Fast Initial Authentication, 
a New Mechanism to Enable Fast 
WLAN Mobility

Konstantinos Georgantas

September 2011

Master of Science Thesis
Stockholm, Sweden 2011

TRITA-ICT-EX-2011:197
Fast Initial Authentication, a Mechanism to enable Fast WLAN Mobility

Konstantinos Georgantas
MSc Communication Systems
School of ICT
Royal Institute of Technology
Sweden
kgeo@kth.se

September 2011
Abstract

The growth of 802.11 wireless networks over the last decade has been tremendous. Through Wi-Fi® they have penetrated every market all around the world due to their performance as well as cost efficiency. However, there is still much work to be done when it comes to mobility handling in such networks. Handovers between same or different wireless architectures should be seamless and not affect any established connections.

The aim of this document is to firstly describe the current 802.11 WLAN architectures and then propose solutions that could possibly face the challenge of reducing the time spent during the link setup. This effort is focused on the authentication process and shall be achieved by either modifying and enhancing the existing standard or by introducing the newly formed Host Identity Protocol in its Diet Exchange version. After critically evaluating all the proposed solutions and proposing any new ones the most feasible one shall be considered for implementation and testing. For this purpose a testing setup of a WLAN (in enterprise mode) is described as well in detail.

Lastly this report follows and contributes to the IEEE 802.11 TGai task group in the sense that some of the described proposals were presented there and taken into account for further study.
Sammfattning

Tillväxten av 802,11 trådlösa nätverk under det senaste decenniet har varit enorm. Via Wi-Fi® de har penetrerat alla marknader runt om i världen på grund av att deras resultat samt kostnadseffektivitet. Det finns dock fortfarande mycket arbete att göra när det gäller mobilitet hantering i sådana nät. Överlänningar mellan samma eller olika trådlösa system bör vara smidig och inte påverka några etablerade anslutningar.

Syftet med detta dokument är att dels beskriva den nuvarande 802,11 WLAN arkitekturer och sedan föreslå lösningar som skulle kunna möta utmaningen att minska den tid under länken installationen. Detta arbete är inriktat på autentiseringen och skall uppnås genom att antingen ändra och förbättra den befintliga standarden eller genom att införa det nybildade protokollet världens identitet i sin kost Exchange version. Efter att kritiskt granska alla förslag till lösningar och föreslå eventuella nya de mest genomförbara en skall övervägas för implementation och test. För detta ändamål ett test installation av ett WLAN (i Enterprise-läge) beskrivs också i detalj.

Slutligen denna rapport följer och bidrar till IEEE 802.11 TGai arbetsgrupp i den meningen att några av de beskrivna förslagen presenterades där och beaktas för vidare studier.
Acknowledgements

During the last months, when I was conducting my thesis, there were people that supported and trusted me. Hereby I would like to thank them and guarantee my life-long appreciation. More specifically, Professor Andrei Gurtov from Helsinki Institute for IT (HIIT) for giving me the chance to conduct my internship there, Professor Markus Hidell from Kungliga Tekniska högskolan (KTH) for the provided assistance and useful advices, Mr Flinck Hannu and Mr Janne Tervonen from Nokia Siemens Networks (NSN) for their support and interest. Last but not least I would like to express my sincere thanks and respect to Robert Moskowitz for his guidance and his willingness to present a part of this thesis report at IEEE TGai meeting at Palm Springs CA.
# Table of Contents

1 Introduction 5
   1.1 Use cases 5
       1.1.1 Vehicular 6
       1.1.2 Non-Vehicular 6
   1.2 Problem statement 7
   1.3 Report organization 7

2 802.11 Standard 9
   2.1 802.11 architecture 10
       2.1.1 802.11 security history 11
       2.1.2 802.11 services 12
   2.2 802.11 Authentication and Key Management (AKM) 14
       2.2.1 pre-RSNA authentication methods 15
       2.2.2 RSNA authentication methods 15
       2.2.3 Extensible Authentication Protocol (EAP) 16
       2.2.4 802.11 Dynamic Key Generation 19
       2.2.5 4-Way Handshake 21
   2.3 802.11r fast BSS transition mechanism 22
       2.3.1 PMK caching 23
       2.3.2 PreAuthentication 23
       2.3.3 Fast BSS Transition (FT) 23

3 Host Identity Protocol 25
   3.1 The HIP Basic EXchange (BEX) 25
   3.2 The HIP Diet EXchange (DEX) 26

4 Design 29
   4.1 IEEE TGai Design Approaches and Evaluation 29
       4.1.1 Removal of Open System authentication 30
       4.1.2 Remove EAP messages on Association Request/Response messages 31
       4.1.3 Piggybacking upper layer message exchanges on Association Request/Response messages 31
   4.2 HIP Design Approach and Evaluation 32
       4.2.1 Lightweight Authentication and Key Management with HIP DEX 32

5 Implementation 37
   5.1 Linux Code Structure 38
       5.1.1 Supplicant side 38
       5.1.2 AP side 40
       5.1.3 Code Path 40
List of Tables

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>802.11 security standards</td>
</tr>
<tr>
<td>2</td>
<td>HIP DEX message length with a fixed 160 bits ECDH key exchange</td>
</tr>
<tr>
<td>3</td>
<td>802.11 states and frame Classes</td>
</tr>
</tbody>
</table>
List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Signal strength downgrade in a typical indoor environment (extracted from EDX planner study).</td>
<td>9</td>
</tr>
<tr>
<td>2</td>
<td>Open System authentication roundtrip</td>
<td>15</td>
</tr>
<tr>
<td>3</td>
<td>Shared Key authentication</td>
<td>16</td>
</tr>
<tr>
<td>4</td>
<td>A generic link setup process</td>
<td>17</td>
</tr>
<tr>
<td>5</td>
<td>Four way handshake</td>
<td>21</td>
</tr>
<tr>
<td>6</td>
<td>Complete EAP exchange</td>
<td>22</td>
</tr>
<tr>
<td>7</td>
<td>HIP Basic EXchange</td>
<td>25</td>
</tr>
<tr>
<td>8</td>
<td>HIP Diet EXchange</td>
<td>27</td>
</tr>
<tr>
<td>9</td>
<td>Mobile STA Authentication-Association states</td>
<td>30</td>
</tr>
<tr>
<td>10</td>
<td>HIP DEX authentication network architecture</td>
<td>33</td>
</tr>
<tr>
<td>11</td>
<td>HIP Fast Initial Authentication</td>
<td>34</td>
</tr>
<tr>
<td>12</td>
<td>Kernel architecture of the wireless subsystem</td>
<td>37</td>
</tr>
<tr>
<td>13</td>
<td>Graph of functions for supplicant’s authentication/association</td>
<td>39</td>
</tr>
<tr>
<td>14</td>
<td>Graph of functions for supplicant’s association after skipping authentication</td>
<td>43</td>
</tr>
<tr>
<td>15</td>
<td>Open System authentication duration</td>
<td>49</td>
</tr>
<tr>
<td>16</td>
<td>Open System authentication + Association duration</td>
<td>50</td>
</tr>
<tr>
<td>17</td>
<td>IEEE 802.11 standard state machine</td>
<td>51</td>
</tr>
</tbody>
</table>
1 Introduction

The technology advances of the last decade have contributed the most to the development of portable mobile devices. Laptops, smartphones, netbooks and tablet PCs are parts of our everyday life. The IEEE 802.11 standard [1] was the major reason that these devices gained our acceptance. It was the major technology to provide Wireless Local Area Network (WLAN) connectivity to such devices. 802.11 is the main standard Wi-Fi® is relying on. The latter is already considered to be a mandatory specification when buying a portable device. According to the Wi-Fi® Alliance in 2014 the amount of shipped Wi-Fi® enabled handsets will reach the number of half a billion [2].

At the same time Voice over IP (VoIP) is gaining acceptance, especially in wireless environments due to the cost effective communication it provides. According to Juniper Research dual mode (Cellular/VoIP over Wi-Fi®) mobile devices will dominate the market [3]. Analyzing further these two modes, it should be mentioned that cellular networks are bandwidth limited; thus VoIP experiences significant constraints when used over them. On the other hand 802.11 is attractive to VoIP because of the high data rates it can provide and hence the uninterrupted services it can support.

However the latter statement is not true in mobile environments in which mobile STAtions (STAs) are moving from one Access Point (AP) to another and especially when they experience a short dwell time within an AP coverage area i.e. mobile STAs attached on vehicles. It is also not true when a large amount of users simultaneously enter for the first time an AP i.e. train stations and subways. The reason of this incompatibility is the overhead that the current initial authentication process introduces to a mobile STA when it firstly enters an Extended Service Set (ESS) as well as the whole link establishment process.

Fast Initial Authentication (FIA) is what mobile stations need in order to merge in real mobile services. The cost of it seems really low as it does not include hardware changes but only some modifications of the 802.11 standard which can possibly be integrated to or form a new amendment of the existing standard.

1.1 Use cases

There are a lot of scenarios that FIA could possibly improve or even satisfy. As mentioned in the previous section, FIA can offer better mobility and that is why all of the use cases refer to mobile services. Reviewing some of them would provide some justification about the importance and need of FIA. The use cases can be divided in two main categories; vehicular and non-vehicular. As we will see STAs’ speed and density and the AP coverage are three important factors that affect the communication’s QoS [4].
1.1.1 Vehicular

As mentioned the first category refers to vehicular services. A mobile STA is usually attached onto a vehicle and is referred to as the On-Board Units (OBU). There are also Road-Side Units (RSUs) which are APs across a path (usually a road). Some of the RSUs hang over the path and some may be attached to poles along the path.

As expected vehicular use cases force the mobile stations to experience short dwell times\(^1\) (some seconds or even less) when passing from one AP coverage to another. The coverage of an 802.11 AP in such cases is not the usual one. Vehicles travel across the road and this implies the need for directional antennas instead of the omnidirectional ones. It may also be desired that each lane of the road has its own AP something that limits the coverage radius to 5 meters approximately or less. Assuming that a car moves with 140 km/h which means 38.9 meters/sec the corresponding interval that an STA has available in order to scan, establish a link and exchange data is 0.128 sec per AP coverage. Therefore latency is the main challenge. Link setup should be really fast in such cases and FIA can help in order to achieve this.

Some of the above mentioned use cases are [4]:

- **Roadside to vehicle communication** Work zone warnings, rail intersection warnings, road condition warnings, traffic information, curve speed warning, navigation, etc.
- **Vehicle to vehicle communication** Sudden front car brake detection, etc.
- **On board payments** Toll collection, gas and drive-through payments, parking lot payments, etc.
- **On board emergencies** Collision avoidance, emergency vehicle signal preemption, emergency video relay, etc.
- **On board data transfer** Browsing, map updates, music, VoIP telephony, program updates, etc.

