WIRELESS COMMUNICATIONS AND MOBILE COMPUTING
Wirel. Commun. Mob. Comput. 2009; **9**:1–23
Published online XX Month Year in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/wcm.916

RESEARCH ARTICLE

Network infrastructure extension using 802.1D-based wireless mesh networks

Rui Campos*, Ricardo Duarte, Filipe Sousa, Manuel Ricardo and José Ruela

INESC Porto, Faculdade de Engenharia, Universidade do Porto, Rua Dr. Roberto Frias, 378, 4200-465 Porto, Portugal

ABSTRACT

Ubiquitous Internet access is becoming a major requirement for end-users due to the increasing number of services and applications supported over the Internet. Extending the coverage of current Wi-Fi infrastructures installed in companies, universities and cities, has been considered a solution to help in fulfilling this requirement, namely when it comes to wireless and nomadic Internet access.

This paper describes and analyses a new and simple solution, called Wi-Fi network Infrastructure eXtension (WiFIX), aimed at extending current Wi-Fi infrastructures. WiFIX is based on standard IEEE 802.1D bridges and a single-message protocol that is able to self-organize the network, and it only requires software changes in IEEE 802.11 access points (APs); no changes to IEEE 802.11 stations are needed. Overhead analysis and experimental results show both the higher efficiency of the solution compared to the IEEE 802.11s draft standard and its good performance as far as data throughput, delay and packet loss are concerned. Copyright © 2009 John Wiley & Sons, Ltd.

KEYWORDS
wireless mesh networks; multi-hop; bridging; self-management

***Correspondence**
Rui Campos, INESC Porto, Faculdade de Engenharia, Universidade do Porto, Rua Dr. Roberto Frias, 378, 4200-465 Porto, Portugal.
E-mail: rcampos@inescporto.pt

UNCORRECTED PROOFS

1. INTRODUCTION

The Internet is becoming the global communication infrastructure for accessing multimedia services and data and, as such, it is the infrastructure users are willing to be permanently connected to. With its rapid growth as well as the demand for broadband services, access networks have been receiving intense investments in recent years. In this context, the IEEE 802.11 [1] standard, also known as Wi-Fi (the two terms will be used interchangeably), is assuming a prominent role. Wi-Fi infrastructures are being widely deployed around the world in university campus, corporations or even in entire cities, as a means of providing wireless and broadband access to the Internet. The success achieved by the Wi-Fi technology is related to three aspects: the use of unlicensed frequency bands, high bandwidth capabilities and low deployment cost. In addition, the integration of the technology within portable devices, from laptops to smart phones, has decisively contributed to further strengthen its success. Indeed, Wi-Fi is becoming a true universal wireless technology for accessing the Internet worldwide.

Due to its success and the increasing number of devices supporting the technology, the demand for Wi-Fi coverage is growing everyday. Since Wi-Fi has limited radio range, the coverage of large geographical areas can only be achieved by installing multiple Wi-Fi Access Points (APs) that, however, need to be connected to the wired backhaul Ethernet infrastructure. In some cases, this may require the installation of a large number of APs and wires, which introduces complexity, may imply high costs, and may involve considerable technical manpower to deploy the network. Also, the need to connect each AP to the wired infrastructure provides limited flexibility; often, the point of attachment of the APs is constrained by the deployment of wires and Ethernet sockets, besides the fact that in some environments (e.g. outdoor) the installation of wires may be harder and pricy. The 802.11 standard already defines a way for interconnecting APs wirelessly using the so-called Wireless Distributed System (WDS) technology [1]. However, manual configuration of the network topology is required, which precludes the automatic set up of the network as well as automatic adaptation to topology changes. As such, the rapid, easy and cost-effective deployment of Wi-Fi infrastructures, using the current Wi-Fi standard, is unfeasible, and new solutions need to be defined.

802.11-based Wireless Mesh Networks (WMNs) are considered the cost-effective solution to extend wired network infrastructures wirelessly. WMNs may be defined in different ways. Herein, we define a WMN as a wireless network consisting of static Mesh Access Points (MAPs) that perform multi-hop bidirectional forwarding between the wired network infrastructure and wireless clients, e.g. IEEE 802.11 stations (STAs). MAPs are assumed to have two Network Interface Cards (NICs), one that is used for connecting to the WMN and the other used to serve STAs; the terms MAP and Mesh Point (MP) will be used interchangeably. Intensive research is being carried out to define new mesh networking solutions. Within the IEEE 802.11s Task Group (TG)* a new standard is being specified [2] to overcome the limitations of the current WDS technology, namely the lack of self-configuration. In order to address the infrastructure extension scenario, a tree-based routing solution is proposed [2]. However, the use of a 6-address frame format, the introduction of a new type of device, the Mesh Point Portal (MPP), to interconnect the WMN with the wired infrastructure, and the explicit and periodic registration of wireless terminals running behind each MAP, make it a complex solution. In References [3,4] Layer-2 solutions for interconnecting multi-hop *ad hoc* networks to the Internet are proposed, but they are tied to a specific IP (Internet Protocol) version. Other proposed mesh networking solutions are based on Layer-3 routing [5,6] but only support a single IP version as well. Layer-2.5 solutions have also been proposed in References [7,8]. Still, they introduce unnecessary complexity in the infrastructure extension scenario.

A new simple and efficient solution, called WiFIX (Wi-Fi Network Infrastrucuture eXtension), based on IEEE 802.1D bridges [9] and a single-message protocol, is proposed in this paper. WiFIX is able to automatically create a tree rooted at the MAP connected to the wired infrastructure, called Master MAP; the terms Master MAP and root MP will be used interchangeably. Also, it supports legacy wireless terminals transparently and does not require their explicit registration. By using overhead analysis the higher efficiency of WiFIX in comparison with the 802.11s tree-based solution is demonstrated. On the other hand, its good performance when it comes to data throughput, delay and packet loss is shown by means of experimental evaluation. WiFIX targets static WMNs and is only focused on connectivity issues. It does not provide any mobility management for mobile STAs. Rather, it relies on current or upcoming 802.11 mobility management solutions [10,11] to support mobile STAs. While WiFIX has been developed having Wi-Fi networks in mind, it is able to support other 802-based networks. Technologies such as Bluetooth, Ethernet or Ultra Wide Band (UWB) can also be used either to interconnect APs or to enable access to legacy terminals using such technologies. The reference scenario for WiFIX is presented in Figure 1.

Our contribution is twofold. Firstly, we show that a simpler mesh networking solution is preferable to more complex solutions that do not bring up benefits for the major application scenario of mesh networks. Secondly, we propose a new mesh networking solution that can support any 802-based technology.

The rest of the paper is organized as follows. Section 2 presents the related work. Section 3 describes IEEE 802.1D bridges, which are the basis of the proposed solution. Section 4 presents the tree-based routing solution described in the IEEE 802.11s draft standard. Section 5 details the WiFIX solution, Section 6 provides the signalling overhead analysis of WiFIX and IEEE 802.11s and Section 7 describes the WiFIX implementation. Section 8 provides the comparison between WiFIX and IEEE 802.11s signalling overheads together with the experimental evaluation of the two solutions. Finally, Section 9 draws the main conclusions of the work.

2. RELATED WORK

This section presents 802.11 mesh networking solutions that have been proposed to extend wired network infrastructures wirelessly and to overcome the single-hop paradigm in current 802.11 networks.

WDS enables 802.11 APs to communicate wirelessly, instead of using the wired backhaul infrastructure, as it is usual in 802.11 networks. WDS uses a four address frame format [1], it works together with the IEEE 802.1D learning bridge mechanism supported by standard 802.11 APs, and it is transparent to IEEE 802.11 stations. For each neighbour AP, an AP creates a WDS link, which appears as a port of the local IEEE 802.1D bridge. Frame forwarding within the WDS is performed using the learning mechanism. The disadvantage is that WDS links between APs need to be created manually. Thus, each modification in the network requires manual reconfiguration.

The 802.11 TG ‘s’ is currently specifying a new standard [2] intended to enable automatic and dynamic creation of 802.11 mesh networks. Even though the standard does not focus specifically on the infrastructure extension scenario, it does define a tree-based routing solution for this case. The tree is rooted at the MP connected to the wired infrastructure, called root MP. Two mechanisms are defined to establish the tree topology [2], Proactive Path Request (PREQ) and Root Announcement (RANN). The use of explicit signalling to establish the path between an MP and the root MP, and *vice-versa*, and the need to perform explicit registration at the root or at the parent MP, are major disadvantages of the 802.11s tree-based routing solution. Additionally, the interconnection of 802.11s WMNs with the wired infrastructure requires the use of a 6-address frame format and a new device, the MPP, which combines an IEEE 802.1D bridge and an

* http://ieee802.org/11

UNCORRECTED PROOFS

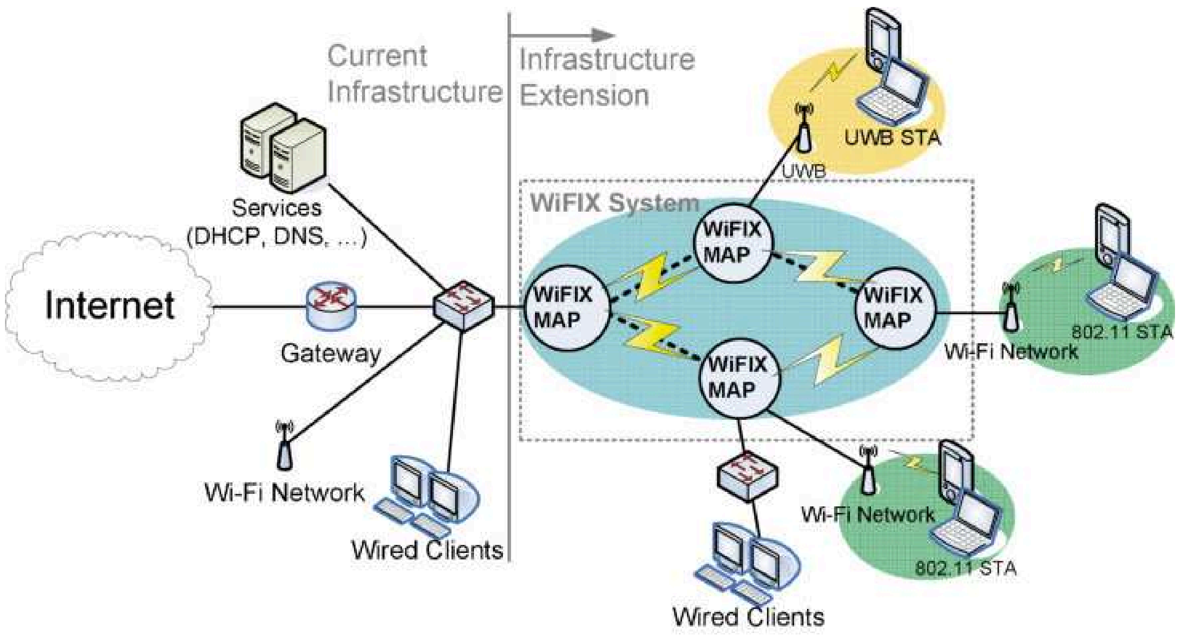


Fig. 1. WiFIX reference scenario.

802.11s MP, to enable internetworking between the WMN and the wired counterpart. On the other hand, 802.11s is a technology-specific network solution–within a mesh network only 802.11 is supported. Considering that we are evolving to a multi-technology scenario this may also be a disadvantage.

In the context of Mobile *Ad hoc* NETworks (MANETs)[†] several mechanisms have been proposed to extend a wired network infrastructure [5]. Proposed solutions can be divided in two different sets. There are solutions based on routing protocols, such as the Optimized Link State Routing (OLSR) protocol [12], which announce the gateway providing access to the Internet within a mesh network and create the path between each node and the gateway. The second set includes solutions that propose mechanisms independent of the *ad hoc* routing protocol in use, where a gateway periodically announces its presence within the network using some specific protocol [6]. The main disadvantage of these solutions is the use of a Layer-3 approach, which requires the definition of new IP auto-configuration mechanisms to distribute global IP addresses. Also, in a communication paradigm where IPv4 and IPv6 will coexist, these solutions are limited since they are mostly IPv4-only or IPv6-only.

Ancilloti *et al.*[3] propose a Layer-2 solution for interconnecting multi-hop *ad hoc* networks to the Internet. The Dynamic Host Configuration Protocol (DHCP) [13] is used for assigning global IP addresses. The OLSR *ad hoc* routing protocol announces the gateway to the Internet and establishes routes within the *ad hoc* network and towards the gateway. Given that there is not a single broadcast domain established within the *ad hoc* network, the solution runs a DHCP Relay on each *ad hoc* node, so that the DHCP protocol can run between each *ad hoc* node and the DHCP server running in the wired infrastructure. The main disadvantage of this solution is that it is tied to IPv4 and its companion protocols, such as DHCP.

Xu *et al.*[4] propose a solution based on Layer-2 bridging that runs within an *ad hoc* network and extends the coverage of an 802.11 AP connected to the wired infrastructure. However, rather than using 802.1D bridges, the authors define their own Layer-2 bridging mechanism; this makes transparent internetworking with installed bridged networks difficult. On the other hand, this solution only works when communication sessions are initiated by *ad hoc* nodes and does not support legacy 802.11 devices; all the devices connected to the *ad hoc* network have to run the proposed solution.

