DIPLOMARBEIT

A Survey of WLAN Security with Focus on HotSpot and Enterprise Environments

Ausgeführt am

Institut für Informationssysteme
Abteilung für Verteilte Systeme
Technische Universität Wien

unter der Anleitung von
o.Univ.Prof. Dipl.-Ing. Dr.techn. Mehdi Jazayeri
und
Univ.Ass. Dipl.-Ing. Dr.techn. Engin Kirda
als verantwortlich mitwirkendem Universitätsassistenten

durch

Lorenz Froihofer
8654 Fischbach 69a
Matr.Nr. 9825055

Wien, Februar 2004 ________________________

Acknowledgements

First of all I'd like to thank SIEMENS AG Österreich for the support in finding this thesis through the Siemens TopStudents program and for providing the development environment. Many thanks go to the whole team around Fritz Kasslatter for the many fruitful discussions and the provided support during the writing of this thesis.

Furthermore, I thank Engin Kirda for his advice and guidance whenever questions arose as well as for proof-reading this thesis. Thanks go also to Mehdi Jazayeri who enabled me the writing of this thesis.

Additional thanks go to my parents, my friends and all the people who supported me directly or indirectly during my study.

Finally, I want to thank the Linux Community and the Open Source Movement for doing a great job. Otherwise, this thesis might not have been possible.

Danksagung

Zu allererst möchte ich mich bei der Firma SIEMENS AG Österreich, vor allem bei den Betreuern des Siemens TopStudents Programms, für die Unterstützung bei der Suche nach einem Diplomarbeitsthema bedanken. Weiters bedanke ich mich für die Bereitstellung der Entwicklungsumgebung. Mein Dank gilt auch dem Team von Fritz Kasslatter für die vielen anregenden Diskussionen und den Support während meiner Arbeit.

Weiterer Dank gebührt Engin Kirda für seinen Rat und seine Hilfestellungen, wann auch immer sich Fragen stellten. Zusätzlich bedanke ich mich bei ihm für das Korrekturlesen der Diplomarbeit. Mein Dank gilt auch Mehdi Jazayeri, der mir das Schreiben dieser Diplomarbeit ermöglichte.

Bedanken möchte ich mich auch bei meinen Eltern, meinen Freunden und allen Personen, die mich während meines Studiums direkt oder indirekt unterstützt haben.

Sehr schätze ich auch die Leistungen der Linux Gemeinschaft und der Open Source Bewegung, ohne die meine Diplomarbeit in dieser Art und Weise nicht möglich gewesen wäre.

Abstract

This thesis provides an overview of the current security situation in IEEE 802.11 networks (WLANs). An evaluation of the situation shows the current issues and some requirements for the future.

Furthermore, the thesis provides a preview of the 802.11i standard. 802.11i is regarded to be the final security solution for WLANs because it claims to fix all known security weaknesses of WLANs today. Additionally, Wi-Fi Protected Access (WPA) as the already available security solution based on 802.11i is presented and differences to 802.11i are described.

Based on this information, the migration challenge to the new security mechanism 802.11i is discussed for wireless hotspot operators and large scale enterprises.

Finally, a solution to the migration issue is presented that saves hardware costs and administrative overhead. A prototype implementation of the migration solution is discussed.

Zusammenfassung

Diese Diplomarbeit bietet einen Überblick über die gegenwärtigen Sicherheitsmechanismen in IEEE 802.11 Netzwerken (WLANs). Weiters wird die derzeitige Situation analysiert und daraus Anforderungen an zukünftige Sicherheitserweiterungen abgeleitet.

Nach dieser Analyse der Gegenwart erfolgt eine Einführung in den IEEE 802.11i Standard, der von sich behauptet, dass er alle derzeitigen Sicherheitslöcher in der WLAN Technologie behebt. Weiters wird der bereits vorhandene und auf 802.11i basierende Sicherheitsstandard Wi-Fi Protected Access (WPA) behandelt und Unterschiede zum 802.11i Standard angeführt.

Basierend auf dieser Vorinformation wird das Problem der Migration zu diesen neuen Sicherheitsmechanismen für Internet Provider über WLAN und größere Unternehmen diskutiert.