1.1.2 Non-Vehicular

On the other hand there are use cases that do not impose such a short dwell time during an AP transition but introduce a high amount of incoming mobile STAs (high density within an AP’s coverage). This is the second category and it refers to non-vehicular services. Link setup should be also fast in such cases in order to reduce the load from the network and serve as much mobile stations as possible.

---

\(^1\)The time a mobile STA is being served by an AP.
Some of these use cases are [4]:

- **Pedestrian Internet access** Browsing, listen to music while exercising, etc.

- **Pedestrian Information access or distribution** Bus/train/subway timetables, navigation, warnings, etc.

- **Pedestrians during an emergency** Building evacuation, earthquakes, accidents, etc.

1.2 Problem statement

The above use cases justify the need for fast link setups. FIA is a part of a link setup and will monopolize the rest of this document. Reducing the time of the initial host authentication is the main goal. People dealing with this problem always ask themselves if authentication can be achieved in a single roundtrip. The answer does not really matter as the question is there in order to define what FIA aims for.

There are different ways of dealing with this challenge. The most of them enhance the current IEEE 802.11 based authentication mechanism by reducing the exchanged messages or by piggybacking upper layer information during the authentication phase. That is why before dealing with the problem of FIA there is a need of detailed understanding of the IEEE 802.11 standard and especially its message exchanges. However before starting this thesis project there was a suggestion that the Host Identity Protocol (HIP) could help reducing radically the message exchanges. So in this report there will also be an initial proposal of how HIP can be integrated in the 802.11 standard as an authentication mechanism as well as a suggestion of a new mobility friendly WLAN architecture which could possibly prove to be useful in the above described use cases.

Moreover, on the implementation part, the code related to the IEEE 802.11 standard is vast. Hence, there will be an extraction of the authentication related code, an analysis of it and at last an attempt for a FIA proposal implementation. For testing purposes and educational purposes it will be also interesting to see how an AP and a mobile STA are set up in a WLAN.

1.3 Report organization

The rest of this thesis report is organized as follows. In Chapter 2 there is a detailed analysis of the IEEE 802.11 standard authentication mechanisms and Chapter 3 describes the HIP protocol in its two versions. Directly after this background information Chapter 4 describes and evaluates the approaches towards FIA as well as elaborates on the HIP proposal that was
formed in the context of this thesis. In Chapter 5 there is an analysis of the Linux code related to the IEEE 802.11 authentication process, an attempt of implementing one of the already described solutions and a detailed WLAN setup guide for testing purposes. Finally, Chapter 6 concludes this report and mentions some future work suggestions.
2 802.11 Standard

All WLAN network technologies are defined by standards. Currently 802.11-2007 defines all the details of WLAN networks and all the operations taken in the MAC sublayer of the OSI Data-Link layer. There are also some periodic amendments which try to face specific problems that arise as the time passes. These amendments may or may not be incorporated to the next upcoming standards. Such a standard amendment could possibly refer to FIA.

When referring to 802.11 we assume that the used physical medium is the space. The general concept implies the absence of the physical security a wire can give. Waves are propagated all over the AP coverage area and are available to anyone, they do not have any specific boundaries other than the radio coverage. It is clear that security mechanisms must be employed in order to provide a wired equivalent level of security. Moreover in the wireless medium, wave propagation varies in respect of time and space and there are cases of interference as the 802.11 operates on unlicensed spectrum. Figure 1 shows how the signal strength can be affected in an indoor environment.

Figure 1: Signal strength downgrade in a typical indoor environment (extracted from EDX planner study).
2.1 802.11 architecture

The IEEE 802.11 standard has its own architecture in order to provide transparent to the upper OSI layers mobile services; starting from the Logical Link Control (LLC) sublayer of the OSI Data-Link Layer. The first part of the WLAN architecture are the STAs. They are the final recipient devices of 802.11 services and can be either portable\(^2\) or mobile\(^3\). The second one is the Basic Service Set (BSS). It can be seen as the coverage area or Basic Service Area (BSA) that an AP serves\([1]\). The access point can be considered an STA also but it is part of the BSS. There is also the Independent Basic Service Set (IBSS) which enables two or more STAs to communicate directly without the presence of any AP. This is known as ad-hoc networking.

Generally when an STA enters a BSA it “associates” with the corresponding AP, this procedure is dynamic and in larger networks it is handled by the Distribution System Service (DSS)\([1]\). Distribution System (DS) provides a cluster of BSSs. Not surprisingly, APs are defined as addressable entities that provide STAs with access to the DS\([1]\). In this way BSSs become parts of a larger service area also known as ESS. At the final point ESS makes the STAs’ communication and mobility transparent to the sublayer of the OSI Data-Link Layer.

As stated previously, the BSA is not well-defined as it depends on the propagation characteristics of the covered area. Moreover a possible STA mobility action can affect a lot the communication channel. As it can be seen at the above figure, the signal strength and thus coverage is not uniformly distributed.

Security in wireless 802.11 networks can be a real challenge as mentioned already. When dealing with it, the following issues should be taken into account thoroughly\([5]\):

- Data privacy
- Authentication, authorization, and accounting (AAA)
- Segmentation
- Monitoring
- Policy

Data privacy refers to the encryption that the transmitted data need due to the insecure medium that WLANs are using. AAA is also needed for protecting the network resources and controlling the users who are eligible of having access to that network. Authentication refers to the verification of

\(^2\)The device can move from place to place but it can only be served when in a fixed location.

\(^3\)The device can be served while moving from place to place.
the user’s credentials; authorization refers to the grant of access to network resources and accounting to the recording of the network resources being utilized by the users. Segmentation is another security mechanism which provides the means to further segment users and grant network resources according to more specific policies than simple authentication. Firewalls, VPNs and VLANs are some kinds of segmentation mechanisms. Furthermore, monitoring the wireless network for possible malicious traffic is quite important and can be implemented through a Wireless Intrusion Detection System (WIDS) or a Wireless Intrusion Prevention System (WIPS). Finally, all the above should be regulated by some means of policy which will dictate the rules of use for the above principles.

It is important to provide strong security to WLANs as they usually act as a portal to 802.3 wired networks.

2.1.1 802.11 security history

After a quick review of the IEEE 802.11 standard history, from 1997 up to 2004 there was not a lot of activity as far as security is concerned. Wired Equivalent Privacy (WEP) was the standardized encryption mechanism which proved to be cracked after some time. Some individuals even preferred not to use it at all as it was degrading network performance. Open system authentication and Shared key authentication were the only existing weak authentication methods.

After that Wi-Fi® Alliance, in order to enhance the WEP encryption, came up with Wi-Fi Protected Access (WPA) which supports the Temporal Key Integrity Protocol (TKIP) with ARC4 stream cipher. The key generation is dynamic and the authentication is done by WPA passphrases or Pre-Shared Keys (PSKs) in the Small Office/Home Office (SOHO) mode and by 802.1X/EAP authorization framework at enterprise mode. Additionally IEEE in 2004 published the 802.11i amendment which introduced the concept of Robust Security Networks (RSNs). RSNs use Counter mode with Cipher block chaining Message authentication code Protocol (CCMP), which makes use of the Advanced Encryption Standard (AES), as far as the encryption method is concerned. The key generation is once again dynamic and the authentication is done by either WPA passphrases or PSKs in SOHOs and 802.1X/EAP in enterprise environments. As a result Wi-Fi® Alliance introduces the Wi-Fi Protected Access 2 (WPA2) which is more or less the 802.11i itself. All these changes can be also observed in Table 1 [5].

IEEE 802.11-2007 sums up all the advances that occurred up to 2007. From then and on the security provided by the standard is quite robust up to now, especially when the CCMP/AES encryption method (i.e. enterprise mode) is used.
### Table 1: 802.11 security standards.

<table>
<thead>
<tr>
<th>802.11 Standard</th>
<th>Wi-Fi® Alliance</th>
<th>Authentication</th>
<th>Encryption Cipher</th>
<th>Key Gen</th>
</tr>
</thead>
<tbody>
<tr>
<td>Legacy</td>
<td>Open System or Shared Key</td>
<td>WEP</td>
<td>ARC4</td>
<td>Static</td>
</tr>
<tr>
<td>WPA SOHO</td>
<td>WPA PSK</td>
<td>TKIP</td>
<td>ARC4</td>
<td>Dynamic</td>
</tr>
<tr>
<td>WPA Enterprise</td>
<td>802.1x/EAP</td>
<td>TKIP</td>
<td>ARC4</td>
<td>Dynamic</td>
</tr>
<tr>
<td>802.11-2007</td>
<td>WPA2 SOHO</td>
<td>CCMP TKIP</td>
<td>AES ARC4</td>
<td>Dynamic</td>
</tr>
<tr>
<td>802.11-2007</td>
<td>WPA2 Enterprise</td>
<td>802.1x/EAP</td>
<td>CCMP TKIP</td>
<td>AES ARC4</td>
</tr>
</tbody>
</table>

#### 2.1.2 802.11 services

The IEEE 802.11 standard defines services. According to the architectural components described above there are two service categories, the **Station Services (SSs)** and the **Distribution System Services (DSSs)** [1]. The SSs are services provided by STAs (both AP and mobile stations) and the DSSs by the DS. More specifically IEEE 802.11 defines the following set of services:

- Authentication
- Association
- Deauthentication
- Disassociation
- Distribution
- Integration
- Data confidentiality
- Reassociation
- MAC Service Data Unit (MSDU) delivery
- Dynamic Frequency Selection (DFS)
- Transmit Power Control (TPC)
- Higher layer timer synchronization
Fast Initial Authentication

802.11 Standard

- QoS traffic scheduling

SSs include Authentication, Deauthentication, Data confidentiality, MSDU delivery, DFS, TPC, Higher layer timer synchronization and QoS traffic scheduling. Respectively DSSs include Association, Disassociation, Distribution, Integration, Reassociation and QoS traffic scheduling. All these services are used by MAC sublayer entities [1].

Before explaining some of the above critical services it would be advisable to analyze the mobility types a mobile STA can be involved into [1]:

1. No-transition: Static or local mobility in a BSA
2. BSS-transition: Handover between two different BSSs but within the same ESS.
3. ESS-transition: Handover between two different ESSs where there is a high probability for service disruption. Please keep in mind this type of transition as it will monopolize the rest of this document. FIA attempts to provide a solution towards seamless handovers during ESS-transitions.

The Distribution service is responsible for the right message distribution within a DS. The required information in order the DS to accomplish the above service are provided by the Association, Reassociation and Disassociation services. However, IEEE 802.11 standard does not define any specific DS implementation but it only suggests the use of the Wireless Medium (WM) as far as the DS Medium (DSM) is concerned.

The Integration service takes care for the safe delivery of messages from the DSM to an integrated LAN.