Layer-2.5 solutions [7,8] have been proposed to form *ad hoc*/mesh networks to either extend wired infrastructures or deploy stand-alone mesh networks. These solutions also target the establishment of a single logical link within the mesh network. However, both introduce unnecessary complexity concerning the extension of wired infrastructures, since in this scenario a tree rooted at the node connected to the wired infrastructure defines the optimal active topology. Furthermore, the solution proposed

[†] http://www.ietf.org/html.charters/manet-charter.html

UNCORRECTED PROOF

in Reference [7] requires hacks for each specific upper layer protocol to be supported, for example DHCP and Address Resolution Protocol (ARP) [14]; this means that, for instance, if the support of IPv6 is required, the solution has to be modified to take this and the companion protocols into account. The solution proposed in Reference [8] introduces its own addressing scheme, which requires new features to map the new Layer-2.5 addresses into the Layer-2 addresses of the underlying wireless interfaces.

3. IEEE 802.1D BRIDGES

IEEE 802.1D bridges [9] are ubiquitous in Ethernet switched Local Area Networks (LANs). In IEEE 802.11 networks, 802.11 APs also act as 802.1D bridges between the 802.11 link and the wired infrastructure, in order to enable transparent internetworking between the wired and wireless domains. 802.1D bridges are also known as learning bridges since they do not define any explicit signalling messages to build the local forwarding table. Rather, they learn the path to stations from the source address of data frames passing through. However, the use of this mechanism requires an active topology without loops, i.e. a tree. The Rapid Spanning Tree Protocol (RSTP) [9] has been defined to guarantee that the active topology is loop-free and to ensure proper operation of IEEE 802.1D bridges. RSTP deals with the automatic and dynamic configuration of an active tree network topology while ensuring redundancy.

The learning and forwarding algorithms employed by IEEE 802.1D bridges are briefly described next. For further details please refer to Reference [9]. The forwarding algorithm is simple. If the bridge does not have any entry in its forwarding table corresponding to the destination of the current data frame, the frame is transmitted on all ports, except the one on which it was received. Conversely, if there is an entry for the destination address, the frame is forwarded to the specified port, unless this is the incoming port, in which case the frame is dropped. The learning algorithm analyses the source address of all frames received and compares it against the information in the local forwarding table. If the source address is not found in the table, a new entry is created with the source address and the port number on which the frame was received; in addition, the lifetime of the entry is reset. If an entry corresponding to the frame source address already exists in the local forwarding table, and the table indicates this address was last seen on a different bridge port, the port number for the entry is modified accordingly. If the information is the same, only the lifetime of the entry is reset. An entry is removed from the forwarding table when its lifetime has expired. The lifetime can be set by configuration; in current Ethernet networks it is usually set to 30 s. This simple mechanism allows 802.1D bridges to learn the paths to each station connected to the network without exchanging explicit signalling.

4. TREE-BASED ROUTING IN DRAFT IEEE 802.11s

The IEEE 802.11s draft standard [2] defines a tree-based routing solution for the scenario of wired infrastructure extension. The Hybrid Wireless Mesh Protocol (HWMP) is the default routing protocol specified in 802.11s. It addresses the extension of the wired infrastructure to STAs by means of a tree-based routing solution. A node in the mesh network is configured as a special MP, called root MP. The root MP is usually connected to the wired infrastructure and announces itself to the other MPs, leading to an active topology defined by a tree rooted at the former. HWMP defines two mechanisms for creating the tree: the PREQ mechanism and the proactive RANN mechanism.

In the proactive PREQ mechanism the root MP periodically and proactively broadcasts a PREQ message with increasing sequence numbers. Upon receiving the PREQ message, an MP creates or updates the path to the root MP in its local forwarding table and, if the PREQ received allowed the creation of a better or a newer path to the root MP, it broadcasts an updated PREQ (TTL, hop count, path metric) to its neighbours. By using this mechanism, the PREQ originated at the root MP is propagated to all nodes in the mesh network. After the reception of the PREQ, an MP may need to send a Path Reply (PREP) in response to the PREQ (registration mode) in order to establish a path in the opposite direction, i.e. to the root. The need to send this response is defined by a flag included in the PREQ message. If the flag is not set, a tree of paths from all MPs to the root MP is set up but the MPs are not registered proactively at the root. A source MP may alternatively send a gratuitous PREP before sending the first data frame, in order to register its address at the root and create a path in the opposite direction; nonetheless, this mechanism only works for communication sessions initiated in the mesh.

The proactive RANN mechanism represents the second tree-based routing solution. The root MP periodically broadcasts a RANN message with increasing sequence numbers. Unlike what happens in the PREQ mechanism, the RANN message is only used to disseminate the metrics of the paths from the root MP to the MPs, across the mesh; it is not used to create or update any path in the forwarding table. Upon receiving the RANN message, an MP sends a unicast PREQ requesting a path to the root MP. This message is sent to the neighbour from which the RANN message with lowest metric value has been received. The root MP sends a PREP message in response to the unicast PREQ. This solution ensures the establishment of the best path between the current MP and the root MP. Besides registering at the root MP, an MP broadcasts an updated RANN (TTL, hop count, path metric) to its neighbours. This way, the information on path characteristics (metric) to the root will be disseminated to all nodes in the mesh network.

UNCORRECTED PROOFS

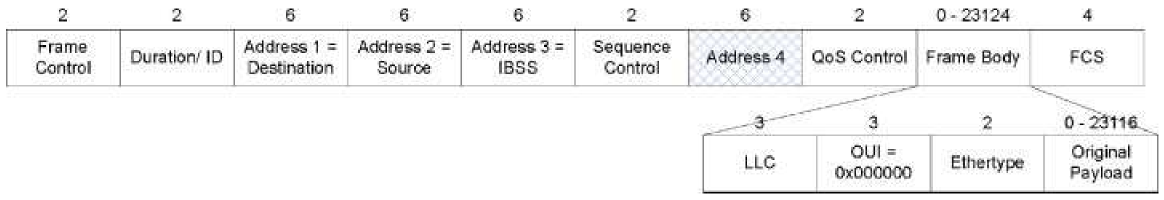


Fig. 2. 802.11 frame format.

5. Wi-Fi NETWORK INFRASTRUCTURE eXTENSION (WiFIX)

Taking into account the limitations and the excessive complexity of existing solutions, a new solution, called Wi-Fi network Infrastructure eXtension (WiFIX) is proposed. WiFIX targets the main application scenario for mesh networks–the extension of the wired infrastructure for providing pervasive Internet access. The solution is simple and conservative. WiFIX reuses solid and widely used concepts such as 802.1D bridges and their simple learning mechanism for frame forwarding, and it is based on a single-message protocol that enables the self-organization of the WMN. Also, it avoids the complexity of 802.11s and even supports the creation of 802 mesh networks instead of 802.11 mesh networks only; 802 technologies other than 802.11 can be used either within the WMN or to enable access to legacy terminals using such technologies, as illustrated in Figure 1. In this paper we focus on 802.11 mesh networks only, however. Therefore, the objective of WiFIX is to extend the wired network infrastructure to 802.11 STAs, such as smart phones, Personal Digital Assistants (PDAs), and laptops, connecting to the MAPs. An MAP can provide both wireless and wired connection, depending on the type of NIC used. One NIC is used for communicating with peer MAPs while the other is used to serve 802.11 STAs. It is assumed that a proper wireless channel is configured in each NIC in order to avoid radio interference between intra-mesh communication and the communication between 802.11 STAs and the MAPs. For an 802.11 STA, an MAP appears as a standard 802.11 AP and, as such, no modifications to 802.11 STAs are required.

In what follows, we present the encapsulation method used in WiFIX to allow multi-hop forwarding within a mesh network based on legacy IEEE 802.1D bridges, the mechanism used to create the active tree topology rooted at the node connected to the infrastructure, and the forwarding process of both unicast and broadcast data frames within a WiFIX mesh network.

5.1. Eo11 encapsulation

In order to enable multi-hop forwarding four MAC addresses are required as defined in the 802.11 WDS format. Two addresses are needed to store the original source and the final destination of a data frame and other two are required to identify the intermediate source and the intermediate destination at each hop; otherwise, frame forwarding does not work. The WDS format could be considered in WiFIX. However, WDS links need to be manually configured for each peer MAP. A possible solution to overcome this limitation would be to define a new mechanism to automatically establish the WDS links. Yet, that approach would have problems, namely with respect to reconfigurations when the mesh topology changes, since each MAP would have to perform time-consuming 802.11 scans to find its neighbour MAPs and establish new WDS links. On the other hand, the WDS format is not supported by all off-the-shelf hardware, which is a disadvantage. To overcome these problems, WiFIX configures the NICs of each MAP used for intra-mesh communication in 802.11 *ad hoc* mode. This has two advantages: (1) *ad hoc* mode is supported by all 802.11 hardware; (2) in *ad hoc* mode the 802.11 network is maintained in a distributed manner and is identified by a Service Set Identifier (SSID), which avoids 802.11 scans. Nevertheless, the use of 802.11 *ad hoc* frame format (Figure 2) has a limitation: only two MAC addresses are available for frame forwarding purposes. Since multi-hop forwarding using a single NIC requires four MAC addresses, the other two addresses need to be stored somewhere in the data frames.

The approach adopted was to define a new header type and encapsulating data frames using a tunnelling mechanism, called Ethernet-over-802.11 (Eo11). Figure 3 shows the Eo11 frame format. Thus, the original source and destination addresses are stored in the inner header of the data frame and the outer header contains the intermediate source and destination MAP addresses, which are changed at each intermediate MAP. The endpoints of a tunnel are located at neighbour nodes in the active topology; as such, the path between each MAP and the Master MAP can include multiple Eo11 tunnels. In this configuration each mesh node forces a frame to go through the next hop (intermediate destination).

5.2. Active tree topology creation

The active tree topology rooted at the Master MAP is created using a new mechanism called Active Topology Creation

UNCORRECTED PROOFS

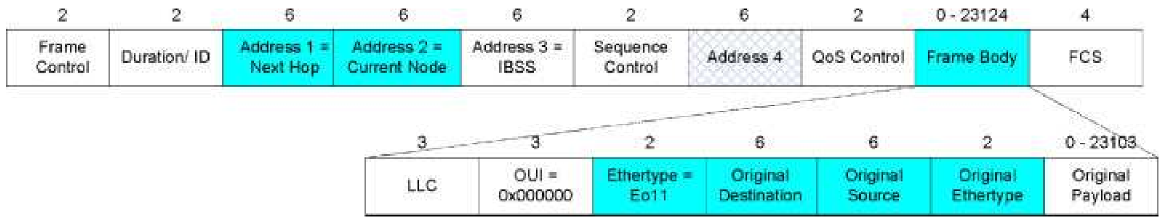


Fig. 3. Eo11 frame format.

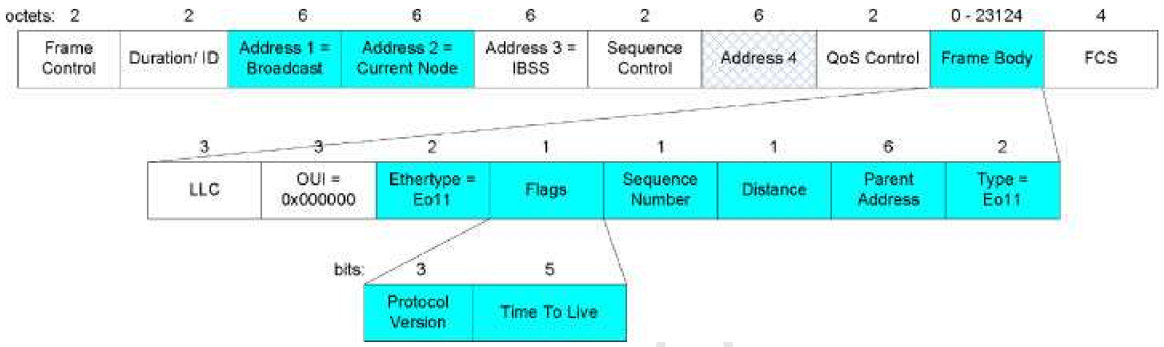


Fig. 4. Topology refresh message.

and Maintenance (ATCM). In ATCM, a Topology Refresh (TR) message, whose format is shown in Figure 4, is sent periodically by the Master MAP. The other MAPs forward this message, upon updating the required fields (e.g. parent address, TTL, distance). The TR message allows each MAP to select the parent providing best connectivity towards the Master (in number of hops), while enabling (1) the parent to be informed about new child nodes that have chosen it and (2) the establishment of Eo11 tunnels between the parent and its children.

The relevant information carried by the TR message is the following: (1) distance (number of hops to the Master); (2) sequence number, to avoid loops; (3) parent address; (4) protocol version; (5) time-to-live, which reflects the maximum number of hops to be traversed by this message. This information is complemented by the current node address and Independent Basic Service Set (IBSS) address, carried in the regular 802.11 frame header.

The ATCM mechanism is similar to the mechanisms of the IEEE 802.11s draft standard described in Section 4. Still, it includes new features, namely regarding the creation of the tunnels between parent and child nodes. Also, in ATCM, the TR message is simultaneously used (1) to announce the Master MAP and (2) to notify upstream neighbours that a given child has selected it as its parent node in the tree; in the 802.11s mechanisms two messages are used instead. The ATCM mechanism is illustrated in Figure 5 for Node C and it is formally defined below;

UNCORRECTED PROOF

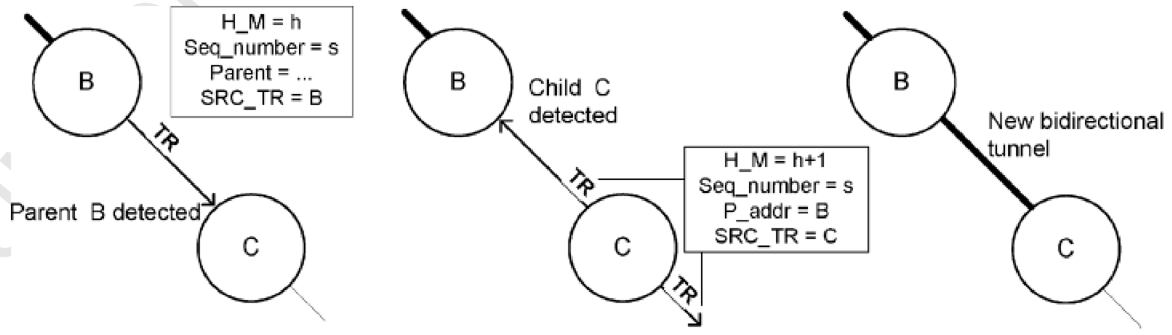


Fig. 5. Virtual link setup.

