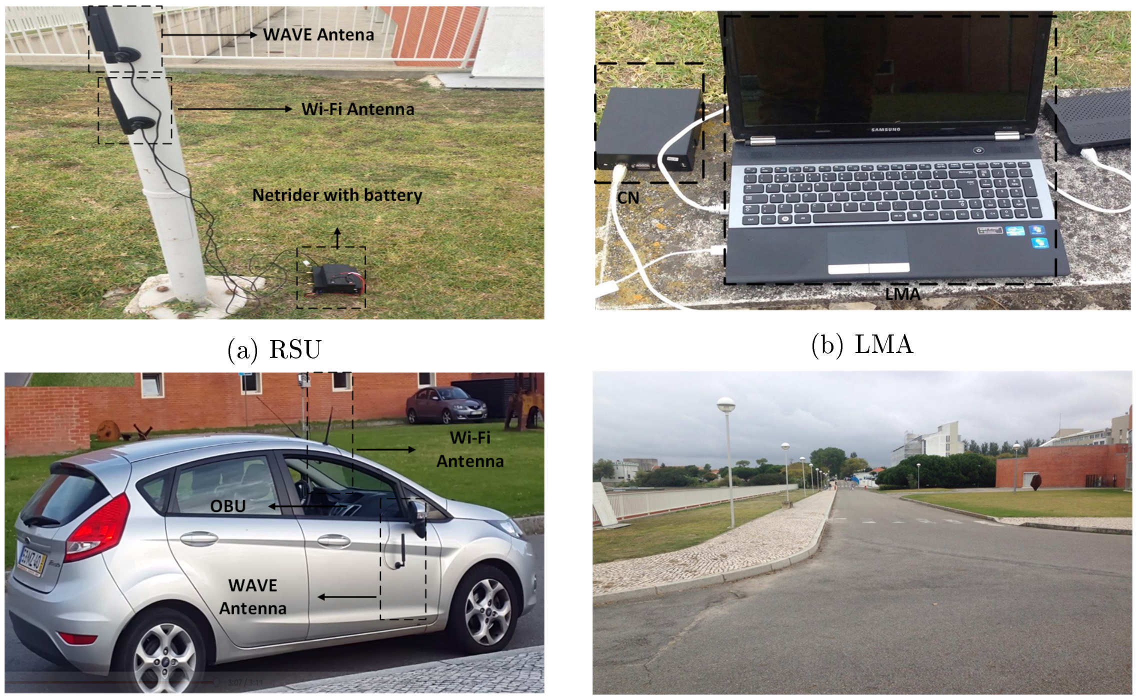
In the figure 6.11 it is illustrated the phases of the performed experiments (labeled from one to three). First, in the label 1, the car is only connected to one MAG. Then the car moves and finds a good wireless connection and performs the connection procedure, maintaining the previous connection. Finally, the car continues to move away from the MAG 1 reaching a point where the signal quality of the network shared by the MAG 1 is bad, and the connection is lost.

Figure 6.12 shows the equipment and the road used in the real world tests.

6.3.3.1 Technical specifications

In this subsection it will be described the technical specifications related with the road tests performed.

First, the distance between the MAG 1 and the MAG 2 is approximately 70 meters. To link the LMA and the MAG 2 it will be used a Wi-Fi network due to the limited length of



(a) RSU (b) LMA

(c) OBU (d) Road used to perform the tests

Figure 6.12: Pictures from the testbed implemented in real environment

---

Ethernet cables available. The connection between the MAG 1 and the LMA is performed by normal Ethernet cables, to avoid the use of a virtual interface, because the MAG 1 will be sharing a Wi-Fi network in some experiments.

The car/cars (cars in case of multi-hop) will start at 0 km/h in the position 1 and end in position 3, reaching an average speed of 20 km/h in the handover tests and a top seep of 32 km/h in the multihoming tests.

To generate traffic and obtain feedback about the delay and the amount of throughput, it is used the D-ITG [59] tool.

In these tests, it were used two different handover approaches: the unoptimized approach, where the connection to the first PoA is lost, and only then the handover is performed; and the optimized approach where, before the connection is lost, the node will connect to the other PoA, and then the handover will be performed.

The tests are repeated 3 times in order to have a good amount of samples to validate the results, but it will be presented only one set of results, because each test depends on values of signal quality, and in each test the handover/disconnect message occurs at different times resulting in different graphs, but with similar behavior.

6.3.3.2 Tests description

Since the goal is to evaluate the disconnect message, it is important to compare the performance with and without this special message. The next items will describe the performed tests:

Test 1 : Using the real world testbed 1, the MAG 1 is announcing a Wi-Fi network, and the

MAG 2 a WAVE network. The CN is sending 15 Mbps of UDP traffic, with a packet size of 1250 bytes to the mMAG, and it is used the unoptimized handover mechanism.

The aim of this test is to evaluate the performance of the first handover mechanism implemented in our group.

In the figure 6.13 it is shown the available throughput at each instant. Besides the fact that the CN is sending 15Mbps, the Wi-Fi connection can only achieve a maximum value of 6.9 Mbps. Later on, while the car moves way from the Wi-Fi network, it is performed the handover from the Wi-Fi to the WAVE network. As can be seen in the figure, this implies a loss of connectivity. Finally, while the car is connected to the WAVE network, the throughput can achieve a maximum value of 12.9 Mbps.

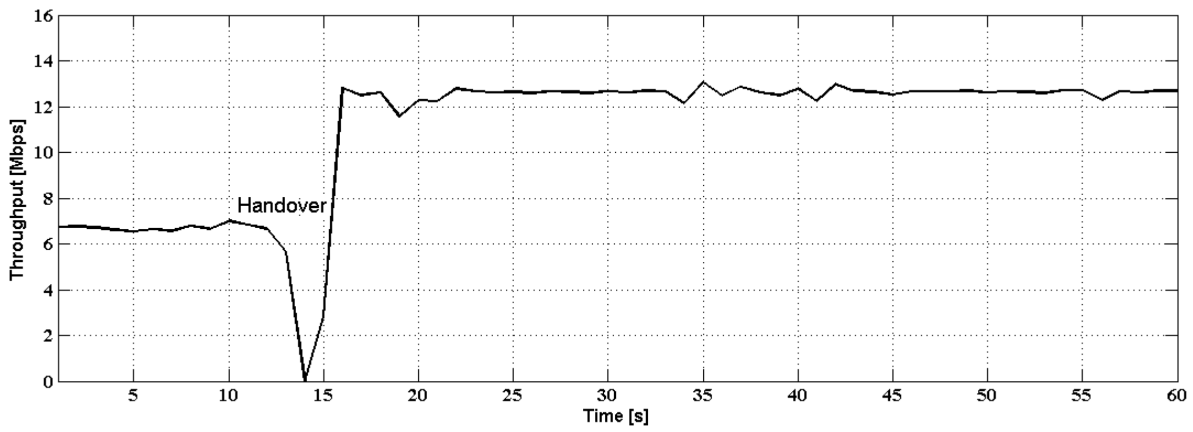


Figure 6.13: Unoptimized handover from WiFi to WAVE network

Test 2 : Using real world testbed 1, the MAG 1 is announcing a WAVE network, and the MAG 2 a Wi-Fi network. The CN is sending 15 Mbps of UDP traffic with a packet size of 1250 bytes to the mMAG, and it is used the optimized handover mechanism.

The aim of this test is to evaluate the behavior of the simplest handover mechanism, but in this case, from Wi-Fi to WAVE.

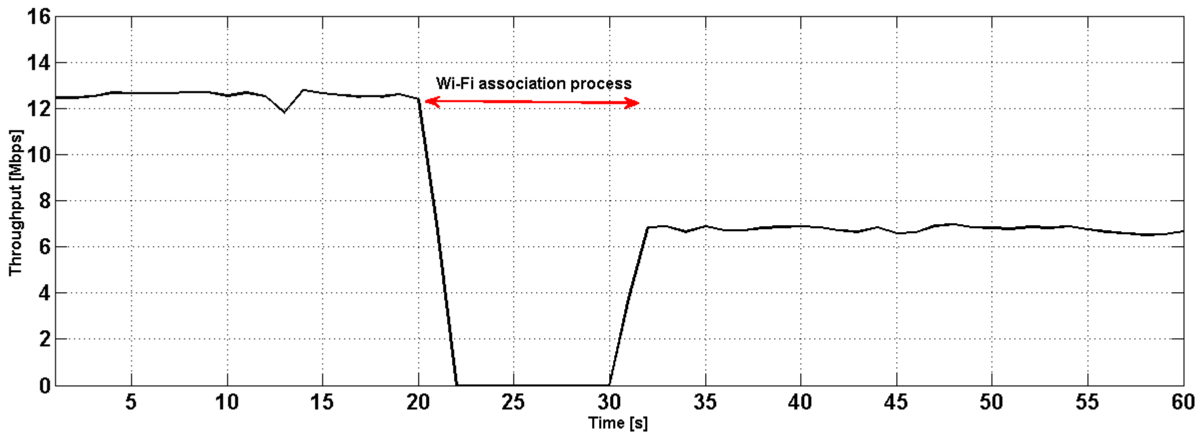


Figure 6.14: Unoptimized handover from WAVE to Wi-Fi network

As it would be expected, the throughput available in the beginning of the test is about 12.9 Mbps, the limit of the WAVE MAG. Latter on, it is performed the handover to the Wi-Fi network. In this mechanism, and as can be seen in the figure 6.14, when the handover is performed from WAVE to Wi-Fi, the mMAG looses connectivity for 7-8 seconds. This happens because the Wi-Fi association process only begins when it is detected the loss of connectivity from the original network. As can be seen, that approach implies a huge loss of performance. After the handover process was done, the throughput reaches the maximum MAG Wi-Fi value of 6.9 Mbps, as it would be expected.

Test 3 : Using real world testbed 1, the MAG 1 announces a Wi-Fi network, and the MAG 2 a WAVE network. The CN is sending 15 Mbps of UDP traffic, with a packet size of 1250 bytes to the mMAG, and it is used the optimized handover mechanism.

This test was performed in order to see the improvements that from the normal to the improved handover mechanism.

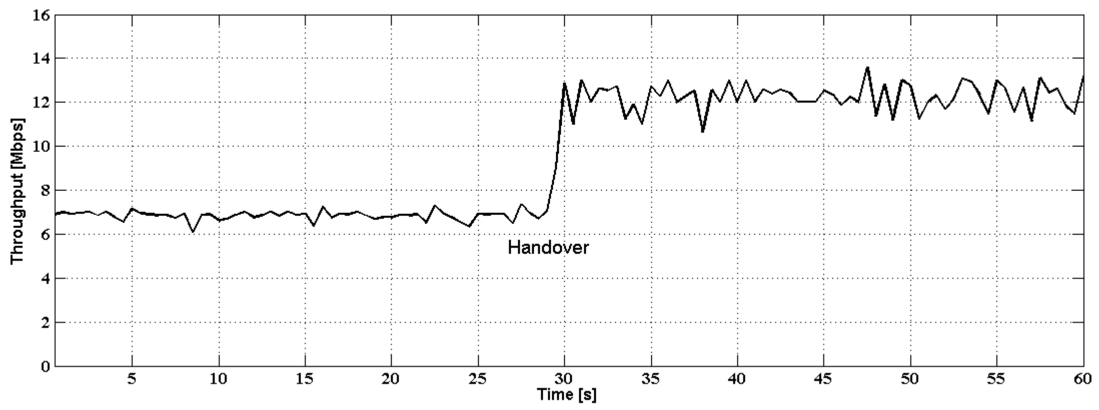


Figure 6.15: Optimized handover from Wi-Fi to WAVE network

From the analysis of the figure 6.15, it is possible to conclude that the handover procedure does not imply a momentary loss of connection anymore. To avoid the loss of connectivity, when the WAVE network is detected, the IPv6 tunnel is created between the LMA and the mMAG, and when the mMAG looses the connection to the Wi-Fi MAG, the traffic is immediately routed to the WAVE IPv6 tunnel.