As we previously mentioned before the exchange of any data in the Distribution service the Association service should precede. It actually binds a mobile STA with the AP it belongs to [1]. In RSN networks however, the RSN Association (RSNA) is somehow different. There is a IEEE 802.1X Port that maps to each association and vice-versa, which determines whether to allow data traffic from an STA into the DS or not. More specifically the IEEE 802.1X Port is divided into the IEEE 802.1X Controlled Port (CP) and the IEEE 802.1X Uncontrolled Port (UP). The CP blocks any traffic by the mobile STA until the latter gets authenticated over the UP. After the authorization the CP unblocks and is used for exchanging protected traffic. Association is always initiated by the mobile STA.

In parallel Reassociation service is responsible for moving one association from one AP to another and thus keeps the DS updated. This means it is the service that handles the BSS transitions within an ESS. In RSNs however

A STA cannot be associated with more than one APs but an AP can be with more than one STAs.
it is not possible to move an association from one AP to the other. Instead RSNAs are being torn down and set up every time. Reassociations are initiated by the mobile STA.

The Disassociation service is invoked when associations should be terminated. It can be initiated by both the AP and mobile STA entities.

In order to achieve wired equivalent security IEEE 802.11 introduces the Authentication and Data confidentiality services. The former one is introduced instead of the wired media physical connection whereas the latter one instead of the data physical existence only in the wired media and not over the air. It should be already clear that Association depends on the Authentication success. The IEEE 802.11 standard defined two Authentication methods, mostly known as pre-RSNA authentication methods. The first one is the Open System authentication and the second one the Shared Key authentication. The Open System one is actually not an authentication method at all as it accepts any STA to join the DS. The Shared Key one makes use of the WEP encryption key an STA possesses. However, IEEE 802.11-2007 and RSNAs introduce the IEEE 802.1X authorization framework as well as the PSKs, keeps the Open System authentication and disallows the Shared Key authentication [1].

The PreAuthentication service allows the Authentication and the Association services to be separated. This means that a mobile STA can be authenticated to another AP than the one that it is currently associated. However the opposite is not possible. PreAuthentication aims to decrease the time of the Authentication that is required during Reassociation and increases the BSS transition mobility performance. Deauthentication service is invoked during the termination of an existing Open System or Shared Key authentication. It involves also the Disassociation service and it should not be refused by either communicating entity. Practically Deauthentication causes the 802.1X CP to block any traffic.

At last the Data confidentiality service role is to protect the data traffic. This is achieved by using either WEP, TKIP or CCMP. WEP and TKIP rely on the ARC4 encryption algorithm whether CCMP on AES.

As it can be understood most of the services IEEE 802.11 invokes relate to the 802.1X authorization framework which is further explained in the following section.

2.2 802.11 Authentication and Key Management (AKM)

The general concept of IEEE 802.1X is already explained briefly. However the whole procedure is rather complicated and demanding. That is why 802.11 authentication methods shall be explained thoroughly.
2.2.1 pre-RSNA authentication methods

In IEEE 802.11-2007 standard Open System authentication was the only pre-RSNA security mechanism to be kept. As shown in Figure 2 it is more like a “hello” exchanging mechanism between the AP and the mobile STA. In the first message the mobile STA sends its identity and a request for authentication. In the second message the AP responds to that request. Open System authentication takes place directly after an AP discovery either from passive or active scanning. Note that Open System authentication does not use WEP protocol for protecting the message exchanges. After Open System authentication successful completion, Association takes place. The rest of the steps that follow take care about the security issues.

![Figure 2: Open System authentication roundtrip](image)

Shared Key authentication is another pre-RSNA authentication method which uses WEP in order to authenticate STAs. It assumes that a static WEP key has been installed to both the mobile STA and AP. In this way the mobile STA begins a 4-way handshake with the AP as illustrated in Figure 3. In the beginning an authentication request reaches the AP and in turn the AP sends back to the mobile STA a clear text challenge. The client replies to the challenge by encrypting the clear text and then the AP replies back by authenticating (or not) according to the decryption result of the received challenge response. Someone would think that Shared Key authentication provides a better authentication mechanism than Open System authentication but in reality it imposes severe security problems. It is trivial to obtain a static WEP key by eavesdropping the sent by the AP challenge and then the reply of the mobile STA. By performing the same operation as the AP in the final step (i.e. decryption) the eavesdropper can easily derive the static key that is applied to all the BSS clients, provided that they have responded correctly.

2.2.2 RSNA authentication methods

On the other hand IEEE 802.11-2007 Enterprise and RSNAs rely entirely on the 802.1X authorization framework and the Extensible Authentication
Fast Initial Authentication

802.11 Standard

Figure 3: Shared Key authentication

Protocol (EAP). 802.1X is a port based network access control standard which provides the means for authentication and authorization of network devices [6]. The standard defines a set of entities such as the Authenticator, the Supplicant and the Authentication Server (AS). Every AP includes an Authenticator and each STA a Supplicant. In a more formal way:

- **Supplicant** is an entity that requests network access and needs to be authenticated according to its credentials.

- **Authenticator** is an entity that holds two authentication ports the uncontrolled one that allows EAP authentication traffic to pass by and the controlled one that blocks any traffic until the Supplicant gets authenticated.

It also defines an Authentication Server (AS). The AS is responsible for validating the Supplicant’s identity and further informing the Authenticator about it. In 802.11 WLANs the use of Remote Authentication Dial-In User Service (RADIUS) server is quite common as it supports AAA processes and can be integrated into the AP or to WLAN controllers.[5]

Hereby note that 802.1X is not exclusively a wireless standard and can be applied to LANs also.

2.2.3 Extensible Authentication Protocol (EAP)

EAP is a Layer 2 authentication protocol. There are quite many different versions of it either proprietary or standardized. Some support one-way authentication and others mutual (that is authentication of the AS side also) [7]. EAP messages are encapsulated in EAP Over LAN (EAPOL) frames. More specifically there are five EAPOL frame types [6]:

---

[5] More details about the above 802.1X standard can be found at [6].
- EAP-Packet: This is a normal EAP frame. The majority of the frames are of this type.
- EAPOL-Start: With this frame the supplicant starts the EAP process.
- EAPOL-Logoff: With this frame an EAP session can be terminated.
- EAPOL-Key: With this frame dynamic keying parameters can be exchanged.
- EAPOL-Encapsulated-ASF-Alert: Enables the delivery of alerts.

In an EAP exchange the two main parts that take part in it are the Supplicant and the AS. The Authenticator acts more like a proxy between them, however note that it controls the Control and Uncontrolled ports. 802.1X/EAP procedure starts directly after the Open System authentication and Association. Figure 4 shows a generic (not version specific) EAP exchange.

Figure 4: A generic link setup process

As it can be seen there are several main steps:
1. The Supplicant passes the Open System authentication.

2. Then the Association phase takes place. The controlled and uncontrolled ports on the Authenticator are blocked.

3. The Supplicant initiates the EAP process by sending an EAPOL-Start frame to the Authenticator. This is however an optional step.

4. The Authenticator then sends an EAP-Request asking the Supplicant for its identity.

5. The Supplicant replies with an EAP-Response and provides the Authenticator with its identity in clear text. The Uncontrolled port unblocks and allows EAP traffic to pass through.

6. The Authenticator encapsulates the EAP-Response into an Access Request packet and sends it to the AS.

7. The AS checks the Supplicant’s identity in its database and sends to the Supplicant an Access Challenge packet.

8. The Authenticator acts as a proxy and sends the EAP-Access Challenge to the Supplicant in a EAP frame.

9. The Supplicant hashes the received challenge and sends an EAP-Access Response back to the AS.

10. The Authenticator again acts as a relay and forwards an Access Response to the AS.

11. The AS compares the hashed supplicant version of the challenge with the proper one and responses with an Access Response stating if the authentication was successful or not.

12. The Authenticator forwards an EAP Access Response frame and the supplicant is authenticated if every previous step was completed successfully.

13. A 4-way Handshake takes place between the Supplicant and the Authenticator in order to derive the dynamic encryption keys that will be used during the data exchange.

14. When every step has been completed the Controlled port unblocks and the Supplicant can obtain an IP address by Dynamic Host Configuration Protocol (DHCP) and start using the network resources.

As someone may notice the fact that the challenge is sent in clear text in combination with the not encrypted hash of it, may pose a security threat. That is why some of the EAP versions introduce tunneled authentication.
EAP versions are quite a lot. Some of them are proprietary and some other not. Among the weak EAP methods are EAP-MD5 and EAP-LEAP. Among the successful ones are EAP-PEAP, EAP-TLS, EAP-TTLS, EAP-FAST. PEAP however is the most common EAP protocol that provides acceptable security during the EAP message exchange.

PEAP stands for Protected Extensible Authentication Protocol, its main feature is the creation of a TLS tunnel which is established in order to protect the Supplicant, by encrypting its identity as well as the EAP-Access Challenge EAP-Access Response phases. But PEAP itself is further divided to different kind of protocols such as:

- EAP-PEAPv0 (EAP-MSCHAPv2)
- EAP-PEAPv0 (EAP-TLS)
- EAP-PEAPv1 (EAP-GTC)

PEAP actually uses two EAP authentication protocols at the same time and that is why it is called "EAP inside EAP". The outer one (PEAPv0, PEAPv1, PEAPv2) is the one that creates the TLS tunnel and the inner one the one that is used within the TLS tunnel [5]. The key difference of PEAP is the server-side certificate that is introduced for the TLS-tunnel establishment.

PEAP generally, and irrespectively from the inner EAP protocol it uses, supports a specific philosophy. In the fourth step of the EAP exchange, previously presented, the Supplicant does not send its real identity in clear text to the AS but instead a bogus-invalid one. The Uncontrolled port at the Authenticator opens allowing EAP traffic to flow. In this way the AS is informed that there is a Supplicant that requests to be authenticated. The AS in turn sends its server certificate to the Supplicant who can authenticate the AS. At this point an encrypted TLS tunnel can be established and the AS requests the real Supplicant’s identity. The Supplicant is supposed to reply and then it is time for the challenge phase to take place where the inner EAP protocol is used. Note that the real communication between the Supplicant and the AS is hidden because of the tunnel.

2.2.4 802.11 Dynamic Key Generation

The 802.1X/EAP’s purpose is to provide authentication and authorization. However it can also provide the needed seeding material for dynamic encryption key generation [F]. Static keys are a trustless and not scalable solution. TKIP/RC4 (optional) and CCMP/AES (default) encryptions are the two

---

September 26, 2011 19
Fast Initial Authentication

802.11-2007 defined and RSNA compatible solutions. The key for each security association is derived after a 4-Way Handshake and it is called Pairwise Transient Key (PTK). PTK is used for unicast traffic encrypting purposes whether the Group Temporal Key (GTK) is used for encrypting broadcast or multicast traffic.

The ideal situation would be to support only RSNAs and thus have RSNs. However, each mobile STA may have different capabilities and thus Transition Security Networks (TSNs) are introduced in order to support RSNAs and pre-RSNAs such as the use of static WEP. The RSN capabilities of each STA are advertised in the RSN Information Element (RSNIE) field of certain 802.11 management frames such as the beacon frames, the probe response frames, and the (re-)association request frames. RSNIE not only defines the encryption mechanisms that an STA supports but also the authentication ones. In this way APs and mobile STAs can inform each other about their security capabilities. The attentive reader will notice that the encryption keys generations depend on the Authentication process and the Authorization is not completed until the encryption keys have been derived.