Table I. Notations used in the description of the ATCM mechanism.

Notation	Description
T_{TR}	TR broadcast interval (seconds)
C_P	Current parent mesh node in the tree rooted at the Master
H_M	Distance (number of hops) to the Master
SRC_{TR}	TR message source address
M_{addr}	Address of the Master
C_{addr}	Address of the current mesh node
P_{addr}	Parent address in TR message
SN_R	Set of nodes receiving the TR message either directly from the master or from some upstream neighbour
B_{UP}	Set of best upstream neighbours
D_P	Decision period (seconds) $= 3 \times T_{TR}$
CS_N	Current sequence number
PS_N	Previously transmitted sequence number
H_{max}	Maximum number of hops a TR message can traverse
L_P	Lifetime of the association established with C_P
$L_C[i]$	Lifetime of the association established with child i
SC_i	Set of children of node i

Table I summarizes the parameters considered in the definition.

5.3. Unicast frame forwarding

The tunnel endpoints created using the ATCM mechanism act as ports of a conceptual 802.1D learning bridge. The Eo11 encapsulation makes it possible for a node to send a unicast frame to the neighbour at the other end of the tunnel, while still maintaining the original source and destination addresses. Moreover, the bridge does not need to work in promiscuous mode, since a unicast frame destined to a node multiple hops away from the source will have as outer destination MAC address the next hop in the path towards the destination. The original unicast frame is decapsulated at every node. The learning bridge algorithm assumed by WiFIX will then process a unicast frame according to the following rules:

- Upon receiving a unicast frame from a tunnel endpoint, add a pair (frame source address on Eo11 header, port corresponding to the tunnel with endpoint equal to Address 2 of 80211 header) to the bridge forwarding table.
- If the unicast frame destination address exists in the bridge forwarding table, send the frame to the tunnel endpoint stated in the bridge forwarding table (the frame is afterwards encapsulated with Eo11 and sent to the associated endpoint address).

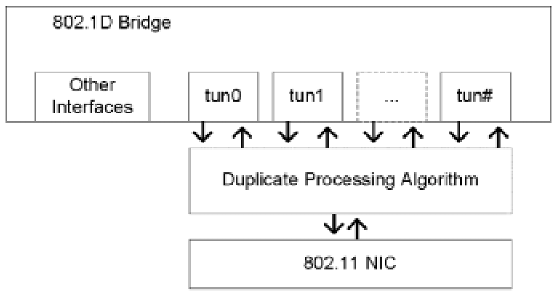


Fig. 6. DPA processing between the real interface and the tunnels.

- If the unicast frame destination address does not exist in the bridge forwarding table, send the frame to all tunnels, except the one the frame was received from.

The rules are essentially those stated in the original 802.1D learning bridge algorithm presented in Section 3. The differences (which are underlined) have mostly to do with the fact that the ports of the 802.1D bridge are actually tunnel endpoints, as shown in Figure 6.

5.4. Broadcast frame forwarding

If the destination address of a frame received by the IEEE 802.1D bridge is the broadcast address, the bridge will send the frame to all ports, except the one the frame was received from. This means that to forward broadcast traffic inside the mesh network the encapsulation of the broadcast frames into a unicast frame per neighbour is required. Considering that a node has a tunnel per neighbour, the broadcast frame will be sent *n*-1 times in unicast encapsulated with Eo11, where *n* defines the number of nodes forming the mesh network. While this mechanism works, it is highly inefficient due to the number of frames involved in each broadcast per radio link, herein defined as the physical space in which nodes are all in radio range of each other.

In order to provide a more efficient solution, the broadcast frame forwarding mechanism has been optimized. A new algorithm, called Duplicate Processing Algorithm (DPA), has been defined, in order to avoid the transmission of unnecessary duplicate broadcast frames. The algorithm is applied between the bridge and the physical Wi-Fi interface (see Figure 6). Using DPA each mesh node forwards just one broadcast frame per radio link. Instead of sending a copy of the broadcast frame to every neighbour (except the originator), a single copy of the frame is sent to the broadcast address, per radio link. The frame is still encapsulated in Eo11, and the information about the original source and destination is kept. The 802.11 source address is set to the current node address. The DPA algorithm uses information about the tree topology (parent and child) as a means to further improve broadcast forwarding. Only the nodes having more than one neighbour forward a broadcast frame; the nodes having a single connection (leaf nodes), transmit

UNCORRECTED PROOFS

ATCM Mechanism for creating the active topology

```
for each $T_{TR}$                                   % TR message creation
      $H_N \leftarrow 0$
      $C_p \leftarrow$ undefined
      $SRC_{TR} \leftarrow M_{addr}$
      MASTER_TR_BROADCAST()                         % Master sends TR msg to broadcast addr through wireless link
      $S_N \leftarrow S_N + 1$
end
for each i $\in SN_R$                               % TR message reception
      if $SRC_{TR} = C_p$
            if $CS_N > PS_N$                        % new TR message received from current parent mesh node
                  $L_p \leftarrow$ MAX_LIFETIME     % MAX_LIFETIME = 3*$T_{TR}$
                  if $(H_N + 1 \le H_{max}) \vee (H_{max} = 0)$
                        $H_N \leftarrow H_N + 1$
                        $P_{addr} \leftarrow C_p$   % $P_{addr}$ used by upstream neighbour to detect current node as child
                        $PS_N \leftarrow CS_N$
                        $SRC_{TR} \leftarrow C_{addr}$   % src addr of TR message changed to current node's address
                        BROADCAST_MOD_TR()          % broadcast modified TR message
                  end
            else
                  DROP_FRAME()                      % duplicate TR message received
            end
      elseif $P_{addr} = C_{addr}$                  % child selected current node as parent
            if $SRC_{TR}$ is unknown
                  TUNNEL_ENDPOINT($SRC_{TR}$)       % Associate tunnel endpoint to $SRC_{TR}$
            end
            $L_C[SRC_{TR}] \leftarrow$ MAX_LIFETIME
      else                                          % Possible parent information collection
            Q $\leftarrow$ ENQUEUE($SRC_{TR}$, $CS_N$, $H_N$)   % Q contains the set of upstream neighbours
      end
end
for each $D_p$
      for each i $\in SN_R$                         % Parent Selection
            $B_{UP} \leftarrow$ BEST_UPSTREAM_NEIGHBOUR(Q)
            if $|B_{UP}| > 1$                       % there may be more than one upstream neighbour with same $H_N$
                  if $C_p \in B_{UP}$
                        $C_p \leftarrow C_p$
                  else
                        $C_p \leftarrow$ LOWEST_MAC($B_{UP}$)     % upstream neighbour with lowest MAC address
                        CREATE_TUNNEL_ENDPOINT($C_P$) % creates tunnel endpoint for chosen parent
                  end
            else
                  if $B_{UP} \ne C_P$
                        $C_P \leftarrow B_{UP}$
                        CREATE_TUNNEL_ENDPOINT($C_P$) % creates tunnel endpoint for chosen parent
                  end
            end
            DELETE(Q)                               % deletes all entries in queue
      end
end
for each i $\in SN_R$
      for each second                               % timeouts
            $L_P \leftarrow L_P - 1$
            if $L_P = 0$
                  DELETE_TUNNEL_ENDPOINT($C_P$)     % deletes tunnel for current parent
                  $C_P \leftarrow$ undefined
            end
            for each $j \in SC_i$
                  $L_C[j] \leftarrow L_C[j] - 1$
                  if $L_C[j] = 0$
                        DELETE_TUNNEL_ENDPOINT($j$) % deletes tunnel to child $j$
                  end
            end
      end
end
```

this frame only if they are its originator or if the frame is entering the mesh through it. In general, a considerable amount of message retransmissions can be avoided with this improvement.

The total number of retransmissions (R) associated with one broadcast message, considering a tree topology with n nodes, is defined by

$$R = \begin{cases} n - n\mathrm{L}, & \text{if originated in a non leaf node} \\ n - n\mathrm{L} + 1, & \text{if originated in a leaf node} \end{cases}$$

where n_{L} represents the number of leaf nodes in the tree; $n - n_{\mathrm{L}}$ relay nodes (the nodes with more than one neighbour) will always forward the frame.

The DPA algorithm deals with two types of duplicates: (1) duplicate frames coming from the local bridge; (2) duplicate frames that arrive from the neighbours through the established tunnels. Each node keeps track of the number of duplicate frames it must drop in the future, either per node or locally (originated by the bridge); these values are called Local Drop counter (LD_C) and Neighbour Drop counter (ND_C), respectively. LD_C is set per frame and ND_C is set per frame and per neighbour. If the broadcast frame is originated locally, the bridge will create unwanted duplicates, one duplicate per neighbour. In this case, DPA sets LD_C to *Number of Neighbours* -1 for the current frame. This means that only one frame out of the set received from the bridge will be sent to the network; the other duplicates will be dropped till LD_C reaches zero. If the broadcast frame is received from another node, the bridge will originate a duplicate for each neighbour except the one that sent the frame. In this case, LD_C is set to *Number of Neighbours*-2. For each broadcast frame sent to the network, the current node will receive in the future one duplicate per neighbour, except for the one that originally sent the frame. The ND_C for each neighbour is increased by one for each frame sent. If the current node receives a broadcast frame whose corresponding ND_C is greater than 0, the frame is dropped and ND_C is decreased by one; if ND_C is 0 or undefined the corresponding frame is retransmitted. Two lifetime values are considered for defining the validity time of the LD_C and ND_C counters: Short Term Lifetime (STL) and Long Term Lifetime (LTL). STL is used for the LD_C counters, since local duplicates, originated at the bridge, generally arrive more quickly. LTL is used for duplicates originated at neighbours.

DPA Algorithm for dealing with duplicate broadcast frames

```
if $BF_{i,NIC}$                                     % broadcast frame received from the NIC
    if $ND_{C,i,k}$ = undefined                     % it is the first time frame i is received from neighbour k
        SEND_FRAME_TO_BR(i)                         % pass frame i to the bridge
    elseif $ND_{C,i,k} > 0$
        DROP_FRAME(i)
        $ND_{C,i,k} \leftarrow ND_{C,i,k} - 1$
        if $ND_{C,i,k} = 0$
            DELETE_ENTRY(i,k)                       % delete table entry related to frame i and neighbour k
        end
    end
elseif $BF_{i,BR,l}$                                % broadcast frame i generated locally received from bridge
    if $LD_{C,i}$ = undefined
        $LD_{C,i} \leftarrow N - 1$
        SEND_FRAME_TO_NIC(i)
        for each $j \in SN$
            if $ND_{C,i,j} \neq$ undefined
                $ND_{C,i,j} \leftarrow ND_{C,i,j} + 1$
            else
                $ND_{C,i,j} \leftarrow 1$
            end
        end
    end
    elseif $LD_{C,i} > 0$
        DROP_FRAME(i)
        $LD_{C,i} \leftarrow LD_{C,i} - 1$
        if $LD_{C,i} = 0$
            DELETE_ENTRY(i)
        end
    end
elseif $BF_{i,BR,r}$                                % broadcast frame i generated remotely received from bridge
    if $LD_{C,i}$ = undefined
        $LD_{C,i} \leftarrow N - 2$
        SEND_FRAME_TO_NIC(i)
        for each $j \in SN$
            if $j \neq k$                           % k is neighbour from which broadcast frame i was received
                if $ND_{C,i} \neq$ undefined
                    $ND_{C,i,j} \leftarrow ND_{C,i,j} + 1$
                else
                    $ND_{C,i,j} \leftarrow 1$
                end
            end
        end
    elseif $LD_{C,i} > 0$
        DROP_FRAME(i)
        $LD_{C,i} \leftarrow LD_{C,i} - 1$
        if $LD_{C,i} = 0$
            DELETE_ENTRY(i)
        end
    end
end
```

Table II. Notations used in the description of the DPA mechanism.

Notation	Description
$\mathrm{BF}_{i,NIC}$	Broadcast frame *i* received from the NIC
$\mathrm{BF}_{i,BR,l}$	Broadcast frame *i* generated locally received from the local bridge
$\mathrm{BF}_{i,BR,r}$	Broadcast frame *i* generated remotely received from the local bridge
SN	Set of one-hop neighbours of the current node
$\mathrm{ND}_{C,i,j}$	ND_C counter for frame *i* and neighbour *j*
$\mathrm{LD}_{C,i}$	LD_C counter for frame *i*
N	Number of neighbours of the current node

A broadcast frame is characterized by the original source address and a hash of the frame (e.g. MD5). Each node has a table that keeps track of the number of frames that must be dropped within a lifetime period, per neighbour ($\mathrm{ND_C}$ counters), and the frames forwarded by the local bridge that should not be sent to the radio link ($\mathrm{LD_C}$ counter). The DPA algorithm is formally described above; Table II indicates the notation used in the description.

