Faculdade de Engenharia da Universidade do Porto

Virtual One-hop Ad hoc Networks

Daniel Marcos Fernandes Sousa

Thesis

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical and Computer Engineering

Dissertation under the supervision of
Prof. José António Ruela Simões Fernandes,
Departamento de Engenharia Electrotécnica e de Computadores,
Faculdade de Engenharia da Universidade do Porto

_____
(President of the Jury)

Porto, 6th March 2008

Resumo

O trabalho descrito neste documento aborda a questão de encaminhamento de tráfego em redes *ad hoc* IEEE 802.11, aplicadas a uma solução para expansão geográfica de redes infra-estruturadas. A solução proposta usa exclusivamente mecanismos de nível 2, sendo completamente transparente tanto para o utilizador como para o administrador. Este modo de operação permite criar a ilusão que todos os nós da rede *ad hoc* são adjacentes a nível IP (distância de um *hop*). Apesar disso, esta solução pode ser usada exclusivamente para encaminhamento de tráfego dentro de redes *ad hoc* embora não esteja optimizada para esse efeito. Uma vez que a referida solução cria uma topologia em árvore, única para toda a rede, e dá conexão a uma rede infra-estruturada, a optimização é feita no sentido de fornecer aos nós os caminhos mais curtos para aceder ao exterior da rede *ad hoc*. Em particular, neste documento é descrito o estudo e a implementação de um mecanismo para optimização do reencaminhamento de tráfego *broadcast*, na solução referida.

Podem ser encontradas aqui breves descrições dos mais recentes protocolos dedicados a redes *ad hoc*, sendo que a maioria ainda se encontra em desenvolvimento. É também feita uma apresentação geral do funcionamento do protocolo *One Hop Ad hoc Protocol* (OHAP), que serviu de base à realização deste trabalho. As propostas elaboradas para a resolução do problema associado ao *broadcast* são detalhadas e discutidas, apresentando-se uma análise comparativa final, com resultados a apoiar a opção por um deles em detrimento do outro. É descrita a implementação do mecanismo desenvolvido, com particular ênfase no mecanismo de detecção de duplicados associado. Finalmente, a solução é avaliada e comparada com protocolos de encaminhamento para redes *ad hoc*, *Ad hoc On-demand Distance Vector* (AODV) e *Optimized Link State Routing* (OLSR), que são referências no que diz respeito a este tipo de protocolos.

Palavras-chave: redes *ad hoc*, nível 2, *broadcast*, expansão geográfica, topologia em árvore, detecção de duplicados

Abstract

The work described in this document addresses the problem of traffic forwarding in IEEE 802.11 *ad hoc* networks, applied to a solution for geographical expansion of infrastructure networks. The proposed solution operates exclusively at layer 2 and is completely transparent for both users and administrators. This operation mode allows creating the illusion that all the nodes in the *ad hoc* network are one IP hop away from each other. Besides, that solution can be used exclusively for forwarding inside the *ad hoc* network despite it is not optimized for that purpose. Considering that the solution creates a tree topology, unique for the entire network, and that it gives connection to an infrastructure network, the optimization done provides shortest paths for the nodes to gain access to the outside of the *ad hoc* network. In particular, this text describes the study and implementation of a broadcast forwarding mechanism to optimize the mentioned solution.

This text includes brief descriptions of the most recent protocols designed for *ad hoc* networks, some of them being currently in development. An overview of the One Hop Ad hoc Protocol (OHAP), which was the starting point for this work, is also presented. The proposed solutions to handle broadcast traffic are presented, discussed and compared in detail, and the results obtained were used to justify the choice of one of them over the other. The implementation of the selected mechanism is also described, emphasizing the associated duplicate detection mechanism. Finally, the solution is evaluated and submitted to a comparison with Ad hoc On-demand Distance Vector (AODV) and Optimized Link State Routing (OLSR), two reference routing protocols in *ad hoc* networks.

Keywords: *ad hoc* networks, layer 2, broadcast, geographical expansion, tree topology, duplicate detection

To my father and my mother,
who longed for this moment
as hard as I did.

Acknowledgements

This work would have never been possible without the support of many people. I would like to thank my adviser, Prof. Dr. José Ruela, for his patience reviewing this work and for his advices full of wisdom. I am also grateful to Ricardo Duarte and Filipe Sousa, who lend me their knowledge and experience on important moments. I would like to leave a word of appreciation to Rui Campos, for its availability discussing ideas and for the material that he put to my disposition. Thanks to João Viana for being supportive and always ready to debate thoughts, ideas and troubles. Finally, and most important of all, many, many, thanks to my beloved brother Rui Sousa, my girlfriend and best friend Cátia Barbosa and my great and longtime friend Leandro Silveira, for their words and gestures along the hard path to this wonderful destination.

Table of Contents

Chapter 5

Chapter 6

Chapter 7

Index of Figures

Index of Tables

Chapter 1

1 Introduction

The present work was developed with the clear purpose of improving a solution already existent. This solution is intended to use an *ad hoc* network to geographically expand an infrastructure network, with resource to a protocol specifically developed for that purpose. However, given the inefficiency of this protocol when it comes to deal with broadcast traffic, there is the need to find an efficient mechanism to handle that type of traffic. This document describes the work performed to overcome this problem.

1.1 Goals

The goals set for the present work focus, in general, on the development of protocols and mechanisms able to provide efficient and transparent forwarding of IPv4 and IPv6 packets, over multi-hop *ad hoc* networks. Furthermore, the mechanisms when applied to these *ad hoc* networks should provide optimized paths for the communication process between the *ad hoc* nodes and the outside of the *ad hoc* network, specifically the Internet.

The specific objectives to achieve are the following:

1. The mechanisms to develop must be transparent for the upper layers, more precisely the network and application layer.
2. Broadcast, multicast and unicast traffic should be supported by the developed solution, hiding its specificities in order to give the illusion that all nodes are connected to the same link, thus complying with the previous point.
3. The developed solution should be validated as a feasible choice to expand geographically infrastructure networks, in a robust and simple way.

Since the starting point for the present work is a protocol, implemented by the author in a previous work [1], that already comprises some of the specified goals, namely the ones in points 1 and 2, the main contribution described in this thesis is its improvement by adding features that optimize the protocol operation. The main improvement is the

enhancement of the broadcast handling mechanism, in order to make the solution more efficient.

1.2 Relevant results

The proposed and validated solution shows that the original protocol benefits a great deal from the improvements introduced, in what concerns handling of broadcast traffic. It is shown that the new mechanism developed for this purpose boosts the performance of the previous solution to almost the double, in terms of bandwidth exploitation, considering communication at 1 hop distance. For greater distances the improvement is not so remarkable, due to the number of *ad hoc* nodes used for testing, but still the gains obtained are undeniable. On mobility scenarios, it is also shown that both solutions handle topology changes swiftly, in the presence of a broadcast storm. Nevertheless, the new solution presents better performance values.

In terms of real world operation, in the presence of unicast TCP and UDP traffic, both solutions show to be able to rival with Ad hoc On-demand Distance Vector (AODV) [2] and Optimized Link State Routing (OLSR) [3] protocols. AODV and OLSR are worldwide references as routing protocols for MANETs (Mobile Ad hoc Network).

1.3 Structure of the thesis

The present document is organized in seven chapters. Each chapter is divided into subsections, when deemed necessary, to give the reader a greater degree of separation between the subjects discussed.

In this first chapter, the work developed is introduced. The goals established for the work are presented along with a summary of the most relevant results and contributions.

The second chapter describes the most recent advances in what concerns routing and forwarding protocols especially designed for *ad hoc* networks. Most of the protocols described are still in development. The chapter will provide the basis to the reader in order to understand the development and validation of the present work.

Chapter 3 describes the previous work done within the scope of the solution developed. In particular, it describes the One Hop Ad hoc Protocol (OHAP) [1], a protocol

for establishing an ad hoc network as an expansion for an infrastructure network, including its particular forwarding mechanisms and general operation.

The fourth chapter describes briefly the development environment in which the developed solution was build up, referring to the tools that were used and justifying their use.

The fifth chapter is related to the actual work performed. In a first phase, two solutions are presented and discussed, based on the results obtained with simulation for each one of them. After this, it is described how the chosen solution, out of the previous two, was integrated in the OHAP protocol to improve it.

Chapter 6 is dedicated to the presentation and discussion of results. The developed solution will be compared with the original OHAP [1] and also with AODV [2] and OLSR [3] protocols. Here the reader can find plots that illustrate the performance of the protocols tested, which will provide greater insight into the analysis.

Finally, chapter 7is dedicated to conclusions. It comprises a summary of all the work developed and of the results obtained. Furthermore, it contains a brief discussion of future work that can be done to improve even more the solution here presented.

Chapter 2

2 State of the art

In this chapter the reader can get acquainted with some of the most recent protocols designed specifically for mobile ad hoc networks (MANET), some of them still in development. The protocols described will provide the necessary information to understand some of the choices and procedures performed in the course of the work described in the next chapters. Moreover, one of these protocols is described not because it was an inspiration for the work developed but rather by its similarity with the solution presented, in terms of general traffic forwarding. It is the case of Lightweight Underlay Network Ad-hoc Routing Protocol (LUNAR) [4]. The descriptions should not be regarded as substitutes of the protocols own publications, but as a simplified introduction to them.

2.1 LUNAR

The Lightweight Underlay Network Ad-hoc Routing Protocol (LUNAR) [4] protocol diverges from traditional *ad hoc* routing protocols, since it does not have mechanisms capable of route repairing, route caching, route maintenance and packet salvation. Nevertheless, according to Tschudin *et al* [4], its performance rivals with AODV's. LUNAR's goals are to explore a different approach to routing protocol schemes and making it simple, which means, having low complexity, so it could be easily implemented in many different environments. It presents a hybrid routing style, having both, reactive and pro-active features. Reactive since the protocol only discovers paths when required and pro-active because it reconstructs active paths regularly (every 3 seconds) even if their state has not changed. LUNAR is limited to a 3-hop radius area. The authors refer [4] that this decision is based on their experience with standard MANET protocols. LUNAR is located below IP, which inhibits IP stack's awareness of *ad hoc* routes and thus, prevents interaction with IP routing table benefiting the protocol. This way, it will seem that any two nodes communicating are one IP-hop away from each other. Underlying LUNAR is a SelNet network [4], which offers a demultiplexing mechanism, dispatching the incoming packet based on a "selector" - a single header field packet. While

forwarding a packet, each path traversed leg will have a different "selector", written by a routine handler inherent to SelNet network, resembling what happens in IP forwarding that rewrites the Ethernet address for each subnet crossed. A special selector value is defined for XRP (eXtensible Resolution Protocol) messages. These messages are used by SelNet, in a response and reply fashion, in order to resolve Ethernet addresses so forwarding can be possible. These messages are constructed over the standard Ethernet frames (as shown in Figure 2.1), with a new ethertype so it can be possible to differentiate SelNet frames.

Figure 2.1 - LUNAR frame format

LUNAR's *modus operandi* consists in capturing all IP control traffic and translating it to XRP messages. This way, ARP request is translated into Route Request (RREQ) and Route Reply (RREP) is translated into ARP reply. Likewise, IP broadcasts are also translated into XRP messages.

2.2 AODV

Ad hoc On-Demand Distance Vector (AODV) is MANET specific routing protocol developed by IETF, published as RFC 3561 [2]. It is based on classic distance vector protocols and belongs to the reactive protocols class. In this class of protocols, routes are calculated on demand as route requests are sent and on the instant they are sent. Then, information about destination and distance associated to it is delivered to the node willing to send data which will be used for sending purposes. This type of protocols holds the advantage of using network resources only when needed. As a disadvantage, there is the fact of flooding occurrence, possibly throughout the entire network, when a request for a route is made, thus delaying data transmission.

Before analyzing protocol operation, some useful characteristics to understanding it will be presented. At first, it is necessary to decide if there is the need to issue a route request. This implies that a node always has to verify if there is any known route to the intended recipient, prior to sending any information to it. If such route is not known, then

the node must send a route request message. The response to the route request issued, unlike it, which is sent in broadcast, is sent back in unicast. All nodes between source and destination, record routes to both as route requests and route replies are being exchanged. The present protocol requires symmetric links to function properly. Each node participating in AODV has a sequence number so that other nodes can decide about route freshness towards it. This sequence number is associated with the routing table entry that other nodes keep to reach that particular node. Moreover, it is used to ensure loop free operation. This is achieved by having all nodes verifying if the offered route to a certain destination has a sequence number at least as great as the one in cache. Only sequence numbers in these conditions are accepted as valid ensuring for one side that the route is fresh and for another, that the path is loop free, since all nodes in between have the same route information.

AODV [2] defines four types of messages in order to discover and establish routes between MANET nodes. Such route discovery process is initiated by a Route Request message (RREQ), which is the message sent by a node willing to communicate with another one. Each RREQ has a RREQ ID field, a sequence number that, along with originator address also present in RREQ, uniquely identifies it. This unique ID is used to prevent duplicates processing, since RREQ messages are sent in broadcast. A RREQ is sent whenever a route to the desired destination is unknown, has expired or is considered to be invalid. In addition, a time to live (TTL) value is associated to each RREQ, set on the TTL field of header of the IP packet carrying the message, in order to control the range of RREQ propagation. When the destination node receives this message, it must issue a Route Reply message (RREP) to the RREQ originator. In fact, if an intermediate node has a valid route to answer to the RREQ it can send a RREP on behalf of the destination node. In case this occurs, the intermediate node must send another RREP to the destination node, announcing a route to RREQ originator. RREP carries a lifetime value used to notify other nodes for how long they should consider valid the route to the destination. Should the destination be a router for a subnet, RREP offers the possibility to announce it to other nodes in a special field for that purpose. This field indicates the subnet's prefix size based on the destination node address, allowing other nodes to know that any destination with that prefix is reachable through that node. Upon receiving RREP, a Route Reply Acknowledgement (RREP-ACK) is sent to destination node, acknowledging that a route has been established. Should a route break at any point of its extent, a Route Error message (RERR) should be sent to inform the link breakage and destinations no longer accessible

through the node sending the RERR. RERR should also be sent in the case of receiving either a data packet, for a destination to which there is no active route or a RERR from a neighbor, associated to one or more active routes. RERR can be delivered either in broadcast, if the number of nodes to warn is high, or in unicast, if there is only one node to inform. If a small number of nodes are to be advised and broadcast is unsuitable, iterative unicast may also be used.

For protocol operation explanation, consider Figure 2.2 and assume that the network represented is stable, meaning that no node moves enough to provoke link breakage.

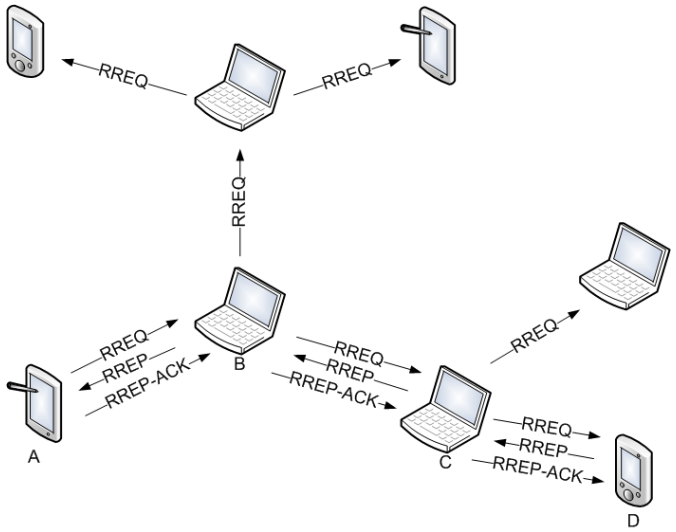

Figure 2.2 - Hypothetical AODV network establishing a route

In these conditions, assume that node A wishes to communicate with node D but has no route to reach it. To acquire a route, node A issues a RREQ which is divulged to the entire network. Meanwhile, every node stores a route to node A upon RREQ receiving. When RREQ reaches node D or an intermediate node with a "fresh" route to it, originates a RREP to be sent in unicast to node A. The reason why unicast reply is possible is the fact that all nodes between source and destination nodes recorded a route back to the source, as explained above. Exactly as happened with RREQ, as RREP follows to node A all nodes in the path record a route to destination node (node D). At this point there is already a route established between nodes A and D. But before communication can take place, node A must acknowledge RREP receipt. After node D receives the RREP-ACK, traffic flows normally between the two nodes. Now, assume node C moves away from node B, enough to break the link between them or at least, to lose symmetry. When this fault is detected, it

will cause a RRER to be issued by nodes B and C. RERR will be then, disseminated notifying the other nodes about the link failure and as a result, route loss to destinations accessed through those nodes.

AODV [2] only becomes active when it is necessary to find a route or to inform about link crashing. In normal operation, as long as there are active routes, AODV plays no role in forwarding traffic.

2.3 PacketBB

Generalized MANET Packet/Message Format (packetBB) [5] is a general purpose multi-message packet format specification currently in development at IETF, especially designed for information exchange between MANET routers. The specification describes the packet format, permitting zero or more messages to be held within it. A packet with no messages in it might be useful if the only information to send can be carried in the packet header. The messages format is also described. Detailing a bit, each one is constituted by a header and a body, where the header provides itself enough information to make forwarding decisions, avoiding the need of inspecting and processing the message body. The body contains attributes associated with the message or its sender and blocks of addresses with associated attributes. The format of these blocks is defined in such a way that it permits to represent them in a compressed form. The attributes follow a generic type-length-value (TLV) format and a number of attributes may be used to characterize a packet, a message, an address or a set of addresses.

There are several advantages associated with this specification. It is possible to extend packets and messages defined by protocols that use the packetBB [5] specification, by merely defining new message types. Also, existing protocols that use this specification are capable of identifying the new messages and TLVs and skip them, processing only known messages and TLVs. Common bit sequences in addresses contained in the address block are used to compress their representation. Message header information may be used to separate messages forwarding and processing, so that no message body analysis is required. The packet format is illustrated in Figure 2.3

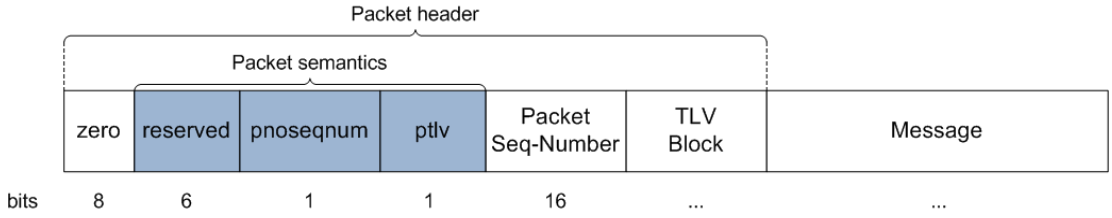

Figure 2.3 - PacketBB packet format

Zero is a one octet field, with all bits set to zero (0), which indicates if a packet starts with a packet header. Two bits (the righmost) of the packet-semantics field are used to define whether the packet header has the packet-seq-number and tlv-block fields. The remaining 6 bits are reserved and must be set to zero (0).

The message format is illustrated in Figure 2.4

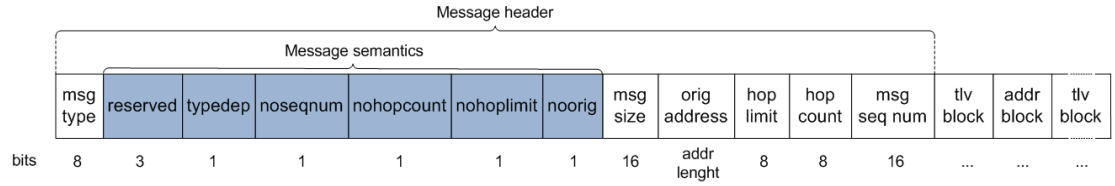

Figure 2.4 - PacketBB message format

Msg-semantics contains 5 bits (the rightmost) which define whether or not the fields in the message header information are included in the message header along with other information related to these fields. The remaining 3 bits are reserved and must be set to zero (0). The orig-address contains the address of the node that generated the message. The msg-seq-number can be type dependent or independent and that dependency relation is specified in the msg-semantics field.