The EAP resulting keying material that we mentioned earlier is called the Master Session Key (MSK) and is at least 512 bits long. The MSK is present on both the AS and the Supplicant after the Authentication process completion. It works as seeding material for a new key, the Pairwise Master Key (PMK). The PMK is just the first 256 bits of the MSK. It is also present on both the AS and the Supplicant. The PMK is unique for every mobile STA and it is generated each time an EAP process is taking place. The AS sends the PMK to the Authenticator (AP) that is responsible for the Supplicant. The Authenticator then generates the Group Master Key (GMK). Now the two entities have the proper material in order to initiate the 4-way handshake which will provide the final keys for data encryption.

The PMK is used in order to derive the PTK. PTKs are unique keys between each single Supplicant and Authenticator in a 1:1 analogy. PTKs are composed by three other keys:

1. The Key Confirmation Key (KCK). It is used for data integrity during the 4-way handshake.
2. The Key Encryption Key (KEK). It is used for data privacy during the 4-way handshake.
3. The Temporal Key (TK). It is used for encryption/decryption of the MSDU payload in 802.11 frames.

The GMK is also used for the generation of the GTK. GTKs are keys shared between all the supplicants and their corresponding Authenticator in a M:1 analogy for broadcast/multicast traffic.
2.2.5 4-Way Handshake

With four EAPOL-Key frame exchanges between the Supplicant and the Authenticator the process of key establishment is completed. Without getting into message generation specific details the four message exchanges are as shown in Figure 5 and as explained below:

1. The Authenticator and the Supplicant derive their nonces, the ANonce and the SNonce respectively. The Authenticator sends its ANonce to the supplicant and the supplicant by using a Pseudo Random Function (PRF) derives the PTK with the knowledge of PMK, ANonce, SNonce and the MAC addresses. The Supplicant holds the key in order to encrypt its unicast traffic.

2. The Supplicant sends its SNonce to the Authenticator and the last one generates the PTK as described in the previous step. The Supplicant hereby also sends its RSNIE and a Message Integrity Code (MIC) which the Authenticator later checks.

3. The Authenticator optionally derives a GTK from the GMK and then it delivers it together with the ANonce, its RSNIE, and a MIC back to the Supplicant.

4. The Supplicant sends a confirmation of the proper key installation to the Authenticator.

Now the Controlled Port can unblock and encrypted traffic can pass through the Authenticator. Keep in mind that there is a Group Key Handshake (GKH) which allows for GTK exchange in case it was not generated during the 4-Way Handshake.

Nonce is a random value generated one time only.
After explaining all the Authentication/Association related steps Figure 6 provides a complete view of all the message exchanges.

Figure 6: Complete EAP exchange

2.3 802.11r fast BSS transition mechanism

The wireless medium itself puts no constraints in the movement of mobile STA as long as there is radio coverage. Mobile STAs can traverse from one BSS to another and maintain their connections by seamless handovers. However, seamless handover is a relative term. Voice over WiFi (VoWiFi) applications have quite stringent requirements as far as handovers are concerned (approximately 150 ms in order to preserve the QoS). On the other hand an 802.1X/EAP Authentication process may last even 700 ms, as we will see in Section 5.3.3. It is obvious that there is a need for faster but also secure roaming. There are ways to support Fast Secure Roaming (FSR) especially after the 802.11i and 802.11r-2008 amendments.

At this point it should be clear that a transition from one AP to another (generally talking only about BSS transitions) requires reauthentication or in other words the establishment of a new Pairwise Master Key Security Asso-
Fast Initial Authentication

which is identified by its Pairwise Master Key IDentifier (PMKID).

2.3.1 PMK caching

PMK caching relies on the ability of the STAs (both mobile and APs) to cache the used PMKs. More specifically a mobile STA can authenticate to an AP. When it roams to another AP then it creates a new PMKSA and hence a new PMK. However, it can cache the previously used PMK and in case it roams back to the first AP it can use the cached PMK in order to avoid the authentication process. This is done by sending a Reassociation that lists in the RSNIE all the cached PMKIDs the mobile STA has used. Then the AP can skip the authentication process and proceed directly with the 4-Way Handshake in order to derive the new PTKs. However, this is not a quite effective solution as it is quite rare to go back and forth between already visited BSSs. That is why PreAuthentication was also introduced.

2.3.2 PreAuthentication

PreAuthentication allows a mobile STA to authenticate to a new BSS before actually performing any transition to it. More specifically, a mobile STA can begin the 802.1X/EAP procedure and establish a new PMKSA with an Authenticator while it is still connected to another one. When the time for handover comes (if it comes) the PMK is already derived and the 4-Way Handshake takes place directly. But again this solution does not seem effective either as it requires all the mobile STAs in a network to preAuthenticate with all the neighboring Authenticators. This poses significant load to the network and especially to the AS.

Of course there are other proprietary FSR mechanisms but they are not standardized by IEEE.

2.3.3 Fast BSS Transition (FT)

802.11r-2008 amendment introduced the FT. The FT is preferably used in a network with a WLAN controller and with controller based APs. The concept applies to transitions only within the same ESS. As a mobile STA enters an ESS it will associate with the first AP and perform the normal 802.1X/EAP Authentication process. Any future BSS transition within the same ESS will utilize FT. What 802.11r defines is multiple layers of PMKs used by different devices. More specifically it is a 3-level key hierarchy.

1. Pairwise Master Key R0 (PMK-R0): The first level key.
2. Pairwise Master Key R1 (PMK-R1): The second level key.
3. Pairwise Transient Key (PTK): The third and final level key.
The PMK-R0 is derived by the MSK and it is cached in the WLAN controller - Authenticator. The second level PMK-R1 key is derived from the PMK-R0 at the controller and sent to the APs where it is cached. Finally, PTKs are also derived from the immediately higher level PMK-R1 keys. The Supplicants are doing the same thing by just knowing the MSK. They cache all the three previously mentioned keys.

According to the above assumptions FT introduces two new Information Elements (IEs). The Mobility Domain Information Element (MDIE) which declares the existence of an FT enabled ESS and the Fast BSS Transition Element (FTIE) which includes information about the FT authentication. These two IEs have their corresponding identifiers also.

As mentioned earlier when a mobile STA initially enters a FT ESS it performs exactly the same processes as it would normally do. The only difference is the use of MDIE and FTIE during the Association Request and Association Response processes. After the initial Association there are two possible methods that a mobile STA can use:

- **Over-the-Air FT**: The mobile STA communicates directly with the target AP that it wants to associate with over the air. The PMK-R1 key is the key to create the seeding material for the final PTK.

- **Over-the-DS FT**: The mobile STA sends an FT Action Request frame through the original AP to the target one over the DS. The target AP also replies over the DS to the mobile STA with a FT Action Response frame. Then the mobile STA Reassociation takes place and the PMK-R1 key is used for the PTK generation.
3 Host Identity Protocol

The Host Identity Protocol (HIP) was introduced by Robert Moskowitz. This protocol’s main purpose is to separate the identifier and locator roles of IP addresses [9]. It is generally designed so that it provides end-to-end authentication and key establishment. HIP introduces a new namespace where only host identities exist. A host’s identity can be represented by either a Host Identifier (HI) or a Host Identity Tag (HIT). The HI is the public key of an asymmetric key-pair. However the HI is not suitable to serve as a packet identifier because public keys’ length varies. Thus the HIT is a 128-bit hashed representation of the HI. Its length as you may notice was chosen deliberately in order to make HIP compatible with IPv6 applications.

HIP traffic is using IPsec in Encapsulated Security Payload (ESP) transport mode. Therefore HIP provides end-host to end-host encryption by establishing a pair of IPsec ESP Security Associations (SAs), one for each direction. Note that the established SAs are bound to HITs therefore enabling dynamic change of IP addresses. There is also no HIP specific data packet format and traffic is normally transported as previously in IPsec ESP mode without introducing any additional overhead [9].

HIP is further divided in two versions according to the message it exchanges. The first one is called Basic EXchange (BEX) and the second one Diet EXchange (DEX). The following sections are an introduction to the HIP protocol itself and the two versions it supports. As we will see in Chapter 4 HIP could possibly provide a new way of authentication with quite little overhead in terms of message exchanges.

3.1 The HIP Basic EXchange (BEX)

The HIP BEX is a cryptographic protocol that uses a SIGMA-compliant 4-way handshake in order to establish a Diffie-Hellman (DH) key exchange and a pair of IPsec ESP Security Associations (SAs) between two entities; the Initiator and the Responder [10].

Figure 7: HIP Basic EXchange
As shown in Figure[7]

1. The I1 packet initiates the 4-way handshake. The I1 packet contains the HIT of the Initiator and optionally the HIT of the Responder.

2. The Responder replies with an R1 packet which contains a cryptographic challenge with adjustable level of difficulty that the Initiator is supposed to solve. The main purpose of this challenge is to make the protocol resilient to Denial of Service (DoS) attacks. The R1 packet also initiates the DH key exchange by attaching the Responder’s public key and the DH key to the R1 packet. R1 also includes a signature covering partially the packet. This lets the Initiator to authenticate the Responder and at the same time allows the use of precomputed challenges.

3. The Initiator has to consume a considerable amount of CPU cycles in order to solve the challenge. Then it also computes the DH session key and creates a HIP association with the derived keying material. The solution of the challenge is attached in the I2 packet together with the Initiator’s DH key and its public authentication key. All the parameters are signed. The Responder then verifies the solution of the challenge, computes the DH session key by using the Initiator’s DH key, creates a HIP association and finally authenticates the Initiator.

4. At the end it informs the Initiator about receiving the I2 packet by sending an R2 packet which also protects the Initiator from replay attacks.

After that the traffic starts flowing between the two entities [11] and the data traffic is encrypted by ESP as the two hosts exchanged Security Parameter Index (SPI) values in the I2 and R2 messages and established the appropriate SAs [12].

3.2 The HIP Diet EXchange (DEX)

HIP BEX can introduce some overhead in memory and processor constrained devices. Thus HIP DEX intends to provide the same level of security as BEX but with the use of as few as possible cryptographic primitives [13]. DEX is a cryptographic protocol similar in philosophy to BEX but with minor changes. The main differences are summarized below:

- The DH key exchange is replaced by Elliptic Curve DH (ECDH) key exchange. RSA/DSA are also replaced by ECDSA [13].
- DEX makes use of AES-CBC encryption algorithm for providing CMAC instead of HMAC.
Fast Initial Authentication

Host Identity Protocol

- The HIT in DEX is also 128 bits as in BEX but the way it is derived is different. DEX does not apply any cryptographic hash on the HI. Instead it uses the left 96-bit of an Elliptic Curve HI, the 4 bits of the HIT suite and the HIP IPv6 prefix that is also used by BEX.