Nach der Darstellung des Migrationsproblems wird eine Lösung dazu präsentiert, die gegenüber herkömmlichen Lösungen eine Einsparung bezüglich Hardwarekosten und administrativem Aufwand bringt. Zuletzt wird eine Prototyp-Implementierung dieser Lösung beschrieben.

Contents

List of Figures

Chapter 1

Introduction

The development of wireless LANs (WLANs) can be considered a big step forward in networking technology because of the several advantages it offers. Users are not bound to a specific place any more. WLANs reduce the cost for cabling and allow access to the network even if there is no wired connection.

However, since the first WLAN implementations became available, there has been much talk about them. Not only once has WLAN technology hit the headlines of magazines because of being inherently insecure. At first it was not realized how much having a wireless network is different from having a wired network.

One of the main differences between wireless and wired networks is that in the case of wired LANs, you have the wires inside your own building and have physical control over it. For WLANs, it is also possible to access the network from outside the building. Therefore, if no measures are taken to restrict the access to the network, anyone with WLAN equipment sitting in the restaurant just across the street is able to browse the Internet by using your Internet connection while drinking a cup of coffee!

By now, there exist mechanisms to reasonably secure WLAN but if latest WLAN equipment is used right out of the box, these mechanisms are not used. The responsibility of securing the network still lies in the user's hand and because of that many different network environments, it is not possible to change this situation.

1.1 Objectives

This thesis focuses on the current mechanisms available and future enhancements for securing the access to a wireless network. After having studied these techniques, the problem of integrating legacy devices that are not capable of the new security mechanisms into environments with new security technologies is addressed.

Integration of so called legacy devices is of special interest to service providers that cannot expect their customers to always upgrade to the latest hardware and software. Therefore, an intelligent environment for providing services to customers with devices that use state of the art security mechanisms and to customers with legacy devices is required. Supporting new devices and legacy devices in an integrated environment has to be done without compromising the overall security of the whole system.

To date, no completely satisfying solutions have been proposed for using legacy devices in an environment with strict security policies. The main goal of this thesis was to develop a solution that removes at least some of the drawbacks of existing solutions.

1.2 Thesis Organization

Chapter 2 gives an overview of IEEE 802.11 (WLAN) networks. It provides a basic introduction to the terminology, deals with the different network types and mainly discusses those aspects that are different from traditional wired networks.

Chapter 3 presents the current security situation of WLANs. Starting with an overview of computer and network security in general, currently available security mechanisms for WLAN are presented. Finally, an evaluation of the current situation is provided.

Chapter 4 mainly considers the draft version of the IEEE 802.11i standard as it claims to close all currently known security holes of 802.11 networks. Furthermore, an introduction and a comparison to Wi-Fi Protected Access (WPA) as an already available subset of the 802.11i standard is provided.

Chapter 5 provides an overview of security measures taken in network infrastructures of wireless hotspots and enterprise environments. Drawbacks of the current situation are discussed and requirements for the future are given.

Chapter 6 gives an introduction to the development environment that was used to implement the prototype of the new migration solution. The basic hardware and software structure of the access point that was used is described in this section. Some details about the existing implementation of WPA on the access point are provided.

Chapter 7 deals with the proposed, novel migration solution. Starting with the problem description and the theoretical solution, the concrete prototype implementation is described.

Chapter 8 provides an evaluation of the migration solution. This evaluation considers the theoretical aspects as well as some practical aspects specific to the prototype implementation. Furthermore, some features that could be added to the prototype in the future are discussed.

Chapter 9 summarizes and concludes this thesis.

1.3 Further Notes

For the rest of this thesis, it is assumed that the reader is already familiar with the OSI[1] 7 Layers Reference Model for Network Communication. The reader should be familiar with TCP/IP networking. Furthermore, understanding of basic network services such as DHCP, DNS or HTTP is required.

[1] OSI: Open Systems Interconnection

Chapter 2

Overview of IEEE 802.11 (WLAN)

This chapter provides the necessary basics needed for understanding the following chapters that deal with the security details of IEEE[1] 802.11 (WLAN) technology. A more thorough overview of WLAN is found in [Gas02b].

2.1 Terminology

- **Station (STA):** *"Any device that contains an IEEE 802.11 conformant medium access control (MAC) and physical layer (PHY) interface to the wireless medium (WM)."* [IEE99]

  Most of the time STAs are regarded as a computing device taking part in the wireless network and used by an end user.