Test 4 : Using real world testbed 1, the MAG 1 announces a WAVE network, and the MAG 2 a Wi-Fi network. The CN is sending 15 Mbps of UDP traffic, with a packet size of 1250 bytes to the mMAG, and it is used the optimized handover mechanism.

First the traffic is being routed through the WAVE network, with an average throughput of 12.9 Mbps, like in the test 1. However, when the handover is performed, instead of the 7-8 seconds connection loss, now the traffic is immediately sent through the Wi-Fi without any packet loss, as shown in figure 6.16. To solve the issue, when the mMAG detects the Wi-Fi connection, it will start the association process, and, in this way, when the handover is performed, the LMA routes the traffic to the IPv6 Wi-Fi tunnel.

Test 5 : Using real world testbed 1, the MAG 1 is announcing a Wi-Fi network, and the MAG 2 a WAVE network. The CN is sending 15 Mbps of UDP traffic, with a packet size of

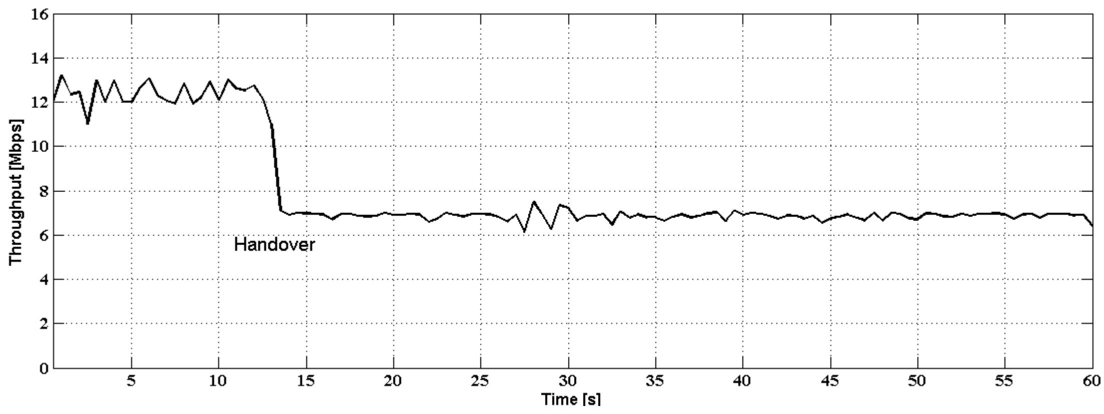


Figure 6.16: Optimized handover from WAVE to Wi-Fi network

1250 bytes to the mMAG, and it is used multihoming capabilities with the Disconnect message.

The aim of this test is to show two things. First, it shows the advantages in terms of throughput by the use of the multihoming. Second, it aims to evaluate the performance of the disconnect message in a real environment.

In the beginning, the mMAG is receiving only 6.9 Mbps due to the Wi-Fi limitation explained before. Later the mMAG, more precisely, the connection manager, finds a suitable WAVE network and performs the connection process. Now, and as can be seen in the figure 6.17, the traffic is sent by the 2 connections available, achieving a total throughput of 15 Mbps, equal to what was been sent by the CN.

Next, the car drives away from the Wi-Fi MAG and the connection manager detects that the Wi-Fi network has poor signal quality and sends the disconnect message to the LMA, that routes all the traffic to the WAVE MAG. As in the laboratory tests, it is possible to notice two peaks soon after the disconnect message is sent, and the explanation is naturally the same. The connection manager is designed to, before the disconnect message is sent, scan for other Wi-Fi networks. While in scanning mode, the interface must run through all the other channels, decreasing the link quality and increasing packet delay.

In the figure 6.18 it is shown an identical test, with the multihoming features but without the disconnect message. As can be seen, when the car looses the connection to the Wi-Fi MAG, since the LMA did not knew the connection was lost, it continues to send the traffic to the Wi-Fi MAG. Only when the IPv6 tunnel that links the LMA and the Wi-Fi MAG expires due time-out (20 seconds), the LMA recalculates the rule and sends all the traffic to the WAVE MAG. Concluding, this road test shows that multihoming improves the throughput significantly compared with the typical handover mechanisms, and the disconnect message allows an almost seamless transition from two to one networks.

Test 6: Using real world testbed 1, the MAG 1 is announcing a WAVE network, and the MAG 2 a Wi-Fi network. The CN is sending 15 Mbps of UDP traffic, with a packet size of 1250 bytes to the mMAG, and it is used multihoming capabilities with the Disconnect message.

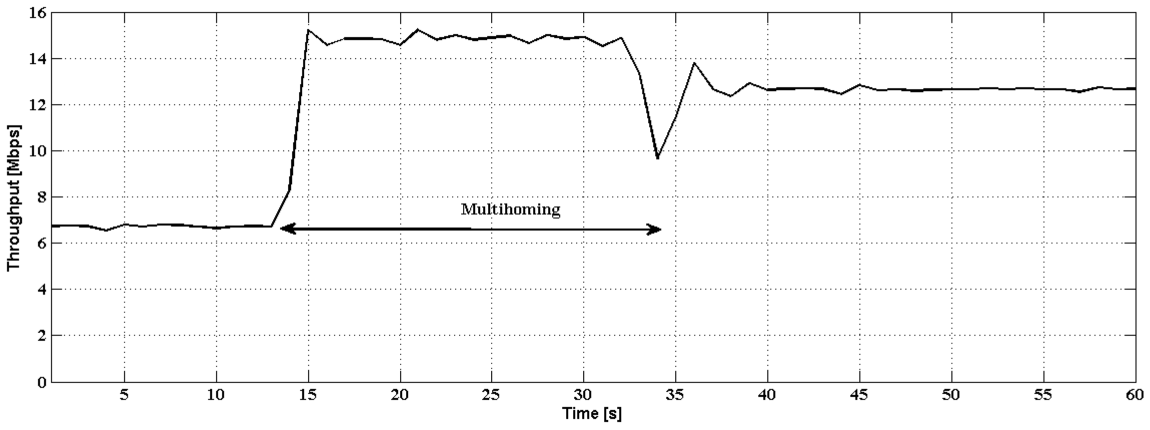


Figure 6.17: Multihoming with disconnect message in Wi-Fi

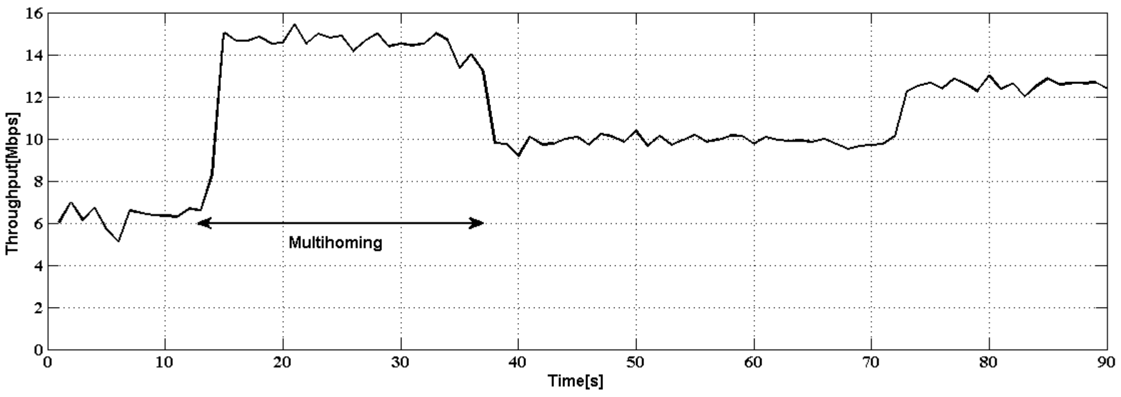


Figure 6.18: Multihoming without disconnect message

This test is performed mainly to evaluate the performance of the disconnect message when the message is sent through a WAVE network.

The behavior of this test is quite similar to the previous test (test 5), and the only difference is when the multihoming ends and the disconnect message is sent. In this scenario, it is possible to observe that the transition from two to one connection is performed without any strange pikes. As in the lab tests, when the disconnect message is sent in WAVE, it is not required the use of scans or associations processes, resulting in a seamless transition.

Test 7: Using real world testbed 2, the MAG 1 is announcing a WAVE network, the MAG 2 a Wi-Fi network and the mMAG 1 communicates with the mMAG2 through WAVE. The CN is sending 5.5 Mbps of UDP traffic, with a packet size of 1250 bytes to the mMAG and it is used multihoming capabilities with the Disconnect message.

This test is performed with two objectives. First, to show the performance of the disconnect message in a multi-hop scenario, and second to prove the explanation about the two pikes in the Wi-Fi disconnect message. In this test, it is sent only 5.5 Mbps due to the WAVE limitations. In a multi-hop scenario, as it was explained previously in this document, the WAVE medium is shared, and the mMAG 1 is using the WAVE

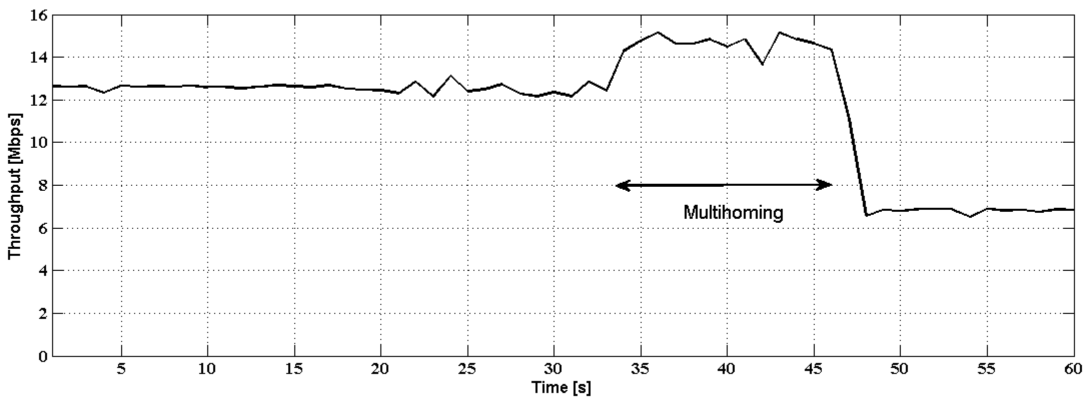
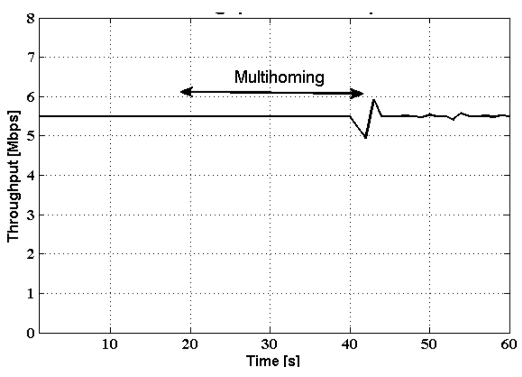
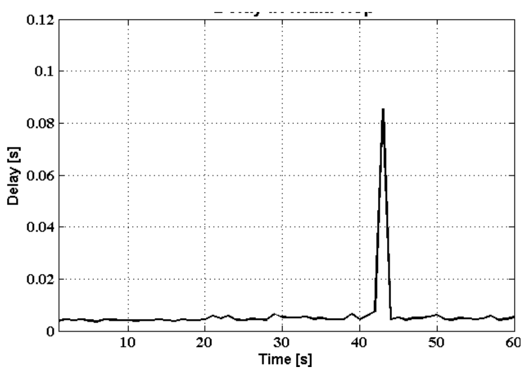


Figure 6.19: Multihoming with disconnect message in WAVE



(a) Throughput in multi-hop



(b) Delay in multi-hop

Figure 6.20: Multihoming with disconnect message, in a multi-hop scenario

interface to receive the traffic from the MAG 2, and to forward that traffic to the mMAG 2, decreasing the available bandwidth.