6. OVERHEAD ANALYSIS

This section analyses the WiFIX overhead and the overhead introduced by the tree-based routing solution considered in IEEE 802.11s. The analysis takes into account the two tree-based routing mechanisms defined in 802.11s: PREQ and RANN. For the PREQ mechanism only the registration mode is considered for the reasons mentioned in Section 4. Mathematical expressions are derived for the overhead of each solution.

Before presenting the overhead expressions, a set of parameters is defined in Table III. The expressions consider the overhead introduced by each solution when only communication between 802.11 STAs and the root/Master (and *vice-versa*) is in place; communication between 802.11 STAs is not taken into account. Below, the term *link* is used either to refer to the virtual link between mesh nodes or the radio link used by STAs to attach to a WMN.

The *overhead for the proactive PREQ mechanism* (O_{PREQ}) can be calculated using the following[Q2] expression:

$$O_{\mathrm{PREQ}}(\mathrm{bytes/s}) = S_{\mathrm{PREQ}} \times \frac{1}{T_{\mathrm{PREQ}}} n + S_{\mathrm{PREP}} \times \frac{1}{T_{\mathrm{PREQ}}} (n-1)\, H_A \qquad (1)$$

The root sends out one PREQ every each T_{PREQ} seconds. The PREQ message is re-broadcast once by each MP forming the mesh network, resulting in n PREQ messages transmitted within the mesh network every T_{PREQ} seconds. Upon receiving a PREQ message, each MP replies with a unicast PREP message in order to register itself and

Table III. Notations used in the overhead analysis.

Notation	Description
S_{PREQ}	PREQ message size (bytes)
S_{PREP}	PREP message size (bytes)
S_{RANN}	RANN message size (bytes)
S_{TR}	TR message size (bytes)
S_A	Average frame size (bytes)
n	Number of nodes forming a mesh network
H_A	Average number of hops between root and MP
n_{STA}	Average number of STAs behind a MAP
T_{PREQ}	PREQ broadcast interval (seconds)
T_{RANN}	RANN broadcast interval (seconds)
T_{TR}	TR broadcast interval (seconds)
L_{FT}	Lifetime of forwarding table entry (seconds)
P_B	Probability of broadcasting a data frame every L_{FT} interval due to absence of forwarding table entry
P_{PD}	Probability of performing path discovery every L_{FT} interval due to absence of forwarding table entry
P	Probability that STA is the endpoint initiating a communication session

the 802.11 STAs behind it (if any) at the root. The PREP message is retransmitted H_A times on average.

The *overhead for the proactive RANN mechanism* (O_{RANN}) can be calculated using the following expression:

$$O_{RANN}(bytes/s) = S_{RANN} \times \frac{1}{T_{RANN}} n + \left(S_{PREP} + S_{PREQ}\right) \times \frac{1}{T_{RANN}} (n-1)\, H_A \qquad (2)$$

The root MP sends out one RANN every T_{RANN} seconds. The RANN message is re-broadcast once by each MP, resulting in n RANN messages transmitted within the mesh network every T_{RANN} seconds. For each RANN message, each MP replies with a unicast PREQ message towards the root MP. In response, the root MP sends a PREP message to the current MP; this procedure is performed every T_{RANN} interval by $n-1$ MPs. Both the PREQ and PREP messages cross H_A MPs, on average. Therefore, they are transmitted H_A times. Q2

The *overhead for WiFIX* (O_{WiFIX}) can be calculated using the following expression (the proof can be found in the appendix)

$$\begin{aligned} O_{WiFIX}(bytes/s) &= S_{TR} \frac{1}{T_{TR}} n \\ &\quad + P S_A \frac{1}{L_{FT}} 2\,(n-1)\,(P_B)^{n_{STA}}\,(n-1)\, n_{STA} \\ &\quad + (1-P) S_A \frac{1}{L_{FT}} 2\,(n-1)\, P_B\,(n-1)\, n_{STA} \\ &= \frac{S_{TR}}{T_{TR}} n + 2 \frac{S_A}{L_{FT}} (n-1)^2\, n_{STA}\, (P\,(P_B)^{n_{STA}} \\ &\quad + (1-P)\, P_B) \end{aligned} \qquad (3)$$

UNCORRECTED PROOFS

WiFIX also considers a root node (the Master) that sends out one TR message every T_{TR} seconds. The TR message is re-broadcast by each MAP, resulting in n TR messages transmitted within the network every T_{TR} seconds. WiFIX does not introduce any further explicit signalling overhead. However, there is additional overhead due to the use of IEEE 802.1D bridges and corresponding forwarding procedure. As mentioned in Section 3, when an 802.1D bridge does not know the path to the destination of the current data frame, it will send the frame to all ports, except the port the frame was received from. If no bridge in the mesh has a forwarding table entry for the destination of the data frame, the frame will be transmitted $n-1$ times in the mesh network; we assume this case in Equation (3), i.e. the worst case. Taking into account that each MAP has two physical interfaces, we have $n-1$ links forming the WMN and other $n-1$ links, one per MAP, to enable the attachment of STAs to the WMN. Thereby, when the path to the destination of a data frame is unknown, the frame will be sent to $2.(n-1)$ links; because there are $n-1$ MAPs and n_{STA} stations per MAP, this procedure is performed $n_{STA}.(n-1)$ times. Nonetheless, the broadcast of a data frame across the mesh network, every L_{FT} interval, by an MAP, will only happen with a given probability, which depends on the destination of the current frame. If the frame is addressed to the Master MAP, the probability of broadcasting the data frame is equal to the probability of having all STAs behind an MAP silent during the L_{FT} interval; this is given by the probability of the intersection of n_{STA} events 'STA silent during L_{FT} seconds'. Since each STA operates independently from the others, the probability of the intersection is equal to the product of the individual probabilities, i.e. $(P_B)^{n_{STA}}$. If a frame is destined to an STA, the probability of broadcasting the data frame is simply equal to P_B. On the other hand, if we assume bidirectional traffic, whether a communication session is initiated by a STA or either by the Master mesh node or some node in the Internet has influence on the WiFIX overhead. We take this into account in Equation (3) by means of the P parameter.

7. WiFIX IMPLEMENTATION IN LINUX

The Linux Operating System (OS) provides the means required to implement WiFIX: an 802.1D software bridge and tools to create virtual interfaces. Using these tools, WiFIX is required to create and delete virtual interfaces, to perform Ethernet-over-80211 encapsulation, and to process the duplicate broadcast frames. The Linux bridge[‡] is flexible enough to support the virtual interfaces. It expects them to behave like Ethernet devices and have a 6-byte MAC address. As the ATCM mechanism used by WiFIX enforces a loop free topology, the spanning tree protocol is disabled. Figure 7 illustrates the modules involved in the implementation, and their interactions. The WiFIX daemon seats between the virtual interfaces, represented by *taps*, and the wireless NIC.

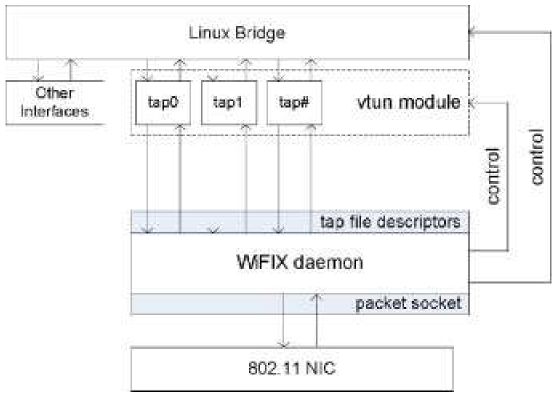


Fig. 7. Interaction between WiFIX and its peer linux modules.

The tunnels to each neighbour are implemented using a *tap* interface, provided by the *vtun* module[§], which are virtual interfaces that behave like Layer-2 Ethernet devices from the upper layers point of view; in the implementation, a *tap* behaves as a tunnel endpoint. A frame sent to a *tap* interface is in fact delivered to a file descriptor on the user space application that created the interface, in this case the WiFIX daemon. The frame will already contain an Ethernet header, with both the source and final destination MAC addresses. It is up to the WiFIX daemon to further process the frame and forward it to the wireless NIC. The kernel will then process this frame in the same way it would process a frame from a physical Ethernet NIC.

The WiFIX daemon accesses the wireless NIC using a Linux packet socket. The packet socket allows the reception of frames with a predefined Ethertype, and it also enables the application using it to specify the source MAC address, the destination MAC address and Ethertype of a frame to be received. These features are valuable for our goals, given that the proposed solution requires that WiFIX processes the Eo11 Ethertype and replaces the MAC address of the frame.

The WiFIX daemon runs the ATCM mechanism in order to detect the parent and child nodes, and create the active tree topology. Each mesh node creates a *tap* interface per child and a *tap* interface concerning the tunnel established with its parent node. All these interfaces are then added to the bridge. The association between the neighbour address and the *tap* file descriptor is maintained in a table, within the daemon. When a frame arrives to the WiFIX daemon from a *tap* file descriptor, the application fetches the neighbour address associated with the *tap*. A new header is then inserted before the original Ethernet frame, with the source address equal to the current node MAC address, the

[‡] http://linux-net.osdl.org/index.php/Bridge

[§] http://vtun.sourceforge.net

UNCORRECTED PROOFS

destination equal to the neighbour MAC address, and the Ethertype set to Eo11. The frame is then sent to the wireless NIC, *via* the packet socket.

The WiFIX daemon receives all the frames encapsulated in Eo11. Only the frames originated at known neighbours are processed, while all the others are dropped. When receiving a frame, the daemon looks-up in the local table for the *tap* file descriptor that corresponds to the source address of the neighbour in the 802.11 frame header. The frame is then decapsulated and sent to the *tap* interface file descriptor. The *tap* will then forward the frame to the bridge. If the current node is the destination of the frame, the Linux bridge will pass it upwards. Otherwise, the bridge will forward the frame according to the 802.1D rules. As a result, some frames can be forwarded to other physical interfaces or *tap* devices that belong to the bridge. The frames passed to the tap devices are then processed by the WiFIX daemon using the method described above.

Broadcast traffic is handled using the DPA algorithm. The WiFIX daemon keeps track of the expected duplicates, and drops them according to the DPA algorithm. The non-duplicate frames are delivered to the *tap* device that corresponds to the 802.11 source address. They are further processed by the bridge, according to 802.1D rules.

To cope with the extended frame length due to the Eo11 header, the MTU of the physical interface was increased by 14 bytes, which can be easily handled by 802.11. The MTU of the *tap* and bridge devices are unchanged to guarantee that the forwarded frames do not exceed the Ethernet MTU.

The source code was compiled for x86 and cross-compiled to MIPS processors. The latter allowed deploying the solution on off-the-shelf 802.11g wireless APs running Linux OS. The Linksys WRT54GL with OpenWRT[‖] WhiteRussian distribution was used.

8. EVALUATION

The performance of WiFIX was studied by means of theoretical analysis and real experiments. In this section, we first evaluate the overhead of WiFIX and compare it with the tree-based routing solution proposed by the IEEE 802.11s draft standard. In addition, we compare WiFIX and 802.11s in terms of reconfiguration speed when a node failure occurs. Finally, we present the experimental results obtained using a test-bed running WiFIX and IEEE 802.11s, considering throughput, delay and packet loss as the performance metrics.

8.1. Overhead evaluation

In order to simplify the overhead calculations, we assume constant values for some parameters in Table III. T_{PREQ}, T_{RANN} and T_{TR} are set to 1 s, according to the recommend

[‖] http://www.openwrt.org

Table IV. Average number of hops between Master and MAPs.

N	H_A
5	1.36
10	1.53
20	1.61
30	1.62
50	1.62

value stated in Reference [2] for T_{PREQ} and T_{RANN}. The values for H_A are a function of the number of nodes n, and were obtained by means of simulations using the Spanning Tree Simulator (STS)[¶], considering the average number of hops over 1000 random topologies (see Table IV). P is set to 0.95, a value that takes into account that it is most probable for an STA to initiate a communication session than to be the endpoint of a session initiated by some node in the Internet; indeed, this is the case in most applications (e.g. web browsing, streaming, file transfer), even concerning unidirectional applications, such as real-time audio/video, where there is always a bidirectional signalling phase before data exchange.

The value of P_B was obtained experimentally by analysing the behaviour of an 802.11 STA in two cases: (1) when the STA is connected to the Internet but there is no user data exchanged; (2) when the STA is connected to the Internet and there is user data transferred between the STA and the Internet. The analysis for the second case was performed by capturing the traffic generated by five users utilizing their 802.11 STAs to read e-mails, browse the web, and communicate using instant messaging. The capture of traffic was performed during periods of 600 s using Ethereal.[#] For the first case we have used the same method and observed that, even though the station did not have any active data flow, there was still some activity due to the exchange of control protocol messages. In practice, this represents the worst case for the 802.1D bridge forwarding procedure; without user traffic the forwarding table entries will expire more frequently and the overhead will be higher. The values of P_B presented in Table V represent the average of 50 samples, 10 samples per user. The value of P_B for a given sample was calculated using expression (4).

$$P_B = \frac{N_i}{\frac{\mathrm{SP}}{L_{\mathrm{FT}}}} = \frac{N_i L_{\mathrm{FT}}}{\mathrm{SP}} \tag{4}$$

where N_i—number of intervals where an STA is silent more than L_{FT} seconds, SP—sample period (in this case 600 s).

Expression (4) reflects the fact that a broadcast will only be sent in N_i intervals out of the total number of L_{FT} intervals composing the sample period; if an STA is silent less than

[¶] http://telecom.inescporto.pt/∼rcampos/software.php

[#] http://www.ethereal.com

UNCORRECTED PROOFS

Table V. Probability of lifetime expiration of the forwarding table entries.