The address block format is illustrated in Figure 2.5

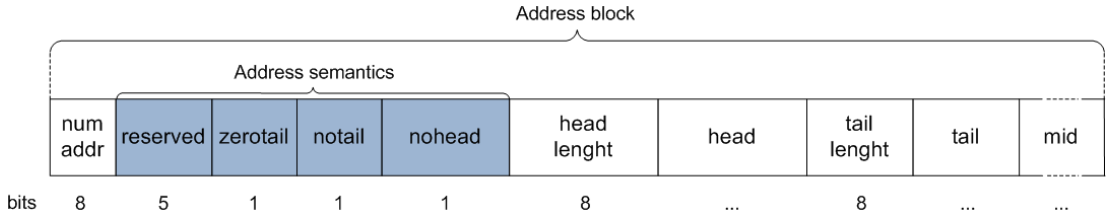

Figure 2.5 - PacketBB address block format

Num-addr indicates how many addresses are represented in the address block. Addr-semantics uses 3 bits (the rightmost) to indicate if there is any head or tail fields. Mid field(s) contains the remaining address bytes that do not belong to either head or tail.

This address representation allows information compression, since addresses sharing the same head and tail do not need to be completely written in the address block. It is only required, in this case, that head and tail are written once and the individual octets from each address are written afterwards in mid fields. If there are no addresses in these conditions, all address octets are sent in mid fields, with neither head nor tail fields.

The TLV block format is illustrated in Figure 2.6

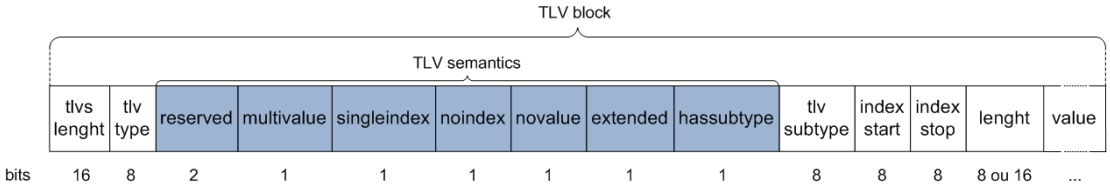

Figure 2.6 - PacketBB TLV block format

Tlv-semantics makes use of 6 bits (the rightmost) to define the presence of subsequent fields. Tlv-subtype indicates a particular subtype specific to the tlv-type. Index-start and index-stop are intended to give an offset position of the information carried. Length, if present, can be an 8 or 16 bits field depending on information in tlv-semantics, and carries the value field length. Value is the actual information to be exchanged.

The specification allows padding to be made, to ensure 32 bit alignment. Octets set to zero (0) must be used to do this.

2.4 NHDP

MANET Neighborhood Discovery Protocol (NHDP) [6] is a neighborhood discovery protocol especially designed for MANETs (Mobile Ad hoc Networks), actually being developed by IETF. Through the exchange of HELLO messages, each node gets to know, up to 2-hops away, local topology information. This information is kept in an Interface Information Base (IIB) and in a Node Information Base (NIB), which are made available to other protocols. NHDP [6] nodes, use local (1-hop) signaling to advertise themselves and their interfaces, as well as to discover links from contiguous nodes, verify

if they are symmetric and, in addition, advertise them as a mean of becoming aware of symmetrical 2-hop neighbors. Besides this, each node uses these messages to maintain its Information Bases up to date.

A Local Information Base (LIB), present in each node but not modified by NHDP, holds IP addresses of all interfaces. NIB is responsible for keeping the Neighbor Set, constituted by Neighbor tuples that records all interface addresses of 1-hop neighbors. Information concerning the node's current and lately lost symmetric 1-hop neighbors is also kept in the NIB. A link must exist between two nodes having each one a tuple in the corresponding NIB associated to the other one. While a corresponding Link tuple exists in the Link set (discussed in the next paragraph), a Neighbor tuple should be maintained in the Neighbor Set. Another NIB obligation is to record, in Lost Neighbor tuples, an interface for each recently lost symmetric 1-hop neighbor. These tuples form the Lost Neighbor Set, which is used to swiftly advertise the addresses of lost links to other nodes, so they can remove these addresses from the 2-hop Sets.

Additionally each node keeps an IIB for each one of its interfaces. Information regarding interfaces' links to adjacent neighbors and 2-hop neighbors reachable through those is preserved in IIBs. Detailing a little bit more, IIB consists in several tuples, called Link Sets, each of which records information regarding interfaces' current and recently lost 1-hop neighbors. Recently lost links are kept with the purpose of being advertised in HELLO messages, permitting faster link removal from 1-hop neighbors' Link Sets that do not matter anymore. A 2-hop Set is maintained also and it holds 2-hop tuples which record a symmetrical 1-hop neighbor interface and an address of a symmetric 2-hop neighbor, reachable via the 1-hop neighbor. Both nodes recorded in a tuple must have a symmetrical link between them.

Signaling in NHDP is undertaken simply by exchanging HELLO messages. Such messages are generated by a node for each of its interfaces, with the purpose of identifying the interfaces and reporting other interfaces of the node. Furthermore, it conveys all Link Set information from IIB and all the information in the NIB. Appended to this, a validity time, for which the information should be considered valid, and an interval time, indicating the regularity of HELLOs, should be included in the messages. HELLOs are exchanged pro-actively and reactively. Pro-activity means that messages are sent at regular intervals, while reactivity is related to the possibility of a node generating HELLOs due to changes in its 1-hop neighborhood or even in itself. The message format used by this protocol is defined in packetBB [5], explained previously (section 2.3).

As said earlier in this text, LIB should not be changed by NHDP. However, it is NHDP responsibility to alter its IIB and NIB, as changes occur in the LIB. These changes should be expected when an interface is added (removed) to (from) a node or when an address is added (removed) to (from) an interface.

2.5 MMARP

The Multicast MANET Routing Protocol (MMARP) [7] is a multicast routing protocol especially designed for MANETs. This protocol has capabilities to handle IGMP so that fixed IP networks can be supported and, therefore, allowing multicast routing between ad hoc nodes and standard IP nodes. This interoperability is attained using Multicast Internet Gateways (MIGs), which are ad hoc nodes standing 1-hop away from the fixed network. Any ad hoc node may become a MIG at any time and several MIGs may coexist within the same MANET. They behave like all the other nodes with one feature added; they must notify access routers about multicast groups with interested receiver nodes within the ad hoc network. A node realizes that it is a MIG because IGMP messages are sent with a TTL value of one and therefore only ad hoc nodes one hop away from the fixed network receive those messages.

The multicast distribution mesh is created in a hybrid fashion. Inside the MANET, multicast routes are discovered on-demand, while routes to multicast sources in the fixed network are established proactively.

The protocol's reactive portion occurs in a response and reply manner. A node broadcasts a MMARP_SOURCE message when it has new information to send. This message is flooded throughout the entire MANET and is used by intermediate nodes to update their forwarding state and the multicast routes. MMARP_SOURCE messages have an identifier that allows nodes to detect duplicate packets. Receiving such a message for the first time causes a node to store the IP address of the previous hop and rebroadcast the packet. When the message gets to a receiver, it broadcasts a MMARP_JOIN message that includes the IP address of the previous hop. If a MANET node detects its IP address in such a message, it becomes aware that it makes part of the route between the source and receiver nodes and consequently activates its Multicast Forwarder Flag (MF_FLAG). Then it rebroadcasts the packet with the next hop address towards the source. By the time the

source receives the MMARP_JOIN message, a multicast route has been established between the source and the receiver. This procedure is illustrated in Figure 2.7.

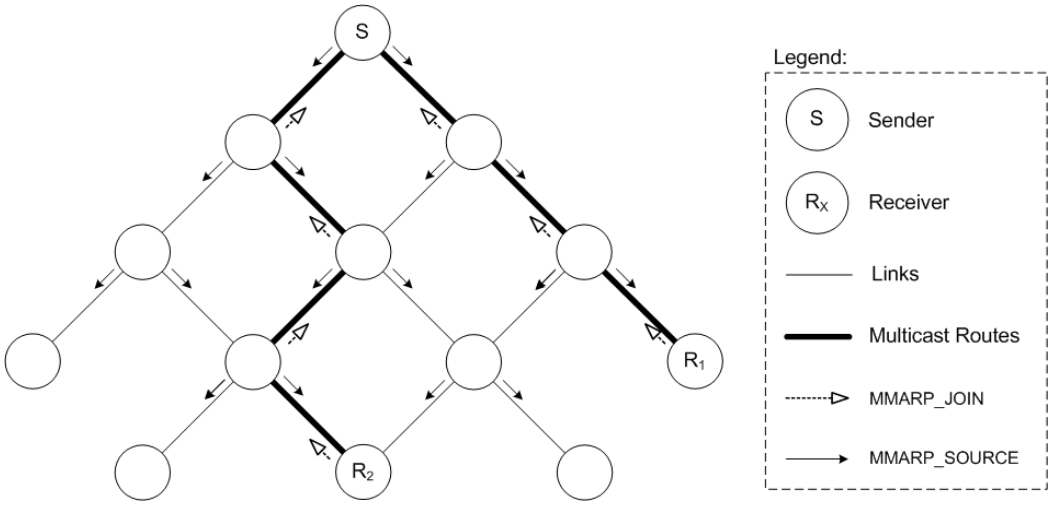

Figure 2.7 - MMARP network establishing a multicast route reactively

The protocol's proactive part just involves sending advertisements regularly. Each MIG sends a MMARP_DFL_ROUTE message advertising that it is the default multicast gateway towards the fixed network. Intermediate nodes are informed, by these messages, about the path to sources in the fixed network. When a MMARP_DFL_ROUTE reaches a receiver, a joining process similar to that in the reactive case takes place. The difference is that the MMARP_JOIN is sent to the MIG, which is responsible for translating this message to the access router and guaranteeing that the multicast traffic generated in the fixed network reaches the interested receivers in the ad hoc fringe. If a standard IP node turns into an active source, the distribution mesh generation takes place similarly to the reactive process, except that standard IP ad hoc neighbor nodes generate MMARP_SOURCE and MMARP_JOIN on its behalf. This might lead to the creation of unnecessary paths, since each neighbor may receive the messages sent by a standard node and process these independently. To overcome this complication there is a field in the message that identifies the node that originated the message, allowing MMARP nodes to recognize duplicates. Figure 2.8 illustrates a protocol’s possible distribution mesh configuration.

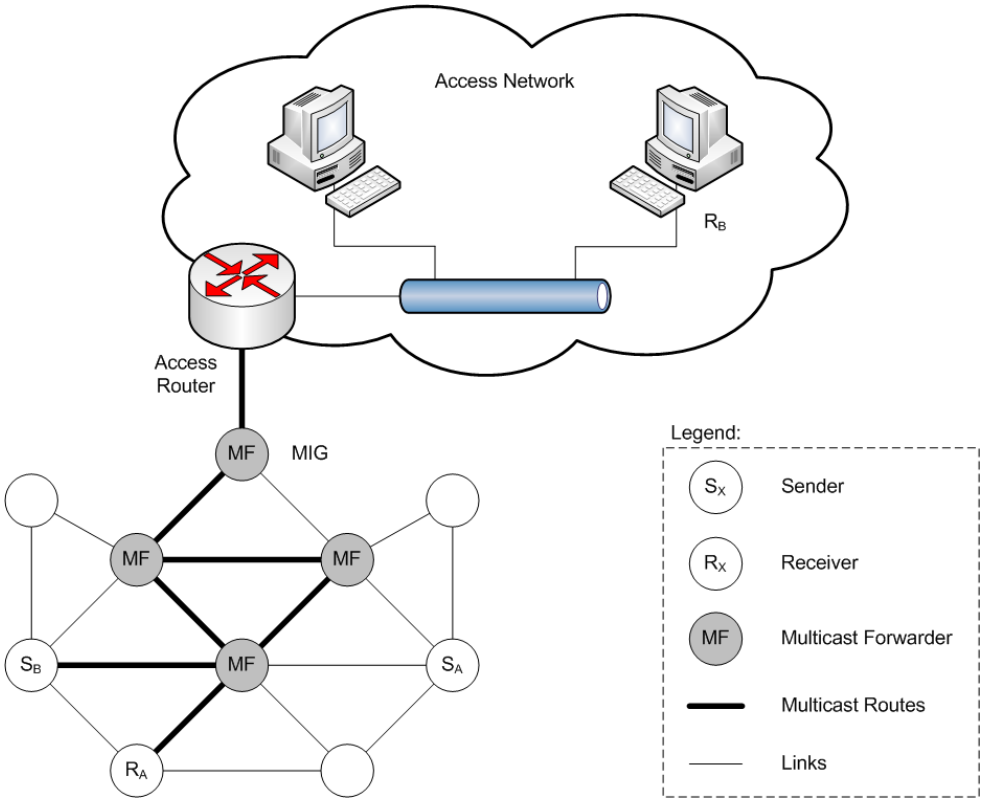

Figure 2.8 - Example of a multicast distribution mesh formed by MMARP

The MMARP [7] protocol requires messages to be sent in a reliable way, which means that in networks like 802.11b, where broadcast messages (which is the case of protocol control messages) are not acknowledged automatically, should have this functionality implemented by the protocol. MMARP [7] uses a method of passive acknowledgment. When one of two communicating nodes has to retransmit the message received, the retransmitted message is used by the first node as an acknowledgment. Otherwise, reception should be explicitly acknowledged.

2.6 SMF

Simplified Multicast Forwarding (SMF) [8] for MANET is a multicast routing protocol currently being developed by IETF. It is located in the network layer and stands on a logic of duplicate packet detection (DPD) mechanisms and reduced relay set designs, in order to provide a best-effort multicast forwarding inside a MANET. The protocol does not specify any particular neighborhood discovery protocol or relay set selection algorithms to operate with, thus meaning that each implementation may use the ones that

seem more suitable. SMF [8] supports Classical Flooding with DPD to accomplish its purpose, eliminating the necessity of having either a relay set selection algorithm or a neighborhood discovery protocol assisting its action. Nevertheless, it is recommended to use a reduced relay set mechanism in typical scenarios.

There are two major handling activities in charge of packet processing and forwarding; processing of outbound, locally-generated multicast packets and reception and processing of inbound packets. The purpose of the former is to resequence the packets generated locally, i.e., each data flow associated to a source and receiver address pair is given its own sequence number span. The later must receive all multicast data and process it for forwarding when necessary. As said earlier in the text, using relay sets is not mandatory despite being preferable.

DPD mechanism is an essential module of SMF's [8] forwarding process. It prevents a packet from being sent more than once, ensuring correct forwarding decisions. The approach for IPv4 consists in maintaining independent sequence number spaces, for each multicast flow between a source and a receiver. Such sequence number must be monotonically increased and is meant to be applied in IPv4 ID header field of all locally-generated multicast traffic, uniquely identifying the packet. A node must keep a history of received and processed packets for comparison with future incoming packets.

IPv6 packets share the same principle, although the process is a bit more complicated. The lack of an ID field in IPv6 packet forces to use IPv6's hop-by-hop options extension header to grant the same functionality. Optionally, the protocol specifies a hash DPD mechanism for IPv6 that also makes use of the hop-by-hop options extension header.

Again, using relay sets is not mandatory despite being preferable. However, being its support one of SMF's [8] goals, implementations must be capable to adapt local multicast packets forwarding rules based on relay set state information, should these be in use. Granting this, enables forwarding to function efficiently in a dynamic topology scenario. Three criteria should be considered in choosing a relay set selection algorithm:

- Robustness to topology changes;
- Localized election or coordination of any relay sets;
- Heuristic support for preference or election metrics.

The SMF [8] specification describes several relay set selection algorithms based upon these criteria.

2.7 DYMO

Dynamic MANET On-demand (DYMO) Routing [9] is a reactive routing protocol for MANETs, currently in development by IETF. It provides multihop unicast routing and is intended to provide routing mechanisms in a stubbed or disconnected MANET. It is particularly interesting for devices with constrained memory, considering that little routing information has to be kept. Although protocol operation is specified for network layer, the algorithm can be used in other layers without changes, except for packet format and layer specific addresses that have to be adequately modified. The two main operations associated with the protocol are route discovery and route management.

The discovery process relies on a response and reply mechanism. When a node needs to know a path to another one, it disseminates a Route Request (RREQ) message through the entire network and every intermediate node records a route to the originator. The target node should respond to this message with a Route Reply (RREP), sent hop-by-hop. Exactly the same process occurs when the response is heading towards the originator; each intermediate node records a route to the target node. By the time the RREP is arriving to the originator node, every intermediate node has a route to it and to the target node, thus establishing a path between them.

As for route management, the nodes extend route lifetimes every time a packet is successfully forwarded, besides monitoring active links to react to topology changes. If a data packet cannot be forwarded because a route is broken or there is no route at all, Route Error (RERR) message is sent to the packet source node, making it aware of the route breakdown. Upon reception of RERR, the source node deletes the route. The next packet towards the same destination originates a new discovery process for that destination.

DYMO messages conform with PacketBB [5], the general packet and message format for MANETs, discussed earlier. Protocol operation requires the use of sequence numbers per node. Sequence numbers are useful to determine routing information freshness and to guarantee loop free topology. To accomplish this, each time a node sends a new routing message (RREQ and RREP), it should increment its sequence number. If this was not done, it might lead other nodes to consider the routing information stale and as a consequence the messages might not reach their target. Sometimes, when a RREQ is sent and a DYMO router in the path between the originator and the target nodes has the routing

information needed to satisfy the incoming message, it can respond on behalf of the target node. Two RREPs should be sent in this case. One of the RREPs is sent to the originator node, containing additional routing information about the target node and the other one to the target node containing additional routing information about the originator node. This way both nodes will know how to get to each other.

There are a few details about routing messages that are worth to be mentioned. When trying to get a route to a certain destination, a node awaits a response for a well determined period after which it may try to send another RREQ. For each retry, the waiting time should be multiplied by 2 with respect to the previous retry waiting time. After a predefined number of retries the destination is declared unreachable. Meanwhile, all packets to the destination being searched are kept in a buffer until a route is found. Should there exist no route, all buffered packets are dropped. A RERR message should be issued not only when a packet cannot be forwarded, but also when, as a result of the monitoring mechanism, a broken link is detected unless it has not been used recently.

DYMO [9] protocol is flexible in what concerns monitoring links. Several mechanisms may be used separately, such as link layer feedback, NHDP and route timeout, as well as a mechanisms made up of more than one of these techniques. Each route in a node should have a timer that is renewed every time a packet is forwarded. The route is removed if the timer expires. For DYMO [9] to function in a stubbed MANET, a DYMO routing capable gateway must exist. This gateway is responsible for sending routing messages on behalf of nodes outside the MANET and maintaining sequence numbers for each one.

2.8 OLSRv2

The Optimized Link State Routing Protocol version 2 (OLSRv2) [10] is a table driven, proactive routing protocol, targeted at mobile ad hoc networks, currently in development at IETF. It works in a completely distributed manner and has no central entity controlling operations.

It is presented as an optimization of the classic link state routing algorithm. The major optimization is related with multipoint relays (MPR), which provide an efficient broadcast scheme for spreading link state information across the entire network, reducing greatly the cost of doing it. Another optimization is that OLSRv2 [10] uses partial link

state information. This way, the number and size of link state broadcasts are reduced, while each node still retains the ability of routing to all destinations in the network through optimal paths, since information about all destinations is available at any time, in spite of limited link information. In fact, if each node advertises its link states only to the nodes that elected it as MPR, this would suffice to provide optimal path routes to the entire network. Routes are established having in consideration the hop count. Furthermore, OLSRv2 [10] does not require sequenced or reliable transmission of control information, reducing the network requirements to a minimum.

Comparing it to its predecessor, OLSR, published as RFC 3626 [3], OLSRv2 [10] provides a few enhancements, more specifically in terms of signaling flexibility and message simplification, including IPv4 and IPv6 addresses compaction.

Before going any further, the reader should be acquainted with the concepts of symmetric 1 hop and 2 hop neighborhood. A node is said to be symmetric 1 hop neighbor of another node when both nodes can communicate directly with each other. Two nodes are said to be symmetric 2 hop neighbors of each other when there is a third node that is symmetric 1 hop neighbor of both and, therefore, is able to provide communication between them.