In the figure 6.20, it is presented the throughput and the delay in the mMAG 2. From the analysis of these two plots, it is possible to obtain the following conclusions. First, when the multihoming process is initiated, the throughput does not suffer any performance drop. Also, it is possible to conclude that the two pikes in the throughput graph are due to the pike in the delay, due to the scans done before the disconnect message is sent. Finally, it is possible to conclude that the disconnect message in single and multi-hop has a similar performance, as it would be expected.

Test 8: Using real world testbed 2, the MAG 1 is announcing a WAVE network, the MAG 2 a Wi-Fi network and the mMAG 1 communicates with the mMAG 2 through WAVE. The CN is sending 5.5 Mbps of UDP traffic, with a packet size of 1250 bytes to the mMAG, and it is used multihoming capabilities with the disconnect message.

This test is similar to the previous test, but now the disconnect message is sent through the WAVE network.

From the analysis of the figure 6.21 it is possible to conclude that, again, the disconnect message has a similar performance in terms of final throughput as when the destination

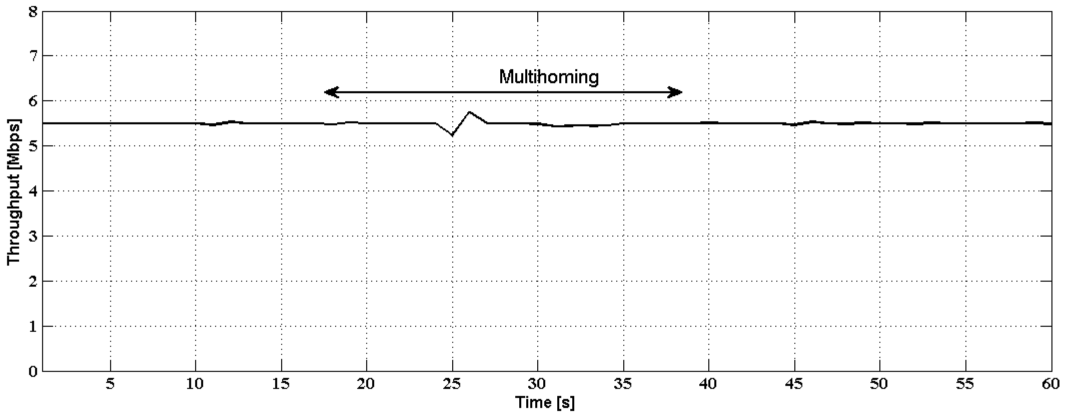
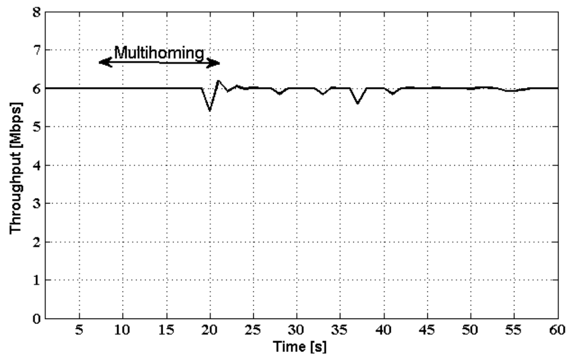
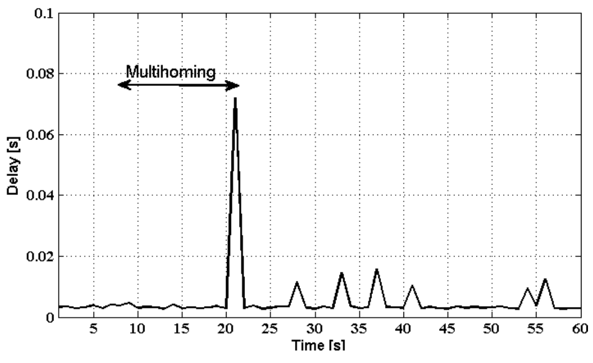


Figure 6.21: Multihoming in multi-hop with disconnect message in WAVE

is in single hop. Again, when the disconnect message is sent in WAVE there is no performance drop. Note that the 2 pikes that are presented in the plot occur, approximately, in the middle of the multihoming arrow, so are not related with the disconnect message. These pikes are probably due to a signal quality drop due to a hole in the road or an abrupt change of direction.



(a) Throughput in single-hop



(b) Delay in multi-hop

Figure 6.22: Multihoming in single-hop with Wi-Fi disconnect message

Test 9: Using real world testbed 1, the MAG 1 is announcing a Wi-Fi network, and the MAG 2 a WAVE network. The CN is sending 6 Mbps of UDP traffic, with a packet size of 1250 bytes to the mMAG, and it is used multihoming capabilities with the disconnect message.

This test is performed to show the amount of packets drops due to the Wi-Fi disconnect message. Besides that goal, it is also showed the throughput and the delay at each second of the test.

From the analysis of the three plots shown in the figures 6.22 and 6.23, it is possible to obtain several conclusions. First, and like in the previous tests where the disconnect message was routed through a Wi-Fi network, when the message is sent there is a drop of throughput. From the analysis of to the packets losses, it is possible to conclude that

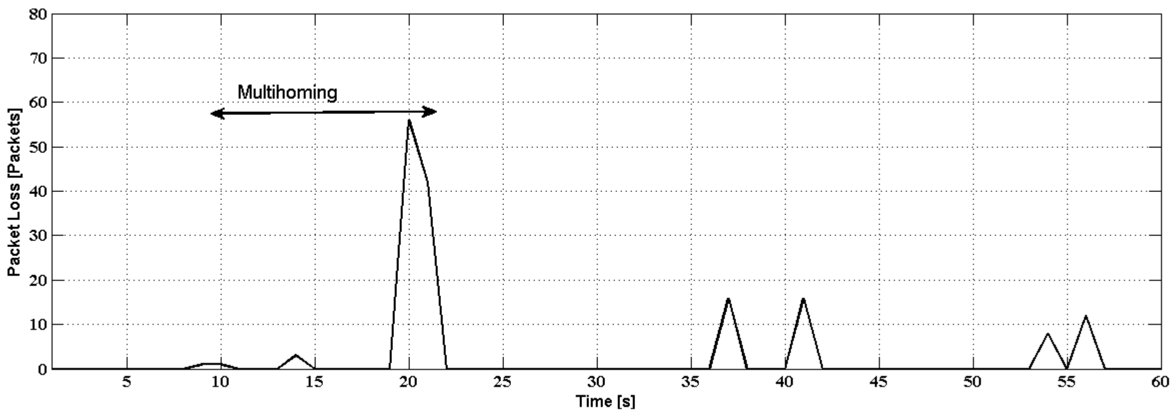


Figure 6.23: Multihoming in single-hop with Wi-Fi disconnect message

packets are lost when the scan is performed; however, the amount of packet loss is small taking into account the amount of packets that are being sent by the CN. From the analysis of the delay, it can be concluded that, when the mMAG does a scan, there is a large increase of the packets delay.

6.4 Network Coding Results

6.4.1 Laboratory Scenarios

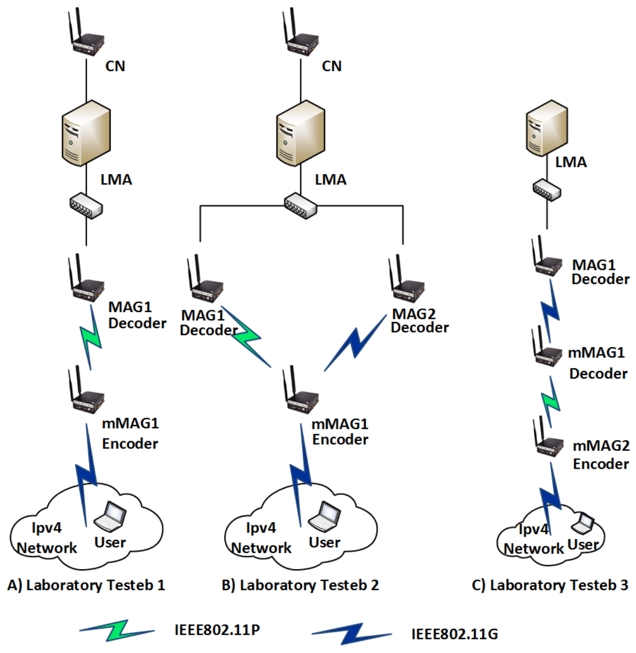


Figure 6.24: Network Coding Testbeds

The testbeds used to validate the network coding integration with multihoming, in real scenarios, are depicted in Figure 6.24. Testbed 1 is a simple single-hop scenario, where the mMAG only was has WAVE network interface available. Testbed 2 is a single-hop scenario with multihoming capabilities, and testbed 3 is a multi-hop scenario.

These testbeds are used for both downlink and uplink scenarios. In the uplink scenarios, the User will send traffic to the LMA. In downlink, the correspondent node will send traffic to the last hop mMAG, meaning that the mMAG does not need to announce any IPv4 Wi-Fi network.

6.4.1.1 Technical specifications

The connection between the LMA and the MAGs/CN are performed by normal Ethernet cables. The MAGs were running the N-PMIPv6 and the extended Gekko program, with the encoder role in case of the downlink tests, and decoder role in case of the uplink tests. The mMAGs are running the N-PMIPv6, the connection manager and the extended Gekko program as decoder in case of the downlink tests, and as encoder in case of uplink tests. Moreover, in the uplink tests, the mMAGs also share an IPv4 network for a normal user to be able to send IPv4 traffic generated by the D-ITG tool to the LMA. In downlink, the traffic is generated in the CN, also using D-ITG tool, and the destination is the mMAG itself. Moreover, the protocol used in these tests is the IPv6. Both in IPv4 and IPv6 the traffic runs through UDP with different rates.

From now on, and in order to make this document cleaner, the algorithm 1 is related to the algorithm that returns the encoder configurations that minimize the amount of packet losses, and the algorithm 2 is related to the algorithm that returns the encoder configurations to achieve a final error lower than a specific value, with the lowest overhead possible.

Each test was repeated three times, and results are shown through the mean and 95% confidence interval.