L_{FT}	P_B (without user traffic)	P_B (with user traffic)
5	0.25	0.19
10	0.34	0.18
20	0.31	0.10
30	0.21	0.03

L_{FT} seconds, a forwarding table entry will still be available and the data frame can be unicasted. The log files used to calculate the values of P_B presented in Table V can be found in Reference [15]. In both cases the local traffic exchanged between the STA and the AP enabling Internet access (e.g. for authentication) was not considered in the calculations.

In the following we provide the results obtained for the WiFIX overhead in comparison with the overhead of the two mechanisms defined in 802.11s. The overhead analysis presented in Section 6 considered overhead in bytes/s. However, in 802.11 systems, it is more suitable to consider overhead expressed in frames/s, since the medium access overhead (i.e. SIFS, DIFS, Backoff) contributes the most to the time required to transmit a frame; the size of the frame is a second order factor. Thereby, the overhead results presented herein are in frames/s and they were obtained by ignoring the sizes of the messages in the mathematical expressions provided in Section 6. The parameters n, L_{FT} and n_{STA} were varied in the calculations.

8.1.1. Comparison of WiFIX and PREQ overhead.

The plots in Figure 8 show the WiFIX overhead normalized to the PREQ mechanism overhead, where the values of P_B correspond to the scenario of no user traffic exchange (see Table V); it represents the overhead introduced by the solutions when the activity of an STA is minimal. The plots show the WiFIX overhead as a function of the number of nodes forming the mesh network, and consider four values for L_{FT}; the difference between the two plots in Figure 8 has to do with the value of n_{STA}, the average number of STAs behind an MAP.

The major conclusion is that WiFIX introduces significantly less overhead than the 802.11s PREQ mechanism, regardless of the lifetime of the forwarding table entries; for instance, when $n = 10$, $n_{STA} = 3$, and $L_{FT} = 5$ s, the normalized WiFIX overhead is 0.53. For $L_{FT} = 5$ s and $L_{FT} = 10$ s, the WiFIX overhead increases linearly with the size of the mesh network, but it is still below the PREQ overhead. For higher L_{FT} values the normalized

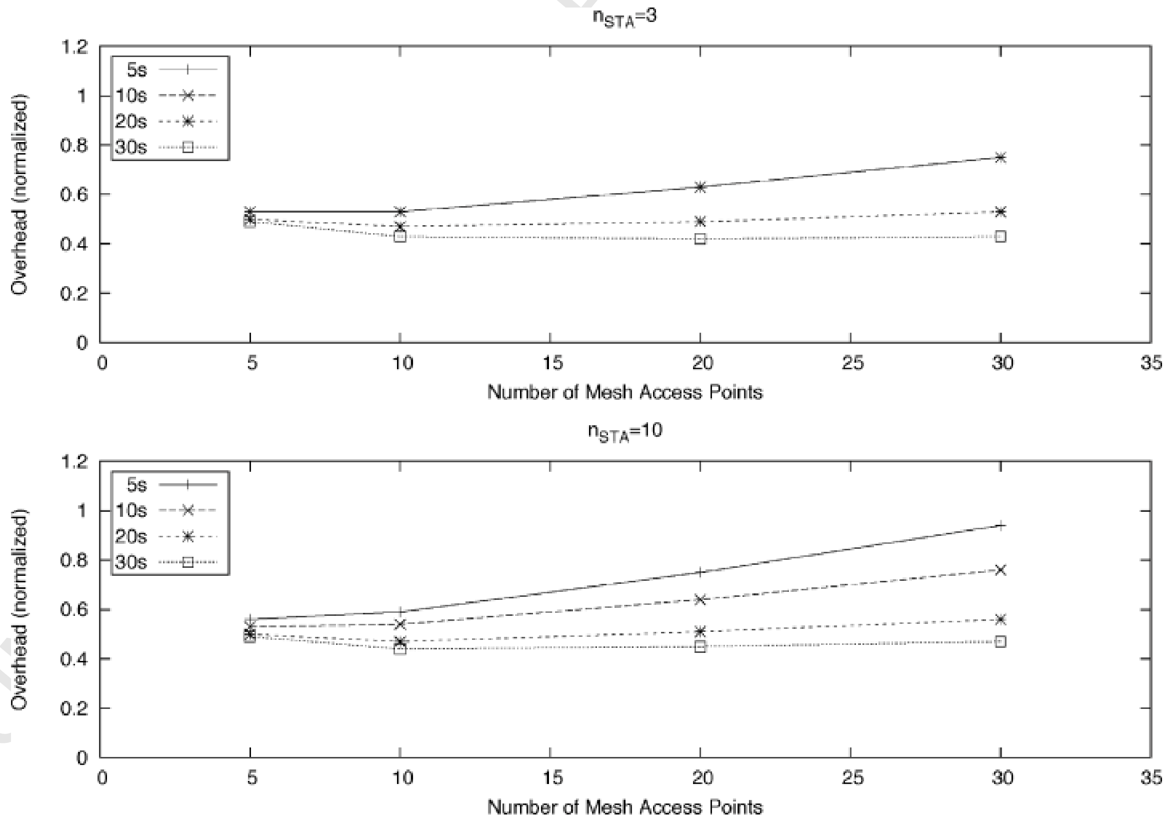


Fig. 8. Overhead of WiFIX normalized to the PREQ mechanism (without user traffic).

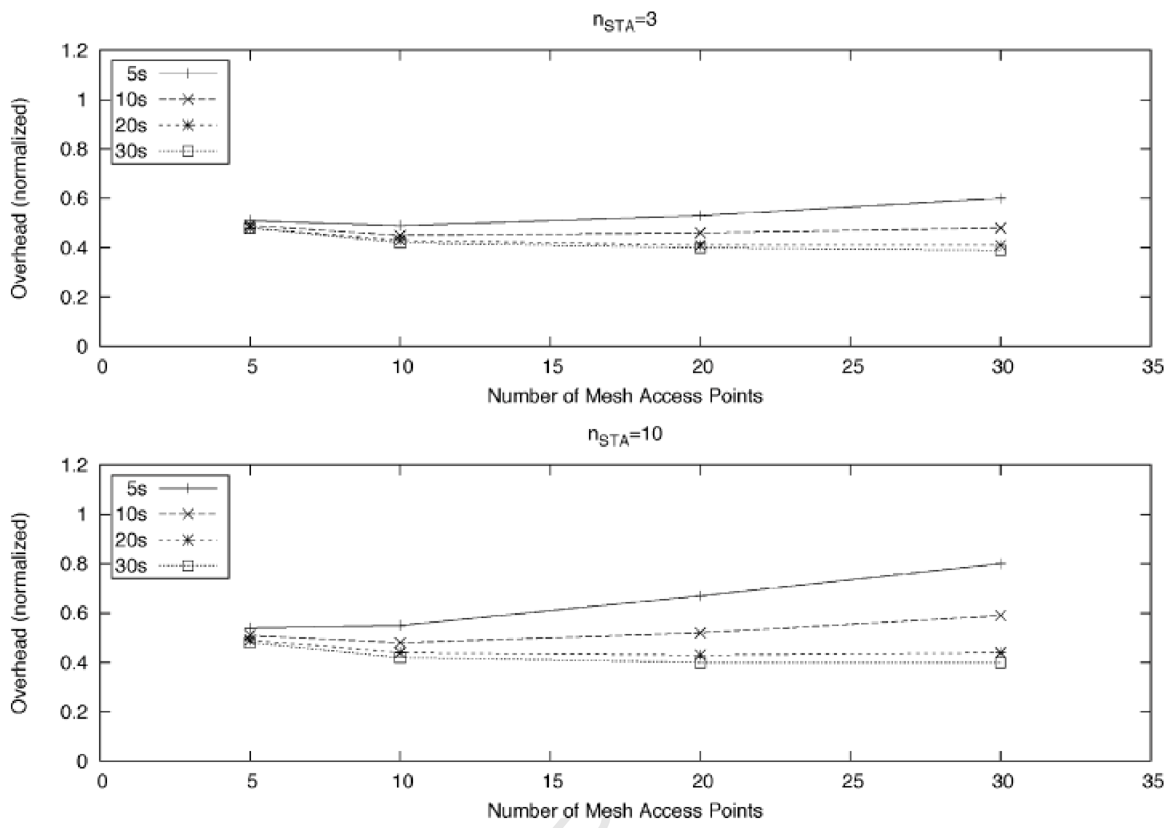


Fig. 9. Overhead of WiFIX normalized to the PREQ mechanism (with user traffic).

WiFIX overhead is kept almost constant with the size of the mesh network. The WiFIX overhead is higher when small values for L_{FT} are considered. This was expected, since the lower the lifetime L_{FT} the higher the number of times the bridges will send traffic using the broadcast mechanism. Furthermore, in general, the increase in the n_{STA} value represents an increase in the WiFIX overhead; this is particularly observed for $L_{FT} = 5$ s and $L_{FT} = 10$ s. If we consider expression (3), this would be expected, since the WiFIX overhead is proportional to n_{STA}.

The plots in Figure 9 show the same information but consider the values for P_B corresponding to the case where there is user traffic (see Table V). By comparing these plots with the plots in Figure 8, we conclude that the WiFIX overhead decreases when the STAs become more active, as expected; greater STA activity means higher probability of having a valid forwarding table entry either for that STA or for the Master MAP (assuming bidirectional traffic).

The bottom line is that the WiFIX overhead is inversely proportional to the activity degree of STAs. The more active the STAs are the lower the WiFIX overhead is, unlike the PREQ mechanism, which has a constant signalling rate independent of the user traffic. In practice, due to the use of a single radio channel to support all MAPs connected to the wired infrastructure, a WMN is expected to be formed by only a few MAPs (let's say $n < 10$), so that minimal data throughput can be guaranteed to the STAs. In such case, WiFIX is significantly more efficient than the 802.11s PREQ mechanism. The reduction in overhead may reach 60%.

UNCORRECTED PR

8.1.2. Comparison of WiFIX and proactive RANN overhead.

The plots in Figure 10 show the WiFIX overhead normalized to the RANN mechanism overhead, considering the values of P_B for the case where the activity of an STA is minimal. The results show that the WiFIX overhead is also considerably lower than the overhead introduced by the RANN mechanism, regardless of the values chosen for L_{FT}. The reduction in overhead may reach about 70%. Here, for low L_{FT} values, the normalized WiFIX overhead also increases linearly with the size of the mesh network. Still, the increasing rate is lower than in the plots of Figure 8; this has to do with the higher signalling overhead of the RANN mechanism when compared to the PREQ mechanism. For higher L_{FT} values ($L_{FT} = 20$ s and $L_{FT} = 30$ s) the normalized WiFIX overhead is kept almost constant as the size of the mesh increases.

The plots in Figure 11 show that with the increase in the activity of the STAs, the normalized WiFIX overhead

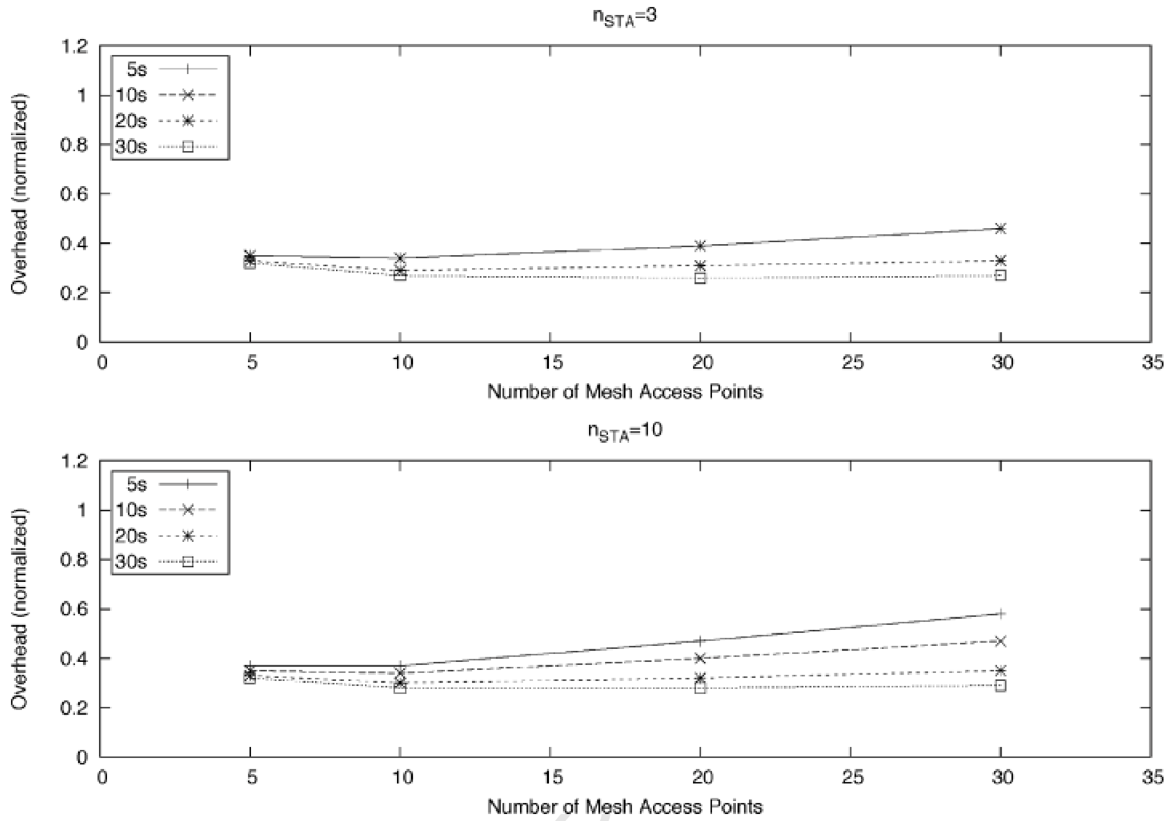


Fig. 10. Overhead of WiFIX normalized to the RANN mechanism (without user traffic).