OLSRv2 [10] makes use of PacketBB [5] and NHDP [6], in order to achieve its purpose. In addition, a TLV to be included in HELLO messages of NHDP for MPR selection signaling, as well as "MPR flooding" mechanism for global information exchange, are characteristics of the protocol. Also, it specifies Topology Control (TC) messages, which are used to disseminate information about link state and gateway role to reach advertised hosts and networks. These messages are regularly sent by each node so that other notes can keep track of network changes. As said before, OLSRv2 [10] requires no sequenced transmission since each TC message has its own sequence number, which is incremented as new messages are generated. Moreover, incomplete TC messages are allowed and interpreted as additional information to previous messages and the missing information being considered unchanged.

Detailing the protocol operation, each OLSRv2 [10] node in the network selects its MPR set. This set is composed of any subgroup of all the nodes in the symmetric 1-hop neighborhood of the selector node, which permits to reach every node belonging to the 2-hop symmetric neighborhood. Each node, also keeps information regarding the symmetric 1-hop neighbors set that have selected it as MPR. These nodes are called MPR selectors.

As an example, in the topology represented in Figure 2.9, nodes A, B, C and D constitute the MPR set for node E and nodes E, F, G, H, I, J, K and L are node A's MPR selectors.

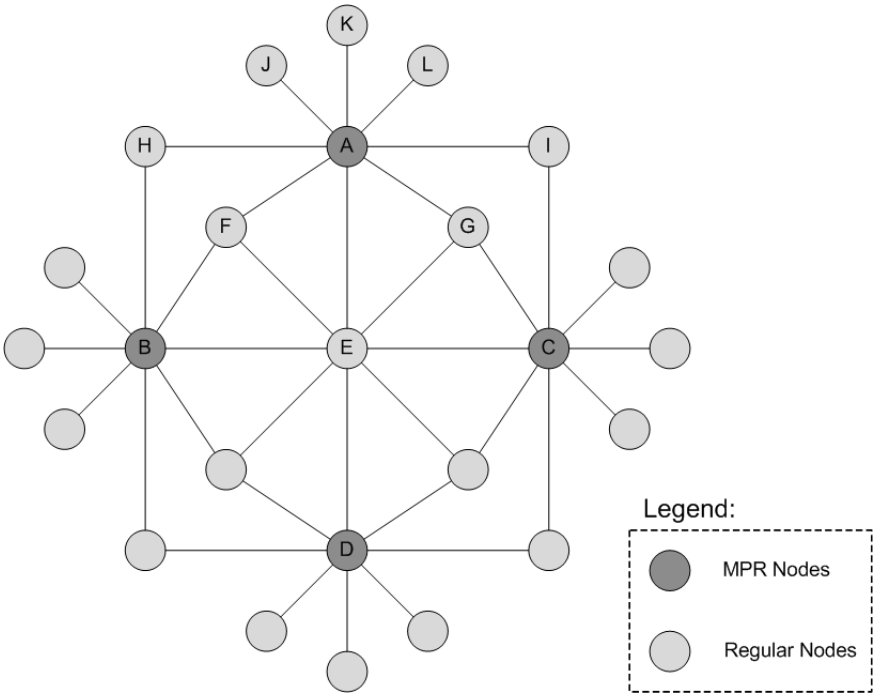

Figure 2.9 - Hypothetical OLSRv2 topology

OLSRv2 [10] admits nodes to use different MPR selection algorithms and still maintains interoperability, as long as some conditions are respected. However these conditions fall off the scope of this description and will not be discussed. Another characteristic of MPR selection has to do with the possibility of each interface belonging to a node to choose its MPRs independently or, alternatively, coordinated with the other interfaces. MPRs must be recalculated every time the selected ones no longer fulfill the required conditions.

Two more sets exist and are maintained by each node; the Relay Set, comprising all the symmetric 1-hop neighbors for which the node has to relay broadcast traffic and the Advertised Neighbor Set that includes the set of symmetric 1-hop neighbors that should see their link state information advertised to the network, in TC messages by their MPRs. Both these sets must include all MPR selectors. Should a node's Advertised Neighbor Set be empty, such node does not generate TC messages, with some exceptions such as gateway reporting.

Several repositories are used to gather information exchanged in OLSRv2's messages. Specifically, there are 4 information bases, which are the Local Information

Base, Neighborhood Information Base, both defined in NHDP [6] and explained before (section 2.3), Topology Information Base and Processing and Forwarding Information Base. The sets referred to in the previous paragraph, among other sets, are accommodated in these information bases. The first two are extended by OLSRv2 [10] that adds some information required for proper protocol operation. The other ones are protocol specific. Topology Information Base holds information used for generation and processing of TC messages, while Processing and Forwarding Information Base maintains data to guarantee that each message is not processed and sent on each interface more than once.

OLSRv2's message and packet formats follow the ones defined in packetBB [5]. Such messages carry the information exchanged between nodes participating in the protocol. Several messages, if sent at the same time, should be combined in the same packet. In addition, messages from different origins or even from different protocols may be combined in the packet, provided that those protocols use packetBB [5] message definition. One of the messages used by OLSRv2 [10] is the HELLO message, defined in NHDP [6]. In addition, each node includes in its HELLO messages, the TLV with information related with MPR selection signaling afore mentioned. Another TLV may be added to these messages, although not required, revealing the node's willingness to be selected as MPR. Should this TLV be missing, a default value is inferred. The other messages used by OLSRv2 [10] are the TC messages. These include the sequence number, among other parameters that OLSRv2 [10] requires to determine whether topology information is up to date.

2.9 Summary

Throughout the chapter, the reader got to know the basics about the protocols described, enough to understand the subsequent work. The LUNAR protocol was described for its similarity with the solution presented in this text, namely the fact that it also operates at layer 2, although the rest of its operation is somewhat different. PacketBB gives an insight on the type of messages that some of the described protocols use to exchange information and is regarded as an extension to the proposed solution, in terms of future work. AODV [2] and OLSR [3], for their key role in the *ad hoc* networking scenario, will be used as references for analyzing the work developed. The remaining protocols were involved in the development of the proposed solution either as an

inspiration or as additions considered in a preliminary phase of study as the reader will realize in the chapters ahead. MMARP [7] with and SMF [8], fit the first situation, since they inspired the duplicates detection mechanism developed. DYMO in association with NHDP were the considered for additions, since they were regarded to lend their routing and information services, respectively, to the broadcast scheme developed.

Chapter 3

3 Previous work

This chapter is dedicated to the previous work done in the context of the actual developed solution. The protocol described is the basis for the work performed until the final solution. Every approach made to find solutions for the work developed was based on the mechanisms and *modus operandi* of the protocol. It deploys an *ad hoc* network with a tree topology and overlays a functional forwarding mechanism, optimized to follow the branches of the tree topology and access the exterior of the *ad hoc* network (Internet). The entire operation of the solution is accomplished in layer 2. There is no interaction with protocols on layer 3, and above. It will be given an overview of the concepts and mechanisms involved in the topology creation, as well as a detailed description of the forwarding operation that the protocol imposes, supported by some illustrations to ease the understanding of the concepts involved.

3.1 OHAP

One Hop Ah hoc Protocol (OHAP) (1) is a forwarding protocol for MANETs implemented by the author in his final year project at the School of Engineering of the University of Porto, under the theme "Ad hoc networks as a way of expanding the classical infrastructure network". It was based on the theoretical work, still in development, done by two researchers at INESC Porto. The proposed solution provides a way to expand an infrastructure network, employing only wireless nodes connected in an 802.11 ad hoc network. By expansion, one means to allow that Internet services and contents are reachable by anyone who connects to the ad hoc network. The solution provides a forwarding mechanism completely independent from the network layer, meaning that no interaction with level 3 protocols exist in order to forward traffic. It fully operates at level 2. Consequently, all nodes appear to be one IP hop away from each other. Furthermore, operation on this network is transparent for both, users and network administrators, since they can only see a standard IEEE 802.11 network. The solution is based on a protocol that establishes a tree-based ad hoc network topology, with the root acting as a "master node”

and behaving as a level 3 gateway to interconnect to the infrastructure network; this will be discussed ahead in detail. The use of a tree topology is understandable, since it is desired to provide Internet access and thus, the tree provides optimal paths towards it.

To fully understand how this solution works, some prior knowledge of the concepts involved is necessary. The two main concepts are virtual connections and IEEE 802.1D bridges. Virtual connections emulate traditional wired networks, working as point-to-point connections. Each node in the ad hoc network has a virtual connection to each of its neighbors for sending traffic destined to it. Each virtual connection corresponds to a virtual interface associated to a neighbor. When a frame is received by the wireless interface, it is injected in the virtual interface corresponding to the node that sent it. The virtual connections concept is depicted in Figure 3.1.

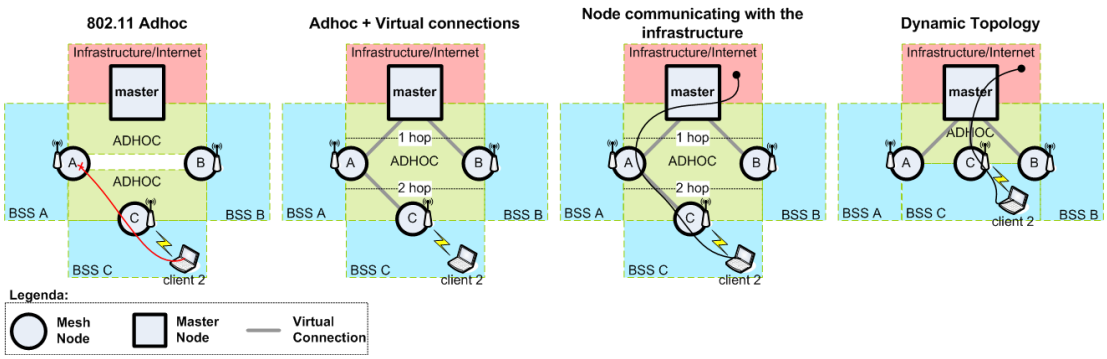

Figure 3.1 - IEEE 802.11 *ad hoc* network with virtual connections as used by OHAP

All the virtual interfaces are inserted in a software-based 802.1D bridge implemented by each node, whose forward mechanism takes charge of redirecting incoming frames Figure 3.2.

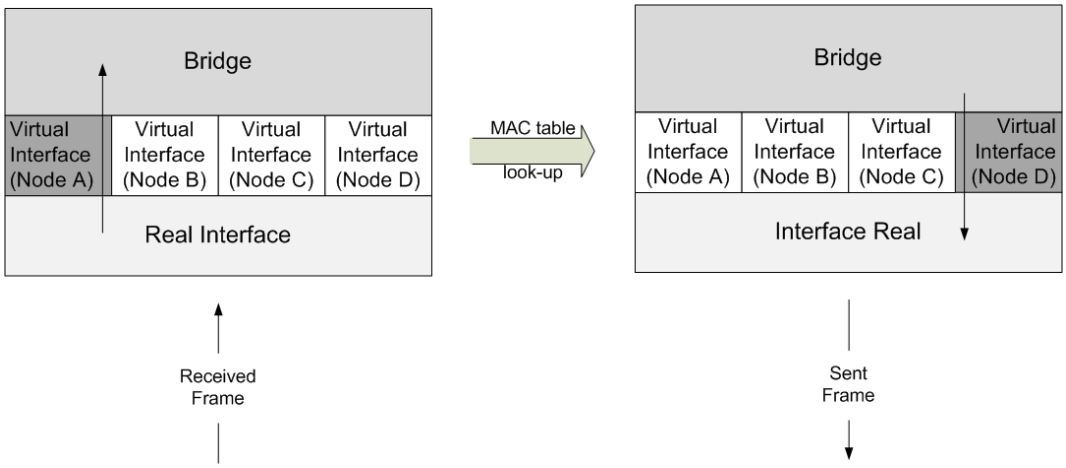

Figure 3.2 - OHAP forwarding scheme using IEEE 802.1D bridges and virtual interfaces

This works as if each node was in a different LAN. Communication is attained passing the information hop-by-hop, since the IEEE 802.11 protocol only provides communication between nodes in the same radio link, throughout the network until it reaches its destination. In order to manage hop-by-hop transmission, besides the bridge forwarding mechanism, each frame is encapsulated in another level 2 frame that has its header changed in each node it traverses to guarantee proper functioning of the 802.11 control frames.

The operation of the proposed solution, as said earlier, is supported by a protocol developed to establish connections between neighbor nodes in order to form a tree topology. Such topology is built by exchanging HELLO messages, sent by every node attached to the tree. The messages are Ethernet II frames slightly modified, as easily seen in Figure 3.3. There are two additional fields in the frame, which are used to send protocol

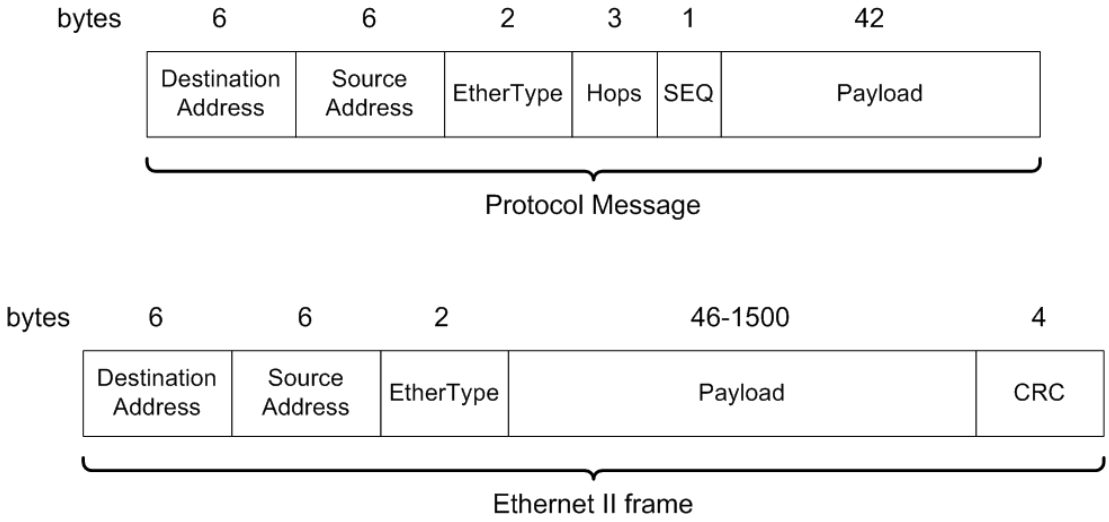

Figure 3.3 - OHAP protocol message format compared to an Ethernet II frame

information. The Hops field holds the number of hops the sending node is away from the "master node" and the Seq field is a sequence number used to avoid undesired retransmissions. Furthermore, the Destination field keeps the neighbor's physical address that the sending node has chosen as "up neighbor", a concept that will be discussed briefly, and EtherType is the protocol ID (0xFFFF, for now) so that each node can identify protocol messages. Finally, the Source field holds the sending node's physical address. These messages are sent encapsulated (Figure 3.4) in a regular Ethernet II frame with the Destination Address set to the broadcast address and the Source Address set to the sending node's physical address. EtherType is set to protocol ID, since this EtherType identifies

both protocol messages and protocol frames. As said earlier, every frame is sent hop-by-hop, thus encapsulation has to be made at all times and not only HELLO messages but all messages exchanged by the nodes that implement the protocol, are sent with EtherType set to protocol ID.

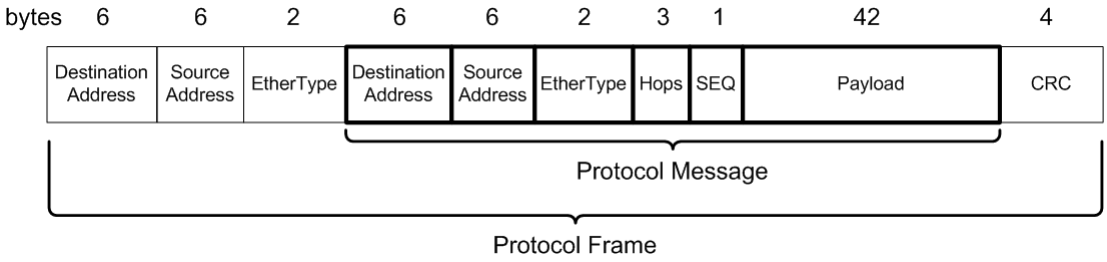

Figure 3.4 - OHAP encapsulation method

The "master node" aforementioned is the tree root and is the node that provides connection to the infrastructure network and, for that reason, it is responsible for generating the HELLO messages. This means that if this node fails, the tree attached to it vanishes. HELLO messages are generated by the “master node” every TIMEOUT_VAL seconds and must be forwarded by the nodes belonging to the tree. TIMEOUT_VAL is a value inherent to protocol operation.

In order to belong to the tree, every node, except the master node, has to choose an “up neighbor” already connected to the network, based on the information present in HELLO messages. This requires collecting HELLO messages for three TIMEOUT_VAL periods, after which the choice must take place. Having made the choice, a virtual interface to the chosen “up neighbor” is created and the node sends an HELLO to inform other nodes of its choice. Upon receiving this HELLO, the corresponding “up neighbor” recognizes its address and creates a virtual interface to its new neighbor. This occurs at every three TIMEOUT_VAL intervals, forcing nodes to update neighboring information and guaranteeing that potential topological changes do not affect the connectivity. If a node changes its “up neighbor”, the previous “up neighbor” eliminates the virtual interface associated to it upon receiving the HELLO and discovering that it is no longer the chosen “up neighbor”. Alternatively, if no HELLO is received after five TIMEOUT_VAL periods passed from virtual interface’s creation, it will lead to its removal. Each node can only have one “up neighbor”, although this can be different every time a node has to choose one. This imposition guarantees that the tree topology is kept. Only connected nodes are

able to communicate in the ad hoc network. When forwarding an HELLO message, a node must update information in it (Destination, Source and Hops) to match its own. This mechanism is illustrated in Figure 3.5.

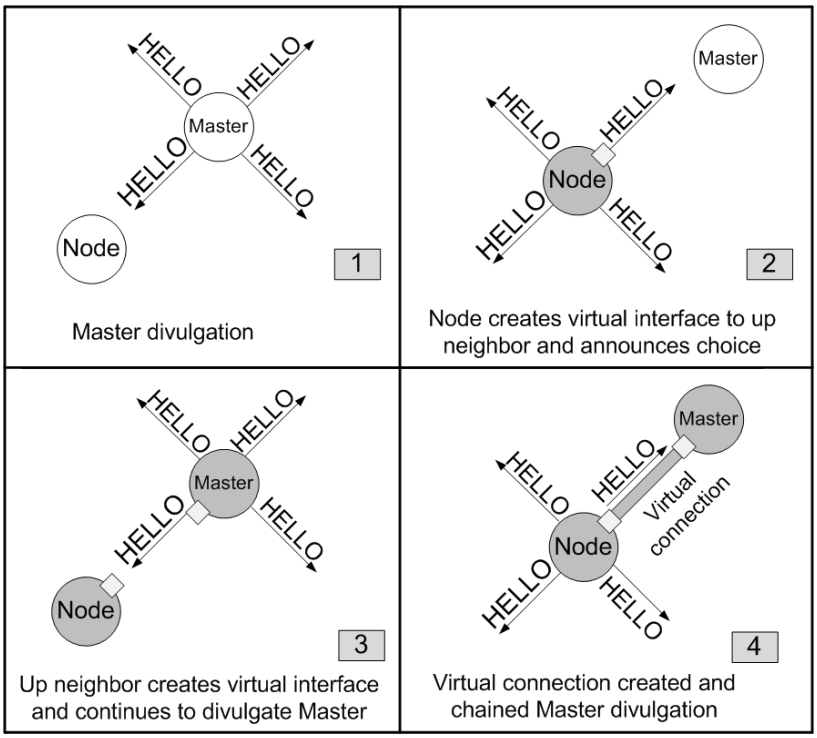

Figure 3.5 - OHAP implementing nodes setting up a virtual connection

Once a node is connected to the network, it is ready to communicate with other connected nodes. Any frame that has to be sent is input to the bridge, which redirects it to the correct virtual interface. When this frame is received at the virtual interface, encapsulation is made, as explained earlier, and the frame is sent. This occurs to every single frame in transit.

Unicast and broadcast communications are native to protocol operation and both follow the procedure explained above.