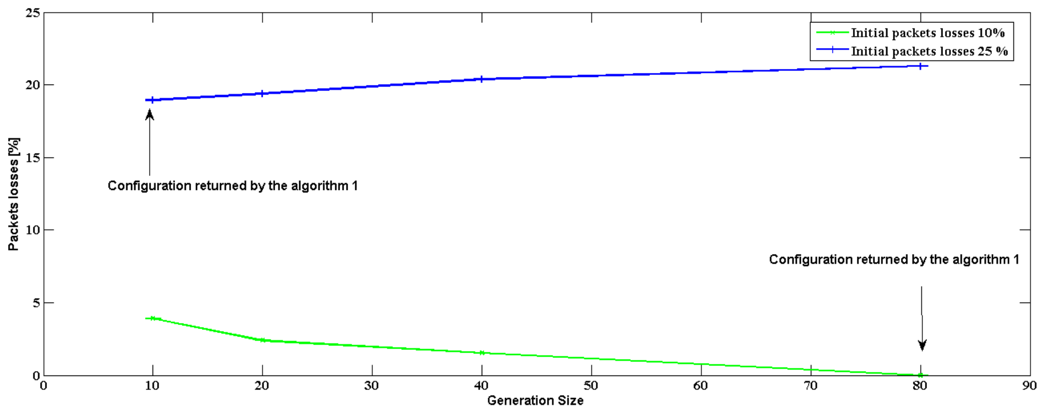
Finally, the algorithm 2 is configured to assure a final packets losses equal to 10% and with a maximum overhead of 25%, with a step value of 16 (except in test 2). Moreover, by default each test lasts for 3 minutes with a decoder window size equal to 5.

6.4.1.2 Experiments and results

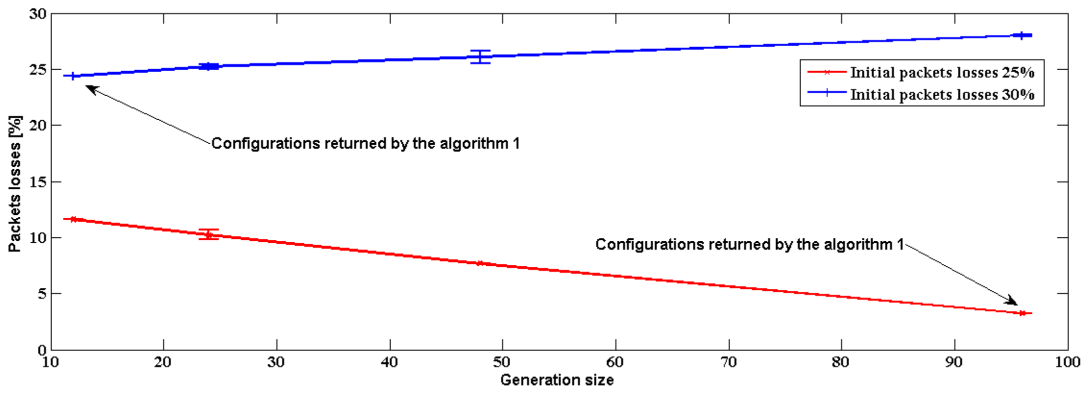
Test 1 : The aim of the first test is to evaluate how the encoder configurations affect the packet losses recovery. Moreover, this test aims to show that algorithm 1 can indeed find the best solution for packet losses recovery. This test will be performed in the testbed 1. The correspondent node sends 300Kbps in downlink, and the network coding is configured with different overheads, packet losses and generation sizes. In this test, it is used algorithm 1 for each different value of packet loss incurred in the network. The 0.1% threshold has been removed to enable the algorithm to find the best solution.

Figure 6.25 presents a comparison of packet losses recovery, when applied network coding configuration with the same amount of overhead. Figure 6.25a presents results with network coding configurations with an overhead equal to 25%, in two cases, when the network packet losses were 10 and 25%. In figure 6.25b it is presented presented results with network coding configurations with an overhead equal to 50%, with network packet losses equals to 25 and 30%.

From the results we observe that the configuration that assures the best solution for packet loss recovery is not always the same. Moreover, for a low percentage of packets



(a) Network Configurations with 25 % of overhead



(b) Network Configurations with 50 % of overhead

Figure 6.25: Packet losses recovery in Algorithm 1 depending on the encoder configurations

---

losses (when compared with the amount of overhead introduced), it is better to have a large generation size: when there is a large value of packet loss, if the generation size is small, the number of packet loss inside each generation will also be small. We also observe that the extra overhead in the network has a key impact in terms packet loss recovery: higher overhead and coding returns in lower packet loss.

Test 2 : The aim of this test is to verify if the algorithm 2 is able to find the configuration that assures the a final packets losses lesser or equal to the desired threshold, with the lower overhead. In this test it is used the laboratory testbed 2. The CN sends 300Kbps in downlink, with different overheads, packet losses and generation sizes. In this test, it is used the algorithm 2 configured with a maximum overhead of 50%, with a step equal to 32 and to achieved a final packet loss lower than 10%.

Figure 6.26 presents the results. The red bar represents the results when using the configuration returned by the algorithm 2. The blue bar represents the next possible algorithm 2 configuration (configuration with plus one step overhead from the optimal overhead found). The green bar represents the previous algorithm 2 configuration (configuration with less one step overhead from the optimal overhead found). In figure 6.26, the previous algorithm 2 configuration (configuration with less one step overhead) has in all cases a final packet loss larger than 10%, but uses the lowest overhead values.

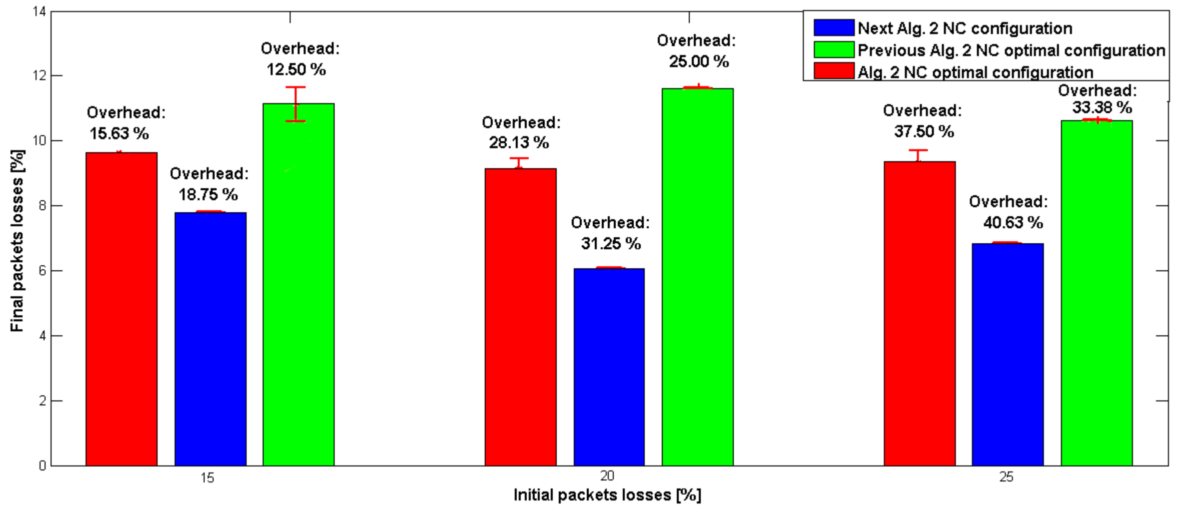


Figure 6.26: Algorithm 2 proof of concept

Table 6.1: Algorithm 2 prof of concept - configurations and iterations

p (%)	15		20		25	
	M/N	**Iterat.**	**M/N**	**Iterat.**	**M/N**	**Iterat.**
Alg. 2	5/32	9	18/64	13	24/64	16
Prev. conf	2/16	-	16/64	-	22/64	-
Next conf	12/64	-	20/64	-	26/64	-

On the other hand, it is possible to see that the next possible algorithm 2 configuration (configuration with plus one step overhead) has the best packets losses values, but naturally, it has the highest value of overhead. Finally, and analyzing the algorithm 2 optimal configurations results, algorithm 2 always achieve a final packet loss lower than 10%, with the lowest overhead.

Table 6.1 shows the configurations used in this test, and the number of iterations that the algorithm 2 needs to find the optimal configuration.

Table 6.2: SNC impact on packet loss for different uplink scenarios configuration

		Figure 6.24 (A)			Figure 6.24 (B)			Figure 6.24 (C)		
	p	**None**	**Alg. 1**	**Alg. 2**	**None**	**Alg. 1**	**Alg. 2**	**None**	**Alg. 1**	**Alg. 2**
Packets Loss (%)	**5**	5.4	0.54	2.43	5.15	0.42	2.36	5.56	0.54	2.32
	15	14.8	3.79	8.0	14.9	3.70	7.40	15.0	3.81	8.0
	25	24.6	18.6	18.7	24.7	18.4	18.4	24.9	18.6	18.6

Test 3 : The aim of this test is to verify if the network configuration affects the final value of packet loss. Using all the laboratory testbeds related with the network coding (from testbed 1 to 3) in uplink and downlink, it is sent 300Kbps of UDP traffic, in single/multi-hop and in uplink/donwlink. It is simulated a packet loss percentage of 5, 15 and 25%, and it is analyzed the final packet loss percentage without Network Coding, using

Table 6.3: SNC impact on packet loss for different downlink scenarios configuration

		Figure 6.24 (A)			Figure 6.24 (B)		
	p	**None**	**Alg. 1**	**Alg. 2**	**None**	**Alg. 1**	**Alg. 2**
Packets Loss (%)	**5**	5.15	0.42	2.36	5.39	0.70	2.33
	15	14.9	3.7	7.4	14.84	3.50	7.89
	25	24.7	18.4	18.4	24.0	17.93	18.7

algorithm 1 and using the algorithm 2.

From the analysis of the table 6.2 it is possible to conclude that the testbed configuration does not have any impact in the Network Coding results.

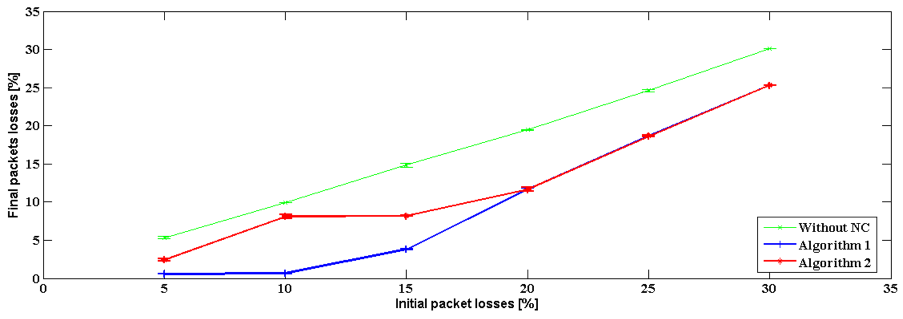
It is also possible to conclude that, as expected, the algorithm 1 will outperform the algorithm 2, in terms of packets loss. It also can be noted that, when there are 25% packet losses in the network, the two algorithms have similar results. This happens because, when the algorithm 2 is unable to satisfy the desired error, it will always try to use the configuration that ensures the minimum error possible.

Test 4 : The fourth test, again with testbed 1, aims to study the performance of each algorithm in terms of extra overhead, packet loss recovery, and the number of iterations of the algorithm to return the best N and M values. In this test, algorithm 1 uses the 0.1 % threshold as a stop condition. The User sends 300Kbps in uplink, and it is increased the simulated packet losses from 5% to 30%, with 5% interval. It is compared the final packet losses without network coding, using algorithm 1 and algorithm 2. In this test, the step value for the algorithm 2 is 16.