UNCORRECTED PR

becomes lower; this holds for every L_{FT} value. It is interesting to note that, in general, the normalized WiFIX overhead slightly decreases with the size of the mesh network, when the number of stations behind each MAP (n_{STA}) increases. This is explained by expression (3). The values of P_B in this case are smaller than in the case when there is no user traffic exchanged (see Table V). The lower P_B values tend to reduce the contribution of the second term in expression (3) to the total WiFIX overhead, especially as n_{STA} increases. In contrast, the RANN overhead is independent of P_B and grows faster with the size of the mesh network. These two opposite effects contribute to the observed reduction in the normalized WiFIX overhead.

We then conclude that WiFIX is more efficient than the RANN mechanism too, regardless of the L_{FT} value selected. The overhead reduction may reach more than 70%. Overall, WiFIX introduces significantly lower overhead than both PREQ and RANN, while providing immediate path availability; even in the cases where there is no path available the data frame is immediately sent to the destination using broadcast.

8.1.3. Broadcast optimization.

In this section we evaluate the performance of broadcast traffic for WiFIX in comparison with both the 802.1D and 802.11s approaches. While the performance of the 802.1D and 802.11s broadcast mechanisms are only influenced by the number of nodes in the mesh, for WiFIX the network topology and the tree set up also come into play. A larger number of active leaf nodes improve the efficiency of WiFIX as far as the broadcast mechanism is concerned. In order to assess the real benefits of WiFIX, we used the STS simulator to generate hundreds of random network topologies and obtain the average number of leafs as a function of the number of MAPs. It was thus possible to find the number of frame retransmissions generated by a broadcast frame inside the mesh network. Figure 12 shows the number of retransmissions, per broadcast frame, for each solution. The curve for the 802.1D bridging mechanism is not shown; it is the same as for the WiFIX worst case.

The 802.11s standard defines that broadcast frames are flooded on the network by making each node retransmit one time a broadcast frame that was received or generated by itself. Loops are avoided by using sequence numbers and TTL. This means that each frame broadcast into a network composed of n mesh nodes will originate n retransmissions. The overhead of this solution increases linearly with the size of the mesh, as shown in Figure 12. The 802.1D standard states that each node receiving a broadcast frame should transmit it on all the other local interfaces except

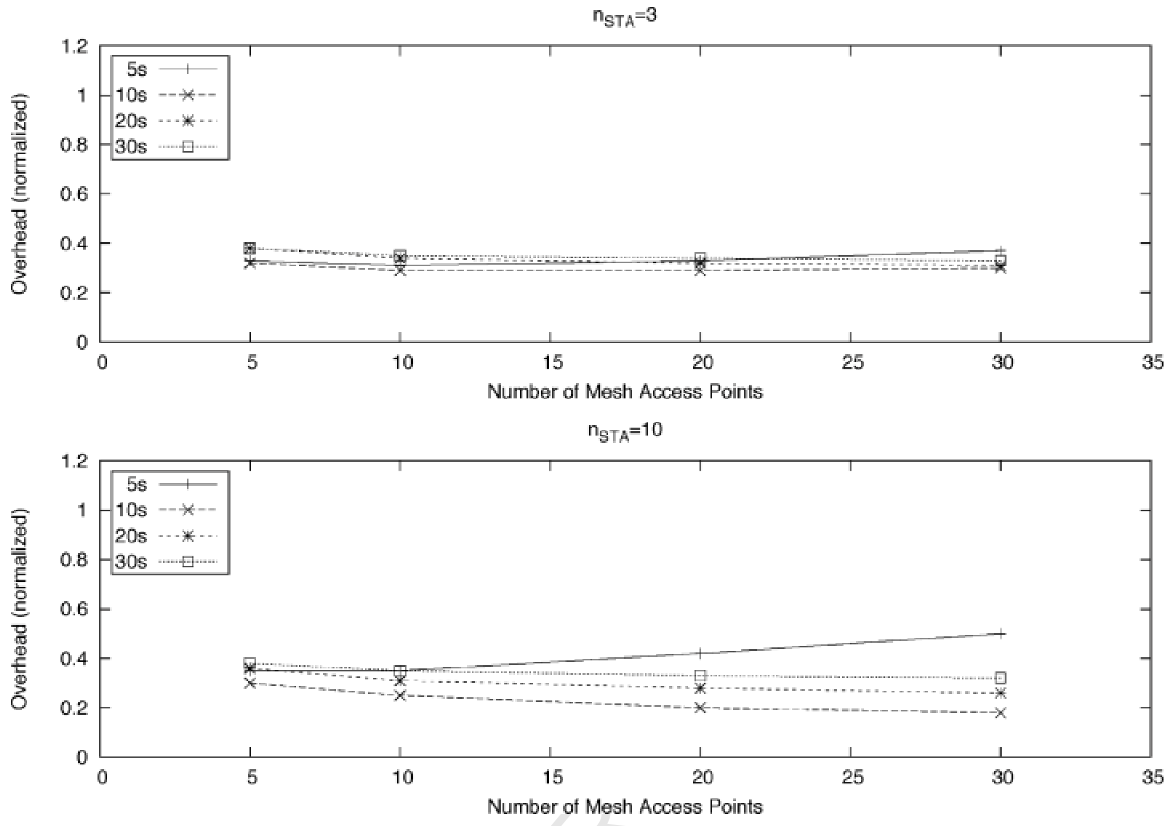


Fig. 11. Overhead of WiFIX normalized to the RANN mechanism (with user traffic).

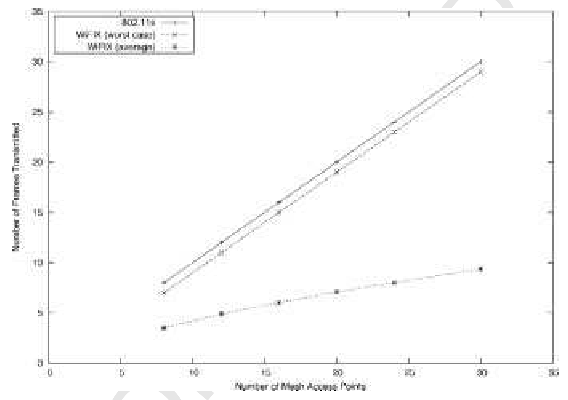


Fig. 12. Number of frames transmitted per broadcast frame as a function of the number of mesh nodes.

the incoming one. In our context, this will mean that each MAP would have to transmit a unicast frame to all its neighbours except the neighbour that originated the frame. A broadcast frame would originate a retransmission on each tree branch. For a network with n mesh nodes, $n-1$ frames would have to be transmitted. WiFIX partially behaves like 802.11s, since each node will originate at most one frame retransmission per broadcast frame received. But, unlike 802.11s, it prevents leafs from further retransmitting the broadcast frame, and does not require any additional information in the headers. Loops are avoided by using intelligence in the nodes. The curve shown in Figure 12 was obtained by applying the equation and considering the average number of leafs obtained for each number of mesh nodes using the STS simulator. The worst case for WiFIX happens when there is only one leaf node; for a network with n nodes, $n-1$ retransmissions will be required. Still, this is already better than the 802.11s approach. In general, the number of leafs will be higher than in the worst case. By using the simulator, we found out that the number of retransmissions increases logarithmically with the size of the mesh. This means that, on average, WiFIX will originate a smaller number of retransmissions per broadcast frame, and the difference to 802.11s will increase with the size of the mesh. This means smaller overhead and lower radio resources consumed.

8.2. Reconfiguration time after node failure

In this section we compare WiFIX and 802.11s when it comes to the reconfiguration time needed after a node failure occurs. The current open80211s implementation, used as a basis for our experimental evaluation (see subsection 8.3), does not support the 802.11s proactive mechanisms yet. In that sense, it was not possible to compare the two solutions experimentally. The analysis presented herein focus on the theoretical lower and upper limits of the reconfiguration time for each solution. Below, we ignore the propagation delays associated to the signalling messages involved in the various mechanisms under analysis, since they are at least two orders of magnitude lower than the refresh periods involved in the mechanisms being studied. Our goal is to have a rough understanding of the reconfiguration times involved in each solution.

The 802.11s PREQ and RANN mechanisms define that a PREQ and RANN message, respectively, shall be sent periodically every second by the root MP [2]. Upon receiving a new PREQ/RANN message, and after a propagation delay to wait for additional PREQ/RANN coming from other paths, each MP selects its best upstream neighbour. This mechanism is equally employed to establish new paths between each MP and the root MP after a node failure occurs, as referred to in the last version of the 802.11s draft standard [2]: '*when a proactive path selection protocol is used, MP failure and information on the new whereabouts of an MP are disseminated during triggered and periodic path update rounds*'. This means that the reconfiguration time is almost instantaneous, if the failure occurs right before the refresh period timeout, and about 1 s, if the failure occurs immediately after the periodic PREQ/RANN message was received by all MPs in the WMN. In this case, the failure will only be detected after one refresh period. The MPs affected by the node failure will not receive any PREQ/RANN message in the next refresh period from their current upstream neighbour. Then, they will select a new upstream neighbour, based on further PREQ/RANN messages received through other paths. Based on the new PREQ/RANN message and the registration made by each MP at the root MP, new paths are created and the reconfiguration of the tree active topology is accomplished.

The WiFIX solution uses a more robust mechanism to select the paths between each MAP and the Master MAP. In order to account for possible signalling message failures, it waits 3 times the interval between two consecutive TR messages until it takes a decision regarding the selection of the best upstream neighbour; this period is called the decision period (D_P) of the ATCM mechanism, which is set to $3 \times T_{\text{TR}}$. In the best case, the reconfiguration time due to the ATCM mechanism will be about D_P seconds, if the node failure occurs before the first TR message within a D_P period is sent by the Master MAP. In the worst case, it will take about $5 \times T_{\text{TR}}$ seconds to complete the reconfiguration of the active topology using ATCM, if the node failure occurs just after the first TR message is received within a D_P period by all MAPs. In this case, the node failure will not be detected by the end of the D_P period, since just two TR messages were lost. As such, only after an additional D_P period the node failure will be detected and a new active topology will be configured accordingly.

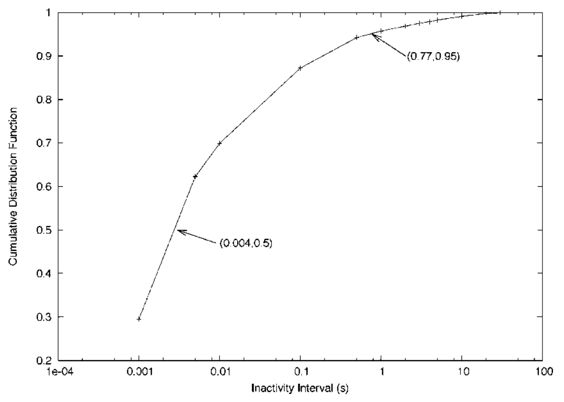


Fig. 13. Cumulative distribution function for the inactivity intervals associated to an STA (WiFIX).

In WiFIX, the STAs running behind each MAP are not explicitly registered in the Master MAP, as it happens in the proactive 802.11s mechanisms. As such, in order to completely evaluate the WiFIX reconfiguration speed we also need to consider the time required until the new paths towards the STAs are found. This is related to the inactivity intervals of an STA. Using the same experimental results considered for the overhead analysis presented in subsections 8.1.1 and 8.1.2, we could find the Cumulative Distribution Function (CDF) for the inactivity intervals associated to an STA when there is user traffic exchanged. The CDF of Figure 13 results from the average of 50 samples, 10 samples per user.

The plot of Figure 13 shows the median (4 ms) and 95th percentile (0.77 s). These values show that the inactivity interval associated to an STA contributes marginally to the reconfiguration time of the WiFIX solution. After the new active topology is established by the ATCM mechanism, the new paths either towards the STAs or the Master MAP will be found quickly.

We then conclude that the WiFIX reconfiguration time is higher than the reconfiguration time achieved using the proactive 802.11s mechanisms, although they are of the same order of magnitude. The reason for the quick reconfiguration times of the proactive 802.11s mechanisms is the immediate change in the paths between each MP and the root MP, every time the affected MPs fail to receive the next PREQ/RANN from their current upstream neighbour within a refresh period. While this represents a good mechanism for detecting node failures, in normal operation, this may cause instability in the paths between each MP and the root MP; a simple message loss may erroneously lead to the immediate change of the current path. On the other hand, a node failure is not envisioned to occur that frequently. Usually, the active topology will be stable. Node failures will only occur due to sporadic problems or due to human intervention for maintenance purposes. Thereby, the higher WiFIX reconfiguration time is not seen as a relevant problem.