- **Access Point (AP):** *"Any entity that has station functionality and provides access to the distribution services, via the wireless medium (WM) for associated stations."* [IEE99]

  One can think of an AP as a computing device acting independently as a bridge between the wired and the wireless medium. Sometimes APs are also regarded as STAs if only the part of network interaction is considered - other parts (mainly computational) are handled separately in that case.

[1] Institute of Electrical and Electronics Engineers

- **Wireless Medium (WM):** is used to transmit data between STAs or a STA and an AP.
- **Distribution System (DS):** interconnects APs for enabling a greater area of physical wireless network coverage. The APs have to exchange messages about which STA is connected to which AP to be able to deliver messages to the right STA. This inter-access point communication is what the DS is mainly for. A special version of this DS is the *Wireless Distribution System (WDS)* that uses the WM for exchanging information.

  Currently, there is no defined standard for the communication between APs and some APs therefore use proprietary protocols. Another possibility is that the APs use Ethernet as broadcast domain and forward network packets unconditionally between the wired and the wireless network. Therefore, the Ethernet can be considered to be the distribution system. Because of this situation, IEEE is working on standardizing an *Inter Access Point Protocol (IAPP)* . For details on IAPP, see [IEE03a].
- **Medium Access Control (MAC) Protocol Data Unit (MPDU):** *"The unit of data exchanged between two peer MAC entities using the services of the physical layer (PHY)."* [IEE99]
- **Medium Access Control (MAC) Service Data Unit (MSDU):** *"Information that is delivered as a unit between MAC service access points (SAPs)."* [IEE99]

  The difference between an MPDU and an MSDU might not be clear out of the definitions. An MSDU is in a layered view slightly above the MPDU. One MSDU may result in one or more MPDUs due to fragmentation.

2.2 Network Types

The *Basic Service Set (BSS)* is the basic building block of a wireless network. It consists of a group of STAs communicating with each other. Communications take place in the so called *Basic Service Area (BSA)* of the BSS. This is the area within a STA must be for being able to communicate with other STAs of the same BSS.

Two different network types can be distinguished by whether they contain an AP or not: independent networks and infrastructure networks. Both types have one thing in common, the *Service Set Identifier (SSID)*. This SSID is set by the system administrator or network administrator and is the same for all STAs within an independent or an infrastructure network. Because of this, the SSID is sometimes also called the *network name*. Note that a STA is only allowed to operate in either an independent or an infrastructure network.

2.2.1 Independent Networks

Independent Networks sometimes are also called ad hoc networks and are built up of individual STAs communicating directly with each other. A BSS of this type is called *Independent BSS (IBSS)*. Figure 2.1 shows an example of an IBSS.

The BSA of an IBSS cannot be exactly defined as two STAs may be part of the same IBSS, but cannot communicate with each other because of being out of reach.

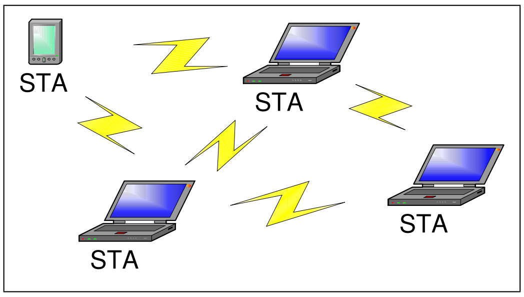


Figure 2.1: Independent BSS

2.2.2 Infrastructure Networks

Infrastructure Networks are characterized by the use of APs. The APs are used for all kinds of communication - even between STAs that are able to receive the physical signals of each other. Therefore, the sender STA has to send the message for the receiver STA to the AP, which sends it to the receiver STA. A BSS of this type is called an *Infrastructure BSS*. An infrastructure BSS is never called IBSS[2].

[2] There is no special abbreviation required for infrastructure BSSs because they are most often mentioned in the context of ESSs, which consist of several infrastructure BSSs.

The BSA of an infrastructure BSS is defined as the area within the reach of the AP. Note that an infrastructure BSS has exactly one AP. An example of an infrastructure BSS is provided in Figure 2.2.