- DEX does not provide anonymity.

- DEX cannot provide perfect forward secrecy.

- DEX was designed to operate in environments with high packet loss. Therefore it supports an aggressive retransmission practice for the messages sent by the Initiator. I1 and I2 messages should be sent every delta msec until the Initiator receives R1 or R2 packets respectively.

As shown in Figure 8:

1. The first packet I1 initiates the HIP exchange as in BEX.

2. The Responder replies with an R1 packet in which it attaches a cryptographic challenge and states the cryptographic algorithms it supports.

3. In the next packet the Initiator presents the solution of the challenge and attaches also a DH key wrapper that carries a key for the Responder. This key is half of the final session key. At this point if there is a password based configured authentication the Initiator performs the appropriate actions in order to attach an authentication response to the I2 message. The I2 packet is MACed by the Initiator.

4. The R2 packet contains a DH key wrapper for the Initiator which further contains the other half of the final session key.

---

8Property stating that a short-term session key cannot be compromised if one of the long-term private keys (from which it is derived) is compromised.

---

**Figure 8: HIP Diet EXchange**

---

September 26, 2011 27
Note that there is no signature available in the above packets as with BEX. For HIP packet authentication purposes there is one DEX parameter in I2 and R2 packets which is a CMAC based message authentication code.

As stated in [13], DEX procedure is equivalent to 802.11-2007 Master and Pair-wise Transient Key (PTK) generation but in DEX it is handled in a single exchange. HIP DEX establishes two SAs. The first one for the DH derived key (Master key equivalent) and the other for the session key (PTK equivalent). The DH derived key is used to secure DEX parameters as well as authenticate the HIP packets [13]. The session key is used for traffic security and authentication.

Besides, HIP is also proposed to act as a mobility management protocol and seems to provide better results than MIPv6 [15] but this kind of HIP capabilities are beyond the scope of this report which focuses mainly on the Authentication process.
4 Design

FIA aims in three basic amendments of the IEEE 802.11 standard [16]:

1. Support of high number of simultaneously entering mobile STAs in an ESS.

2. Support of small dwell time (because of high velocity and small cell areas) within an Extended Service Area (ESA).


Moreover it should be highlighted that FIA’s scope is only related to the Authentication and Association phases. According to [17], eleven out of sixteen of the message exchanges during link setup are consumed on Authentication and two out of sixteen on Association processes. This makes a rough total of 80% of the message exchanges. Thus the reduction of the total roundtrips depends mostly on the Authentication process.

4.1 IEEE TGai Design Approaches and Evaluation

According to [18], the IEEE TGai presents three main ways of improving the performance of the Authentication process in EAP based (or RSNA enabled) wireless networks. They mostly rely on the existing authentication mechanisms and try to reduce the exchanged packets by modifying the 802.1X/EAP framework.

More specifically,

1. there is a lot of doubt about the existence of the Open System authentication which is considered to be a pre-RSNA Authentication process not acceptable any more in contemporary wireless networks.

2. There is also a doubt about how much useful the three first messages of the EAP process are.

3. Finally another proposal is introducing upper layer information on Association Request/Response messages in order to speed up the process of link establishment.

These suggestions will be further elaborated, enhanced according to new ideas that came up during the course of this thesis and finally critically evaluated according to their feasibility.

Moreover, a totally new approach towards authentication in WLAN networks will be presented based on the already described in Chapter 3 HIP protocol. The proposal is made based on personal research within the scope of this thesis report and some guidance from Robert Moskowitz; the initiator of the HIP protocol.
4.1.1 Removal of Open System authentication

Open System authentication is a null authentication algorithm as the IEEE 802.11 standard states. Any STA requesting Open System authentication can be authenticated [1].

When studying the IEEE 802.11-2007 standard one may ask what is the purpose of the above statement or why someone needs a null authentication algorithm. The answer lays on the pre-RSNA wireless networks and the vulnerabilities of WEP encryption as described in Section 2.2.1. For pre-RSNA WLANs today it is considered more secure to authenticate any mobile STA and then proceed with pre-shared WEP encryption rather than challenging it with a clear-text nonce in order to authenticate it and then use WEP encryption. The reason for this is that the latter policy allows for an easy WEP-key extraction because of the cleartext challenge.

Moreover, Open System authentication itself does not provide any security. It’s only purpose is to maintain backward compatibility with the IEEE 802.11 state machine [1].

As wireless networks will evolve in the following years they will obviously merge in RSNA-like architectures. Open System authentication seems to be a not so useful roundtrip which poses some additional overhead in the already time consuming EAP process. Figure 9 shows in a simple way the Authentication and Association states that a mobile STA should undergo during the link setup. The thicker line shows how the removal of the Open System authentication will affect the final state machine [1] [18]:

![Figure 9: Mobile STA Authentication-Association states.](image)

However, a statement from the IEEE 802.11-2007 standard [1] about ESS transitions mentions that authentication is a prerequisite for associa-
tion. Moreover, any act of Deauthentication in RSNA networks shall cause the STA to be disassociated, the IEEE 802.1X Controlled Port for that STA to be disabled and the deletion of the Pairwise Transient Key Security Association (PTKSA).

This statement implies that Deauthentication is an important function in RSNs and moreover that association is closely tightened with authentication. In other words the solution of removing the Open System authentication is a reasonable one and more or less should be incorporated in the next upcoming standards. However, there is a need of decoupling authentication from association. The state machine is posing considerable constraints on this approach.

Additionally, and as it will be later explained in Chapter 5, the Wi-Fi® implementation is not exactly following the principles of the IEEE 802.11 standard. A possible removal of the Open System authentication would require a big amount of code restructuring. This statement is justified based on the fact that a lot of other processes rely on the success of Authentication.

4.1.2 Remove EAP messages on Association Request/Response messages

As before mentioned a possible solution for FIA is to remove the three first messages of the EAP process. These messages are:

1. the EAPOL Start (optional message),
2. the EAP Request-ID and
3. the EAP Response-ID

In this way the mobile STA does not have to prove its identity, which in any case is sent in clear-text, to the AP. Generally this proposal seems capable of improving the whole Authentication process but there is not enough certainty about any security threats it may cause. Nevertheless, there are not any issues arisen up to now [19]. The main disadvantage of this approach is that IEEE 802.1X is not an entirely IEEE 802.11 related standard. This means that 802.1X is used in other standards also and a possible change of it would require an introduction of a new 802.11 oriented protocol rather than the modification of 802.1X itself.

4.1.3 Piggybacking upper layer message exchanges on Association Request/Response messages

Another interesting approach is piggybacking other time-consuming processes and perform them in parallel with the Association process during the
link setup. One of them is the DHCP IP address assignment. DHCP generally consumes a considerable amount of time of the link establishment which can be eliminated by introducing concurrency to this higher layer process.

This proposal however does not really improve the Authentication process itself, rather it is an intermediate way to accelerate the link establishment procedure by introducing a concurrent process. Unfortunately, IEEE has not yet provided the appropriate framework in order to develop such a solution but a proposal only.

Moreover, even if letting DHCP running over Association frames may be quite promising, it is a clear violation of the layer hierarchy \[20\]. In any case it is hard to imagine a network layer process to be completed by MAC layer protocol operations.

### 4.2 HIP Design Approach and Evaluation

When evaluating all the above mentioned proposals some doubts arise about their efficiency or their ability to be adopted either by the IEEE 802.11 standard or the Wi-Fi\textsuperscript{®} Alliance. Robert Moskowitz suggested that HIP and especially the DEX version has all the appropriate characteristics in order to be used as a key exchange mechanism in a MAC layer security protocol \[13\]. In the context of this degree project, the author developed a method that presents how DEX could probably be used in order to provide a lightweight authentication mechanism to 802.11 networks. The basic principles of this proposal were also presented in IEEE TGai meeting at Palm Springs in California (10 of May 2011) \[21\] as an alternative solution for the Group’s main goal, which is Fast Initial Link Setup. A detailed version of this presentation is being documented hereby as well as in a recently IEEE ICC 2012 submitted conference paper.

Having described all the background of the concepts involved in a HIP based Authentication process, it should be clear by now that there can be direct application of DEX in IEEE 802.11 standard for entity authentication and key generation. What remains to be defined is how HIP can be integrated in the current standard and act as a Key Management System (KMS). Note that HIP’s BEX version has been already tested in 802.11 networks but does not seem to achieve the desired results for real-time applications \[22\].

#### 4.2.1 Leightweight Authentication and Key Management with HIP DEX

One of the main advantages of HIP (both BEX and DEX) is that it fits directly into the key model that 802.11 standard has introduced (MK, PTK, GTK). The first thoughts about integrating HIP into such a process is to let HIP datagrams run over 802.11 Authentication frames \[23\]. The MK and PTK keys would be delivered as already mentioned in Section \[3.2\] The
GTK could be delivered on an Association Response frame as a reply to an Association Request frame that contains a HIP UPDATE datagram. The HIP UPDATE can generally act as a rekeying mechanism when needed.

Hereby, we should mention that terms like mobile STA, Supplicant and Initiator may be used interchangeably from now on and depending to the context. The same applies for Authenticator and Responder terms.

A suitable deployment could consider a central wireless controller to act as a HIP Responder and its assigned APs as relays of traffic between the HIP Initiator and the HIP Responder [22]. The APs may introduce a Port based Network Access Control as the one in use by 802.1X framework for ensuring that only authorized Supplicants may have access to the network. By adding a new Information Element to the beacon and Authentication frames we can firstly announce the HIP capabilities of the network [22] and secondly distinguish HIP traffic. In this way only HIP traffic will be allowed to flow between the Initiator and the Responder until the Supplicant gets authenticated.

![Figure 10: HIP DEX authentication network architecture](image)

The above scheme introduces a much simpler architecture and seamless handovers within the same ESS. More specifically the established HIP DEX SAs are preserved during handovers within the same ESS as the SA establishment is valid between the mobile STAs and the wireless controller. Thus the controller and only the controller, which has the appropriate level of trust by the AS, should be responsible for communicating with the AS. Hereby we assume that the Initiator and the Responder share long lived SAs and credentials with the AS and create SAs between them according to the described protocol. The SA between the Initiator and the AS is used for mutual authentication purposes. Figure 10 shows how this entities could possibly interact according to their roles.
The procedure should be rather simple. According to Figure 11, the basic steps are:

1. The APs transmits beacon frames that advertise the HIP capabilities of the network as well as the Responder’s address (alternatively the mobile STA could perform active scanning and begin a HIP message exchange to Responder’s link-local address or pre-defined multicast address [22]).

2. The Initiator performs Open System authentication and Association. Both ports of the AP are blocked in the beginning.

3. The Initiator (acts as Supplicant) starts a DEX exchange with the Responder (acts as Authenticator) where the AP act as relay of traffic. The uncontrolled Port unblocks in order to let HIP traffic flow to the Responder.