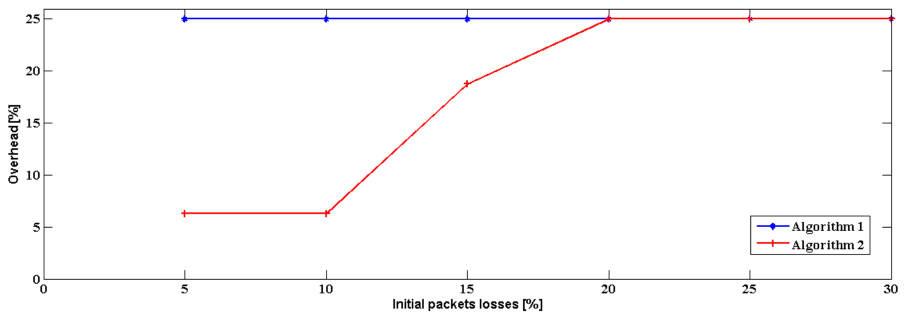
From the analysis of figure 6.27a, we observe that, effectively, network coding has a positive impact in the network packet losses. Considering algorithm 2, between the 10% and 15% of initial packets losses, the final packet losses remain virtually unchanged. The answer for this peculiar result can be seen in the figure 6.27b. Between 10% and 15% of initial packet losses, the overhead increases from 6.25 to 18.75%, but the packet losses in that interval has also increased by 5%. Moreover, it can be noticed that, when the initial packet losses increase, the distance between the red and blue lines to the green line starts to decrease. This means that the amount of packet losses recovered decrease with the increase of the network packet losses, if the amount of overhead is maintained.

We also observe that algorithm 1 outperforms algorithm 2 for values of initial packet losses lower than 20%, and then the algorithm 2 matches the algorithm 1 in terms of packet losses recovery. This happens because, when the initial packet loss reaches 20%, the algorithm 2 can no longer assure a value of final packet losses lower than 10%, and will use the maximum overhead possible. In the figure 6.27b, it is possible to see the extra overhead (M/N) introduced by each algorithm, depending on the initial packet losses percentage: the better algorithm 1 results come with a cost, since this algorithm will, in most of the times, use the maximum overhead possible.

Table 6.4 presents the network coding configurations and the number of iterations that each algorithm requires to return N and M values, depending on the initial packet losses. The number of iterations presented is the number of times that each algorithm needs to calculate the packets losses, using the expression 5.1. The results show that the algorithm 2 is lighter than the algorithm 1, as expected. This happens because, in



(a) Packet losses



(b) Overhead

Figure 6.27: Packet losses recovery and overhead depending on the encoder configurations

the worst case scenario it will increase the overhead from 1/step (1/16) until reaching the maximum value of overhead (25%), and then only performs more 4 iterations in the search for the best configuration with the overhead obtained in the previous cycle. In the algorithm 1 case, if there is no configuration that assures a packet loss lower than 0.1%, the algorithm will test every possible combination with an overhead smaller than 25%. Moreover, when comparing the results obtained in this test with the test 2 (table 6.1), it is possible to obtain more conclusions about the algorithm 2 performance. In the test 2, the algorithm was configured with a maximum overhead threshold of 50% and a step value of 32. When comparing the results from tables 6.4 and 6.1, it is possible to see that, when the step value increases, the algorithm will be able to find configurations that require less overhead; however, the number of iterations necessary to return the encoder configuration increases. Taking as example the case when there is an initial packet loss of 15%, in the test 2, the algorithm assured an error lower than 10% with an overhead of 15.63%, requiring nine iterations. The same experiment was repeated in this test, but now the algorithm returns a configuration that introduces in the network an overhead of 18.75%, requiring seven iterations. Please note that, although the overhead limit used in these tests was different, this does not affect these conclusions.

Moreover, when the overhead limit threshold is increased, the algorithm will be able to return configurations that assure a final packet loss lower than 10%. In the test 2, the algorithm was able to assure a packet loss lower than 10% with values of initial packet loss equal to 25%, and in this test, the limit was 15%.

Test 5 : The aim of this test is to compare two network coding approaches, the centralized

Table 6.4: Algorithm 1 and Algorithm 2 comparison - output and iterations

p (%)	5		10		15		20		25	
	M/N	**Iterat.**	**M/N**	**Iterat.**	**M/N**	**Iterat.**	**M/N**	**Iterat.**	**M/N**	**Iterat.**
Alg. 1	4/16	32	13/52	365	16/64	549	16/64	549	2/8	549
Alg. 2	4/64	4	1/16	4	12/64	7	16/64	7	2/8	7

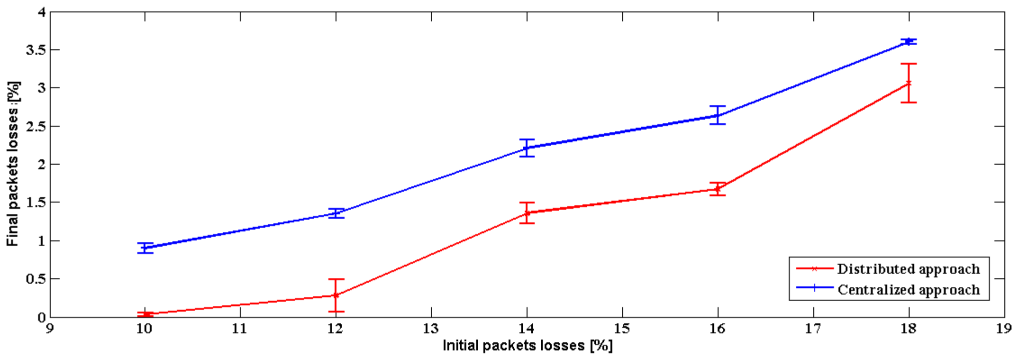
and the distributed. In the centralized approach, as the name indicates, there is a centralized node responsible for encoding the packets, based on the average of the packet loss percentage of each available network. A good example of this approach would be the LMA encoding traffic to a decoder presented in a mMAG, that is connected to two networks provided by two MAGs. The LMA, using the information about each network packet losses percentage, would use the average of those values to obtain the encoder configuration. In the distributed approach, each PoA is responsible for encoding its own packets. A good example of this approach would be the MAGs to encode their own traffic and send it to the mMAG. Using laboratory testbed 2, the user sends 300Kbps of UDP traffic to the LMA. Maintaining packet losses of 2% for the Wi-Fi connection, it was increased the packet loss of the WAVE connection, starting in 10% and ending at 16%, with 2% increment. Two results were taken, the first was using the algorithm 1 to calculate the encoder parameters with the real WAVE and Wi-Fi packet losses. In the second result it was used the same algorithm and the packet loss percentage in each connection, but now the two encoders presented in the mMAG will use the average of the sum of the percentage of packet losses of each connection.

From the analysis of the figure 6.28a, it is possible to conclude that the distributed approach always outperforms the centralized one, in terms of packets losses.

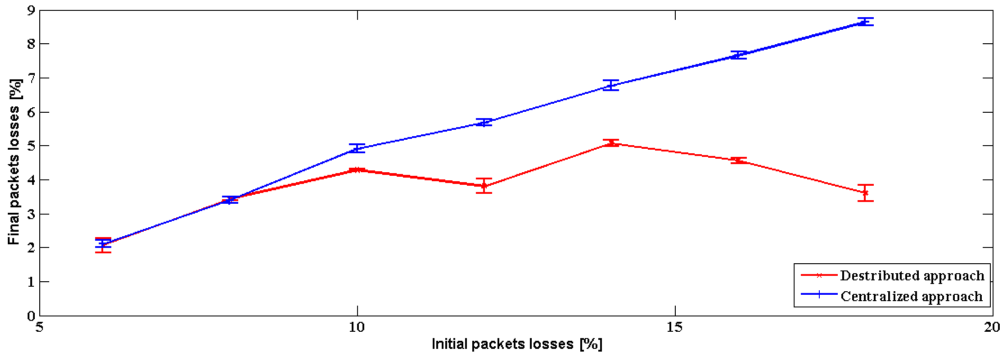
Test 6 : The aim of this test is to compare the two approaches, but now with algorithm 2. To do so, it is repeated the previous test, but now using algorithm 2.

From the analysis of the figure 6.28, it is possible to conclude that the distributed approach is better that the centralized one. It can be also noticed that the distance between the centralized line and the distributed one increases with the increase of the difference between the two connections. This happens due to the minimum overhead feature of the second algorithm. For example, if the packets loss percentage in the Wi-Fi connection is 2% and in the WAVE connection is 18%, the centralized approach considers an average of those values to calculate the encoder parameters; in other works, it will consider 10% of packets loss percentage for the two connections. In the case of the Wi-Fi connection, there is no significant problem because it is a value below 10%, but the same does not happen in the WAVE connection. The true WAVE connection packet losses percentage is 18, but what is passed to the algorithm 2 is 10%, meaning that it will select a configuration with an overhead lower than the maximum value (25%). Concluding, due to its particularities, the algorithm 2 should not be used in a distributed approach due to its poor performance. As shown in the two plots, the algorithm 1 has a better behavior in both approaches. It can be also concluded that it is always better to have a distributed approach than a centralized one.

Test 7 : The aim of this test is to understand how the windows and the generation size affect the network final delay. To do so, it is used testbed 2, with a static packet losses



(a) Algorithm 1



(b) Algorithm 2

Figure 6.28: Comparison between distributed and centralized approaches

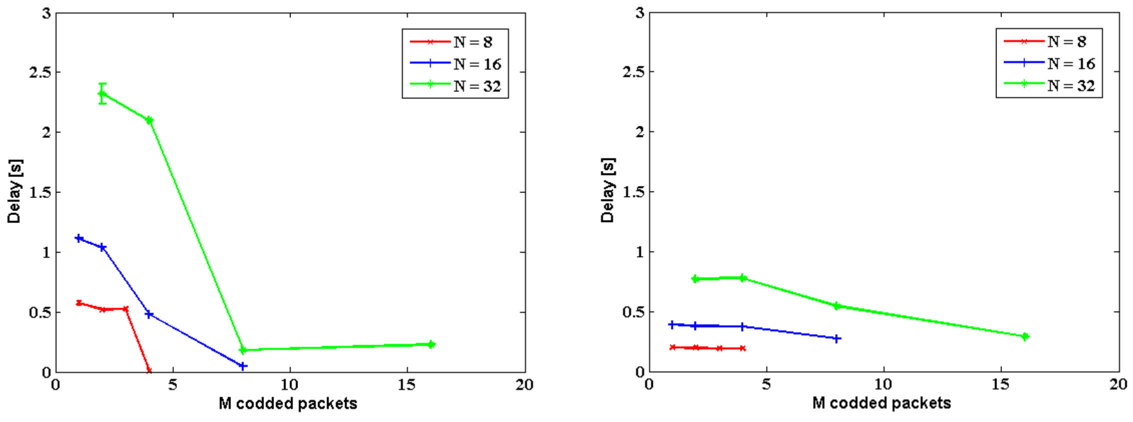
percentage of 5% and a window size of 5 generations. It is sent 300Kbps in uplink, and it is increased the normal packet sent (N) from 8 to 32, increased the coded packets of each generation to a maximum value of network overhead (M/N) of 50%.

In the figure 6.29a, it can be seen the delay with a 5 generations window and with an initial packet losses percentage of 5%. From a first look, it can be concluded that the delay is very high, specially if compared with the 2.7 milliseconds of obtained delay without network coding. The answer for these high values resides in the out of order recovery mechanism of the Gekko program. What happens is the following: when the decoder is unable to recover the lost packets (did not receive at least N packets), it will wait 5 generations until sending the packets to the decoder. The problem is that it never recovers from this situation, and now all the packets will wait 5 generations to be sent. From the analysis of the red line presented in the figure 6.29a, while the overhead is 37.5%, the delay is high, and then suddenly decreases for values closer to the scenario without network coding. This happens because when network coding uses an overhead of 50%, with an initial packet losses of 5%, the final value of packet loss is 0.00%, and therefore the out of order recovery mechanism is not used and the delay does not increase. Another point to have in mind is that these tests were performed in the lab, with the boards close to each other, reducing drastically the probability of out of order packets.