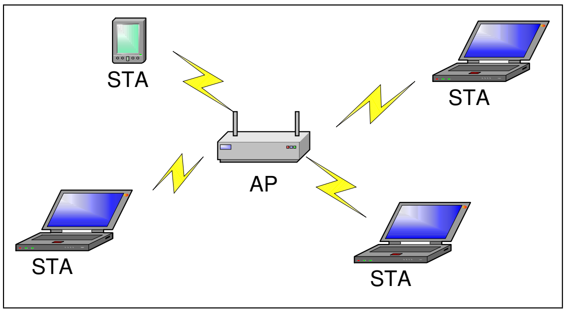


Figure 2.2: Infrastructure BSS

Extended Service Set

An *Extended Service Set (ESS)* is created by grouping several infrastructure BSSs together. This is achieved by connecting the APs of the different BSSs via the Distribution System. Figure 2.3 shows an ESS consisting of two BSSs chained together via the DS.

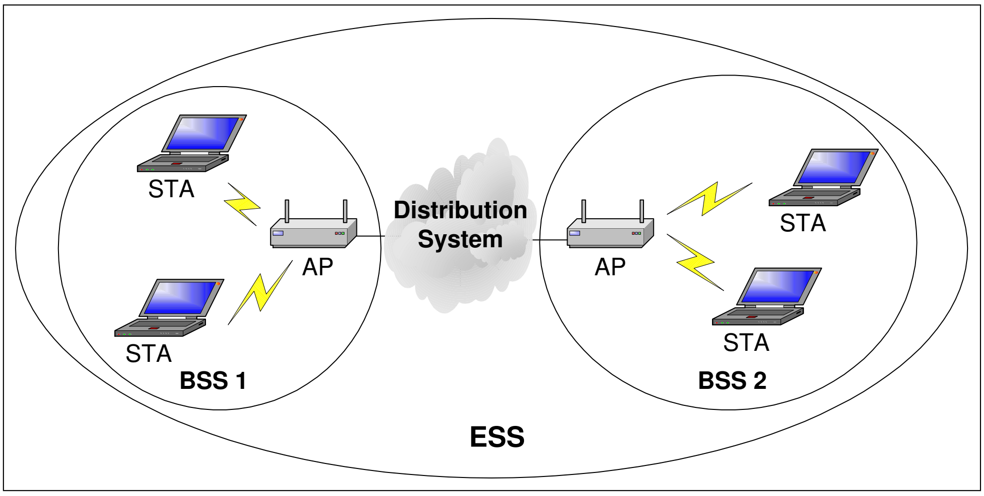


Figure 2.3: Extended Service Set

To specify which STAs belong to the ESS, all STAs of the ESS have the same SSID. APs of the ESS broadcast this SSID within some network packets to allow STAs to find the AP belonging to the specific ESS with the broadcasted SSID.

2.3 Distribution System

The main purpose of the Distribution System (DS) is to keep track of which STA is communicating through which AP and to deliver frames for STAs accordingly. This allows the provision of link layer mobility - a STA is able to move from one BSS to another BSS of the same ESS without losing the connection if the first and the second BSS overlap.

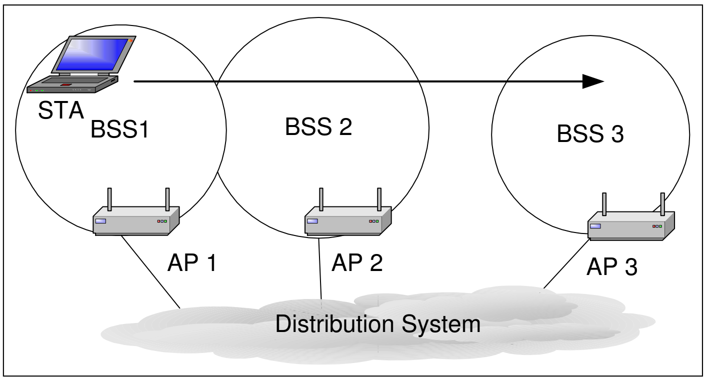


Figure 2.4: Mobility in ESS

In Figure 2.4 exist three BSSs. The service areas of BSS 1 and BSS 2 overlap while the ones of BSS 2 and BSS 3 do not. Suppose that a STA is moving out of BSS 1 into BSS 2. Because the service areas overlap, the STA simply stops communicating through AP 1 and starts communicating through AP 2 without interrupting any open connections. The DS takes care that from now on the frames for the STA are delivered through AP 2.