4. The Responder after the reception of the I2 message communicates with AS in order to authenticate the Initiator and replies accordingly back.

5. Setup of ESP SAs.

6. Flow of ESP protected traffic (no HIP overhead).

Figure 11: HIP Fast Initial Authentication

The wireless controller could support tens of APs. However, in the case of ESS transition (that is translated to wireless controller transition) there
should also be defined a HIP based mobility solution. Mobility could either include rekeying or not and should use the HIP UPDATE message in order to inform peers about the change of address.

In this way DEX certainly promises the reduction of the Authentication process messages, not to mention the fast transitions during BSS handovers. As shown in Figure [11], the Authentication process roundtrips are reduced to two and this is the reason we believe that DEX can provide delays that can be tolerated by most of the time sensitive applications. More specifically, comparing to the current IEEE 802.11 WPA2 AKM the solution we hereby propose can prove to be quite efficient. In terms of exchanged data the HIP DEX based approach needs no more than 550 bytes to complete the AKM process. Table [2] summarizes the DEX messages’ length [24]. Taking into account that every DEX message is encapsulated into an Authentication frame (approximately 40 bytes) this makes a total of 542 bytes for the whole 4-way handshake.

Table 2: HIP DEX message length with a fixed 160 bits ECDH key exchange

<table>
<thead>
<tr>
<th>Message</th>
<th>Length (bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>I1</td>
<td>40</td>
</tr>
<tr>
<td>R1</td>
<td>92</td>
</tr>
<tr>
<td>I2</td>
<td>148</td>
</tr>
<tr>
<td>R2</td>
<td>102</td>
</tr>
<tr>
<td>Total</td>
<td>382</td>
</tr>
</tbody>
</table>

On the contrary a WPA2 operation may require the exchange of up to 1300 bytes until the whole AKM process is completed. Although this number may fluctuate depending on the used WPA2 security specific mechanisms, the advantage of the above described solution is the seamless BSS handovers and the quite low overhead DEX pauses on the wireless controller during the ESS transitions.

What HIP DEX can offer is AKM in 2 roundtrips. This is quite impressive in terms of number of message exchanges. However, as discussed in [22] HIP’s BEX challenge may introduce some additional time during the calculation from the receiver side point of view. This makes BEX not the ideal candidate for time sensitive applications. Of course the purpose of DEX is to simplify and eliminate the above constraints. The real performance of DEX is not yet known and it remains to be tested in order to know if this promising solution could be adopted for FIA purposes.

Additionally, it would be a big change for IEEE 802.11 standard to adopt a completely new concept for AKM operations. Mostly the willingness of the vendors can define the potentiality of this migration as soon as they are
convinced that HIP can make the difference. But it seems not feasible at this moment.
5 Implementation

The before mentioned approaches towards FIA require modification of the Linux kernel code. As shown in Figure 12 there are different subsystems in the kernel code which are involved in the IEEE 802.11 standard implementation and the drivers’ support [25]:

- mac80211: Kernel subsystem responsible for implementing shared code for soft-MAC/half-MAC wireless devices. It contains the MAC Layer Management Entity (MLME).
- cfg80211: Layer between userspace and mac80211 drivers. It mainly performs sanity checking and protocol translations.
- nl80211: Userspace access to cfg80211 operations. Used by wpa_supplicant and hostapd.
- wpa_supplicant: Userspace module supporting cfg80211.
- hostapd: Userspace module implementing the AP MLME. Closely tightened with nl80211.
- userspace Station Management Entity (SME): Support of separate Authentication/Association commands.

![Kernel architecture of the wireless subsystem.](image)

The actual place where everything begins is the WPA Supplicant, which is normally the wireless control unit for every mobile STA device. In Linux it is implemented by the wpa_supplicant module which is responsible for the key negotiation of the Supplicant with the Authenticator and controls the handovers as well as the Authentication and Association phases [26]. The corresponding module for the AP or Authenticator functions is the hostapd [27]. Both of the modules share some common code and they directly interact with the lower level subsystems through the nl80211 interface.
5.1 Linux Code Structure

When dealing with massive amount of code a human only analysis and comprehension of it is almost impossible. For this reason there are documentation systems like Doxygen which can generate a documentation browser of several html files [28]. There is also the possibility of forming function caller-callee graphs which further simplifies the analysis process of the code. Appendix A provides a quick guide in Doxygen and configuration details according to the dissertation specific needs. Note that the documentation build process of the above modules may take up to 24 hours depending on the used system’s performance.

Note that all the following code analysis and modifications were done in the context of this thesis project.

5.1.1 Supplicant side

What the extract of Doxygen revealed about the code performing the AKM operations from the supplicant’s point of view is illustrated in Figure 13.

The wpa_supplicant_event() function is critical for any code modification from now and on. The actual Authentication/Association exchanges begin here.

The supplicant’s code works basically with events coming up from the drivers. The first action that a mobile STA performs when asked to connect to a network is to scan for the available APs. As soon as it receives the beacon frames or a Probe Response depending on the type of scanning it supports, it generates an EVENT_SCAN_RESULTS event in the wpa_supplicant_event() function and continues normally further to the wpa_supplicant_scan_results() where it picks the requested network. Then it transfers to the wpa_supplicant_associate() function through the intermediate wpa_supplicant_connect(). All the next AKM operations continue by following the same idea. There are events for every process like

- EVENT_AUTH,
- EVENT_DEAUTH,
- EVENT_ASSOC,
- EVENT_DISASSOC,
- EVENT_EAPOL_RX

and others. In any case and depending to the modifications someone wants to do, it is advisable to start from this point in order to understand the overall AKM process.

When further analyzing wpa_supplicant_associate() someone can observe that it is a step to two different ways of implementing the wpa_supplicant
Fast Initial Authentication

Implementation

Figure 13: Graph of functions for supplicant’s authentication/association
operations. According to the header file `driver.h` where all the driver flags are defined, there is a flag that specifies if the supplicant will authenticate and associate with a single command or separate ones. In other words this is the place where the code defines whether user space SME is supported or not.

If the flag is set the code will follow the branch of SME functions in `sme.c` source file starting from the `sme_authenticate()` function. Otherwise the code will continue up to another driver flag in order to check if the driver supports user space MLME. According to the statement’s result association starts from `ieee80211_association()` function or directly from `wpa_drv_associate()` function which is “closer” to the driver’s level.

```
define WPA_DRIVER_FLAGS_USER_SPACE MLME 0x00000004
```

/* Driver provides separate commands for authentication and association (SME in wpa_supplicant). */
```
define WPA_DRIVER_FLAGS_SME 0x00000020
```

### 5.1.2 AP side

Almost the same event driven concept applies from the AP’s point of view. The `ieee802_11_mgmt()` function is the place where all the incoming management frames (from the driver) are being processed. The possible management frame types listed below are defined in `ieee802_11_defs.h` header file.

```
define WLAN_FCSTYPE_ASSOC_REQ 0
#define WLAN_FCSTYPE_ASSOC RESP 1
#define WLAN_FCSTYPE_REASSOC_REQ 2
#define WLAN_FCSTYPE_REASSOC RESP 3
#define WLAN_FCSTYPE_PROBE REQ 4
#define WLAN_FCSTYPE_PROBE RESP 5
#define WLAN_FCSTYPE_BEACON 8
#define WLAN_FCSTYPE_ATIM 9
#define WLAN_FCSTYPE_DISASSOC 10
#define WLAN_FCSTYPE_AUTH 11
#define WLAN_FCSTYPE_DEAUTH 12
#define WLAN_FCSTYPE_ACTION 13
```

### 5.1.3 Code Path

The `wpa_supplicant` module as well as the `hostapd` one are highly depending and cooperating with the system’s drivers. That is why in order to cover different driver types the code has concluded to be complex and not flexible in the name of backward compatibility. In the case of this report Appendix B summarizes the hardware and software specific details that were used during the implementation and testing phases.

According to these details the code documented in this report is following the SME path which will be further explained in the following section.
5.2 Removal of Open System authentication

As discussed in Section 4.1.1 theoretically the removal of the Open System authentication is desirable. In this Section there will be discussion of its feasibility in a practical level.

5.2.1 Code Analysis

As soon as the wpa_supplicant reaches the point of \textit{wpa_supplicant\_associate()} function it continues its execution with the functions that are defined in \textit{sme.c}. This means that the next under call function is the \textit{sme\_authenticate()} which is responsible for implementing the Open System authentication. Its main role is filling the appropriate parameters that will be later passed by \textit{wpa\_drv\_authenticate()} function and its callee to the driver.

On the other side and as soon as the authentication request is sent to the AP the hostapd module receives the frame from the driver and finds out that its type is of \textit{WLAN\_FC\_STYPE\_AUTH} and invokes the \textit{handle\_auth()} function in order to send an authentication response.

Back to the mobile STA and upon reception of the AP response frame an \textit{EVENT\_AUTH} event comes up and the \textit{sme\_event\_auth()} function is called. The latter calls the \textit{sme\_associate()} which depending on the result of the authentication response, begins the association with the AP. This is done by updating or initializing all the required parameters and calling \textit{wpa\_drv\_associate()}.

5.2.2 Code Modifications

Unfortunately a direct call of the \textit{sme\_associate()} or \textit{wpa\_drv\_associate()} functions after the scanning process is completed would not work. This means that the code would not send an association request frame; instead it would generate a segmentation fault. All the functions are interconnected with a lot of others on which they depend. Parameters that are retrieved during the authentication are used in the association phase also.

Any possible changes should take place in a low level in order to assure that all the other inter-dependent processes are not affected by the changes of the code. According to this policy a suitable modification would be to invoke the \textit{sme\_associate()} function through the \textit{sme\_authenticate()}. Such a modification would skip the authentication directly before invoking the \textit{wps\_drv\_authenticate()} and would start by associating the mobile STA to the AP. Hereby the first arisen question is how much the delay is if we follow the authentication path in order to initiate the association process.

Such a change still requires the pass of several parameters that the wpa_supplicant would otherwise acquire after the authentication process has been completed. For instance some of these parameters are the bssid or in other words the MAC address of the \textit{authenticated} AP, the ssid or in other
Fast Initial Authentication

Implementation

words the name of the authenticated network, its string length, the used frequency in MHz, any vendor specific IEs (which in this case should be passed in another way), the management frame protection option etc. The existence of Open System authentication at this point has already started justifying itself at least at its practical level especially when it comes to the vendor specific IEs.