The operation of the algorithm is simple. There is a TTL associated to each virtual interface. Every time TIMEOUT_VAL period elapses, all TTLs are decremented until the value of zero is reached. At this point, all virtual interfaces whose TTL is zero are removed. TTL values are renewed every time an HELLO concerning a particular virtual interface is received. When a node receives an HELLO, it verifies whether it has been chosen as an “up neighbor” or is no longer chosen, and creates/renews a virtual interface or

deletes it, respectively. An “up neighbor” choice is made every three TIMEOUT_VAL. In between these TIMEOUT_VAL periods, all node’s interfaces, not only virtual but also real wireless interfaces associated to protocol operation, are monitored for incoming and outgoing frames. During this period HELLO messages are sent, received and collected and common traffic is processed. Figure 3.6 and Figure 3.7 illustrate the block diagram and the operation of the protocol, respectively.

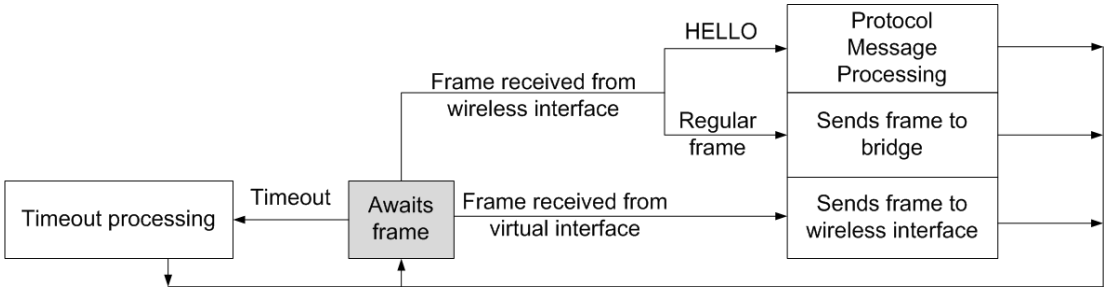

Figure 3.6 - OHAP block diagram

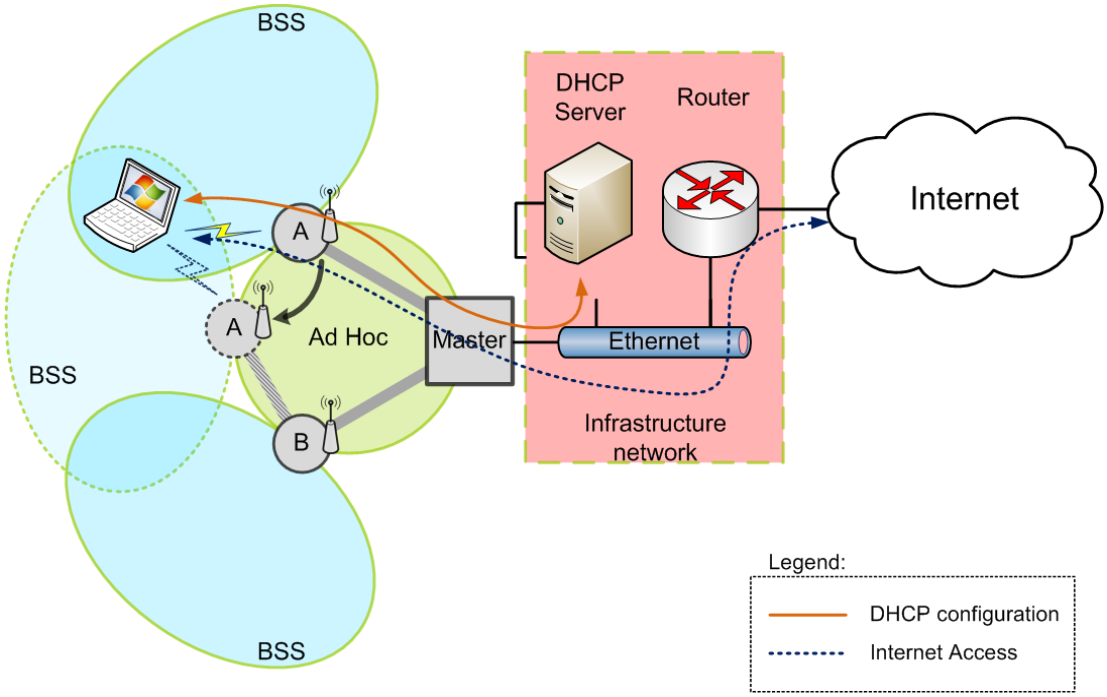

Figure 3.7 - OHAP operation

Figure 3.6 is pretty much self-contained. As for Figure 3.7, it illustrates a legacy node connecting to the *ad hoc* network implementing OHAP and acquiring configuration information from the DHCP server for later Internet access through the router in the fixed

network. It is also shown node A moving and, consequently, changing “up neighbor”. Nevertheless, connection between the legacy node and node A is not affected and the later can still continue to communicate

As a final remark, this solution is able to support legacy devices.

Chapter 4

4 Development Environment

The present work was developed on standard laptop computers equipped with CISCO AIRONET 350 SERIES wireless interface PC cards from Cisco Systems, Inc. These cards are compliant with the IEEE 802.11b standard. All interfaces were used in *ad hoc* mode, since the developed solution emphasizes the use of an *ad hoc* network.

All the computers, were running the operating system UBUNTU Gutsy Gibbon (v.7.10) with kernel version 2.6.22-14. Considering that the One Hop Ad hoc Protocol (OHAP), which stands as the departing point for the present work, was developed in a similar LINUX platform (UBUNTU Edgy Eft (v.6.10) kernel version 2.6.17-11), the option for this operating system becomes obvious. Moreover, this operating system is open source, which means the possibility of performing any changes to serve one's interests, and free of charge. The network API of these types of platforms is very straightforward when compared those of proprietary operating systems. Another important reason for the choice, is the fact that most network devices lay on UNIX-based platforms, as is the case of LINUX. Finally, the native tools for configuration and control of network interfaces and resources and the level of depth they offer are other strong arguments to justify the choice. Some of the commands used are described in Table 4.1.

Command	Description
ifconfig <iface> <address> netmask <mask>	Associates an IPv4 address to an interface
ifconfig <iface> [up|down]	Brings an interface to an up or down state
iwconfig <iface> essid <name>	Associates an interface to a wireless network
iwconfig <iface> mode <managed|ad-hoc|monitor>	Specifies the operation mode of an interface
iwconfig <iface> key <sec_key|off>	Specifies the security key to access the wireless network that an interface is associated with
iwconfig <iface> channel <number>	Specifies the channel on which the interface will operate

Table 4.1 - Summary of the commands used

The Parallel Simulation Environment for Complex Systems (PARSEC) simulation tool [11] is a C-based simulation language for execution of discrete-event simulation

models. It was used to develop a program to simulate the behavior of one of the solutions proposed to improve OHAP, prior to implementing the developed solution. Still for simulation purposes, the source code of Shortest Tree Simulator [12], an application for path calculations in tree topologies referenced in [13], was made available by its author, in order to be used and modified as needed. It was changed to generate the mesh and tree topologies used in the simulation models developed. The mesh topologies generated were saved in topology files, with the data format required by the Parsec program developed. Later these files were submitted to the simulation program, to examine and validate the performance of the proposed scheme over several topologies. As for the tree topologies, the algorithm for examination and validation of the proposal was embedded in the above mentioned application.

The source code of the original OHAP prototype was also used for the present work. The implementation of the proposed solution was built over OHAP's source code, so the resulting prototype can be considered as OHAP compliant with added features. The improvements made consist in the application of the simulation results to the existing OHAP solution.

Chapter 5

5 Implementation

In this chapter the reader will be able to follow all the work done from its early stages until the complete implementation. It will describe the mechanisms developed to solve the problem in hands and the options made concerning them, based on simulation results and illustrations to give greater insight. The actual implementation phase is described afterwards. It is followed by the explanation of the details related to the practical application of one of the above mentioned mechanisms, to the protocol implementation being improved.

5.1 Problem identification and proposed solutions

The departure point for the present work is One Hop Ad hoc Protocol (OHAP) explained in the previous chapter. The initial OHAP implementation suffers from a few problems. Among them, broadcast inefficiency is the most limitative, because, in spite of supporting it, it does so by encapsulating broadcast frames in unicast frames. This means that broadcast and unicast frames are exchanged in the same way. The problem remains in the fact that when a broadcast message is received in the bridge, such message is sent on every virtual interface. Consider that a node has N virtual interfaces. This means that an arriving broadcast frame will be sent N-1 times in unicast, using the encapsulation method. This situation resembles the classical flooding technique, in which each node retransmits a broadcast message every time a new one is received, as depicted in Figure 5.1. Classical flooding is highly ineffective due to the number of messages involved in each broadcast and, consequently, high bandwidth consumption. Current work is focused on solving this problem.

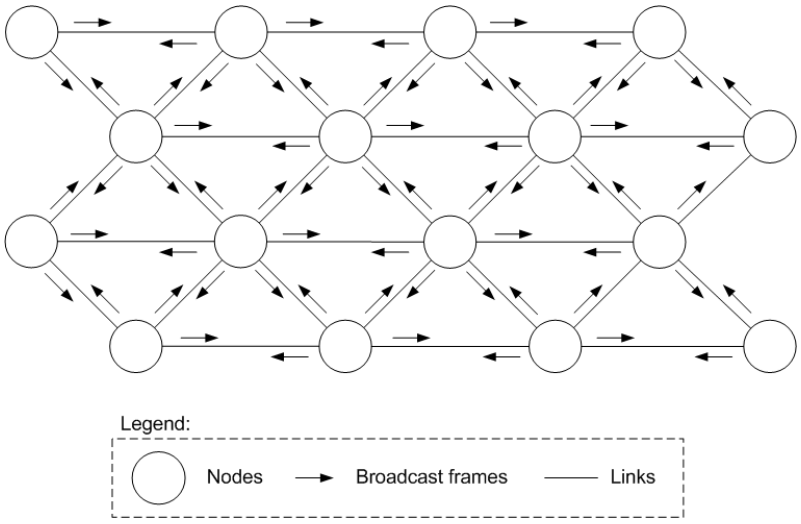

Figure 5.1 - Classical flooding mechanism

Two different approaches were considered. The first one, although focused in solving the broadcast problem, would also solve another OHAP limitation: communication inefficiency between nodes inside the protocol's ad hoc network. The tree topology used by OHAP is optimized to provide access to the outside of the ad hoc network. If two nodes inside the network were to communicated with each other, their messages would have to climb the tree up to the common node and, from there, descend until they reach the destination. Figure 5.2 illustrates this situation, considering the communication between nodes A and B.

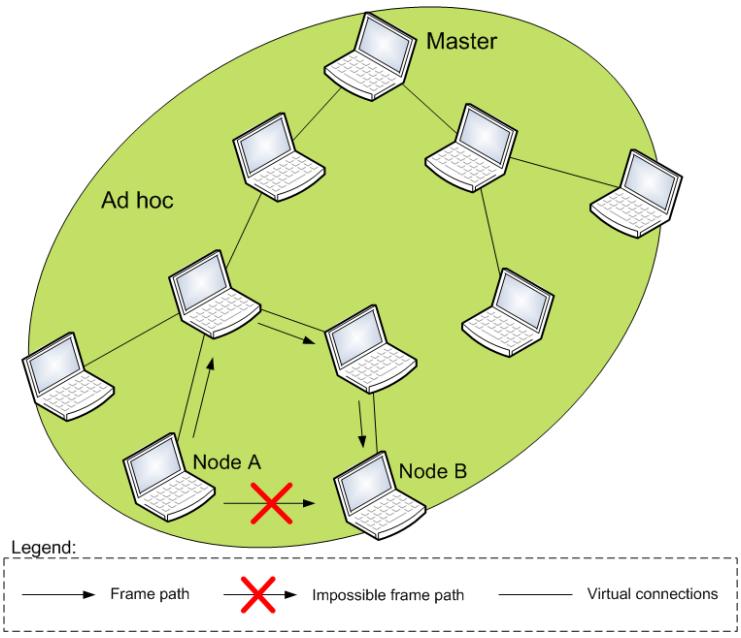

Figure 5.2 - Inefficiency of the tree topology providing inbound node communication

To overcome this drawback, the existing solution would have to be complemented with an inbound forwarding mechanism that could ignore the imposed tree topology, which means to ignore virtual connections, for inbound communications. Hence, at this point there are two problems to solve. Based on the assumption that a layer 2 adaptation of Dynamic Manet On-demand (DYMO) [9] routing protocol could be added to the current protocol implementation, since its algorithm is layer independent, a broadcast mechanism was proposed. DYMO would solve the inbound communication inefficiency and at the same time, because it uses Neighborhood Discovery Protocol (NHDP) [6], would make available neighborhood information up to 2 hops. This information, present in an NHDP's Interface Information Base (IIB) would be used to decide whether a node should forward a broadcast frame or not. IIB, as described in chapter 2, section 2.4, holds two sets of nodes: the Link set, which records 1 hop neighbor nodes and the 2 hop set, which registers 2 hop symmetric neighbors and 1 hop symmetric neighbors through which the 2 hop symmetric neighbors can be reached. Combining this information, forwarding decisions could be made by the algorithm described below.

The algorithm is as follows:

```
Node sends a broadcast frame and waits for neighbors' broadcast frames
If a broadcast is received
        Checks if it is a new broadcast
        If yes
                Checks if all 1 hop neighbors are also sender's 1 hop neighbors
                If not
                        If no common 1 hop neighbors and node has more than 1 neighbor
                                Forward broadcast
                        Else if there are common 1 hop neighbors
                                If node has more 1 hop neighbors than common neighbors have
                                        Forward broadcast
                                Else if tied in number of 1 hop neighbors
                                        If MAC address bigger than neighbors'
                                                Forward broadcast
```

Each time a node forwards a broadcast frame, a message counter is incremented. When all nodes finish transmissions, the counter holds the total broadcast frames sent for a given topology. This algorithm was tested against several topologies with different number of nodes.

Figure 5.3 depicts the application of this algorithm to a hypothetical topology.

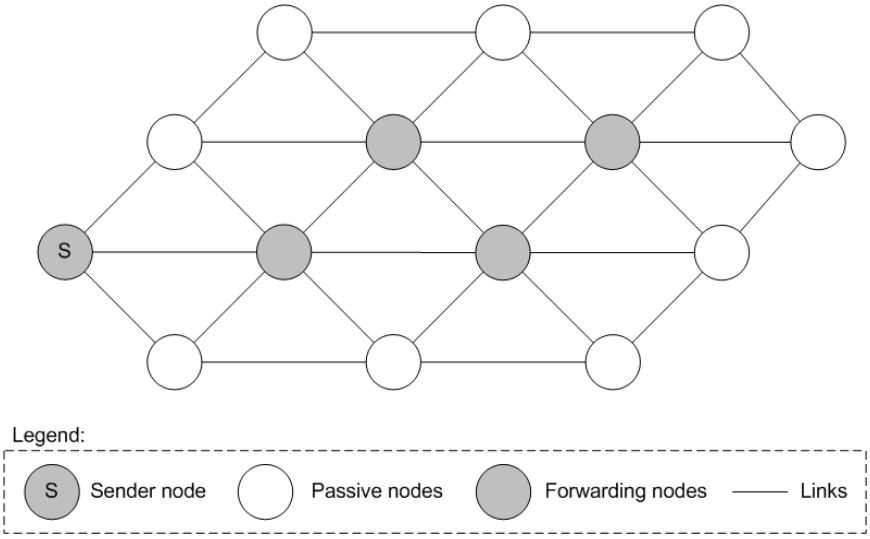

Figure 5.3 - Application of the NHDP based mechanism to a hypothetical network

The second approach has in consideration that the communication between ad hoc nodes is vestigial and therefore, the tree topology will not reveal to be a problem. This consideration is based on the fact that the key purpose of the existing protocol is to establish an ad hoc network to extend an infrastructure network rather than provide efficient communication between ad hoc nodes. In fact, the idea of using a tree topology is to provide optimal paths towards the root node that provides connection to the infrastructure network. Furthermore, the entire process of creating the tree topology follows the established tree branches themselves, which coincides with the direct paths to the root node. The only case in which it occurs is precisely when two random nodes, other than the root node, try to communicate with each other. On the light of this assumption, neither DYMO nor NHDP would be necessary to the existing protocol and another broadcast scheme is required. Moreover, maintaining NHDP just for broadcast sake could become problematic due to the quantity of messages to be exchanged. With this in mind, to consider that the tree topology is to be maintained in this scenario is a realistic approach. Being so, there is no alternative left than to use it to make broadcasts circulate in the network. However an improvement, with respect to encapsulation is essential to achieve efficient broadcasting. To improve the efficiency, each ad hoc node must send just one copy of a broadcast frame to all neighbors instead of delivering one copy to each neighbor. This implies a slight modification in the encapsulation. When a broadcast frame is to be transmitted by a node, it is necessary that it is sent with broadcast address set in the frame's Destination Address field, so all neighbor nodes can receive it. In addition, as bridges

forward broadcast frames in every interface except the one where the broadcast is received, it is necessary to allow just one of these copies to be sent. All other forwarded copies must be dropped. Another particularity is that only nodes having more than one virtual connection need to forward broadcast frames. Nodes having only one virtual connection are leaf nodes, with respect to the tree topology. Since leaf nodes are on the edges of the tree, they do not forward broadcast frames unless they are the originator. A considerable amount of messages could be spared with this improvement, especially when there are nodes with many others attached.

The algorithm to calculate the number of messages exchanged is very simple. Consider a tree topology with N nodes. Allow L to be the number of leaf nodes in the tree and R the total number of retransmissions associated to one broadcast, including the original transmission. In these conditions, there are at least N-L nodes that, no matter what, will have to forward the broadcast frames. Detailing, if one broadcast frame is sent, the number of retransmissions will be:

$$(1) \qquad R = \begin{cases} N - L, & if\ originated\ in\ a\ non - leaf\ node \\ N - L + 1, & if\ originated\ in\ a\ leaf\ node \end{cases}$$

Considering that each node transmits a broadcast frame, locally generated, the total will be:

$$(2) \qquad R = (N - L) \cdot (N - L) + (N - L + 1) \cdot L$$

Expanding:

$$(3) \qquad R = N^2 - N \cdot L + L$$

The algorithm to use in order to compute the total number of broadcast frames exchanged is simply formula 3.

In summary, assuming vestigial communication between nodes inside the MANET a slight improvement in the broadcast mechanism could greatly improve the overall performance of the protocol.

5.2 Validation of the Proposed Solutions

The solutions afore mentioned were tested by simulation and the results are compared against each other. Moreover, comparison with classical flooding mechanism is also presented to determine the gains obtained by each broadcast scheme. The comparison is made in terms of the number of messages transmitted versus the number of nodes in the MANET. For testing purposes, it is assumed that each node in the network sends one broadcast frame. The measurements made only take in account the number of frames exchanged in the simulated network topologies, for a total of 500 topologies per network size. The presented results are a mean over all simulated topologies.

Simulations occur in two steps. The first step consists in generating random topologies, with a proper application and the second step calculates the total number of messages that were circulating in the network.

For the first approach, the one that uses DYMO and NHDP, the application is used to generate random graphs. The generated graphs are recorded in topology files that are later submitted to the Parsec simulation algorithm, which is the one explained above. Based on the information in the topology file, the algorithm constructs the Link Set and the 2 Hop Set, from NHDP's Interface Information Base, which establishes the communication rules between nodes. Then it computes the total number of messages exchanged for each generated topology.

In what concerns the second approach, the same application is used to generate topologies, which in this case are shortest path trees (SPT), extracted from generated graphs by Dijkstra's algorithm. For each generated tree, the leaf nodes are accounted and the total number of messages exchanged in that particular tree topology is computed applying formula .3.