From the analysis of the green line, what was said makes even more sense. First, the delay is large due to the high values of generation size, and since the final packet drop percentage is not 0.00%, the previous effect, with high generation values will be significant. When the overhead reaches 25%, the delay decreases significantly. Now the problem is in the encoder side, because the coded packets are sent all at the same time, so the encoder must encode 8 packets and then send them to the decoder.

Test 8 : This test is similar to the previous one, but in this case the packet losses are increased to 10%, and the window size is reduced to 2.

From the analysis of the figure 6.29b, it is possible to reinforce the conclusions obtained in the previous test. First, with a lower window size the delay is decreased. It is possible, however, to notice that in the figure 6.29a, there are some samples where the delay is better than the delays presented in figure 6.29b. The explanation for this is simple: due to higher values of initial packet loss percentage (10% against 5%), the final value never reached 0%, and the out-of-order recovery mechanism is always used, delaying the information by two generations in all cases.



(a) Delay when windows size 5 (b) Delay when windows size 2

Figure 6.29: Evaluation the of impact that the windows size has in the packets delay

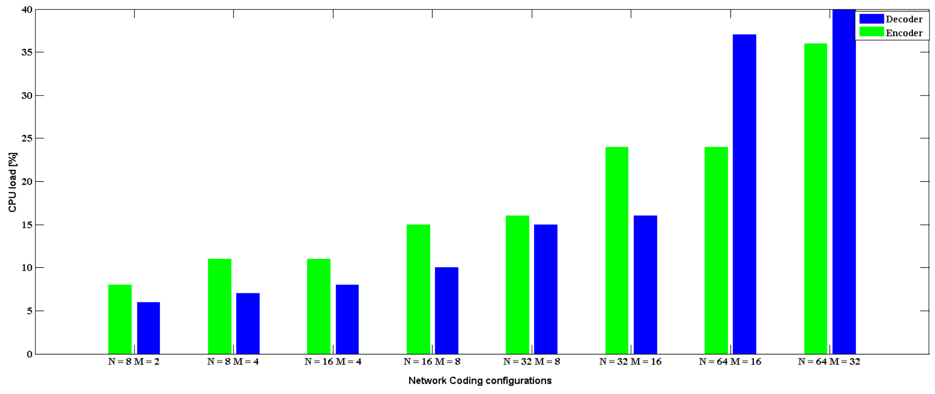
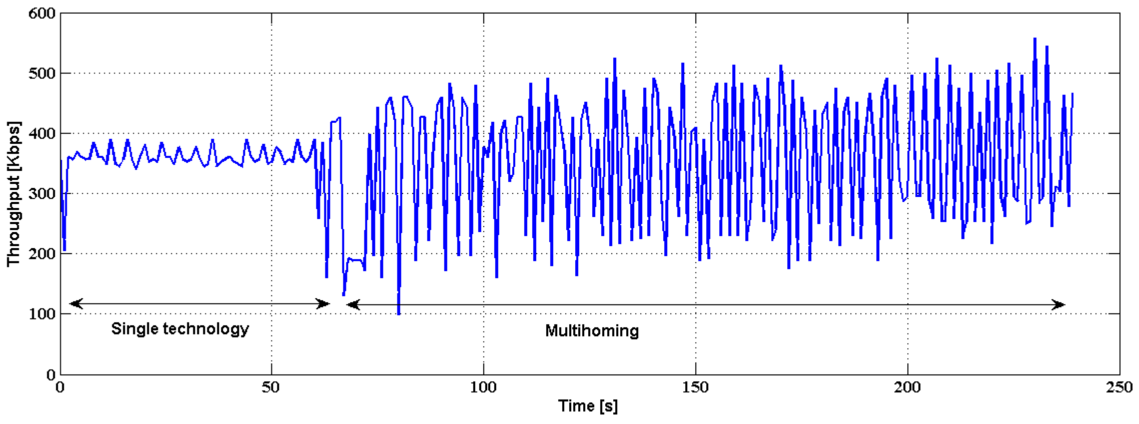


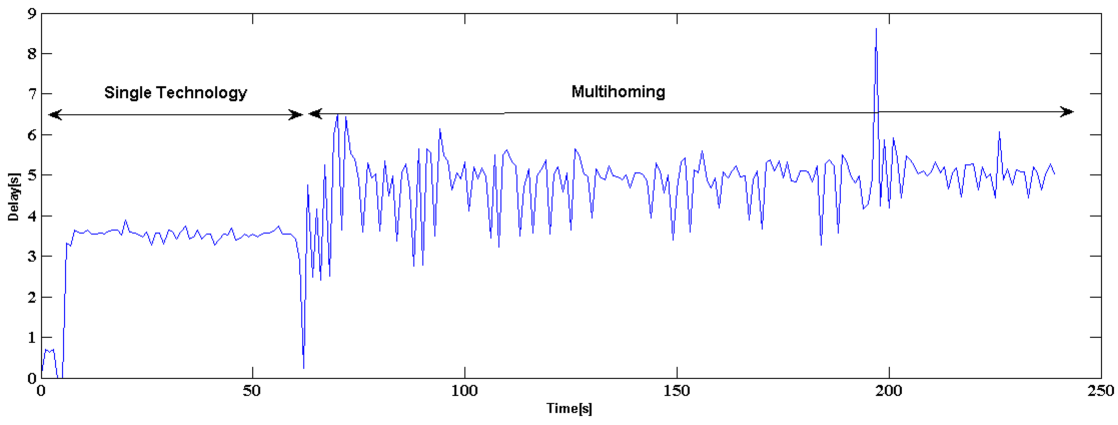
Figure 6.30: CPU load depending on the encoder configurations

Test 9 : In order to understand how the CPU usage is affected by the generation size, it is used laboratory testbed 1, in uplink. The generation size is increased from 10 to 96, and it is analyzed how the percentage of CPU usage changed. To perform this test, the encoder configurations are sent directly to the encoder, without the use of any algorithm. Moreover, it is used a window size of 2 generations, an initial packet loss percentage of 10%, and each test has a duration of 1 minute.

From the analysis of the figure 6.30, it is possible to conclude that the network coding has a major impact in the Netrider CPU usage. The major impact in the CPU is caused by the generation size, instead of the number of coded packets. This happens due to the network coding matrices operations. If the generation size increases, the size of the matrices that the CPU will have to process will increase too, increasing the CPU load.



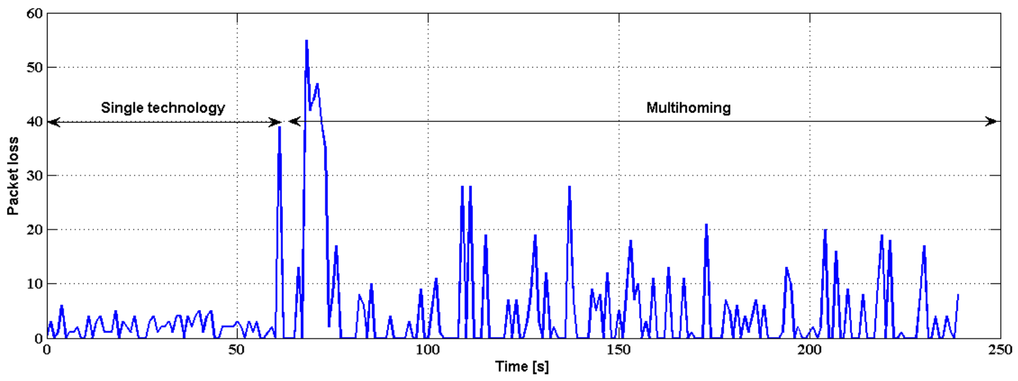
(a) Throughput



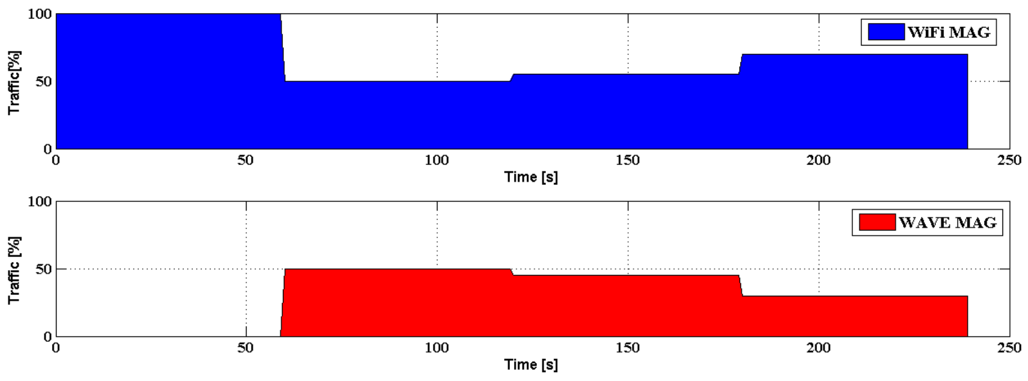
(b) Delay

Figure 6.31: Throughput and delay in mobility

Test 10 : The tenth test is performed to show that, in the network coding perspective, the mobility is seamless. Testbed 2 is used with downlink traffic, running Algorithm 1, with an initial packet loss percentage of 15% in the Wi-Fi and 12% in the WAVE link, and the correspondent node sends 350 Kbps of UDP traffic to the mMAG. First, the mMAG is connected only to the Wi-Fi MAG. A minute later, it connects to the WAVE MAG. As this test goes on, it will be increased the load of the WAVE MAG two times. To do so, the LMA will send, first 15 Kbps, and then 150 Kbps in the last 2 minutes of the test. Another important fact is that the two MAGs have the same capacity.



(a) Packet losses



(b) Traffic division through available MAGs[%]

Figure 6.32: Packet losses and traffic division

Figures 6.31 and 6.32 show the throughput, delay, packet losses and the traffic split between technologies over time. From these results, it is possible to obtain several conclusions about the network performance. First, and taking into account the throughput graph, it is possible to conclude that, when network coding is applied in multihoming, the throughput becomes unstable. This happens due to the out of order packets recovery mechanism and the need to have two decoder threads in parallel. Moreover, and because the WAVE and Wi-Fi links have different packet loss percentages, the configuration of the encoder/decoders will be naturally different. This last aspect will increase the delay, due to the different generation sizes: the traffic will arrive at the decoder with different rates, and the traffic will be sent with different delays (encoding time) to the decoder, and will be doubly exposed to the out of order recovery mechanism already explained in this document. The delay graph validates this conclusion: when the mMAG starts to use multihoming, the delay increases, in average for 50%. Although it is shown that the current network coding has poor performance in a delay point of view, it is interesting to see that the final packet loss is 3.67%, similar to cases where the initial packet loss was 15% (see figure 6.27a). In terms of packet loss recovery, it is possible to conclude that the performance is not significantly affected.

Moreover, network coding is being used while the mMAG is in movement, and with different amount of throughput in each MAG. Another important conclusion is that, for the network coding, the amount of traffic that goes through each MAG has no impact, and the key aspects are just the multihoming and the amount of packets losses.

Test 11 : The last test aims to evaluate how network coding behaves in a multi-hop scenario, where the single and multi-hop connections have different packet loss percentages, using multi-hop testbed in uplink (testbed 3). In this test it will be changed the amount of packet losses percentage in the single and multi-hop link. The user will send 300 Kbps to the LMA and the encoder will use the algorithm 1 to obtain the M and N values.