The below part of the code reveals the parameters that the `sme_associate()` depends on as well as their dependency from the parameters passed by the `sme_event_auth()` function.

```c
322 void sme_associate(struct wpa_supplicant *wpa_s, enum wpas_mode mode,
323 const u8 *bssid, u16 auth_type)
324 {
325     struct wpa_driver_associate_params params;
326     struct ieee802_11_elems elems;
327     os_memset(&params, 0, sizeof(params));
328     params.bssid = bssid;
329     params.ssid = wpa_s->sme.ssid;
330     params.ssid_len = wpa_s->sme.ssid_len;
331     params.freq = wpa_s->sme.freq;
332     params.wpa_ie = wpa_s->sme.assoc_req_ie_len ?
333         wpa_s->sme.assoc_req_ie_len :
334         wpa_s->sme.assoc_req_ie_len;
335     #ifdef CONFIG_IEEE80211R
336         if (auth_type == WLANAUTH_FT && wpa_s->sme.ft_ies) {
337             params.wpa_ie = wpa_s->sme.ft_ies;
338             params.wpa_ie_len = wpa_s->sme.ft_ies_len;
339         }
340     } #endif /* CONFIG_IEEE80211R */
341     params.mode = mode;
342     params.mgmt_frame_protection = wpa_s->sme.mfp;
343     if (wpa_s->sme.prev_bssid_set)
344         params.prev_bssid = wpa_s->sme.prev_bssid;
```

When analyzing the code there are dependencies to be resolved as the ones mentioned above and the structure of the code needs to be changed accordingly in order to just skip the Open System authentication roundtrip. This explains partly the fact that such a process is still kept into the standard. It would require some effort to resolve all the existing possible dependencies on the 802.11 standard implementations (both pre-RSNA and RSNA ones). Figure 14 shows the basic idea behind the change that occurred in the `wpa_supplicant` side in order to skip Open System authentication.

For this purpose a new similar to the `sme_associate()` function, called `my_sme_associate` was developed in the context of this thesis. Its main purpose is to substitute the former one by avoiding the parameter passing from the `sme_event_auth` caller function. This passing used to appear after each `EVENT_AUTH` (the reception of the authentication response). Moreover the call of the `my_sme_associate()` function should occur directly after the invocation of `sme_authenticate()`. The reasons why `sme_authenticate()` is still kept is because of the parameters and the system checks that take place in it.
Fast Initial Authentication

Implementation

Figure 14: Graph of functions for supplicant’s association after skipping authentication
In the hostapd module there is only one check in the `handle_assoc()` function that needs to be removed. More specifically the function, before allowing a mobile STA to associate, checks if an STA has been previously authenticated (mandatory by the standard). If it has it proceeds to the association and if not it fails (see the part of code below). Such a check would always abort a link setup process that has not previously undergone an Open System authentication.

```c
if (sta == NULL || (sta->flags & WLAN_STA_AUTH) == 0) {
    hostapd_logger(hapd, mgmt->sa,
                   HOSTAPD_MODULE_IEEE80211,
                   HOSTAPD_LEVEL_INFO, "Station tried to 
                 "associate before authentication 
                 "(aid=%d flags=0x%x)",
                 sta ? sta->aid : -1,
                 sta ? sta->flags : 0);
    send_deauth(hapd, mgmt->sa,
               WLAN_REASON_CLASS2_FRAME_FROM_NONAUTH_STA);
    return;
}
```

The above code modifications should be enough in order to allow the wpa_supplicant and hostapd modules to interact with each other by skipping the Open System authentication. Although these changes may be valid in the user space and cannot be questioned it is still unknown if they are compatible with the lower level subsystems. For this purpose in the following Section there will be a detailed description of the testing environment setup and the testing procedure as well, in order to explore further the RSN WPA2 operations and finally the implementation results.
5.3 Implementation Evaluation

The above implementation cannot be tested unless a real testing environment is set up. The basic idea is to build an AP with an integrated RADIUS server and try to authenticate through a mobile STA to it. In this Chapter there will be detailed description of how to set up such a testing environment as this process can be quite frustrating when it comes to RSNA WPA2 Enterprise WLANs.

The very first part of the testing process is to setup the devices involved in it. That means the configuration of the AP and the mobile STA with the hostapd and the wpa_supplicant modules respectively.

5.3.1 Setup of hostapd

Starting from the AP side, the first step is to download the source files of the hostapd module:

1. wget http://w1.fi/releases/hostapd-0.7.3.tar.gz
2. tar xzvf hostapd-0.7.3.tar.gz
3. cd hostapd-0.7.3/hostapd

In order to compile them and build the module there is a need to adjust the configuration parameters according to the driver specifications of the wireless Network Interface Card (NIC). For this purpose there is a defconfig file in ./hostapd-0.7.3/hostapd/ directory which needs to be modified and saved as a separate configuration .config file. A .config file can be created:

4. cp defconfig .config
5. vi .config

In this case the AP driver is the ath5k one which uses the nl80211 kernel interface. So it is important to enable the CONFIG DRIVER NL80211 configuration option. From that point and on there might be some library updates or additions that should take place depending on the used Linux distribution. A sample .config file used in the thesis related systems is attached together with all the other thesis related files (note that it is a hidden file as it starts with a dot).

After this the source files can be compiled with the help of the Makefile by simply running:

6. make clean
7. make

Furthermore, it is important to adjust the AP characteristics according to the desired security levels. As already mentioned WPA2 Enterprise is a more complex security policy which requires a RADIUS authentication server. For this purpose there is a hostapd.conf file which includes all the possible network specific settings. As with the .config file there is a need for configuration of these various settings. Some of them are:

- **interface**: Setting the wireless interface name
• **driver**: Setting the driver interface type

• **ssid**: Setting the name of the *Service Set Identifier (SSID)*

• **hw_mode**: Setting the operation mode of the used interface as well as the allowed channels. Depends mostly on the hardware capabilities.

• **channel**: Setting the hostapd operating channel

• **auth_algs**: Setting the used authentication mechanism (Open System, Shared Key)

• **wpa**: Setting the used WPA version (WPA1, WPA2)

• **wpa_key_mgmt**: Setting the key management algorithms

• **wpa_pairwise**: Setting the cipher suites (encryption algorithms)

• **ieee8021X**: Enabling the 802.1X authorization

• **eap_server**: Use of the integrated EAP server instead of RADIUS server

• **eap_user_file**: Path to the EAP server user database

• **ca_cert**: Path to the CA certificate for EAP-TLS/PEAP/TTLS

• **server_cert**: Path to the server certificate for EAP-TLS/PEAP/TTLS

• **private_key**: Path to the private key matching with the server certificate

• **radius_server_clients**: File name of the RADIUS clients configuration (if this setting is commented out, the RADIUS server is disabled)

• **radius_server_auth_port**: The UDP port number of the RADIUS server

As stated on the last bullets there should be a RADIUS server in order to achieve WPA2 Enterprise security levels. In the context of this thesis work EAP-TLS was the used EAP protocol. The configuration of the integrated EAP server requires the use of certificates for support of mutual authentication. These certificates should be signed by a *Certification Authority (CA)* which in this case does not need to be any of the well-known ones but the owner of the AP. All the needed server keys and certificates can be obtained by running the openssl based `crtgen.sh` script [29]. After that the EAP database (hostapd.eap_user file) should be configured in order to much the users to the corresponding version of EAP. For TLS inserting an entry like:
Fast Initial Authentication

Implementation

1) TLS

should be enough in order to much all the users with EAP-TLS. Finally the hostapd.radius_clients file matches the AP IP addresses to passwords in order to authenticate them to the RADIUS server. An entry like:

1 | 0.0.0.0/0 password

would set a common password for all the APs that want to communicate with the RADIUS server.

All the above mentioned configuration files and scripts are also included together with all the other thesis related files. As soon as the AP configuration has been completed the following commands should enable it and let it run.

1) sudo service network-manager stop
2) sudo ./hostapd ./hostapd.conf -d # -d for debugging info

5.3.2 Setup of wpa_supplicant

From the mobile STA’s side, a similar procedure should be followed. The source code can be downloaded by running:

1) wget http://wl.fi/releases/wpa_supplicant-0.7.3.tar.gz
2) tar xzvf wpa_supplicant-0.7.3.tar.gz
3) cd wpa_supplicant-0.7.3/wpa_supplicant/

As in the hostapd case in order to compile and build the module there is a need to adjust the configuration parameters according to the driver specifications. For this purpose there is a defconfig file in ./wpa_supplicant-0.7.3/wpa_supplicant/ directory which needs to be modified and saved as a separate configuration .config file. A .config file can be created as previously with:

4) cp defconfig .config
5) vi .config

In this case the mobile STA’s driver is the iwlan one and uses the nl80211 kernel interface. So it is important to enable the CONFIG_DRIVER_NL80211 configuration option. From that point and on there might be some library updates or additions that should take place depending on the used Linux distribution. A sample .config file used in the thesis related systems is attached together with all the other thesis related files (note that it is a hidden file as it starts with a dot).

After this the source files can be compiled by simply running:

6) make clean
7) make

It is also important to adjust the mobile STA’s characteristics according to the desired security levels. For this purpose there is a wpa_supplicant.conf file which includes all the network specific settings, Appendix includes all
the needed configuration used in the context of this thesis report. As with the `.conf` file there is a need for configuration of these various settings. Some of them are:

- **ssid**: Name of the desired *Service Set Identifier (SSID)* to connect with.
- **proto**: Desired protocol to be used (WPA/WPA2).
- **key_mgmt**: Key management algorithms.
- **pairwise**: Unicast cipher suites (encryption algorithms).
- **group**: Broadcast cipher suites (encryption algorithms).
- **eap**: List of accepted EAP methods.
- **ca_cert**: Path to the CA certificate for EAP-TLS/PEAP/TTLS
- **client_cert**: Path to the client certificate for EAP-TLS/PEAP/TTLS
- **private_key**: Path to the private key matching with the client certificate

The mobile STA should also have the proper certificates and keys installed. The `openssl` based `certgen.sh` script has been already mentioned provides them for the client (mobile STA) side as well.

All the above mentioned configuration files and scripts are included together with all the other thesis related files. As soon as the `wpa_supplicant` configuration has been completed the following commands should enable it and let it run.

The last set of commands:

```
8 sudo cp wpa_supplicant.conf /etc/
9 sudo cp wpa_cli wpa_supplicant /usr/local/bin
10 sudo service network-manager stop
11 sudo ifconfig wlan0 up
12 sudo ./wpa_supplicant -dd -iwlan0 -c/etc/wpa_supplicant.conf
```

allow the `wpa_supplicant` to start the link setup process.

### 5.3.3 A Normal WPA2 operation

As soon as the link setup process has started the two involved entities (AP and mobile STA) exchange all the proper messages in order to be able to communicate with each other as described in Sections 2.2.2-2.2.5. As the process proceeds both the `hostapd` and `wpa_supplicant` modules print debugging messages about the status of the link setup process. `debugHOSTAP.pdf` and `debugWPAS.pdf` files include all of this debugging information.
Results regarding a usual RSN WPA2 operation can now be drawn. The basic tool that was used in order to observe this operation was Wireshark [30]. Wireshark was always running after the initialization of the hostap module and before the wpa_supplicant in order to be able to capture all the exchanged packets.