Simulation results are in the graphic shown in Figure 5.4 and Figure 5.7 and in Table 5.1 and Table 5.2, ahead. As expected, both proposed broadcast schemes behave better than classical flooding, since classical flooding is a very inefficient and expensive broadcasting mechanism. However, the gains obtained with the first approach, are small. A close look to the plotted values shows that a network using the proposed scheme would behave like a MANET with 2 nodes less and using classical flooding, for networks up to 15 nodes. For networks with a larger number of nodes the gains are even smaller. Adding the control traffic used by NHDP to fill the necessary structures for protocol operation

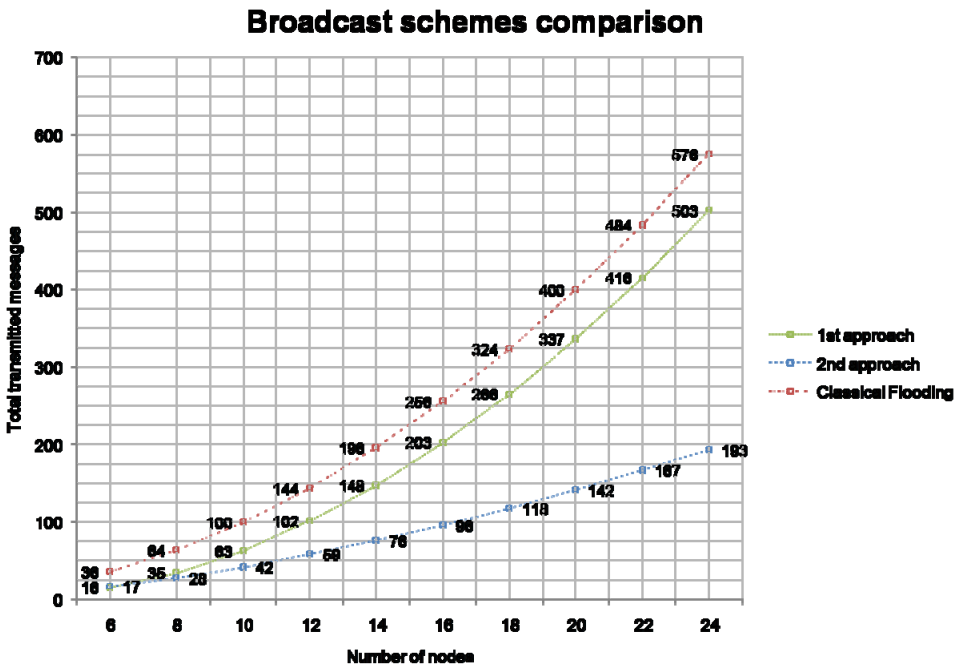

Figure 5.4 - Comparison between proposed broadcast mechanisms and classical flooding

might even nullify any gains. Thus this mechanism, does not improve the existing solution, at least to a degree that makes worth the implementation.

In addition, all mechanisms under analysis present a quadratic growth, but the second approach has a much slower increase rate. It is easily observed that the broadcast scheme based on the tree already implemented by OHAP is, by far, the one with better performance. Furthermore, this approach has its foundations already laid, since the forwarding decisions for broadcast frames are based on information that already exists in the OHAP participating nodes, unlike the first approach. In fact, if a broadcast frame is received by a leaf node, it is injected in the bridge by the only existent virtual interface and, since it is the only interface, no forwarding will be carried out. The gains obtained with this approach are considerably high. For a 24 nodes network implementing this broadcast mechanism there is a gain of about 8 nodes compared to the first mechanism and a 10 nodes gain compared to the classical flooding scheme. By expressing gains in terms of number of nodes, one means that a network that has a smaller number of nodes and implements another broadcast mechanism, exchanges as many messages as the mechanism under analysis.

In terms of number of messages exchanged, a network using the second mechanism sends, on average, 57% of the broadcast frames that the first scheme would. Moreover, except for the case of a 6 node network, the second broadcast scheme always retransmits

fewer frames than the first one. Another trend that can be observed is that the more nodes a network has, the bigger is the difference between the numbers of frames retransmitted by each mechanism.

Data concerning to each approach is represented in Table 5.1 and Table 5.2. For each topology set, the confidence interval for the mean number of messages retransmitted is calculated. Upper and lower limits, in both tables, correspond to interval boundaries. Confidence level is set at 99%. With this in consideration and given the values on the tables, one can see that amplitude of the intervals compared with the average number of transmissions is very small, which reveals high simulation accuracy. These results reinforce the conclusions taken from Figure 5.4.

Nodes	1st approach			
	Upper limit	Lower limit	Mean	Standard deviation
6	17	15	16	9
8	37	33	35	17
10	66	60	63	27
12	107	97	102	40
14	154	142	148	54
16	211	195	203	69
18	276	256	266	86
20	349	325	337	104
22	430	402	416	123
24	519	487	503	142

Table 5.1 - Results for the first mechanism

Nodes	2nd approach			
	Upper limit	Lower limit	Mean	Standard deviation
6	17	17	17	4
8	29	27	28	8
10	43	41	42	12
12	61	57	59	16
14	79	73	76	22
16	99	93	96	29
18	122	114	118	37
20	147	137	142	45
22	173	161	167	54
24	200	186	193	64

Table 5.2 - Results for the second mechanism

A last comparison can be made. For both broadcast schemes, the best and worst possible topologies are the same and are illustrated in Figure 5.5 and Figure 5.6.

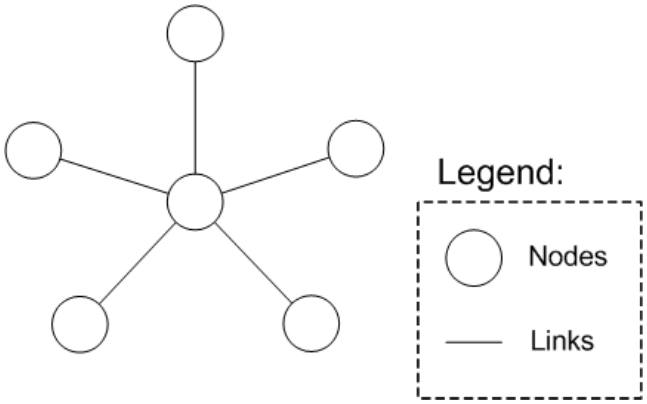

Figure 5.5 - Best possible topology

The best possible topology is the one in which N-1 out of N nodes composing the topology are connected to the N^{th} node and to that node only, becoming leaf nodes. With this topology a broadcast from any of the N-1 leaf nodes will require only two retransmissions, including the original transmission, to reach all nodes in the network. On the other hand, if the N^{th} node, the one to which all the others are connected, originates a broadcast no retransmission is necessary, besides the original transmission.

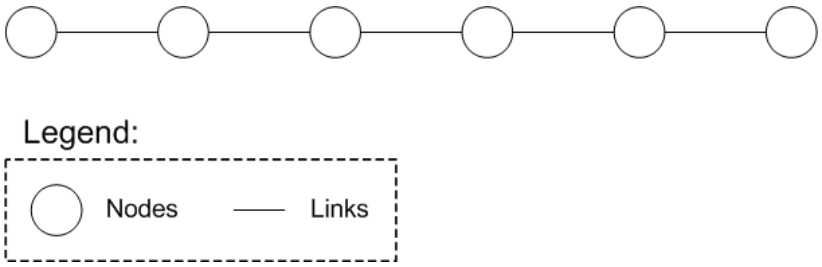

Figure 5.6 - Worst possible topology

The worst case is a chain topology. In this case, no matter the number N of nodes composing the network, there are only 2 leaf nodes. This means that if one of those leaf nodes transmits a broadcast frame, it will be retransmitted N-1 times: the original transmission plus N-2 retransmissions from non-leaf nodes. If one of those non-leaf nodes transmits one broadcast, it will be retransmitted N-2 times including the original transmission.

Plotting the points for these cases, the results are shown in Figure 5.7.

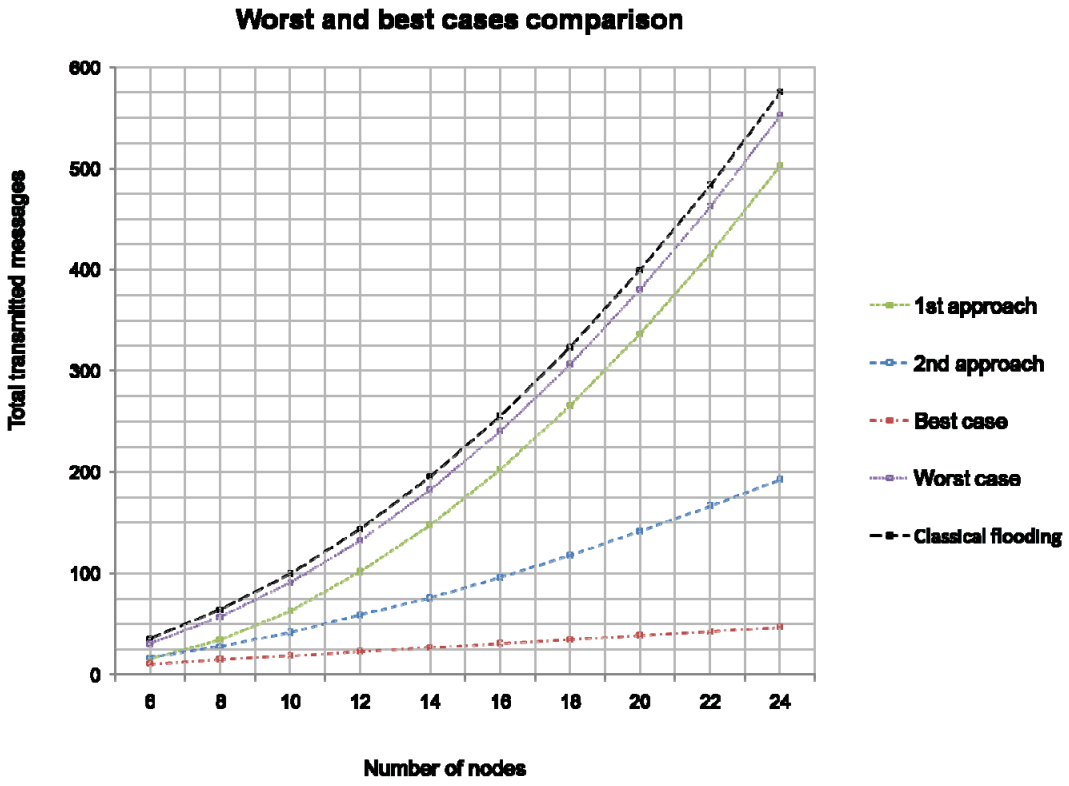

Figure 5.7 - Comparison between the proposed solutions, classical flooding and extreme cases

It is clear that even in the worst case, the performance of either proposed approach would still be better than classical flooding. However, the first approach is dangerously near the worst case limit, while the second approach mechanism is unambiguously closer to the best case performance. This demonstrates that following the already existing tree topology is much better in terms of real world implementation, since a large number of topologies will lead to results close to the best case.

A real world OHAP implementation with more than 15 or 16 nodes is deemed very unrealistic. But even in that situation, the second broadcast mechanism is, with no doubt, more advantageous than any other mechanism here in discussion.

5.3 Implementation

According to the simulation results, the broadcast scheme best suited for implementation, is the one that takes advantage of the tree topology already used by One Hop Ad hoc Protocol (OHAP).

One of the main issues related with this technique is the detection of duplicate frames. MANET oriented protocols that use broadcasting need to implement duplicate detection mechanisms in order to keep the wireless medium free from unnecessary retransmissions. The great majority of these protocols rely on unique identification (UID) values sent along with the information exchanged, so that communicating nodes can determine whether they are or are not processing information that has been already processed. In chapter 2, the duplicate detection mechanisms of Multicast MANET Routing Protocol (MMARP) [7] and Simplified Multicast Forwarding (SMF) for MANET [8] were mentioned and both use the UID technique afore mentioned. MMARP [7] trusts in a sequence number, as UID value, that is included in all control messages. Not only is it used to prevent duplicates but also for avoiding loops in the routes. On the other hand, SMF, given its forwarding only nature, makes use of the IPv4 and IPv6 headers to carry its UIDs. For both IPv4 and IPv6, SMF [8] defines a sequence number like MMARP does. Moreover, for IPv6, SMF [8] provides an alternative for the sequence number based duplicate detection: a hash value. The immutable information and header options of IPv6 multicast header plus the data in the packet are used for hash calculation. Another value is also used in hash calculation and SMF [8] defines it as Hash Assist Value (HAV). HAV is sent on the IPv6 hop-by-hop header option and behaves like a sequence number. A table of source addresses and associated hash values is maintained in order to verify whether incoming frames are duplicates.

Initially, for the developed solution, it was considered using the sequence number UID technique to implement a duplicate detection scheme. However, the base solution is implemented to operate at layer 2, thus neglecting upper layers specificities. This means that no interaction with layer 3 protocols is desired to maintain the independence of the solution. So it was compulsory to find a method to insert a UID into the Ethernet II frames used by OHAP. But without available header fields to insert information, the UID would have to be sent together with data. Remember that data in the protocol frames, since it uses encapsulation, means another frame as explained in chapter 3. Using this method would result in the frame represented in Figure 5.8.

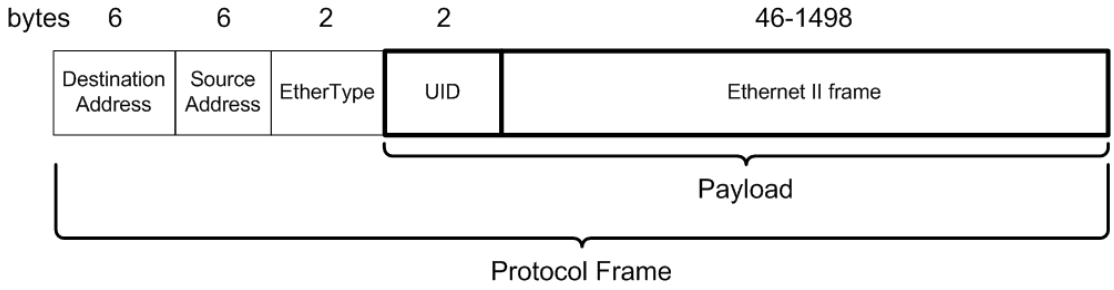

Figure 5.8 - Resulting frame by the addition of a UID

This approach is unrealistic since it might get very difficult to correctly process frames and UID. For better understanding this issue, consider that a broadcast frame has been received by a virtual interface. To send it, it is necessary to encapsulate the frame. So a UID value is set and the frame to forward is appended to it. Both these values compose the data to be encapsulated and so far, no problems have affected the solution. However, if the frame received for forwarding has the maximum length allowed, inserting the UID value will originate data corruption. Reducing maximum transfer unit (MTU) of virtual interfaces to provide space to accommodate UID is possible, but it would result in prejudice for unicast frames depriving them from the bytes that would be used for UID, since they do not have one. Moreover, the MTU of the virtual interfaces is already reduced in OHAP to avoid that the inner frame exceeds the payload length of the outer frame.

A feasible way for avoiding the problem is to use SMF's hash approach, but without the HAV value. In fact, calculating a digest with the frame to be encapsulated as the input for the hash function would result in a very good, per frame, UID. This is the approach taken for implementing the duplicate detection mechanism for the developed solution.

So, when a node has to broadcast a frame it calculates the corresponding hash value and records it along with its own physical address, constituting a pair of values to identify the frame. Then it proceeds to encapsulation, as done in the original OHAP solution except that the Destination Address field of encapsulating frame is set to the broadcast address, and broadcasts it. Neighbor nodes, upon receiving the broadcast, remove the outer frame header, conserving the source address, and calculate the hash for the inner frame. Then, based on the source address and the hash value, they verify if the pair is unique. If it is, they rebroadcast the frame. Otherwise, the frame is dropped. When the originator node receives its neighbors broadcast, it performs as they did. It removes encapsulation, preserving source address and hashes in the inner frame. When the previously calculated

and stored digest is matched with the one for the frame just received, source addresses (the previously recorded and the new one) will fail to correspond, since the originator node kept its own address as the address associated to the digest. Hence, the duplicate frame is detected and dropped. A node's own address is only associated with a digest if the hashed frame was locally generated. Moreover, frames arriving from a node connected to the network, but not participating in OHAP, are treated as locally generated by the node to which the former is connected, except that the originator address is the source address of the received frame (hence, originator address). There is also the possibility of a node sending two or more equal frames, which are not the same frames. To avoid nodes processing the second frame as a duplicate, since this results on digest collision, when two or more consecutive frames are received for forwarding and their associated pairs source address and hash value completely match, the frame is considered as a new frame equal to a previous one and is forwarded. This procedure does not result in unnecessary retransmissions because each node transmits each broadcast frame only once and ignores all duplicates. So, two equal and consecutive frames from the same source mean two different transmissions.

To enable OHAP nodes to have virtual connections with each other, each one has a virtual interface per neighbor. The virtual interfaces are inserted in software bridges, whose forwarding mechanism is used to correctly deliver frames exchanged between the nodes. A broadcast frame not dropped by the duplicate detection mechanism at arrival and, for that reason, eligible for forwarding, will be sent by the bridge to all of its ports except the one from where the frame was received. Thus, all virtual interfaces that receive the frame will try (and succeed if nothing is done) to send the frame to the wireless medium. This situation is clearly the one to be avoided, once it will originate several unnecessary retransmissions. To prevent it, all copies of the broadcast frame except one sent by the bridge to virtual interfaces must be dropped. It does not matter from which virtual interface the broadcast is sent. The main point here is that after one of the virtual interfaces sends the broadcast frame, typically the first to be listened, all other interfaces trying to send it must be silenced. To handle this problem, the duplicate detection mechanism will also check outgoing frames for duplicates when they are delivered to the virtual interfaces, in order to avoid retransmitting the same broadcast frame. This mechanism is illustrated in Figure 5.9.

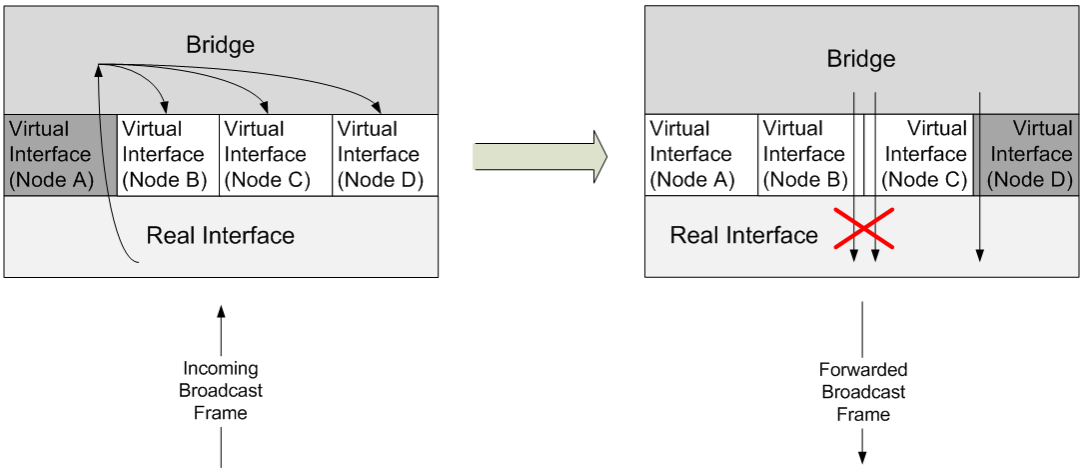

Figure 5.9 - Conceptual mechanism to avoid unnecessary retransmissions

The information needed to perform the duplicate detection can be stored in the type of data structure deemed more convenient and each node participating in OHAP has to maintain such a structure. In the present case, a linked list was used as the supporting data structure. The information stored is only relevant locally. Each entry in the structure refers to a single frame. Before explaining which data has to be stored for the duplicate detection mechanism to work properly, it is important to say that although two or more equal frames from the same origin are treated as different frames, they share the same entry. This avoids keeping more than one entry with the same source address and hash values, which would make the search difficult. When there is a positive match for such a frame the entry is updated immediately.

Each entry must hold the following data:

- SRC_ADDR
- DIGEST
- FWD_COUNT
- FWD
- FRM_COUNT
- VIF
- TIME

SRC_ADDR – stores the physical address of the frame originator. As said earlier, if the frame is locally generated (or has come from a node not participating in OHAP, but it is connected to the network through a node that participates), the field should be set to the physical address of the node in which the frame was generated (the source address field of the frame). It is set to the sender's address on creation.

DIGEST – stores the hash result of the encapsulated frame. It is set to the hash value on creation.

FWD_COUNT – this field is incremented by 1 for every frame forwarded by the node. It can occur more than once, if two or more equal frames from the same origin are received. Otherwise, its value is only incremented once. It is set to 0 on creation.

FWD – holds the number of times a frame must be forwarded. Basically, it counts the number of equal frames received from SRC_ADDR. If two or more equal frames from the same origin are received, is incremented by 1, for each frame. It is set to 1 on creation.