As the STA moves further on, it leaves the service area of AP 2. Because the service areas of BSS 2 and BSS 3 do not overlap, the STA loses connectivity for as long as it takes to get into the service area of BSS 3. After reaching the service area of BSS 3, the STA is connected to the network again. If AP 2 has buffered some frames for the STA, they can be delivered now to the STA via AP 3.

2.3.1 Mobility Types

[IEE99] identifies three types of mobility:

- **No-Transition:** This is because the STA either does not move at all or does not move out of the service area of its associated AP.

- **BSS-Transition:** As in the example above, a STA may move from one BSS to another within the same ESS.
- **ESS-Transition:** is the case that a STA moves from a BSS of one ESS to a BSS of another ESS. This is an allowed movement of a STA, but not supported in any way, especially not in the way of not interrupting open connections. The STA definitely loses all open connections.

2.4 Architectural Services

Nine services are defined in [IEE99]. They are divided into Distribution System Services (DSS) and Station Services (SS).

2.4.1 Distribution System Services

- **Distribution:** The distribution service is used whenever data is sent from or to a STA. This service takes care to deliver a frame to the right destination.
- **Integration:** This service is used if the sender or recipient of a message is a member of an integrated LAN. The integration service has to provide the functionality to enable a message to travel over the border between the DS and the integrated LAN. This might include for example media or address translation.
- **Association:** To use the DS and therefore the network resources reachable via the DS, a STA has to become associated with an AP. This is comparable to the process of registration - the STA registers at the selected AP that tells the DS that it is responsible to handle frames for the specified STA. After that, any frames to or from the STA are transferred via the selected AP. Of course, this only applies to ESSs and is not applicable for IBSSs.
- **Reassociation:** As already described in 2.3, a STA may move from one BSS to another. This also implies that the STA has to associate with the AP of the new BSS. In this case, the service is called reassociation because the STA has already been associated before. After reassociation, the DS has to update the association state for the STA to the new AP.

- **Disassociation:** Disassociation is used to terminate an existing association of a STA. After disassociation, any attempt of the STA to send data to the network or of any other STA to send data to the disassociated STA is unsuccessful.

2.4.2 Station Services

- **Authentication:** Before a STA is allowed to communicate with another STA, it has to authenticate itself to the other STA. This applies to IBSSs as well as to ESSs. In an ESS, a STA has furthermore to become associated with an AP after authentication before being allowed to use the network.

  [IEE99] defines two different types of authentication:

  - *Open System Authentication:* If a STA uses Open System Authentication, any other STA may become authenticated at this STA. This is like having no authentication at all and is the default authentication algorithm.
  - *Shared Key Authentication:* Shared Key Authentication relies on WEP (Section 3.2) and a shared secret key between the two STAs that wish to communicate for performing authentication. This kind of authentication is described in more detail in 3.2.2.

  Preauthentication: If STAs move from one BSS to another BSS in the same ESS, they have to authenticate at the AP of the second BSS before being allowed to associate with this AP. Because authentication may take some time (depending on the used authentication algorithm) and some applications might require the time interval needed for switching from one BSS to another being as short as possible, 802.11 supports a mechanism called *Preauthentication*.

  Preauthentication may be done by a STA that is already associated with an AP but wants to associate with another AP in the near future. Therefore, the STA authenticates at the AP with which it wants to associate while still being associated with its current AP. After that it disassociates at the old AP and associates with the new AP.

- **Deauthentication:** Deauthentication terminates an authenticated relationship between two STAs. In the case of an ESS, authentication is a prerequisite for association. Therefore, deauthentication also terminates an existing association between a STA and an AP. See Section 2.5.2 for further information.

- **Confidentiality[3]:**

  In traditional wired LANs, physical access to the network is relatively easy to control because the wires are in the own building and controlling the wires means controlling access to them. Even after having physical access, sniffing[4] of LAN traffic is not too easy because most network installations make use of switched LANs nowadays. Nevertheless, sniffing of switched LANs is still possible.