UNCORRECTED PROOFS

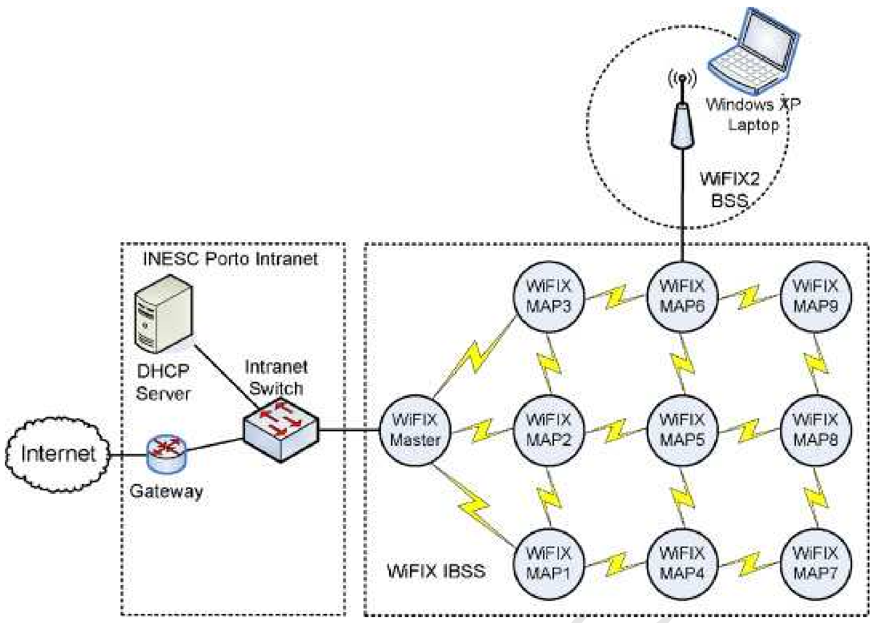


Fig. 14. Test-bed setup.

UNCORRECTED PROOFS

8.3. Experimental evaluation

In order to show its feasibility and good performance, WiFIX was experimentally evaluated using 802.11s as a benchmark. The experimental setup and the results obtained for WiFIX and 802.11s are described in what follows. The existing open-source implementation of 802.11s[**] was used as a basis for our experimental evaluation. In its current version open80211s only supports the on-demand part of HWMP. However, the difference between the 802.11s on-demand and proactive mechanisms exists at the control plane only. The actual paths established between each MP and the root MP using any of these mechanisms are the same, since the same path selection metric (Airtime Link Metric) is taken into account. Thus, the performance results presented in subsection 8.3.2 are those that would be obtained if the 802.11s proactive mechanisms were used to establish the paths between each MP and the root MP.

8.3.1. Experimental setup and tests.

The test-bed deployed to assess the operation of WiFIX in a real environment is shown in Figure 14. Both performance and functional tests were carried out. The INESC Porto's wired infrastructure was extended with 10 WMN nodes, positioned along the third floor of the INESC Porto's building, and forming the topology presented in Figure 14. In each WMN node, wireless NICs with *atheros* chipset were used, in order to be able to run open80211s. For WiFIX, the wireless NICs of each MAP were set up in 802.11 *ad hoc* mode to enable the creation of an 802.11 *ad hoc* network between the 10 WMN nodes. For open802.11s, using the tools provided by the open802.11s project, we have created a mesh network between the 10 WMN nodes. For running open802.11s, it was necessary to compile the latest Linux 802.11 wireless driver (ath5k); the same Linux wireless driver was used for WiFIX, for the sake of proper comparison between WiFIX and 802.11s. The default wireless driver settings were considered in the tests, concerning transmission power, number of MAC retransmissions, and RTS/CTS. The rate of the wireless NICs was set to the maximum bit rate possible (currently 11 Mbit/s, due to the problems related to the Linux wireless driver reported in the official web site[††]). Channel 3 was used in the tests, in order to minimize the interference from INESC Porto's Wi-Fi infrastructure. In addition, the tests were performed over the weekends, so that the influence of people walking around, as well as interference from the INESC Porto's Wi-Fi infrastructure had minimal impact on the results.

The performance tests were made using *mgen*[‡‡] and *iperf*[§§] as measurement tools, with all MAPs (1 to 9) sending traffic to the Master MAP. Both UDP and TCP traffic tests were carried out, considering sessions of 120 s. Ten tests were considered for TCP and for each offered rate in UDP. *mgen* was used for generating UDP traffic, while *iperf* was used for generating TCP traffic. The average throughput, one-way-delay (OWD), and packet loss ratio

[**] http://www.open80211s.org

[††] http://wireless.kernel.org/en/users/Drivers/ath5k

[‡‡] http://pf.itd.nrl.navy.mil/mgen/mgen.html

[§§] http://iperf.sourceforge.net

were measured for WiFIX and 802.11s. The goal was to evaluate the WMN performance when WiFIX and 802.11s are used alternatively. UDP tests were made considering six different constant bit rates (from 120 kbit/s to 720 kbit/s) for the flows between each MAP and the Master MAP. In all tests, the size of the data packets was set to 1500 bytes and, for UDP, the offered rate was set equal for every node.

The functional tests were performed using WiFIX to extend the INESC Porto's Intranet. No changes were made to the network services already deployed. The INESC Porto's access router and DNS servers were used as well. We connected a Cisco Aironet 1200 AP to the Ethernet port of one MAP. This was an unmodified corporation grade AP. It was used to enable the Windows XP laptop to access the WiFIX network and, consequently, the Intranet. After connecting to the WiFIX network, the laptop obtained an IP address from the DHCP server (see Figure 14), and accessed both Intranet and Internet services; services such as web browsing, Internet radio streaming, and VoIP worked without a glitch, in a similar way as if the node was directly connected to the infrastructure using a standard Wi-Fi AP. It was even possible to use more demanding services like network boot from the local Preboot Execution Environment (PXE) service.

8.3.2. Experimental results.

In what follows we refer to the experimental results obtained for WiFIX and 802.11s, taking into account four metrics: average throughput between an MAP and the Master MAP, average throughput per MAP, packet loss ratio, and OWD.

The plot in Figure 15 presents the average throughput for both solutions. WiFIX provides higher UDP throughput when compared to 802.11s, regardless of the offered load. This is explained by the higher packet loss ratio of 802.11s, even for smaller rates, shown in the plot of Figure 16. The higher packet loss is caused by the unstable paths established by 802.11s, using a new metric called Airtime Link Metric (ALM), which is defined as the default path selection metric for 802.11s WMNs [2]. The ALM metric

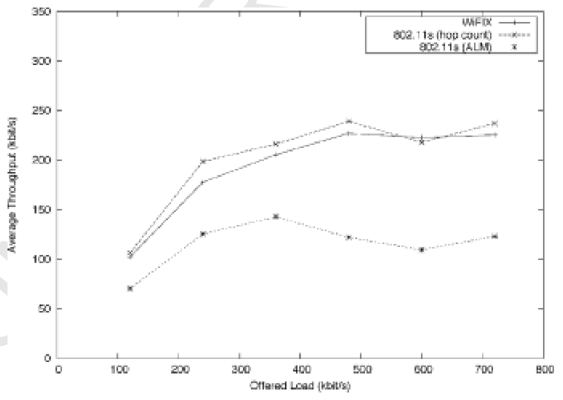


Fig. 15. WiFIX and 802.11s average throughput *versus* offered load considering UDP traffic.

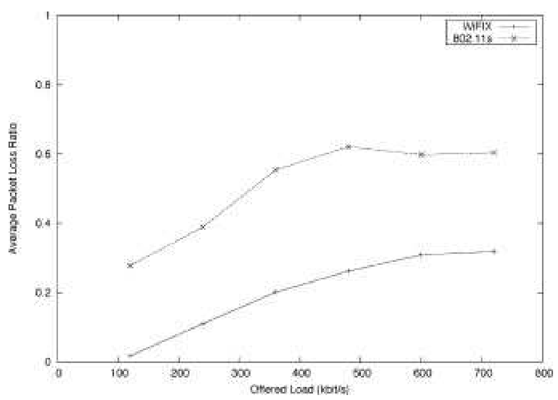


Fig. 16. Average packet loss ratio for WiFIX and 802.11s.

considers two constant and two variable parameters. The variable parameters are the rate at which the wireless NIC is running and the frame error probability for a test frame whose size is considered in the metric too. For all performed tests, the rate was set to constant (11 Mbit/s). As such, only the frame error probability was variable and contributed to the difference between path costs. The 802.11s path refresh mechanism allows the establishment of a new high quality path (in terms of ALM) if the current path becomes worse. A given MP detecting a path with better ALM metric (usually including unloaded links, which exhibit lower frame error probability) selects it as its new path and starts sending its traffic through that path. However, shortly the new path will suffer a metric degradation, since the current node (and maybe other nodes) selected it as the new path. The previous path may now become a good path again and a new change may occur. This leads to frequent oscillations in the paths between each MP and the root MP (as verified during our tests), as well as the concentration of traffic in some paths, which overall contributes to increase the packet loss in the WMN and reduce the average throughput. Indeed, this path oscillation phenomenon was also verified by Garroppo *et al.* in their experimental work presented in Reference [16]. Still, in order to confirm that the worse 802.11s performance does come from the use of the ALM metric, we have made experiments considering hop count as the path selection metric. The results shown in Figure 15 validate this hypothesis. When using the hop count metric for 802.11s the average throughputs of WiFIX and 802.11s are similar, as expected, since the same active tree topology is employed in both solutions.

WiFIX uses the number of hops between each MAP and the Master MAP as the default metric to compute the paths. Based on our experimental results and the results achieved in Reference [16], we conclude that the hop count metric is a better metric than ALM.

The plot in Figure 17 shows the average throughput per MAP when the offered load is 360 kbit/s per MAP. Similar plots would be obtained for higher offered load values. This plot shows that the higher average throughput obtained for WiFIX does not come from lower fairness; the average throughput per MAP is also higher for WiFIX

UNCORRECTED PROOFS

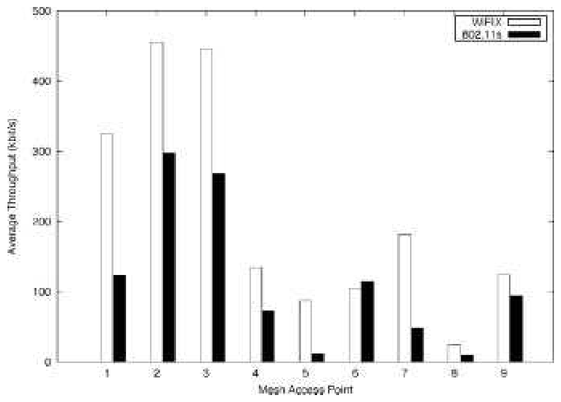


Fig. 17. Average UDP throughput per MAP when the offered load is 360 kbit/s.

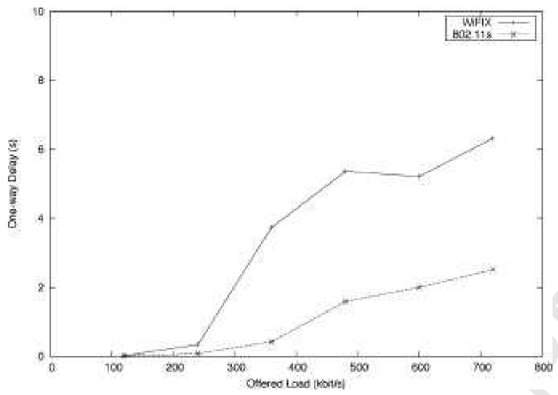


Fig. 18. One-way-delay for WiFIX and 802.11s.

independently of the node considered. These results also illustrate the unfair distribution of the throughput among the WMN nodes, a well-known problem in 802.11 mesh networks [17]. The MAPs closer to the Master have the highest throughput, since their own traffic is enough to almost completely fill up their queues, causing the packets from the farther nodes to be dropped very frequently.

The plot in Figure 18 presents the OWD for WiFIX and 802.11s. In general, a higher delay is obtained for WiFIX. This is due to the higher number of packets transported by the WMN when WiFIX is used. For 802.11s, there is a higher packet loss, which leads to less traffic in the WMN. Therefore, the network delay becomes lower than for WiFIX, as expected. Concerning lower offered rates ($\leq$ 240 kbit/s), which defines the cases where the network is operating below the saturation point, both solutions provide similar network delays.

Table VI presents the average throughput per node and the average throughput between an MAP and the Master MAP for TCP traffic. As for UDP, WiFIX provides higher TCP throughput than 802.11s. Also, the MAPs closest to the Master MAP are able to transmit more TCP traffic, as for UDP. The reasons for the higher throughput obtained when using WiFIX are those mentioned above for the UDP results.

8.4. Discussion

IEEE 802.11s and the other solutions referred to in Section 2 perform explicit signalling to establish a path between each MAP and the root of the tree topology. Conversely, WiFIX takes advantage of the frequent data frames exchanged between each 802.11 STA and the root of the mesh network, even when the STA does not have any active data flow to establish the path. Due to the use of 802.1D bridges, the establishment of a path between a new 802.11 STA and the root of the tree is implicitly performed by the data frames, either user data or signalling messages. The major advantages of WiFIX are: (1) the use of learning bridges and a single-message protocol to create the spanning tree rooted at the Master; (2) the lower overhead when compared to both PREQ and RANN mechanisms.

The applications considered in the overhead analysis mainly generate bursty traffic. If other applications, such as large file transfer and audio/video streaming, are considered, the values for P_B will be lower. Consequently, the WiFIX overhead may in practice be even lower than the values presented herein. On the other hand, the overhead analysis assumed bidirectional traffic. Although there are applications that exhibit unidirectional behaviour, we consider this as a reasonable assumption for three reasons. Firstly, even though those applications exhibit unidirectional data traffic, there is always bidirectional signalling associated with the data stream, which enables the maintenance of forwarding state. Secondly, even when an STA does not exchange any data there is still some signalling messages being sent around, which was confirmed during the experimental analysis performed to find the values of P_B. Thirdly, usually an STA will not have a single data flow active. Bidirectional data flows may be active together with unidirectional data flows, which implicitly help in maintaining 'fresh' forwarding state at the 802.1D bridges.

UNCORRECTED PROOFS

Q3

Table VI. Average[Q3]TCP throughput for WiFIX and 802.11s.