FRM_COUNT – this field is incremented by 1 for each corresponding duplicate frame received. If two or more equal frames from the same origin are received, it is incremented by 1 for each frame, on entry update. It is set to 1 on creation.

VIF – stores the number of virtual interfaces in the node. If two or more equal frames from the same origin are received, it must be incremented, for each frame, by the number of virtual interfaces in the node as on entry update. Basically, counts the number of duplicate frames expected to receive. It is set to the number of virtual interfaces as on creation.

TIME – stores a time stamp for the instant that the entry was created. It is used to determine when the entry is no longer considered valid. If two or more equal frames from the same origin are received, it is reset to current time as on entry update. It is set to the current time as on creation.

Every time a new broadcast frame is to be forwarded by a node, either received from a neighbor or locally generated, an entry with the fields described above must be crated and stored. If an entry already exists, it must be updated.

FWD_COUNT and FWD fields are jointly used, so that the duplicate detection mechanism can decide whether or not a frame should be forwarded. When a frame is received in a virtual interface, if FWD_COUNT is less than FWD the frame is forwarded. Otherwise it is dropped. When the frame is dropped, FRM_COUNT is incremented since the frame is a duplicate. FRM_COUNT is created with the value "1" because the

forwarded frame is also accounted (as a duplicate of itself). When FRM_COUNT is equal to VIF, it means that all duplicates have been detected, so the entry can and should be deleted. Should this fail to happen, most probably due to frame loss in the wireless medium, and actual time exceeds TIME by TIMEOUT the entry is deleted. TIMEOUT is a variable set to indicate for how long an entry should be maintained, to avoid storing stale information and, consequently, rejecting valid frames. It has to be big enough to guarantee that all neighbors have time to receive, process and forward the frame. For testing and debugging purposes, the TIMEOUT value was set to 2000 ms, which is much more than enough. Typically, a good value for TIMEOUT would be around 125 ms, considering full length frames, the lowest transmission rate and a realistic number of neighbors (1518 bytes, 1 Mbps and 5 neighbors, respectively). Furthermore, when a node loses connectivity with the tree or changes its up neighbor, all its entries must be deleted, since they do not keep valid, or at least relevant information anymore.

The broadcast frames are submitted to the SHA1 hash function for digest calculation. Hashes resulting from SHA1 application are 160 bits (20 bytes) long and have an extremely small collision domain, so it is almost impossible to have frames being dropped due to hash collisions.

Based on the description made so far, the following algorithm was created and added to OHAP implementation, with the purpose of endowing the protocol with more efficient broadcast handling capabilities:

```
Frame received from interface
If broadcast frame and is not a protocol message and is from a neighbor
        Calculate hash
        If no entry for received frame
                Create entry
                Send frame to the respective virtual interface
        Else
                If recorded source address matches current frame source address
                        Update entry
                        Send frame to the respective virtual interface
                Else
                        Increment duplicates count
                        If duplicates count match total number of virtual interfaces
                                Remove entry


Frame received from virtual interface
If broadcast frame
        Calculate hash
```

```
If matches an existing entry
        If matches a locally generated frame
                If not duplicate
                        Update entry
                        Encapsulate and transmit frame
                Else
                        Increment duplicates count
                        If duplicates count match total number of virtual interfaces
                        Remove entry
        Else
                If frame has not been forward yet
                        Encapsulate and transmit frame
                Else
                        Increment duplicates count
Else
        Create new entry
        Encapsulate and transmit frame
```

Chapter 6

6 Work Evaluation

The evaluation of the present work required the development of a simple application for broadcast traffic assessment, since it is oriented to solve the broadcast inefficiency. The application is a broadcast traffic generator and was built under the client-server paradigm. It sends dummy broadcast frames with a sequence number on the server side, which are captured and accounted on the client side, based on the sequence number. The output of the application is expressed in terms of bandwidth consumption and total frame loss. It is possible to set the size desired for the frames of the generated broadcast flow, so that one can compare the performances of the implemented solution against the original One Hop Ad hoc Protocol (OHAP), for broadcast flows with different frame sizes.

To evaluate the developed solution for real world operation conditions, it was used the *Iperf* traffic generator. This tool provides a way to stress a network with TCP and UDP traffic, in order to find its limits. Furthermore, the results obtained are compared with the original OHAP, to infer any improvement obtained with the present work. The Ad hoc On-demand Distance Vector (AODV) [2] and the Optimized Link State Routing (OLSR) [3] protocols were also submitted to testing, due to their status as reference ad hoc protocols, so that a comparison could be established between the developed solution and universally proven and accepted protocols.

The tests were made in two phases. The first phase is concerned with broadcast efficiency measurements, with the traffic generator developed, for both the implemented solution and the original OHAP. The second phase refers to real world traffic handling capabilities where all protocols mentioned earlier in this section are compared. In both phases of testing, the measurements are obtained, for 1 hop and 2 hop distances between communicating nodes. The mobility issue, quite common in this type of networks, is also taken into account by forcing the topology to change while performing measurements. The topologies considered for these tests are presented in Figure 6.1**,** where configurations used for both 1 hop and 2 hop tests are depicted.

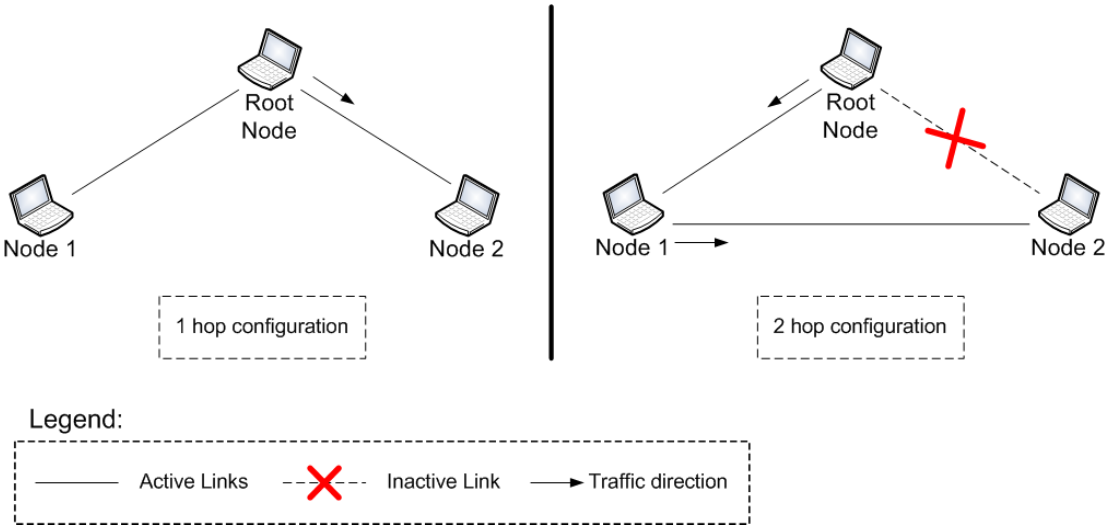

Figure 6.1 - Topologies used for tests

There is a built-in mechanism in OHAP (ported to the new solution) for blocking MAC addresses. It can be accessed by Telnet on any IP address defined in the node on which it is desired to enable blocking. The commands available for doing so are *block* and *accept* followed by the MAC address(es) to block or accept, respectively. In addition, the *maclist* command is defined for listing currently blocked addresses. So, to simulate the 2 hop distance between nodes and node mobility, the Root Node address is blocked on Node 2 forcing the former to create a link with Node 1. For the 2 hop test, the blocking is imposed prior to measurements, while for the mobility test, blocking is forced during measurements. In the AODV and OLSR tests, the iptables application was used for the same purpose.

6.1 Broadcast performance

This set of tests was executed for the original OHAP solution and the present one. The tests consisted in stressing both solutions with three broadcast flows, one at a time, of different frame sizes. The parameters measured were the bandwidth consumption and the percentage of frames lost for the three flows. The chosen frame sizes, in bytes, were 64, 718 and 1500. They correspond to minimum, intermediate and maximum frame size, respectively. The bandwidth results represent the raw bit rate achieved rather than useful data rate.

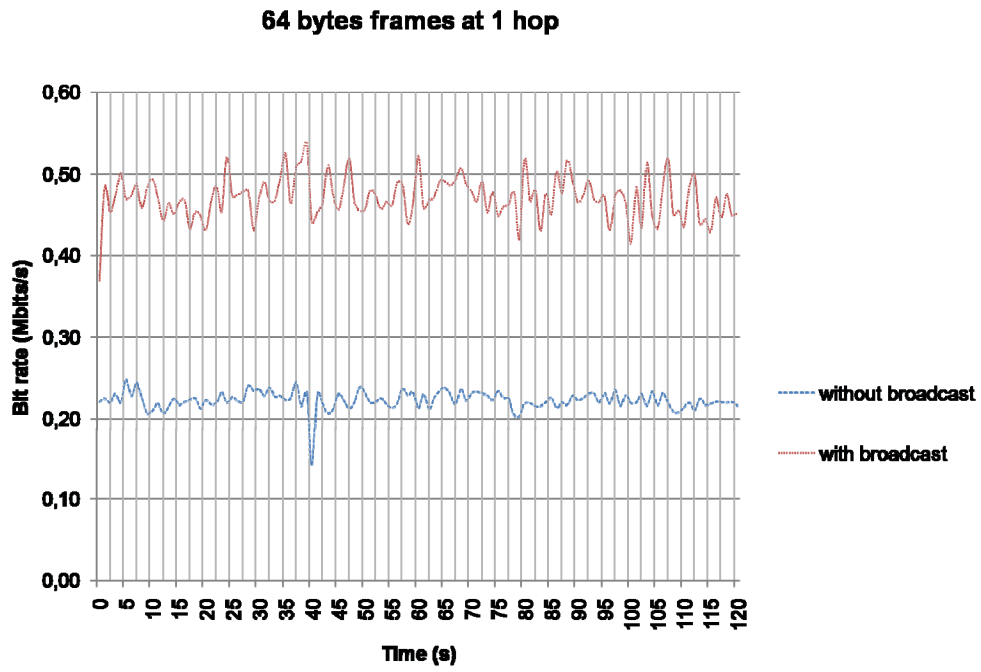

Figure 6.2 - Comparison between original OHAP and the new solution for 64 bytes frames exchanged by nodes at 1 hop from each other

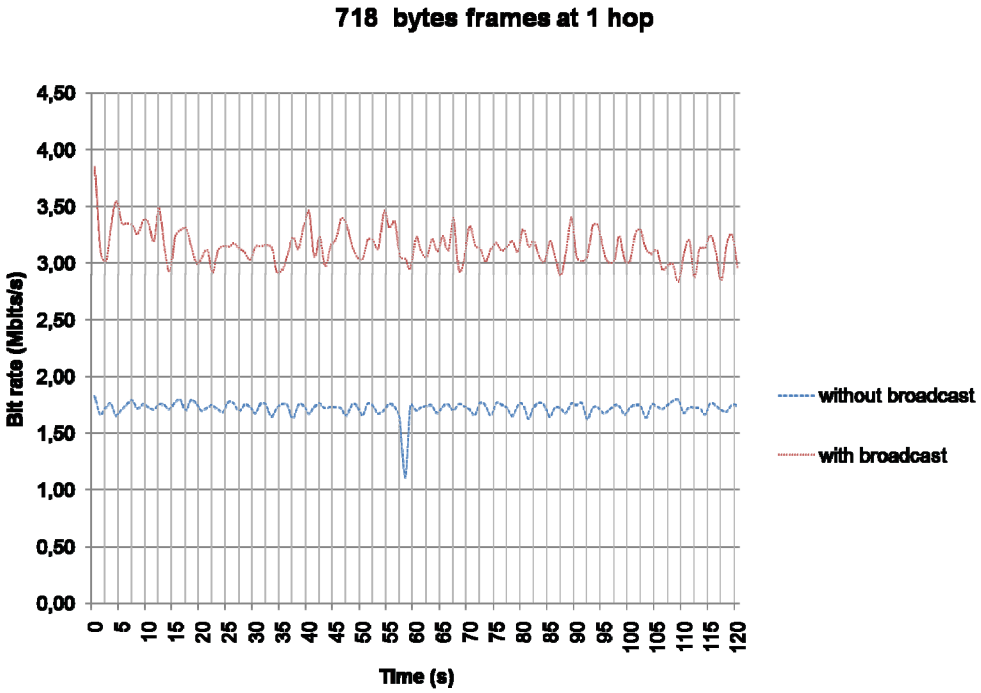

Figure 6.3 - Comparison between original OHAP and the new solution for 718 bytes frames exchanged by nodes at 1 hop from each other

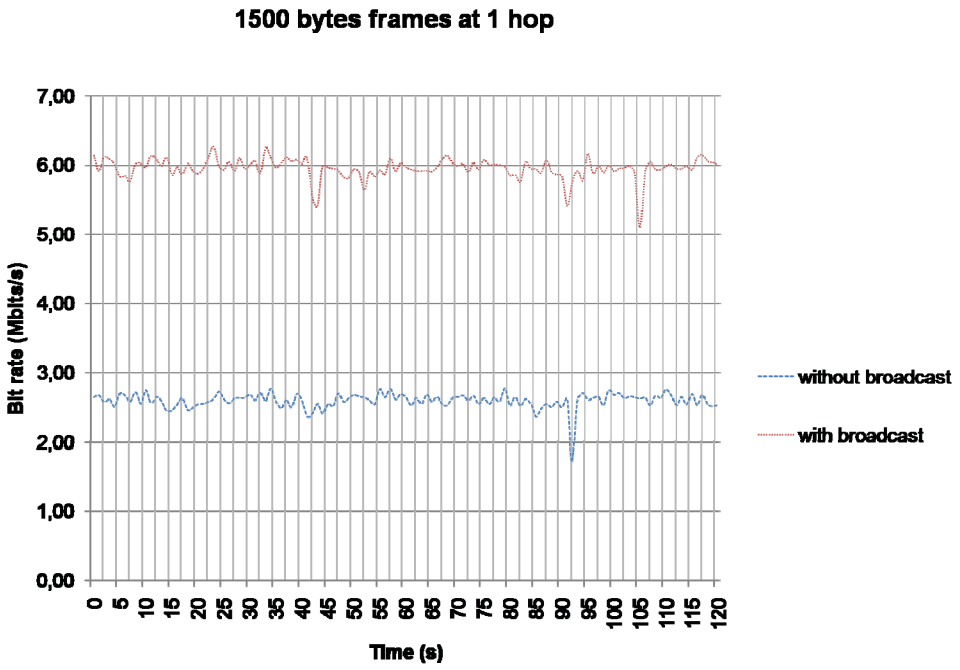

Figure 6.4 - Comparison between original OHAP and the new solution for 1500 bytes frames exchanged by nodes at 1 hop from each other

For the 1 hop test (Figure 6.2, Figure 6.3 and Figure 6.4), it is clear that the new solution is far better than the original OHAP. The bit rate values achieved by the new implementation, in all three cases, are around twice as much as the original OHAP. This happens because the original solution supports broadcast in unicast, due to the encapsulation method used. To be able to deliver frames in the hop-by-hop fashion mentioned in chapter 3, the encapsulating frame has its Destination Address field set to a unicast address. This means that a broadcast frame originates as many unicast frames as the number of virtual interfaces on the node. So, when receiving a broadcast frame, the Root Node generates two copies of it: one copy for Node 1 and another copy for Node 2.

With the new broadcast scheme, the Root Node only sends one copy of the broadcast frame, which is captured by both attached nodes, since the Destination Address of the encapsulating frame is the broadcast address.

Another characteristic observed in the plots is that the frame size influences the maximum achievable bit rate. Bigger frames get higher bit rates, as it is supposed to, since bigger frames result in lesser interframe spacing (IFS) periods. Moreover, if the frames are big, it means larger transmitting periods, which compared to the IFS turns them negligible when compared with the transmitting time. When the frames are small, the transmitting

period is of the same magnitude as the IFS. This situation results in more IFS periods, drastically reducing the maximum bit rate.

	Frame loss (%)	
Frame size (bytes)	Original OHAP	New solution
64	0	0,93
718	0	0,36
1500	0	0,16

Table 6.1 - Loss results for the 1 hop broadcast test

In what concerns to frame loss, the values in Table 6.1 show that losses only occur on the new solution. Unlike unicast frames, broadcast frames are not confirmed in IEEE 802.11 networks, which leads to a certain amount of loss. Anyway, the loss values do not compromise the new solution since they are not significant. The fact that just one of the nodes (the Root Node) was accessing the wireless medium helps this situation.

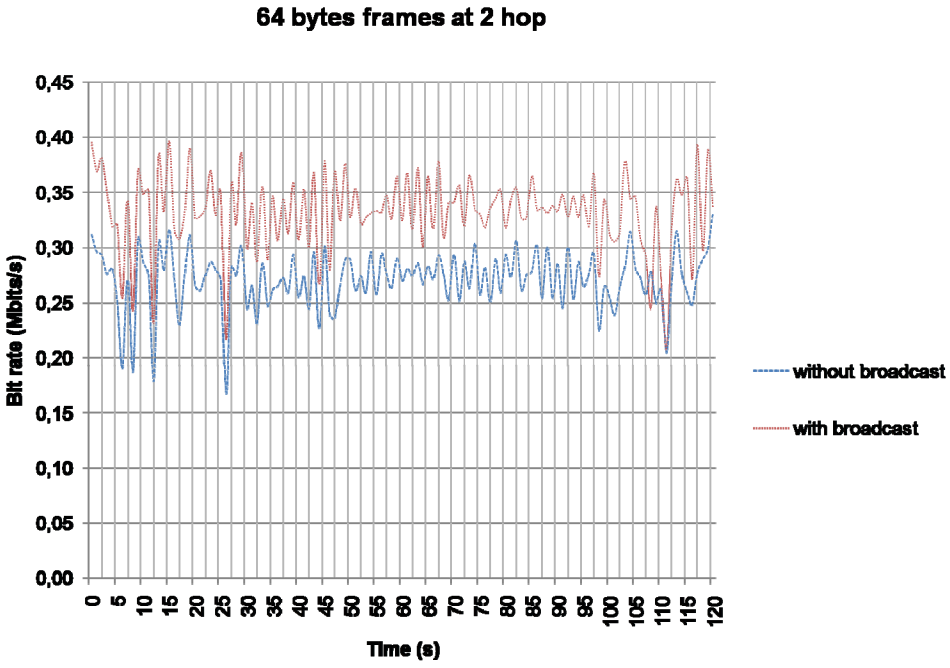

Figure 6.5 - Comparison between original OHAP and the new solution for 64 bytes frames exchanged by nodes at 2 hop from each other

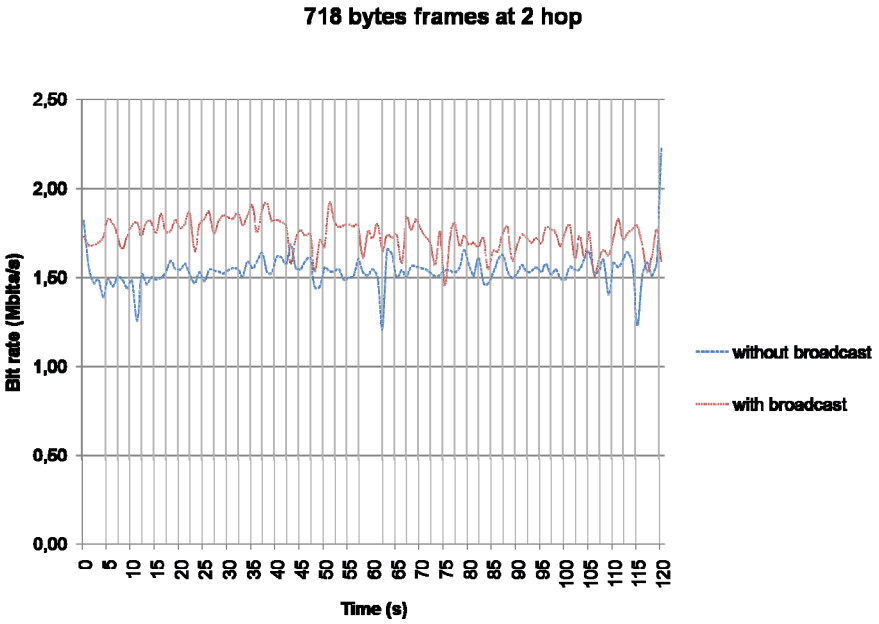

Figure 6.6 - Comparison between original OHAP and the new solution for 718 bytes frames exchanged by nodes at 2 hop from each other

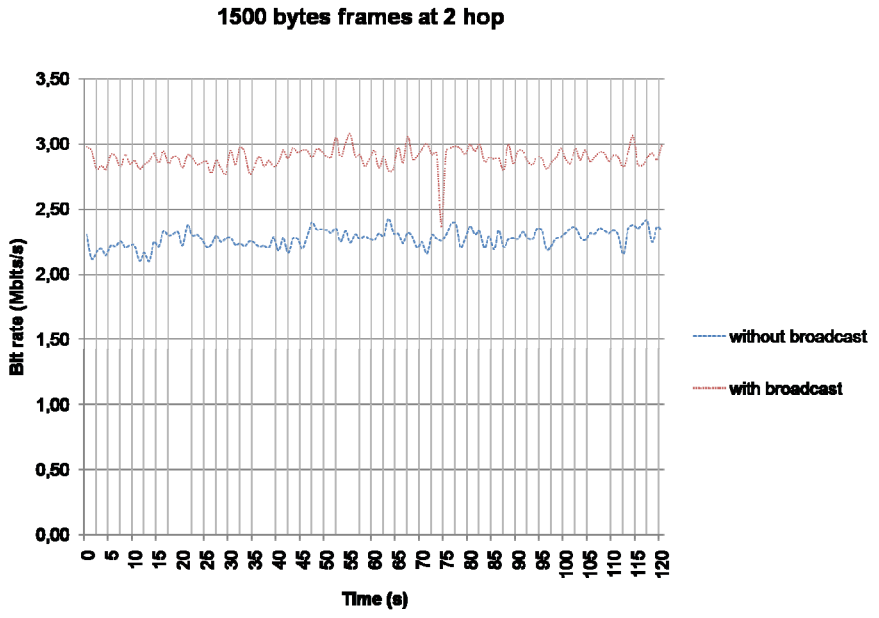

Figure 6.7 - Comparison between original OHAP and the new solution for 1500 bytes frames exchanged by nodes at 2 hop from each other

The values for the 2 hop tests (shown in Figure 6.5, Figure 6.6 and Figure 6.7) show that the performance of the new solution drops to bit rates near the ones for the original OHAP implementation. Still, they are lower for the later. As said before, IEEE

802.11 does not confirm broadcast frames; thus, since the control frames for that effect are not exchanged, there are some small gains for each flow on the new solution. The bit rate dropping was expected, since the number of transmitting nodes has risen. With more nodes accessing the wireless medium there is less bandwidth available for each node. However, this also means that with a bigger network, for example with two more nodes attached to Node 1, the new solution would surpass even further the original OHAP, since in that case, the number of messages in transit in the new solution would still be 2, but the original OHAP would raise this number to 4.