  WLANs are different in two ways: First, you cannot control the wireless medium and second you cannot apply switches. Therefore, sniffing WLAN traffic is rather easy. Because of these reasons, 802.11 (1999) defines a mechanism called *Wired Equivalent Privacy (WEP)* that provides encryption to messages that are transmitted over the WM. The goal of WEP was to provide the same level of confidentiality as in traditional wired networks. WEP is handled in more detail more in Section 3.2.

- **MSDU Delivery:** MSDU Delivery is the service responsible for transferring MSDUs (MAC Service Data Units) between MAC sublayer entities.

2.5 Management Operations

2.5.1 Scanning

Before a STA is able to use a network, it has to find it. The process of searching for a network is called *scanning.* A STA has two possibilities for finding the right network: *passive scanning* and *active scanning.*

[3] The 1999 edition of the 802.11 standard defines Privacy instead of Confidentiality. Latest draft versions of 802.11i are using the term Confidentiality. This is the term used in this thesis.

[4] Sniffing is the process of monitoring and analyzing network traffic.

Passive Scanning

When scanning in passive mode, a STA listens for incoming beacons. Beacons are management frames that contain all the necessary information a STA needs to synchronize with the network that generated the beacon. Furthermore, the SSID is contained in the beacon to enable a STA to identify the network and to check whether the SSID matches the desired SSID of the STA.

Beacons are transmitted repeatedly whenever a configured time interval has elapsed. In an ESS, the APs are responsible for transmitting beacons - in an IBSS, the transmission of beacons is distributed over the participating STAs.

Active Scanning

To perform an active scan, a STA sends a probe request including the SSID of the desired network or a broadcast SSID to find all available networks. In an ESS, an AP that receives a probe request answers with a probe response - in an IBSS the STA that transmitted the last beacon is responsible for transmission of the probe response.

Sending a probe response is only allowed if the SSID in the received probe request matches the SSID of the recipient - or the received SSID is a broadcast SSID that queries for all available networks. The probe response now contains the necessary information for the STA to synchronize with the corresponding network.

2.5.2 Authentication and Association

Authentication and association follows the scanning process of a STA, but a previous scanning process need not be the case. Each STA has to maintain two state variables for each STA with which direct communication is desired:

- *Authentication State:* authorized or unauthorized
- *Association State:* associated or unassociated

These state variables are changed due to successful (De)Authentication and (Dis)Association messages in the way shown in Figure 2.5[5] whereas unauthenticated, unassociated is the initial state.

[5] The diagram uses the UML statechart syntax as defined in [OMG03].

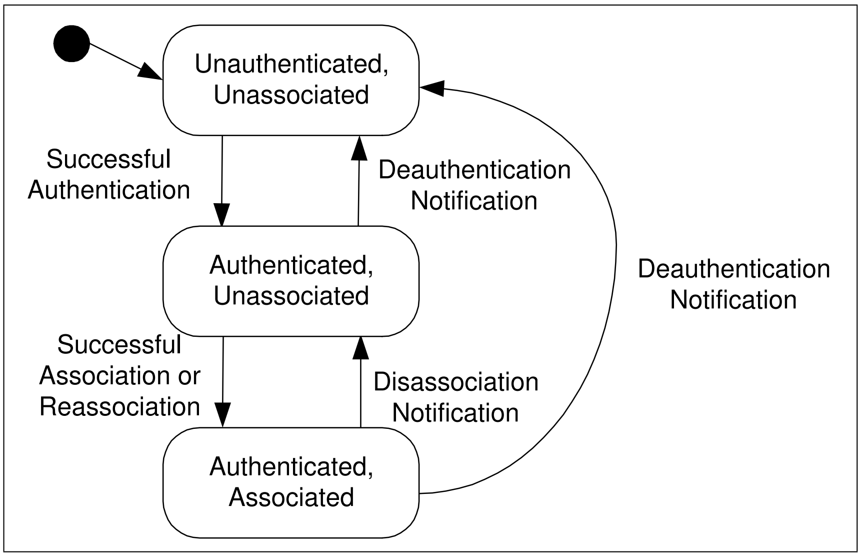

Unauthenticated, Unassociated
Successful Authentication
Deauthentication Notification
Authenticated, Unassociated
Deauthentication Notification
Successful Association or Reassociation
Disassociation Notification
Authenticated, Associated

Figure 2.5: Authentication and Association; details [IEE99], Section 5.5

After successful authentication, a STA becomes authenticated, but still unassociated. This is enough for IBSS networks where STAs communicate directly with each other. In ESSs, successful (re)association with an AP is necessary. Thereafter, the STA is authenticated and associated and able to use the DS and therefore has a working network connection.