Figure 6.33 depicts the packet loss percentage in multi-hop communications. When the encoder only has available the amount of packet loss of one hop, it is better to have the highest packet loss percentage of all hops. When comparing these results with the results shown in figure 6.27a, in terms of network coding, having a multi-hop scenario where each hop has different packet loss percentage is very similar to have a single-hop connection with a packet loss percentage equal to the sum of the packet losses of all hops. Finally, it is possible to conclude that the best performance is always obtained when the encoder node knows the packet loss percentage of the two hops.

Although all previous conclusions are valid, the group of bars in the middle (5%, 15% and 20%) appears to contradict the previous conclusion. The problem here is that, in the encoder point of view, 15% or 20% of packets losses is the same in terms of N and M calculated due to the overhead and N values constrains (in the algorithm 1 the maximum value of overhead and N is 25% and 64, respectively).

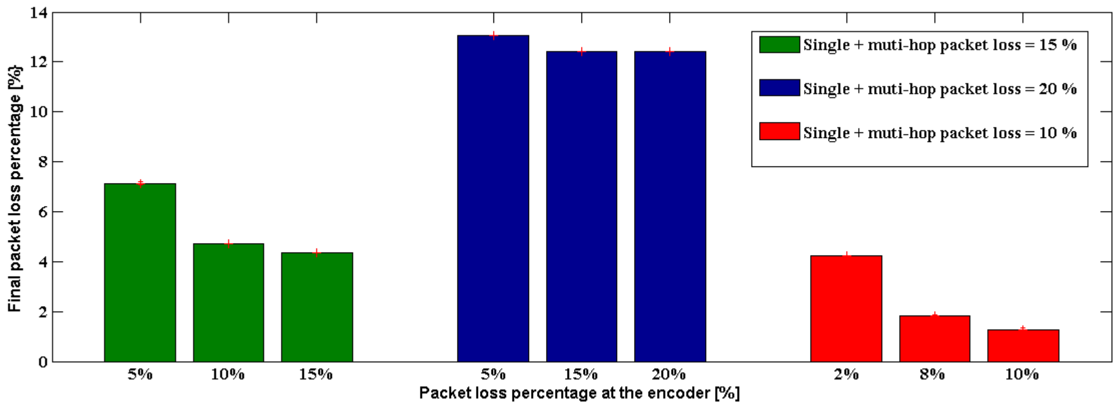


Figure 6.33: Packet loss percentage in multi-hop

6.5 Chapter Considerations

This chapter described the testbeds and tests performed in order to evaluate the work performed in this dissertation.

From the laboratory tests related with the uplink connection manager it was demonstrated that the uplink traffic was successfully differentiated and routed through the available WAVE and Wi-Fi networks. Moreover, it was also demonstrated that the load balancing system routes the uplink traffic through the access network that has the lowest load. Finally, all the developed features work in single and multi-hop.

In the laboratory tests related with the disconnect/reconnect message, it was demonstrated that the use of this message removes packet losses that would occur due to poor signal quality or loss of connectivity.

In order to compare and validate the performance of the disconnect message in a real environment, road tests were conducted. These tests have shown that the disconnect message increases the multihoming framework performance. When compared with the most optimized handover mechanism, the disconnect message alongside the multihoming provides, by far, the best solution in terms of achieved throughput to the end node.

Finally, the network coding tests have shown that the network coding can be used to reduce packet losses. Moreover, the tests also showed that the developed algorithms achieved its purpose. The algorithm 1 can find the best configuration that maximizes the packets losses recovery, but in order to achieve that, it will need high values of iterations and overhead. The algorithm 2 also fulfills its objectives, and it was demonstrated that the step value affects the amount of iterations and overhead needed in each situation. It was shown that the factor that contributes the most for increasing the delay of encoded packet was the window size. In this field, it was demonstrated that the management of this window can and should be improved, in order to reduce the delay. Finally and most important, was shown that the developed network coding supports the same features that the mobility protocol supports, i.e, mobility, multihoming in single and in multi-hop.

Chapter 7

Conclusions and Future Work

This chapter resumes all the major achievements/conclusions and points to new directions.

Section 7.1 will present an overall resume/conclusion about the work done in this dissertation.

Section 7.2 will present the improvements that can be done in order to improve the developed work.

7.1 Conclusion

In the beginning of this dissertation, there were several goals aimed to achieve. The first goal was, using an already implemented VANET, design and implement an intelligent uplink manager that takes into account the type of traffic, and the real time load of each PoA in multihoming. The second goal was to perform some improvements and finalize some gaps in the N-PMIPv6, such as the disconnect message and the multi-hop flag. Finally, the last goal was to integrate the network coding in the multihomed vehicular network, maintaining all the mobility related aspects unaffected, in order to successfully reduce packet losses due to low connections quality.

The next items resume the conclusions in the several areas:

- **Uplink Manager conclusions :** In this area, it was successfully proved that, in a first instance, the traffic is successfully differentiated based on its port, and routed according to that. Moreover, it is also shown that, if there are many WAVE connections at the same time, the uplink manager will chose the best connection available taking into account the achieved throughput of each PoA and the load that will introduce in the PoA (avoid constant jumps between PoA). Finally, it was also demonstrated that, in multi-hop the traffic is successfully managed according to what was proposed.

- **Disconnect/reconnect messages conclusion :** As it was described in this dissertation, it were created two messages to improve overall network efficiency. The results showed (real world tests) that the multihoming with disconnect message outperforms even the most optimized handover mechanism, increasing the multihoming efficiency. The reconnect message, allows an immediate reuse of a connection, that recovers from a bad signal quality, without waiting 20 seconds from the tunnel removal and repeat the association process. Moreover, it was shown, in real world tests, that these messages were also supported in a multi-hop scenario, with identical performance as the single-hop ones.

- **Network Coding conclusion :** It was shown that network coding can improve the amount of packet losses, but the amount of network overhead is increased. Moreover, another price to pay is the delay that network coding introduces, that is even more expressive in a multihoming scenario. Although the major part of the delay is due to an unoptimized out of order packet recovery mechanism, the network coding will always introduced undesired delay.

  It was demonstrated that both the algorithms developed according the specifications. It was demonstrated that algorithm 1 always reaches the minimum packet losses possible and that algorithm 2 assures an error lower than the specified in the algorithm, with the minimum overhead possible. It was also demonstrated that, due to a clever approach, the algorithm 2 is always lighter than the algorithm 1, in terms of iterations.

  It was demonstrated also that network coding is transparent for the mobility protocol. Moreover, network coding can be use in single technology, multihoming, multi-hop and in uplink or downlink scenarios.

  Finally, it was possible to see that the current version of the Netriders does not support a large generation, and that the CPU load increases exponentially with the generation size.

7.2 Future Work

This section will present some future work that can be done in order to improve the current work.

7.2.1 Mobility and Connection Manager

In the next topics it will be presented some future work that can be done, in the N-PMIPv6 and in the connection manager, in order to improve overall performance.

- **Disconnect Message** In this dissertation was improved (WAVE) and implemented (Wi-Fi) a disconnect message, but it still is possible to perform a last improvement.

  As was explained previously, when a network has poor signal quality, a disconnect message is sent through that network. Now a question may arise: does it make sent to send such important message through a network that has bad quality ? The answer is clearly no.

  An improvement that can be made is to send the disconnect message in the network that has the best signal quality and let the LMA process that message and route the traffic through the "good" networks. This improvement would mean a greater tolerance in the configured thresholds.

- **Wi-Fi disconnect scan problem :** In this dissertation it was proved that the scans before sending the disconnect message meant a momentary increase of the delay and also an increase in the amount of packet losses. To solve this problem, and to truly achieve a seamless disconnect, it would be interesting to search for new networks after the disconnect message is sent. To do so, it is necessary a change of approach in the connection manager.

7.2.2 Network Coding

The next topics will provide some future work that can be done, in the network coding, in order to improve overall performance.

- **Improve Network Codding cordination with Connection Manager :** After all work done with the integration of the network coding in the VANET, it is now possible to improve the current connection manager, that can now manage all network coding aspects. It is possible to use Wbest [61] tool, in order to know the amount of packet drop percentage of each available network, and pass the information to the Gekko program, that will find the values of N normal packet and M coded packets per generation that suit each situation. Moreover, it would be interesting that the connection manager knows when it makes sense to use the network coding, having into account all the setbacks that were described in this document (uplink scenario).

- **Improvement out of order packet recover mechanism :** As it was concluded in this work, the current network coding suffers from a delay that is not treatable in some cases. This happens, as it was explained previously in this document, due to the out of order recovery mechanism.

  A possible improvement is to improve the window management, since this is what increases the delay. The idea is, in cases the generation does not have sufficient packets, but if the next generation has the sufficient packets to decode, it would decode and send the information to the decoder. This way, the delay provoked by the generation that is waiting to receive the missing packet will not propagate the delay to the other generations.

- **Perform optimized network coding with multihoming :** Although in this dissertation it was implemented network coding in multihoming, it was done in a partial way: each technology has it own code/decode process. A future work that could be done is to do the same that is done in this dissertation but now using a single code/decode process. This would mean a significant changes at current network coding implementation.

- **Optimaze matrix operations :** While analyzing the Gekko program, more specifically the encoding/decoding process, it was noticed that all the operations related with matrices could have a better efficiency. This could decrease the amount of time spent in this operation and decrease the overall delay.

- **Study other possible application of network coding** The network coding application field is huge. In the state of the art of this work it was presented many possible applications of the network coding with different scopes. It would be interesting to use network coding not only to recover from packet losses, but also increase throughput.

Bibliography

[1] S. Al-Sultan, M. M. Al-Doori, A. H. Al-Bayatti, and H. Zedan, “A comprehensive survey on vehicular ad hoc network,” *Journal of network and computer applications*, vol. 37, pp. 380–392, 2014.

[2] D. Jiang and L. Delgrossi, “Ieee 802.11 p: Towards an international standard for wireless access in vehicular environments,” in *Vehicular Technology Conference, 2008. VTC Spring 2008. IEEE.* IEEE, 2008, pp. 2036–2040.

[3] S. Céspedes, X. Shen, and C. Lazo, “Ip mobility management for vehicular communication networks: challenges and solutions,” *IEEE Communications Magazine*, vol. 49, no. 5, pp. 187–194, 2011.

[4] A. L. Caro, J. R. Iyengar, P. D. Amer, S. Ladha, G. J. Heinz, and K. C. Shah, “Sctp: a proposed standard for robust internet data transport,” *Computer*, vol. 36, no. 11, pp. 56–63, 2003.

[5] D. Katabi, S. Katti, W. Hu, H. Rahul, and M. Medard, “On practical network coding for wireless environments,” in *Communications, 2006 International Zurich Seminar on.* IEEE, 2006, pp. 84–85.

[6] S. Ahmed and S. S. Kanhere, “Vanetcode: network coding to enhance cooperative downloading in vehicular ad-hoc networks,” in *Proceedings of the 2006 international conference on Wireless communications and mobile computing.* ACM, 2006, pp. 527–532.

[7] D. Lopes and S. Sargento, “Network mobility for vehicular networks,” in *2014 IEEE Symposium on Computers and Communications (ISCC).* IEEE, 2014, pp. 1–7.

[8] N. Capela and S. Sargento, “Machine learning for resources prediction in multihoming scenarios,” in *2015 IEEE Globecom Workshops (GC Wkshps).* IEEE, 2015, pp. 1–7.

[9] A. M. T. Martins, “Mobility in vehicular networks with multiple access points to the infrastructure,” Master’s thesis, Aveiro University, 2015.