The bit rate reducing along with the frame sizes, as seen in the 1 hop tests, is also verified for the 2 hop test and for the same reasons.

	Frame loss (%)	
Frame size (bytes)	Original OHAP	New solution
64	2,68	2,27
718	24,03	14,05
1500	29,56	24,41

Table 6.2 - Loss results for the 2 hop broadcast test

As for the loss values (shown in Table 6.2), the original OHAP now shows some losses. The new solution shows a lower percentage of lost frames, which can be justified with the fact that there are less frames in transit with the new solution, since in that case the IEEE 802.11 control frames are not present. Furthermore, in this situation there are two nodes trying to access the wireless medium and thus collisions are expected. For each unicast frame exchanged there are three control messages exchanged as well. So, despite the collision avoidance mechanisms used by IEEE 802.11, more frames are synonym of higher collision probability and bigger frames raises even more this probability, since the wireless medium is occupied for longer periods. When the network is being stressed, as is the case, one should expect relatively high loss values.

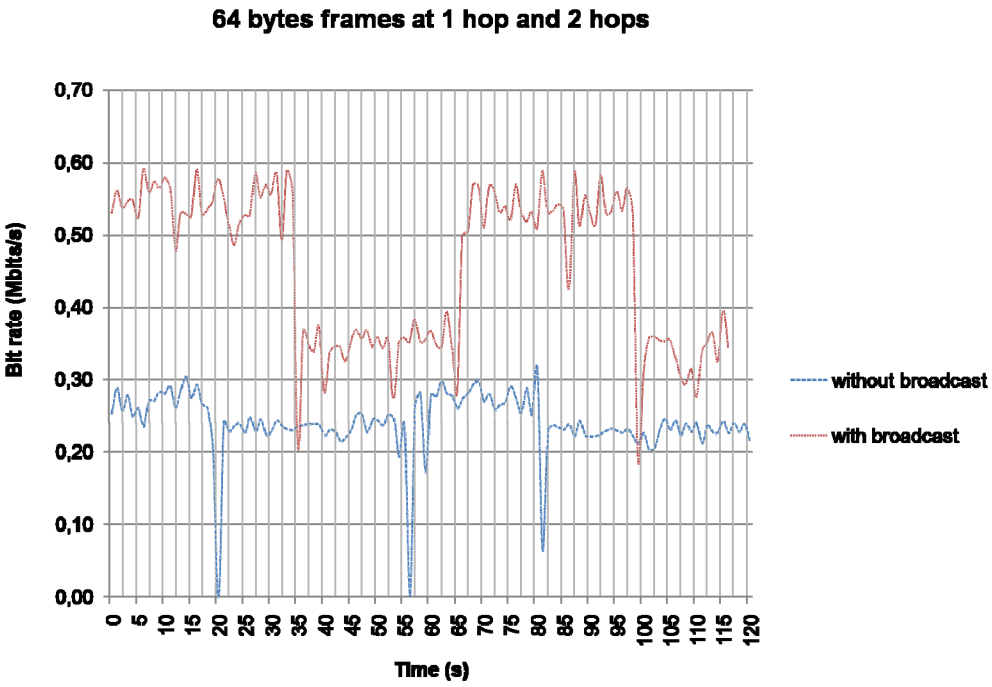

Figure 6.8 - Comparison between original OHAP and the new solution for 64 bytes frames exchanged by nodes in a mobility scenario

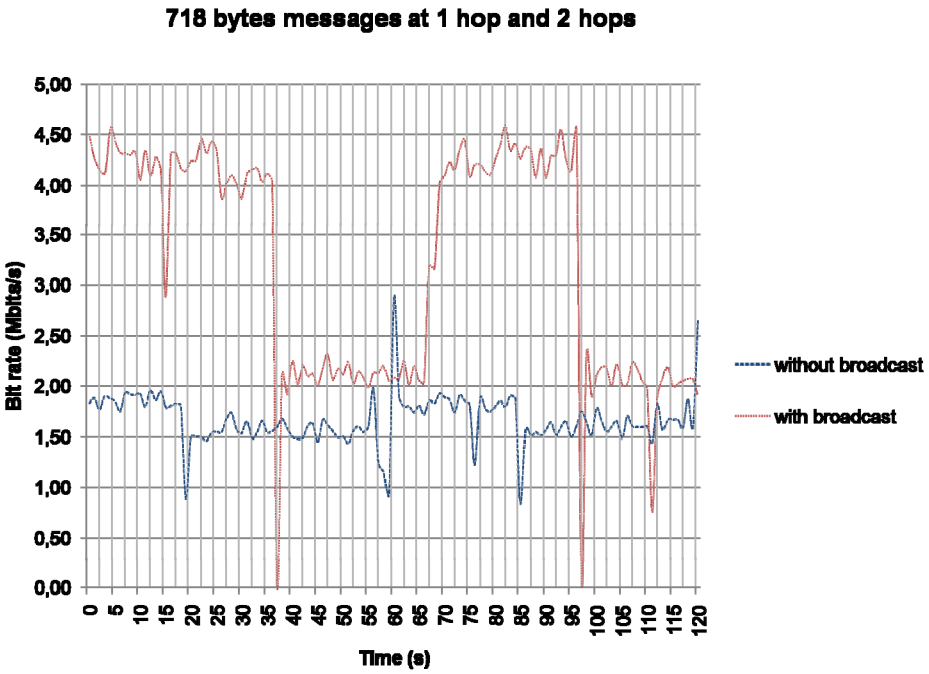

Figure 6.9 - Comparison between original OHAP and the new solution for 718 bytes frames exchanged by nodes in a mobility scenario

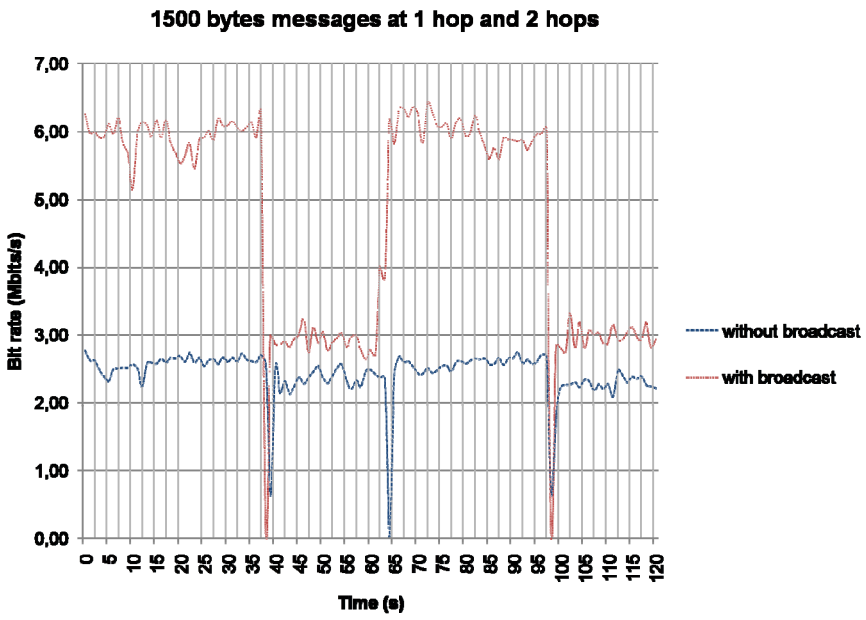

Figure 6.10 - Comparison between original OHAP and the new solution for 1500 bytes frames exchanged by nodes in a mobility scenario

The last three plots of this set of tests (Figure 6.8, Figure 6.9 and Figure 6.10) correspond to the mobility situation. Considering the values for the new solution it is clear in each image that when a node changes its position in the topology, it becomes affected, according to the expectations. Initially, the nodes are connected according to the 1 hop configuration showed in Figure 6.1, in the beginning of this chapter. Then, Node 2 is forced to connect to Node 1 (2 hop configuration of the same image). Some seconds later it returns back to the previous configuration and some more seconds afterwards, it is forced into the 2 hop configuration again. These changes correspond to the three jumps that can be observed in the plots. When Node 2 is connected at 1 hop from the Root Node, the bit rate associated is of the same order of magnitude as the one in the 1 hop test. The same occurs for the 2 hop configuration. So, mobility plots present themselves as an overlap of the 1 hop and 2 hop plots between each transition instant, as expected.

The original OHAP solution has the same behavior although it is not so perceivable. The reason why this occurs is related with the fact that in both 1 hop and 2 hop configurations, for each frame generated, two frames are sent. At 1 hop Root Node sends two copies (one copy for Node 1 and one copy for Node 2) and at 2 hops Root Node and Node 1 send 1 copy each (Root Node sends to Node 2 and Node 2 sends to Node 1). The only reason why the bit rates are not the same in both cases (original OHAP and new

solution) has to do, again, with the detail that only unicast frames are confirmed in IEEE 802.11. In fact, as the frame sizes increase, it is more difficult to define a gap between the 2 network configurations.

The transition instants, discussed earlier, lead to a much higher frame loss in comparison with all other instants, since no traffic is received when the changes occur. Nonetheless, because the adjustment is so swift, these losses barely affect the total loss, as one can see in Table 6.3.

	Frame loss (%)	
Frame size (bytes)	Original OHAP	New solution
64	8,74	2,27
718	13,28	11,94
1500	18,09	9,46

Table 6.3 - Loss results for the broadcast mobility test

Considering the values in Table 6.1, Table 6.2 and Table 6.3, it is easy to conclude that the losses in the mobility case are between the values for 1 hop and 2 hops results. Once again, the total loss values for the new solution are better than the original ones, for the same reasons given in the analysis of the 2 hop tests.

In summary, the improved OHAP solution shows to perform better than the original one, both as far as bandwidth consumption and losses, in the presence of broadcast traffic. There are major gains for 1 hop communication in terms of bit rates, even though the losses are higher. But considering the bit rate gains obtained, the loss values become negligible, since there are twice as much frames being transmitted on the same period.

When analyzing the 2 hops performance, the gains are more modest but still pretty clear. Furthermore, the losses in this case affect more the original OHAP than the new solution.

For the mobility case, considering the methodology adopted to accomplish the tests, it turned out to be a combination of the results of the 1 hop and 2 hop tests. It is so since the change in topology caused by the movement of nodes inside is quick enough and, therefore, the losses occurred in that instant practically do not affect the total loss.

However, considering a flow of real time information that is being transmitted in broadcast, for example video, and assuming that the losses are evenly distributed across

time, the losses values obtained indicate that neither solution is adequate. The losses are assumed to be evenly distributed, because there are no measurements for partial losses and because the bit rate plots are steady enough to consider it so. Anyway, for bursty non-real time broadcast traffic the results are considered good enough to allow acceptable levels of efficiency.

6.2 Real World Performance

The tests performed for this evaluation allow knowing the limits of the developed solution for the most common types of traffic present in today's networks. The results are compared to those obtained with the original solution, AODV and OLSR. The tests consist in stressing the network, for each of the protocols to test, with TCP and UDP traffic and measuring the associated maximum bandwidth, average jitter and total losses. For TCP traffic there is no jitter or loss reports since it is a reliable protocol, meaning that it warrants that all packets are delivered in the same order they were sent. The bandwidth results represent the throughput for each solution.

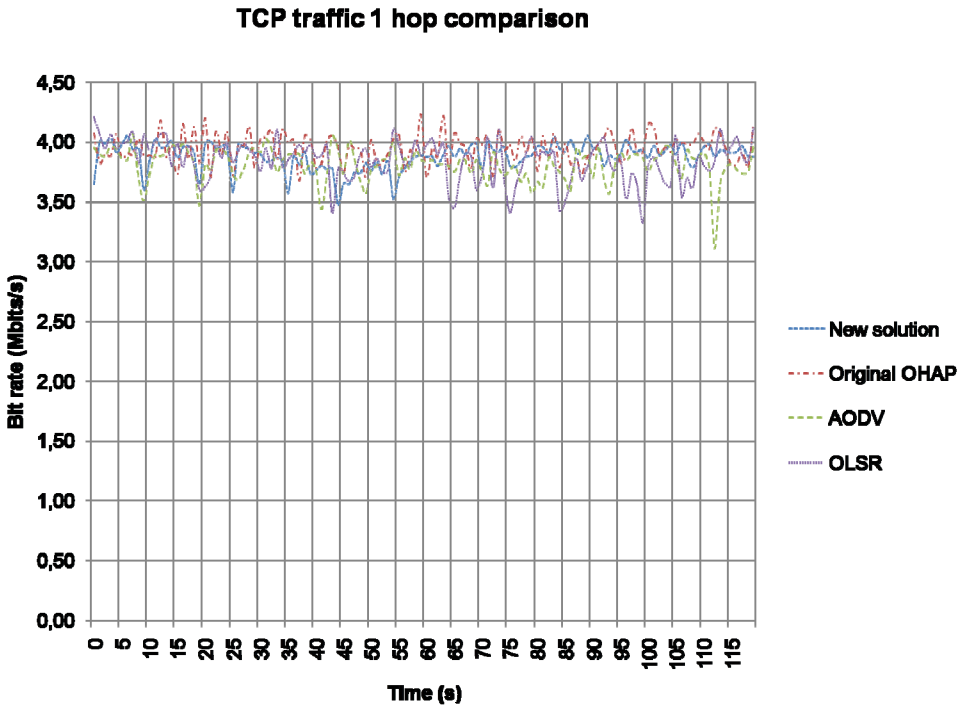

Figure 6.11 - Comparison between several solutions for TCP traffic exchanged by nodes at 1 hop from each other

The TCP traffic plot for the 1 hop configuration (Figure 6.11) comes with no surprises. All protocols present identical performance with this type of traffic. The nearly 4 Mbps mean bit rate value observed is the expected maximum throughput, considering that the tests were made over an IEEE 802.11b compliant ad hoc network. The maximum raw data rate possible in this type of network is 11 Mbps, which leads to a typical throughput around 4.5 Mbps.

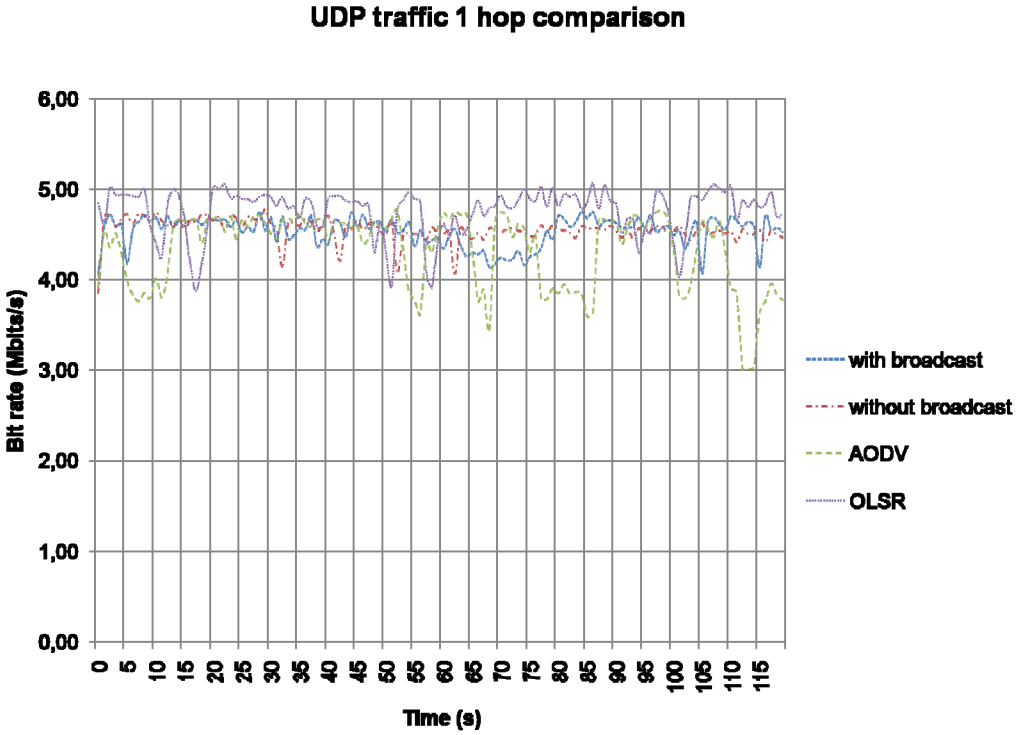

Figure 6.12 - Comparison between several solutions for UDP traffic exchanged by nodes at 1 hop from each other

UDP performance results (Figure 6.12) for the original OHAP implementation and the new one are according to what was expected. The bit rate increase in comparison with the TCP test is understandable since UDP, unlike TCP, is not a reliable protocol. This means that, it is intended for sending short packets that do not require guaranteed delivery. Since the overhead used for TCP traffic confirmation is not present for UDP, the later is faster and more efficient, although some packets may arrive out of order, or not arrive at all, and even appear duplicated due to the lack of confirmation. For these reasons, the results expected for the UDP tests were near 5 Mbps, just like the tests demonstrate.

Protocol	Average Jitter (ms)	Total Losses (%)
New solution	10,09	0,09
Original OHAP	12,96	0,14
AODV	7,82	5,5
OLSR	7,28	0,26

Table 6.4 - Jitter and loss values for the 1 hop test with UDP traffic

Considering the values in, Table 6.4 it is clear that the tested protocols handle the UDP traffic efficiently in the 1 hop configuration, barely getting affected by jitter or losses, exception made to AODV. AODV is more susceptible to losses, due to its reactive nature. In fact, every time an AODV node needs to transmit information, it first needs to establish a route. This route discovery process delays the traffic and leads to some losses if there are no control mechanisms over the information to transmit, as is the case with UDP traffic.