Solution	per MAP (kbit/s)									Average (kbit/s)
	1	2	3	4	5	6	7	8	9	
WiFIX	238	2449	1909	39	13	123	54	14	103	**549**
802.11s	495	1510	1731	17	52	104	160	32	32	**459**

The results presented in subsection 8.1 show the higher efficiency of WiFIX when compared to the 802.11s tree-based solution, regardless of the lifetime value selected for the forwarding table entries (L_{FT}). The recommended value in the 802.11s draft standard is 5 s [2]. Even for that value the WiFIX overhead is significantly below the overheads of the proactive 802.11s mechanisms, namely for the typical case where STAs behind MAPs send/receive data.

Besides being more efficient than PREQ and RANN for unicast traffic, WiFIX is also more efficient when it comes to broadcast traffic, thanks to the use of an intelligent management of broadcast frames. WiFIX takes into account the inherent broadcast characteristic of the underlying wireless technology in use and, in that way, avoids unnecessary retransmissions of broadcast frames. Yet, the DPA algorithm included in WiFIX represents an optimization to the broadcast frame forwarding process. WiFIX still works if only IEEE 802.1D bridges are used to process broadcast traffic.

The higher reconfiguration time required by WiFIX when a node failure occurs is not seen as a major problem in the type of WMNs being considered herein, as they are envisioned to be static and node failures are expected to occur sporadically. On the other hand, the lower reconfiguration time induced by the 802.11s proactive mechanisms is achieved using a less robust mechanism that may induce path oscillations and contribute to further exacerbate the problems arising from the use of the ALM metric.

While the mesh nodes are assumed to be essentially static, the STAs that connect to a WiFIX mesh network may be mobile. WiFIX does not provide any mobility management solution addressing STAs. Rather, it relies on 802.11 mobility management mechanisms already defined (IEEE 802.11f [10]) or being defined (IEEE 802.11r [11]). WiFIX is focused on the connectivity part only, since those mechanisms address mobility of 802.11 STAs.

The experimental analysis performed using the developed prototype shows the good performance of WiFIX when compared to 802.11s. It provides higher throughput, both for UDP and TCP traffic, lower packet loss, and similar delay when the offered load is equal to or lower than the maximum capacity of the WMN. Theoretically, the use of the airtime link as the metric to compute the communication paths is better than the number of hops. When using the number of hops every link is considered to have the same quality. Using the ALM metric links with lower quality may be avoided. Still, from the experimental results presented in subsection 8.3.2 we conclude that the use of a simple metric appears to be preferable. It is important to realize that WiFIX is not tied to a single metric. It can use metrics other than hop count deemed useful in a given scenario or environment, to create the active tree topology. The performance analysis presented herein focused on the WiFIX target scenario–the extension of the wired infrastructure for providing pervasive Internet access. Regarding intra-WMN communications, WiFIX may be worse than 802.11s due to the use of a single active tree rooted at the Master MAP for this type of communication too.

In its current version, WiFIX supports a single Master MAP. However, with a few simple modifications, it can be extended to support multiple Master MAPs. The TR message used to create the active topology has to include an identifier (e.g. the MAC address) of the Master MAP, so that slaves are able to distinguish between different Master MAPs. Upon selecting an upstream neighbour and a Master MAP, based on the set of received TR messages, the current node shall only retransmit the TR message associated to the selected Master. Nonetheless, the upstream neighbour selection policy is the same: to choose the parent node that ensures the lowest communication cost towards the wired infrastructure. Instead of a single active tree topology, multiple trees rooted at a different Master MAPs are created.

9. CONCLUSION

With the need for pervasive global connectivity due to the increased number of multimedia services and applications running over the Internet, IEEE 802.11 mesh networks are gaining momentum as a means to extend the radio coverage of current Wi-Fi networks. Mesh networking proposals have been based on Layer-3, Layer-2.5 or Layer-2 solutions that either have disadvantages in terms of address auto-configuration or are too complex concerning the main target scenario for mesh networks.

In this paper we proposed a new and simple solution, called WiFIX, which is based on legacy technology, exhibits good performance, and is more efficient than the tree-based solution proposed in the 802.11s draft standard. Current Wi-Fi APs can simply be upgraded with the WiFIX software package and be used to extend existing network infrastructures automatically. WiFIX is not limited to 802.11 mesh networks; it can be used for creating 802 mesh networks in general. As future work, we shall implement the support of multiple Master MAPs and consider the use of multiple radio channels to create an enhanced WiFIX wireless mesh network.

10. APPENDIX

The expression for WiFIX overhead, as a function of n, is demonstrated using the mathematical induction method. Our statement is that the WiFIX overhead is defined by

$$O_{\mathrm{WiFIX}}(n) = \frac{S_{\mathrm{TR}}}{T_{\mathrm{TR}}} n + 2\frac{S_A}{L_{\mathrm{FT}}}(n-1)^2 n_{\mathrm{STA}}\left(P\left(P_B\right)^{n_{\mathrm{STA}}} + (1-P)P_B\right) \tag{A1}$$

In order to simplify the expression above we define two constants C_1 and C_2

$$C_1 = \frac{S_{\mathrm{TR}}}{T_{\mathrm{TR}}} \quad C_2 = 2\frac{S_A}{L_{\mathrm{FT}}} n_{\mathrm{STA}}\left(P\left(P_B\right)^{n_{\mathrm{STA}}} + (1-P)P_B\right)$$

UNCORRECTED PROOFS

The simplified expression becomes

$$O_{\mathrm{WiFIX}}(n) = C_1 n + C_2 (n-1)^2 \qquad \text{(A2)}$$

Proof. First, we prove that expression (A2) is true for $n = 1$for[Q4]$n = 1$, the mesh network is composed of the Master only. As such, the overhead introduced by WiFIX is only due to the periodic message sent out by the Master

$$O_{\mathrm{WiFIX}}(1) = \frac{S_{\mathrm{TR}}}{T_{\mathrm{TR}}}$$

Using (A2) we get

$$O_{\mathrm{WiFIX}}(1) = C_1 1 + C_2 (1-1)^2 = C_1 = \frac{S_{\mathrm{TR}}}{T_{\mathrm{TR}}}$$

The expression then holds for $n = 1$.

$n = k+1$

We need to prove that the expression is valid for $n=k+1$, assuming that it is valid for $n = k$. As such, for $n = k$ we get

$$O_{\mathrm{WiFIX}}(k) = C_1 k + C_2 (k-1)^2$$

For $n = k+1$ we have

$$O_{\mathrm{WiFIX}}(k+1) = C_1(k+1) + C_2 k^2 \qquad \text{(A3)}$$

But, $O_{\mathrm{WiFIX}}(k+1)$ can also be defined as a function of $O_{\mathrm{WiFIX}}(k)$

$$O_{\mathrm{WiFIX}}(k+1) = O_{\mathrm{WiFIX}}(k) + C_1 + C_2 (k-1) + C_2 k \qquad \text{(A4)}$$

Expression (A4) represents the overhead introduced by k mesh nodes plus the additional overhead introduced by the insertion of a new mesh node in the network. A new node means that the T_{TR} message will be retransmitted one more time (C_1). In addition, the broadcast sent out by each node when no forwarding table entry exists for a given destination also reaches the $(k+1)$th mesh node. On the other hand, a broadcast message sent out by the Master when there is no forwarding table entry available reaches one more mesh node and the corresponding STAs behind it; this is represented by the term $C_2.(k-1)$. The contribution for the overall overhead by the $k+1$ mesh node is given by $C_2 k$.

Expression (A4) can be simplified in order to prove that it leads to the same result obtained using expression (A3)

$$\begin{aligned} &O_{WiFIX}(k+1) \\ &= C_1 k + C_2 (k-1)^2 + C_1 + C_2 (k-1) + C_2 k \\ &= C_1 (k+1) + C_2 k^2 - 2.C_2 k + C_2 + C_2 k - C_2 + C_2 k \\ &= C_1 (k+1) + C_2 k^2 \end{aligned}$$

Thus, the expression holds for all n.

ACKNOWLEDGEMENTS

The authors Daniel Sousa for his collaboration in the implementation of the WiFIX solution. Also, they thank the valuable comments provided by Jaime Dias which contributed to improve the WiFIX solution and the quality of this paper. Finally, they thank the anonymous reviewers for their fruitful comments that also contributed to improve the quality of the paper.

Q4

REFERENCES

1. IEEE 802.11 Work Group, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications, June (2007).
2. IEEE 802.11s/D1.07, draft amendment to standard IEEE 802.11: Mesh Networking. IEEE, September2007, work in progress.
3. Ancillotti[Q5] E, *et al*. A layer-2 architecture for interconnecting multi-hop hybrid ad hoc networks to the internet, in *Proceedings of the Third Annual Conference on Wireless On demand Network Systems and Services (WONS 2006) Les Ménuires*, France, January (2006) 87--96.
4. Xu S, Papavassiliou S, Narayanan S. Layer-2 multi-hop IEEE 802.11 architecture: design and performance analysis, in *IEE Proceedings-Communications*, Vol. 151, 5 (2004) 460–466.
5. Bernardos C, Calderon M, Moustafa H. Survey of IP address autoconfiguration mechanisms for MANETs, *IETF Internet Draft*, *draft-bernardos-manet-autoconf-survey-03* (work in progress), April 2008.
6. Templin F, Russert S, Yi S. MANET Autoconfiguration, *IETF Internet Draft, draft-templin-autoconf-dhcp-11* (work in progress), February 2008.
7. Tschudin C, Gold R, Rensfelt O, Wibling O. LUNAR: a lightweight underlay network ad-hoc routing protocol and implementation, in *Proceedings of Next Generation Teletraffic and Wired/Wireless Advanced Networking (NEW2AN'04)*, St. Petersburg, February (2004).
8. Draves R, Padhye J, Zill B. The architecture of the link quality source routing protocol, microsoft research, *Technical Report*, MSR-TR-2004-57 (2004).
9. IEEE 802.1D Work Group, IEEE Standard for Local and Metropolitan Area Networks: Media Access Control (MAC) Bridges, June (2004).
10. IEEE 802.11f Work Group, IEEE Trial-Use Recommended Practice for Multi-Vendor Access Point Interoperability via an Inter-Access Point Protocol Across Distribution Systems Supporting IEEE 802.11 Operation, July (2003).
11. IEEE 802.11r Work Group, grouper.ieee.org/groups/802/11/Reports/tgr_update.htm.

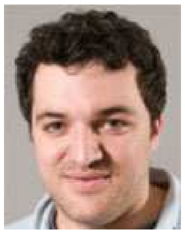
Q5

UNCORRECTED PROOFS

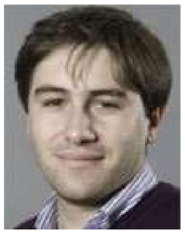
12. Clausen T, Jacquet P. Optimized Link State Routing Protocol (OLSR), IETF RFC 3626,October (2003).
13. Droms R. Dynamic Host Configuration Protocol, IETF RFC 2131,March 1997.
14. Plummer D. An Ethernet Address Resolution Protocol, IETF RFC 826, 1982.
15. Log files, telecom.inescporto.pt/<rcampos/frameCaps.zip.
16. Garroppo R, Giordano S, Iacono D, Tavanti L. On the development of a IEEE 802.11s Mesh Point prototype, in *Proceedings of Tridentcom2008*, Innsbruck, March (2008).
17. Bruno R, Conti M, Gregori E. Mesh Networks: Commodity Multihop Ad Hoc Networks, *IEEE Communications Magazine*, Vol. 43, Issue 3 (123–131), March 2005.

Authors' Biographies

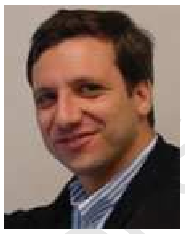
Rui Campos received a Diploma degree in Electrical and Computers Engineering in 2003 from University of Porto, Portugal. He works as a researcher at INESC Porto and he is pursuing his Ph.D. in Electrical and Computers Engineering. His research interests include mobile communications, network auto-configuration and spanning tree algorithms.



Ricardo Duarte received a Diploma degree in Electrical and Computer Engineering in 2004 from University of Porto, Portugal. From 2004 to 2008, he was a researcher at INESC Porto, where he participated in the IST Framework DAIDALOS and VISNET projects. Currently, he works at Fraunhofer Portugal, where he is the head of the IT department. His main research interests are mobile communications, Quality of Service and Ambient networks.



Filipe Sousa received a Diploma degree (2000) and a M.Sc. (2008) in Electrical and Computer Engineering from University of Porto, Portugal. He also received a Master in Business Information from Catholic University of Porto, Portugal in 2004 and the Certificate Cisco Network Professional (CCNP). From 2000 to 2009, he was a researcher at INESC Porto, where he actively participated in several projects in the IST framework. Currently, he works as a researcher at Fraunhofer Portugal. His main research interests are resource management, Quality of Service, network monitoring and mobile communications.



Manuel Ricardo received a Diploma degree in 1988, an M.S. in 1992, and a Ph.D. in 2000, all in Electrical and Computers Engineering from University of Porto, Portugal. He is an associate professor at University of Porto, where he gives courses in mobile communications and computer networks. He also leads the Communication Networks and Services Area at INESC Porto.

José Ruela received the Diploma degree in Electrical Engineering from the University of Porto, Portugal, in 1972, and the Ph.D. degree in Electrical Engineering from the University of Sussex, UK, in 1982. He is an associate professor at the University of Porto, where he gives courses in Data and Computer Communications, and Broadband Networks. He is also research manager of the Telecommunications and Multimedia Unit at INESC Porto. His current research interests are resource management and Quality of Service in high speed networks and mobile networks.

UNCORRECTED PROOFS