For the testing purposes ten packet captures took place. All of them followed the above described procedure. After processing the captured files and applying the appropriate filters in order to exclude any interference from other devices besides the ones involved in the tests, the main results of the experiments are illustrated in Figures 15 and 16.

![Figure 15: Open System authentication duration](image)

As shown in Figure 15, the Open System authentication roundtrip introduces a delay of 1640 µs in average and with a standard deviation of approximately 290 µs. Additionally in Figure 16, someone can observe that the Association roundtrip adds 3160 µs in average to the previous value. That is because the two entities are performing more complex operations as the link setup is in progress. As already mentioned Open System authentication is the lightest exchanged roundtrip in the overall operation.

5.3.4 Evaluation outcomes

After the testing procedure was set up and applied to a normal WPA2 operation, as a reference point, it was time to test the code modifications,
This operation provided a lot of feedback about resolving any faults occurred in the modified code. As the modifications took place in a blackbox programming environment, it was only possible to debug them only by testing.

Unfortunately after trying different adjustments the code keeps generating the below noted nl80211 error message:

“nl80211: MLME command failed: ret=-107 (Transport endpoint is not connected)”

All of the debugging information is included in the debugWPASerror.pdf.

The process of understanding the above mentioned error by the nl80211 interface included a detailed review of the IEEE 802.11 standard literature. More specifically and as it can be seen in Figure 17 there are classes of frames that are strictly related to each state. State 1 allows only Class 1 frames, State 2 allows Class 1 & 2 frames and State 3 allows all the three frame Classes to be exchanged. Table 3 gives more details about which frame types belong to each class. The IBSS frame types are excluded as they are not of particular interest in the context of this report.

When a mobile STA is in State 1 it is allowed to exchange only Class 1 frames. The standard defines that only successful authentication can enable the STA to exchange Class 2 frames and respectively association requests. If the STA does not authenticate to the AP it will remain forever in State 1 unauthenticated and unassociated. In the same way, association enables
Fast Initial Authentication

Implementation

Figure 17: IEEE 802.11 standard state machine

the exchange of Class 3 frames or not. This indication shows that in order to skip the Open System authentication there should be a change of the IEEE 802.11 state machine. This is the reason that the nl80211 is not able to perform the command of association without previously completing authentication.

This observation as already mentioned partly explains the nl80211 received error. To be more specific all subsystems are following the state machine as it is the basic abstract algorithm of the 802.11 standard. Hence, removing Open System authentication requires considerable amount of work in different layers of code implementation. That means that not only the wpa_supplicant should be modified but the cfg80211 and mac80211 layers also. This fact was not clear from the beginning and until exploring and delving into the Linux code.
Table 3: 802.11 states and frame Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Frame type</th>
<th>Frames</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 1</td>
<td>Control</td>
<td>Request To Send</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clear To Send</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ACKnowledgement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ContentionFree-End + ACK</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CF-End</td>
</tr>
<tr>
<td></td>
<td>Management</td>
<td>Probe Request/Response</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Beacon</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Authentication</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Deauthentication</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Announcement Traffic Indication Message</td>
</tr>
<tr>
<td>Class 2</td>
<td>Management</td>
<td>Association Request/Response</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Reassociation Request/Response</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Disassocation</td>
</tr>
<tr>
<td>Class 3</td>
<td>Data</td>
<td>Data subtypes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>QoS data subtypes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ACK</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CF-End</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CF-End + ACK</td>
</tr>
<tr>
<td></td>
<td>Management</td>
<td>QoS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DLS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Block Ack Action</td>
</tr>
<tr>
<td></td>
<td>Control</td>
<td>Power Save</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Block Ack</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Block Ack Request</td>
</tr>
</tbody>
</table>
6 Conclusions and Future Work

As it may have been understood FIA is a real challenge that needs to be addressed. Different ways of facing it have been proposed in a discussion that started on May 2010 and is ongoing up to now. According to the author’s knowledge and by the time this report was submitted, there was not any other effort of implementing any of the hereby described approaches in any platform. Therefore it seems that there is still a lot to be done in the implementation area before evaluation with real measurements takes place.

After this thesis project completion it became obvious how complex IEEE 802.11 standard implementation has become. Different vendors and the regulations IEEE has imposed, have concluded into a rather complex code and in kernel subsystems that are closely tightened together. From the author’s personal point of view the analysis of the Linux code as well as the black box programming required in order to modify it was an interesting, though cumbersome process. Moreover, building a WPA2 Enterprise mode WLAN from the scratch was definitely a useful and quite informative experience as it required different kind of configurations such as driver installations, module compilations, scripts for certificate generations etc. Finally, participating on IEEE TGai weekly teleconferences was a rather interesting experience, not to mention the chance of presenting the HIP based solution at Palm Springs in California.

As far as the future work is concerned having the above described code analysis and testing setup it should be much quicker and easier for anyone to familiarize him- or herself to the Linux IEEE 802.11 implementation and the set up of a WLAN. What remains is a research for further modifications that would allow the link setup process to skip the open System authentication. As far as the rest of the suggestions it would definitely be interesting to see a DEX implementation and performance measurements of the design approach proposed hereby.

But FIA depends mainly on the IEEE TGai and that is why the decisions to come will be rather decisive.
A Appendix

Setting up Doxygen starts by downloading its code and then

1. `wget ftp://ftp.stack.nl/pub/users/dimitri/doxygen-1.7.4.linux.bin.tar.gz`
2. `tar xzvf doxygen-1.7.4.linux.bin.tar.gz`
3. `./configure`
4. `make install`

from that point and on the configuration file should be edited according to the needs of the user and then

5. `doxygen <config-file>`

A sample configuration file used in the context of this thesis report is included with the other thesis related files.
B Appendix

The used for implementation and testing purposes systems were a Wistron CM9 wireless NIC adjusted with a miniPCI adapter on a Linux Ubuntu machine for the AP and a laptop embedded wireless NIC on a Linux Ubuntu machine also. Both OSs were running Ubuntu 10.10 distribution.

AP

Ethernet controller: Atheros Communications Inc. Atheros AR5001X+ Wireless Network Adapter (rev 01)
Subsystem: Wistron NeWeb Corp. CM9 Wireless a/b/g MiniPCI Adapter
Flags: bus master, medium devsel, latency 168, IRQ 18
Memory at dfaf0000 (32-bit, non-prefetchable) [size=64K]
Capabilities: access denied
Kernel driver in use: \texttt{ath5k}
Kernel modules: \texttt{ath5k}

mobile STA

Network controller: Intel Corporation WiFi Link 5100
Subsystem: Intel Corporation WiFi Link 5100 AGN
Flags: bus master, fast devsel, latency 0, IRQ 46
Memory at d4200000 (64-bit, non-prefetchable) [size=8K]
Capabilities: access denied
Kernel driver in use: \texttt{iwlagn}
Kernel modules: \texttt{iwlagn}
C Abbreviations

AAA Authentication Authorization Accounting
ACK ACKnowledgement
AES Advanced Encryption Standard
AKM Authentication and Key Management
AP Access Point
AS Authentication Server
BEX Basic EXchange
BSA Basic Service Area
BSS Basic Service Set
CA Certificate Authority
CCMP Counter mode with Cipher block chaining Message authentication code Protocol
CP Controlled Port
DEX Diet EXchange
DFS Dynamic Frequency Selection
DH Diffie Hellman
DHCP Dynamic Host Configuration Protocol
DS Distribution System
DSM Distribution System Medium
DSS Distribution System Service
EAP Extensible Authentication Protocol
EAPOL Extensible Authentication Protocol Over LAN
ESP Encapsulating Security Payload
ESS Extended Service Set
FIA Fast Initial Authentication
FSR Fast Secure Roaming
FT Fast BSS Transition
FTIE Fast BSS Transition Element
GMK Group Master Key
GTK Group Transient Key
HI Host Identifier
HIP Host Identity Protocol
HIT Host Identity Tag
IBSS Independent Basic Service Set
IE Information Element
IEEE Institute of Electrical and Electronics Engineers
IP Internet Protocol
kmh kilometers per hour
LLC Logical Link Control
MAC Media Access Control
MDIE Mobility Domain Information Element
MIC Message Integrity Code
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MLME</td>
<td>MAC Layer Management Entity</td>
</tr>
<tr>
<td>mps</td>
<td>meters per second</td>
</tr>
<tr>
<td>MSDU</td>
<td>MAC Service Data Unit</td>
</tr>
<tr>
<td>MSK</td>
<td>Master Session Key</td>
</tr>
<tr>
<td>NIC</td>
<td>Network Interface Card</td>
</tr>
<tr>
<td>OBU</td>
<td>On-Board Unit</td>
</tr>
<tr>
<td>PMK</td>
<td>Pairwise Master Key</td>
</tr>
<tr>
<td>PTK</td>
<td>Pairwise Transient Key</td>
</tr>
<tr>
<td>PTKSA</td>
<td>Pairwise Transient Key Security Association</td>
</tr>
<tr>
<td>RSN</td>
<td>Robust Security Network</td>
</tr>
<tr>
<td>RSNA</td>
<td>Robust Security Network Association</td>
</tr>
<tr>
<td>RSNIE</td>
<td>Robust Security Network Association Information Element</td>
</tr>
<tr>
<td>RSU</td>
<td>Road-Side Unit</td>
</tr>
<tr>
<td>SA</td>
<td>Security Association</td>
</tr>
<tr>
<td>SME</td>
<td>Station Management Entity</td>
</tr>
<tr>
<td>SPI</td>
<td>Security Parameter Index</td>
</tr>
<tr>
<td>STA</td>
<td>STAtion</td>
</tr>
<tr>
<td>TG</td>
<td>Task Group</td>
</tr>
<tr>
<td>TKIP</td>
<td>Temporal Key Integrity Protocol</td>
</tr>
<tr>
<td>TLS</td>
<td>Transport Layer Security</td>
</tr>
<tr>
<td>TPC</td>
<td>Transmit Power Control</td>
</tr>
<tr>
<td>TSN</td>
<td>Transition Security Networks</td>
</tr>
<tr>
<td>UP</td>
<td>Uncontrolled Port</td>
</tr>
<tr>
<td>VLAN</td>
<td>Virtual Local Area Network</td>
</tr>
<tr>
<td>VoIP</td>
<td>Voice over Internet Protocol</td>
</tr>
<tr>
<td>VPN</td>
<td>Virtual Private Network</td>
</tr>
<tr>
<td>WEP</td>
<td>Wired Equivalent Privacy</td>
</tr>
<tr>
<td>WIDS</td>
<td>Wireless Intrusion Detection System</td>
</tr>
<tr>
<td>WIPS</td>
<td>Wireless Intrusion Prevention System</td>
</tr>
<tr>
<td>WLAN</td>
<td>Wireless Local Area Network</td>
</tr>
<tr>
<td>WPA</td>
<td>Wi-Fi Protected Access</td>
</tr>
</tbody>
</table>
References


September 26, 2011 58
REFERENCES