[10] M. R. T. Oliveira, “Mobility in vehicular networks with dynamic connectivity and load balancing,” Master’s thesis, Aveiro University, 2015.

[11] T. Narten, E. Nordmark, W. Simpson, and H. Soliman, “Neighbor discovery for ip version 6 (ipv6),” Internet Requests for Comments, RFC Editor, RFC 4861, September 2007, http://www.rfc-editor.org/rfc/rfc4861.txt. [Online]. Available: http://www.rfc-editor.org/rfc/rfc4861.txt

[12] S. Gundavelli, K. Leung, V. Devarapalli, K. Chowdhury, and B. Patil, “Proxy mobile ipv6,” Internet Requests for Comments, RFC Editor, RFC 5213, August 2008, http://www.rfc-editor.org/rfc/rfc5213.txt. [Online]. Available: http://www.rfc-editor.org/rfc/rfc5213.txt

[13] N. Capela, M. Oliveira, A. Martins, F. Castro, and S. Sargento, “Multihoming for seamless handover in real vehicular networks,” 2016. [Online]. Available: http://nap.av.it.pt/mulhomingglobal.html

[14] F. Castro, N. Capela, and S. Sargento, “Multihoming for uplink communications in vehicular networks,” *submitted to Wireless Days 2017*. [Online]. Available: http://nap.av.it.pt/multihomingUplink.html

[15] ——, “Multihoming and network coding in vehicular networks,” 2016. [Online]. Available: http://nap.av.it.pt/networkCodingVanets.html

[16] S. Zeadally, R. Hunt, Y.-S. Chen, A. Irwin, and A. Hassan, “Vehicular ad hoc networks (vanets): status, results, and challenges,” *Telecommunication Systems*, vol. 50, no. 4, pp. 217–241, 2012.

[17] M. Shulman and R. Deering, “Vehicle safety communications in the united states,” in *conference on experimental safety vehicles*. Citeseer, 2007.

[18] S. Yousefi, M. S. Mousavi, and M. Fathy, “Vehicular ad hoc networks (vanets): challenges and perspectives,” in *2006 6th International Conference on ITS Telecommunications*. IEEE, 2006, pp. 761–766.

[19] F. Li and Y. Wang, “Routing in vehicular ad hoc networks: A survey,” *IEEE Vehicular technology magazine*, vol. 2, no. 2, pp. 12–22, 2007.

[20] C. Perkins, “Ip mobility support for ipv4,” Internet Requests for Comments, RFC Editor, RFC 3344, August 2002, http://www.rfc-editor.org/rfc/rfc3344.txt. [Online]. Available: http://www.rfc-editor.org/rfc/rfc3344.txt

[21] S. Gundavelli, K. Leung, V. Devarapalli, K. Chowdhury, and B. Patil, “Proxy mobile ipv6,” Internet Requests for Comments, RFC Editor, RFC 5213, August 2008, http://www.rfc-editor.org/rfc/rfc5213.txt. [Online]. Available: http://www.rfc-editor.org/rfc/rfc5213.txt

[22] V. Devarapalli, R. Wakikawa, A. Petrescu, and P. Thubert, “Network mobility (nemo) basic support protocol,” Internet Requests for Comments, RFC Editor, RFC 3963, January 2005.

[23] K. Zhu, D. Niyato, P. Wang, E. Hossain, and D. In Kim, “Mobility and handoff management in vehicular networks: a survey,” *Wireless communications and mobile computing*, vol. 11, no. 4, pp. 459–476, 2011.

[24] D. Johnson, C. Perkins, and J. Arkko, “Mobility support in ipv6,” Internet Requests for Comments, RFC Editor, RFC 3775, June 2004, http://www.rfc-editor.org/rfc/rfc3775.txt. [Online]. Available: http://www.rfc-editor.org/rfc/rfc3775.txt

[25] A. Udugama, M. U. Iqbal, U. Toseef, C. Goerg, C. Fan, and M. Schlaeger, “Evaluation of a network based mobility management protocol: Pmipv6,” in *Vehicular Technology Conference, 2009. VTC Spring 2009. IEEE 69th.* IEEE, 2009, pp. 1–5.

[26] F. Teraoka and T. Arita, “Pnemo: a network-based localized mobility management protocol for mobile networks,” in *2011 Third International Conference on Ubiquitous and Future Networks (ICUFN).* IEEE, 2011, pp. 168–173.

[27] I. Soto, C. J. Bernardos, M. Calderon, A. Banchs, and A. Azcorra, “Nemo-enabled localized mobility support for internet access in automotive scenarios,” *IEEE Communications Magazine*, vol. 47, no. 5, pp. 152–159, 2009.

[28] R. Stewart, “Stream control transmission protocol,” Internet Requests for Comments, RFC Editor, RFC 4960, September 2007, http://www.rfc-editor.org/rfc/rfc4960.txt. [Online]. Available: http://www.rfc-editor.org/rfc/rfc4960.txt

[29] J. Shi, Y. Jin, H. Huang, and D. Zhang, “Experimental performance studies of sctp in wireless access networks,” in *Communication Technology Proceedings, 2003. ICCT 2003. International Conference on*, vol. 1. IEEE, 2003, pp. 392–395.

[30] M. Atiquzzaman and W. Ivancic, “Evaluation of sctp multistreaming over wireless/satellite links,” in *Computer Communications and Networks, 2003. ICCCN 2003. Proceedings. The 12th International Conference on.* IEEE, 2003, pp. 591–594.

[31] R. Fracchia, C. Casetti, C.-F. Chiasserini, and M. Meo, “Wise: best-path selection in wireless multihoming environments,” *IEEE Transactions on Mobile Computing*, vol. 6, no. 10, pp. 1130–1141, 2007.

[32] L. Ma, F. Yu, V. C. Leung, and T. Randhawa, “A new method to support umts/wlan vertical handover using sctp,” *IEEE Wireless Communications*, vol. 11, no. 4, pp. 44–51, 2004.

[33] K. Habak, M. Youssef, and K. A. Harras, “An optimal deployable bandwidth aggregation system,” *Computer Networks*, vol. 57, no. 15, pp. 3067–3080, 2013.

[34] V. P. Kafle, E. Kamioka, and S. Yamada, “Comoroho: Cooperative mobile router-based handover scheme for long-vehicular multihomed networks,” *IEICE transactions on communications*, vol. 89, no. 10, pp. 2774–2785, 2006.

[35] N. Capela and S. Sargento, “Multihoming and network coding: A new approach to optimize the network performance,” *Computer Networks*, vol. 75, pp. 18–36, 2014.

[36] K. Andersson, C. Ahlund, B. S. Gukhool, and S. Cherkaoui, “Mobility management for highly mobile users and vehicular networks in heterogeneous environments,” in *2008 33rd IEEE Conference on Local Computer Networks (LCN).* IEEE, 2008, pp. 593–599.

[37] C. Xu, F. Zhao, J. Guan, H. Zhang, and G.-M. Muntean, “Qoe-driven user-centric vod services in urban multihomed p2p-based vehicular networks,” *IEEE Transactions on Vehicular Technology*, vol. 62, no. 5, pp. 2273–2289, 2013.

[38] R. Stewart, Q. Xie, M. Tuexen, S. Maruyama, and M. Kozuka, "Stream control transmission protocol (sctp) dynamic address reconfiguration," Internet Requests for Comments, RFC Editor, RFC 5061, September 2007.

[39] P. Nikander, A. Gurtov, and T. R. Henderson, "Host identity protocol (hip): Connectivity, mobility, multi-homing, security, and privacy over ipv4 and ipv6 networks," *IEEE Communications Surveys & Tutorials*, vol. 12, no. 2, pp. 186–204, 2010.

[40] P. Nikander, J. Ylitalo, and J. Wall, "Integrating security, mobility and multi-homing in a hip way." in *NDSS*, vol. 3, 2003, pp. 6–7.

[41] A. Dhraief and N. Montavont, "Toward mobility and multihoming unification-the shim6 protocol: A case study," in *2008 IEEE Wireless Communications and Networking Conference.* IEEE, 2008, pp. 2840–2845.

[42] E. Nordmark and M. Bagnulo, "Shim6: Level 3 multihoming shim protocol for ipv6," Internet Requests for Comments, RFC Editor, RFC 5533, June 2009, http://www.rfc-editor.org/rfc/rfc5533.txt. [Online]. Available: http://www.rfc-editor.org/rfc/rfc5533.txt

[43] A. García-Martínez, M. Bagnulo, and I. Van Beijnum, "The shim6 architecture for ipv6 multihoming," *IEEE Communications Magazine*, vol. 48, no. 9, pp. 152–157, 2010.

[44] C. Fragouli and E. Soljanin, *Network coding applications.* Now Publishers Inc, 2008.

[45] R. Ahlswede, N. Cai, S.-Y. Li, and R. W. Yeung, "Network information flow," *IEEE Transactions on information theory*, vol. 46, no. 4, pp. 1204–1216, 2000.

[46] T. Ho, R. Koetter, M. Medard, D. R. Karger, and M. Effros, "The benefits of coding over routing in a randomized setting," 2003. [Online]. Available: http://www.its.caltech.edu/~tho/

[47] C. Gkantsidis and P. R. Rodriguez, "Network coding for large scale content distribution," in *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.*, vol. 4. IEEE, 2005, pp. 2235–2245.

[48] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft, "Xors in the air: practical wireless network coding," in *ACM SIGCOMM computer communication review*, vol. 36, no. 4. ACM, 2006, pp. 243–254.

[49] M. Sathiamoorthy, A. G. Dimakis, B. Krishnamachari, and F. Bai, "Distributed storage codes reduce latency in vehicular networks," *IEEE Transactions on Mobile Computing*, vol. 13, no. 9, pp. 2016–2027, 2014.

[50] S.-Y. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE transactions on information theory*, vol. 49, no. 2, pp. 371–381, 2003.

[51] T. Ho, M. Médard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Transactions on Information Theory*, vol. 52, no. 10, pp. 4413–4430, 2006.

[52] A. S. M. Rodrigues, "Network coding for access control and loss resilience in multimedia streaming," Master's thesis, Faculdade de Ciências da Universidade do Porto, 2010.

[53] EURECOM. (2012) Openairinterface proxy mobile ipv6. [Online]. Available: http://openairinterface.eurecom.fr/openairinterface-proxy-mobile-ipv6-oai-pmipv6

[54] M. Aramoto. (2012) Umip. [Online]. Available: http://umip.org/

[55] J. Dias, A. Cardote, F. Neves, S. Sargento, and A. Oliveira, "Seamless horizontal and vertical mobility in vanet," in *Vehicular Networking Conference (VNC), 2012 IEEE*. IEEE, 2012, pp. 226–233.

[56] N. Coutinho, "Seamless integration of heterogeneous networks in multicast environments," Ph.D. dissertation, Universidade de Aveiro, 2014.

[57] J. Malinen. (2013) Hosapd. [Online]. Available: https://w1.fi/hostapd/

[58] R. Prior and A. Rodrigues, "Systematic network coding for packet loss concealment in broadcast distribution," in *The International Conference on Information Networking 2011 (ICOIN2011)*. IEEE, 2011, pp. 245–250.

[59] D-ITG. (2013) Distributed internet traffic generator. [Online]. Available: http://traffic.comics.unina.it/software/ITG/

[60] MathWorks. (2015) Matlab. [Online]. Available: http://www.mathworks.com/products/matlab/

[61] M. Li and R. K. Mark Claypool. (2008) Wbest. [Online]. Available: http://web.cs.wpi.edu/~claypool/papers/wbest/