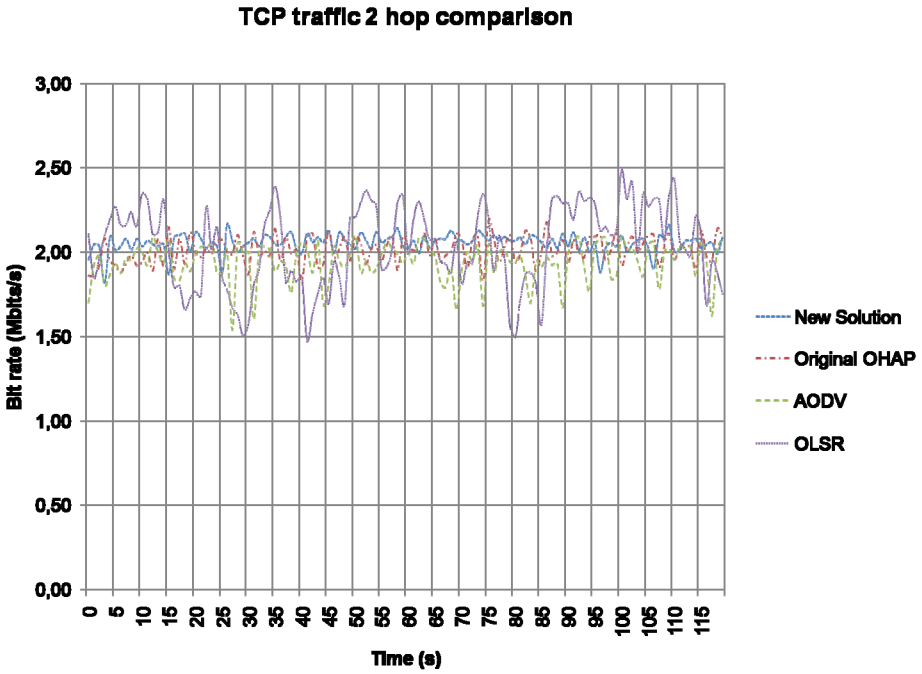

Figure 6.13 - Comparison between several solutions for TCP traffic exchanged by nodes at 2 hop from each other

Like the 1 hop test, the 2 hop test for TCP traffic fits the expectations (Figure 6.13). The average bit rate has lowered to 2 Mbps, half the bit rate for the 1 hop test, which makes sense, considering that there are two nodes accessing the wireless medium.

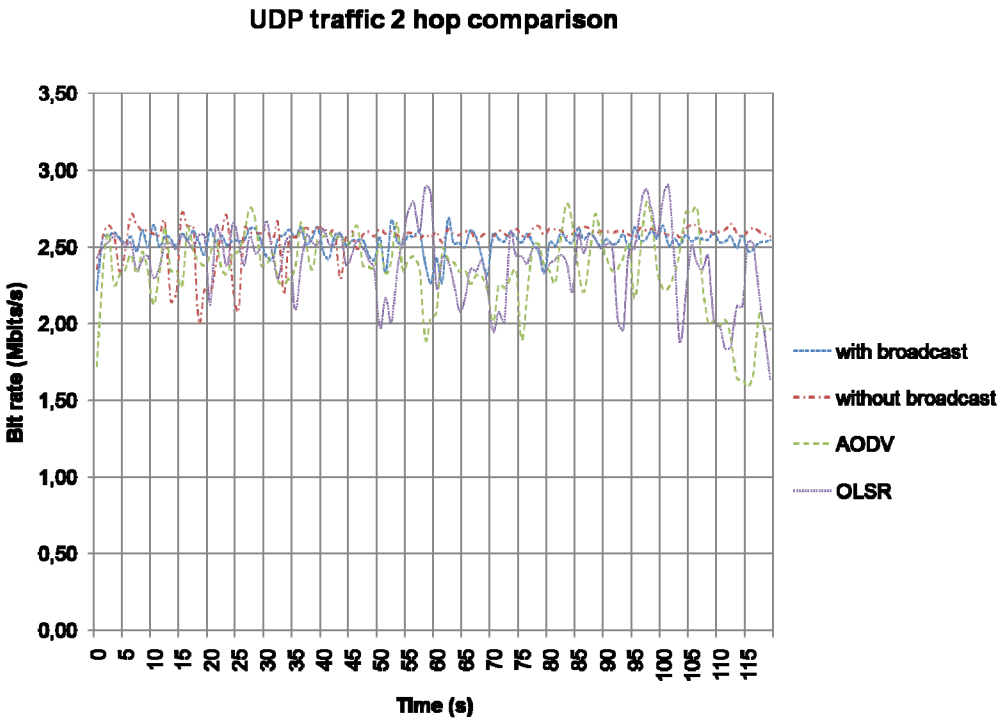

Figure 6.14 - Comparison between several solutions for UDP traffic exchanged by nodes at 2 hop from each other

The plot for the 2 hop test with UDP traffic (Figure 6.14) shows that the bit rate drops to half compared to the 1 hop configuration, just like in the case of TCP traffic.

Protocol	Average Jitter (ms)	Total Losses (%)
New solution	19,36	0,39
Original OHAP	21,32	0,69
AODV	13,19	5,70
OLSR	18,88	13,43

Table 6.5 - Jitter and loss values for the 2 hop test with UDP traffic

As for jitter and loss values (Table 6.5), the jitter has raised in the 2 hop configuration, for all tested protocols, to values approximately two times higher than the ones for the 1 hop test. It is not a surprising fact, since the bit rate has dropped to half, due to the particularity that each packet gets transmitted twice, thus delaying the flow.

AODV losses remain higher than the losses of the new solution and Original OHAP. OLSR behavior in terms of loss, however, is much higher than the remaining protocols tested. The proactive nature of the protocol leads to estimate a low loss value, but

that does not get confirmed. Probably, the exchange of topology information, to maintain freshness of routes, affected the overall performance.

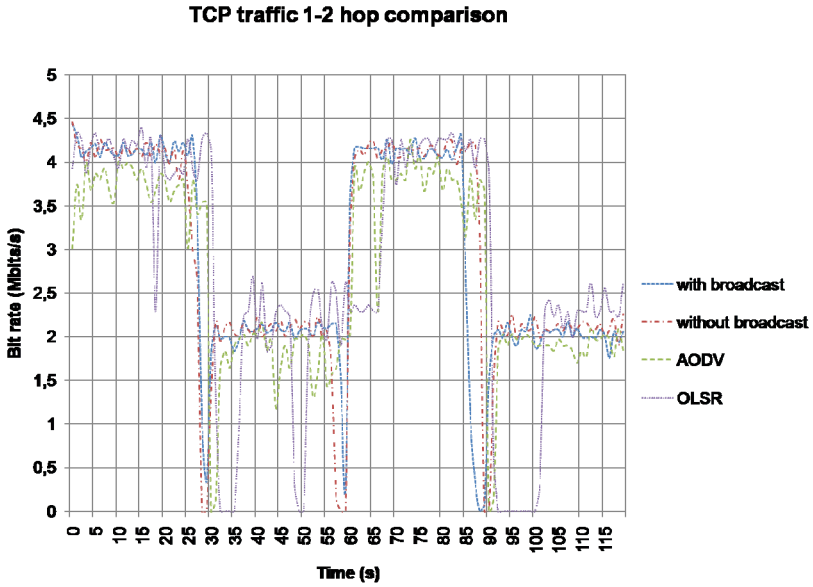

Figure 6.15 - Comparison between several solutions for TCP traffic exchanged by nodes in a mobility scenario

The results shown in Figure 6.15 lead to the same conclusions taken in the previous section, for broadcast traffic during mobility. The resulting plot can be seen as an overlap of the results for the 1 hop and 2 hop tests.

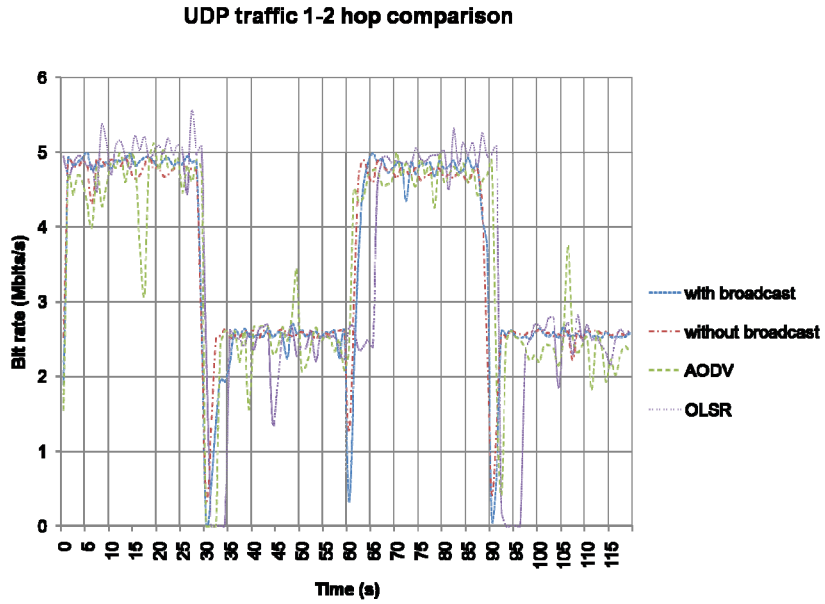

Figure 6.16 - Comparison between several solutions for UDP traffic exchanged by nodes in a mobility scenario

The mobility plot for UDP traffic (Figure 6.16), as the one for TCP, shows the overlap of the plots for 1 hop and 2 hop tests.

Protocol	Average Jitter (ms)	Total Losses (%)
New solution	19,93	6,05
Original OHAP	19,78	4,33
AODV	10,98	4,70
OLSR	10,25	12,00

Table 6.6 - Jitter and loss values for the mobility test with UDP traffic

Table 6.6 shows that node mobility is well tolerated by any of the protocols. Nonetheless, one ought to point out that the new solution and Original OHAP sustain a major loss increase in comparison with the previous 1hop and 2 hop tests. From the table values, one can deduct that both solutions do not handle mobility quite as well as AODV or OLSR. However, this situation can be improved. It is just a question of fine tuning some parameters, such as the TIMEOUT_VAL variable associated to the sending of HELLO messages and to the choice of an "up neighbor", mentioned in chapter 3. Despite this loss increase, both the original OHAP and the new solution present loss values capable of competing with AODV and OLSR. This particularity also affects jitter on both solutions, although not as much as the losses. In fact, jitter values for the mobility test are around the same values observed for the 2 hop test, while AODV and OLSR jitter is somewhere in between the values obtained for the 1 hop and 2 hop tests.

6.3 Summary

In general, all the tests discussed above, show that the changes introduced in the original OHAP implementation arise as valuable assets in what concerns the broadcast handling capabilities. Furthermore, those changes do not affect the performance of the solution in what comes to handling unicast traffic.

The performance values obtained for real world operation shows that the developed solution, as well as the original OHAP, is able to rival with established protocols like AODV [2] and OLSR [3], even though the purpose of both solutions is to expand an infrastructure network, rather than provide a routing implementation for ad hoc networks.

However, that purpose could not be achieved without the development of a forwarding mechanism.

Chapter 7

7 Conclusions

In this last chapter, the purpose and achievements of the presented work will be reviewed and summarised. At first, the work developed will be recalled by conducting the reader, step by step, through the course of tasks taken since the initial study until the final prototype. Then, a summary of the results and contributions will be presented, from the tests performed on the developed solution and the solutions that were put side by side with it. Finally, a brief discussion on future work aimed at the improvement of the developed solution will be carried out.

7.1 Work Review

The development of the solution described in this document, started with the study of broadcast mechanisms that could improve the One Hop Ad hoc Protocol (OHAP), since, although native to the protocol, the broadcast handling is highly inefficient. There were two proposals in evaluation.

The first one was based on the assumption that communication between *ad hoc* nodes was intense and, therefore, it would require an inbound mechanism to route traffic inside the mobile ad hoc network (MANET), since OHAP is established on a tree topology. It was considered using the Dynamic MANET On-demand (DYMO) routing protocol (discussed in chapter 2, section 2.7) to support this function, since it uses the Neighborhood Discovery Protocol (NHDP) (also discussed in chapter 2, section 2.4). The neighborhood information conveyed by NHDP [6] would be used to decide whether or not a node should retransmit a broadcast frame and, at the same time, communication between nodes inside the MANET would be improved by DYMO.

The second proposal was based on the assumption that communication between *ad hoc* nodes was vestigial, due to the global purpose of the solution: to expand an infrastructure network. Talking this into consideration, the only feasible alternative was to take advantage of the already available tree topology. So, it was necessary to override the encapsulation-based forwarding mechanism in OHAP, forcing it to handle broadcast

frames differently from using unicast frames. Each node in the tree, in case it had more than one neighbor would have to retransmit a broadcast frame only once for all neighbors, instead of retransmitting it in unicast, once for each neighbor.

Both suggested mechanisms were submitted to simulation and the second approach proved to have better performance as far as the number of retransmissions, the only parameter evaluated, by the way.

The next step ahead was implementing the mechanism approved in the OHAP prototype. The selected mechanism had to be carefully implemented, since it is a mechanism that strongly depends on duplicate detection schemes. Some techniques were equated, such as the ones used by Simplified Multicast Forwarding (SMF) for MANET [8] and by Multicast MANET Routing Protocol (MMARP) [7]. The hash technique, one of the suggested by SMF [8], was the basis for the mechanism implemented. Basically, the developed mechanism, implemented by every node, would calculate a hash value for each incoming and outgoing broadcast frame and keep track of all the hashes calculated, in order to detect which frames had already been processed and which ones had not.

Having this mechanism, all it was needed was to change the Destination Address of the encapsulating frames to the broadcast address, rather than the unicast addresses of its neighbors, when a node has to forward a broadcast frame and guarantee that only broadcast messages from neighbors (the tree neighbors) are processed, guaranteeing that the tree topology was followed.

7.2 Relevant results and contributions

Based on the results obtained from the broadcast tests performed to characterize the developed solution, it is possible to conclude that the advantages of the new broadcast scheme jump to sight. The results achieved show clearly that the original OHAP provides major benefits due to the broadcast mechanism in simple scenarios like the ones tested. The 1 hop test shows bandwidth consumption gains around 2:1 in comparison with the original OHAP, despite the total losses being higher. However, the loss values are not significant. The gains obtained for the 2 hop test, although not so overwhelming, still demonstrate undeniable gains that favor the new solution. In terms of losses, the new solution outperforms the original, unlike in the 1 hop case, proving to also be superior in this respect. As for mobility issues both solutions handle it swiftly, still with the advantage

on the side of the new solution both in terms of bandwidth consumption and total frame losses. From these results, one can extrapolate that for larger networks the broadcast mechanism implemented may reveal itself a breakthrough in terms of performance when dealing with broadcast traffic when compared with the original OHAP undifferentiated forwarding mechanism. The gains expected in such scenarios are around N:1, being N the number of neighbors that a node has connection with.

In what concerns real world operation, both the new solution and the original OHAP were validated as capable forwarding mechanisms in MANETs, with results similar to the performance of worldwide reference protocols as Ad hoc On-demand Distance Vector (AODV) [2] and Optimized Link State Routing (OLSR) [3]. The results obtained in the presence of TCP traffic show that both solutions, the new one and the original OHAP, present performances as good as AODV and OLSR for every scenario tested (1 hop test, 2 hop test and mobility test). The performance of the developed solution, in the presence of UDP traffic in what concerns bandwidth consumption is the expected. UDP traffic is shown to be well handled by both the new solution and the original OHAP, when compared to AODV and OLSR. In fact, the jitter and loss values, in all tests performed and in comparison with the two reference protocols, show that the solutions are capable of efficiently handling UDP traffic.

On the light of these results, it is fair to say that the developed solution is a valid one and that it reaches all goals set for this work.

7.3 Future work

The work developed had its base on the OHAP solution. OHAP has some problems to be solved. The protocol does not include any mechanism for verification of bidirectional links, and by extent neither the present solution does, since it was tailored not to deploy a full grown solution but rather to verify if its forwarding mechanism principle was a valid one. So, support to verify bidirectional links would be a good addition to the solution with the penalty of creating deceivable links, due to the lack of link symmetry.

Another improvement that could be made is the addition of more metrics to choose a neighbor. Currently, the choice is only based on the distance to the root node of the tree. Metrics like link quality and the number of neighbors hold by a node would provide more efficient choices in terms of overall network performance.

A reformulation of the message format to comply with Generalized MANET Packet/Message Format (packetBB) [5] should also be considered, in order to improve efficiency in the presence of other MANET protocols compliant with packetBB [5].

Finally, adopting the broadcast mechanism for multicast traffic should be regarded as an enhancement in the presence of multicast traffic. This poses as a relatively important improvement, since broadcast is just a particular case of multicast. If this work shows very significant results with the improvement made towards broadcast traffic, those results would also apply to multicast traffic.

References

1. **Sousa, Daniel.** Redes ad hoc como uma extensão da infra-estrutura. [Online] July 2007. [Cited: February 24, 2008.] http://paginas.fe.up.pt/~ee00195/docs/Relatorio.pdf.

2. **Perkins, C., Belding-Royer, E. and Das, S.** Ad hoc On-Demand Distance Vector (AODV) Routing. [Online] July 2003. [Cited: February 16, 2008.] http://www3.tools.ietf.org/pdf/rfc3561.pdf.

3. **Clausen, T. e Jacquet, P.** Optimized Link State Routing Protocol (OLSR). [Online] October de 2003. [Citação: 27 de October de 2007.] http://www3.tools.ietf.org/pdf/rfc3626.pdf.

4. **Tschudin, Christian, et al.** LUNAR: a Lightweight Underlay Network Ad-hoc Routing Protocol and Implementation. [Online] February 2004. [Cited: October 23, 2007.] http://cn.cs.unibas.ch/pub/doc/2004-new2an-lunar.pdf.

5. **Clausen, T., et al.** Generalized MANET Packet/Message Format. [Online] 10 de September de 2007. [Citação: 2 de November de 2007.] http://www3.tools.ietf.org/pdf/draft-ietf-manet-packetbb-09.pdf.

6. **Clausen, T., Dearlove, C. and Dean, J.** MANET Neighborhood Discovery Protocol (NHDP). [Online] June 29, 2007. [Cited: October 30, 2007.] http://www3.tools.ietf.org/pdf/draft-ietf-manet-nhdp-04.pdf.

7. **Ruiz, Pedro M., Gómez-Skarmeta, Antonio F. and Groves, Ian.** Multicast Routing for MANET Extensions to IP Access Networks: The MMARP Protocol. [Online] October 2002. [Cited: October 20, 2007.] http://ants.dif.um.es/staff/pedrom/papers/WS-WP2-MMARP-doc.pdf.

8. **Macker, J.** Simplified Multicast Forwarding for MANET. [Online] June 26, 2007. [Cited: November 28, 2007.] http://www3.tools.ietf.org/pdf/draft-ietf-manet-smf-05.pdf.

9. **Chakeres, I. and Perkins, C.** Dynamic MANET On-demand (DYMO) Routing. [Online] July 5, 2007. [Cited: November 13, 2007.] http://www3.tools.ietf.org/pdf/draft-ietf-manet-dymo-10.pdf.

10. **Clausen, T., Dearlove, C. and Jacquet, P.** The Optimized Link State Routing Protocol version 2. [Online] July 9, 2007. [Cited: January 15, 2008.] http://www3.tools.ietf.org/pdf/draft-ietf-manet-olsrv2-04.pdf.

11. **Takai, Mineo, et al.** PARSEC: Parallel Simulation Environment for Complex Systems. [Online] September 1999. [Cited: November 6, 2007.] http://pcl.cs.ucla.edu/projects/parsec/.

12. **Campos, Rui.** Shortest Tree Simulator. [Online] October 2007. [Cited: November 21, 2007.] http://telecom.inescporto.pt/~rcampos/STS.tar.gz.

13. **Campos, Rui and Ricardo, Manuel.** A Fast Algorithm for Computing Minimum Routing Cost Spanning Trees. *submitted to Elsevier Computer Networks Magazine.* 2007.