Being authenticated and associated, the state of a STA changes to authenticated, unassociated by successful disassociation. Regardless of the current state successful deauthentication always results in the unauthenticated, unassociated state as authentication is a requirement for association.

2.6 MAC Addressing

This section contains the information needed to understand the basics of the wireless network addressing scheme that is slightly different from the one used in traditional wired networks.

2.6.1 MAC Addresses

MAC addresses are contained in the header fields of a network packet. Each address field contains a 48 bit universal LAN MAC address as defined in Section 5.2 of [IEE90].

- **Destination Address (DA):** The destination address contains the address of an individual MAC entity or a group address for a group of MAC entities as the final recipient(s) of the MAC frame.

- **Source Address (SA):** This is the address of the original sender, i. e. the MAC entity that produced the frame.

- **Receiver Address (RA):** The receiver address contains an individual or a group address of the direct receiving STA(s) on the WM. This address is not necessarily the same address as the DA, but might be the same if the final recipient is a wireless station. The RA can be seen to specify an intermediate receiver.

- **Transmitter Address (TA):** This address is used to specify the STA that has transmitted the frame onto the wireless medium.

- **Basic Service Set ID (BSSID):** The BSSID is a 48 bit field like a MAC address and is used to distinguish between different BSSs. In an independent BSS the BSSID is specified as a locally administered IEEE MAC address produced by a random number generator. Locally administered MAC addresses are defined in Section 5.2 of [IEE90] and have the universal/local bit set to 1. Furthermore, the individual/group bit of the BSSID has to be set to 0 indicating an individual address.

Note that the addresses do not have a fixed place in the MAC header of a network packet. The MAC header contains four address fields, address 1 to address 4, whereas the address fields might be omitted if they are not needed.

The presence and meaning of a specific address field depends on the type of frame being transmitted. 802.11 distinguishes between three types of frames: *Control Frames*, *Data Frames* and *Management Frames*.

Further information on the MAC header and MAC frame formats is found in [IEE99], Section 7.

2.6.2 To DS / From DS

The *To DS* and *From DS* bits specify whether a frame is going to or coming from the distribution system. Interpretation of all their combinations is shown in table 2.1.

To DS	**From DS**	**Meaning**
0	0	Frames for direct communication between STAs: Data frames in an IBSS and control and management frames.
0	1	Frames leaving the DS
1	0	Frames entering the DS
1	1	Frames inside a wireless distribution system.

Table 2.1: Interpretation of To DS and From DS

Chapter 3

Current Security Situation

This chapter deals with solutions currently available for securing a wireless network. These solutions have been added one by one after the realization of the necessity to additionally secure wireless networks compared to traditional wired networks. After a short introduction to security and the terminology, the different security mechanisms are described.

3.1 Introduction & Terminology

3.1.1 Security Definition

First of all it is necessary to clarify what the term security means and how it can be defined. A common approach is to define security over the three requirements confidentiality, integrity and availability.

- **Confidentiality[1]:** means that resources are only accessible by authorized entities.
- **Integrity:** requires that information can only be modified by authorized entities. Modification does not only mean changing but also creation or deletion of resources.
- **Availability:** is the property of a resource to be accessible whenever an authorized entity needs access to it.

[1] The term Secrecy is sometimes used instead of Confidentiality, see [Sta98]. [IEE99] uses the term Privacy for this purpose.

3.1.2 Attack Methods

Because this thesis is about WLAN technology, the description of attacks focuses mainly on the network level. [Sta98] classifies attacks to network security into two different categories, *passive attacks* and *active attacks*. This approach is also used here.

Passive Attacks

Passive attacks are characterized by the fact that they do not produce any additional network traffic and are therefore extremely difficult to detect - if ever. They are used to gather some information about a network and the transmitted data on it and are normally performed before any kind of active attack.

- **Traffic Monitoring:** is used to obtain statistics about network traffic and to explore network paths.
- **Sniffing:** monitoring and analyzing of network traffic.
- **Eavesdropping:** obtaining copies of messages without permission.

Active Attacks

Active attacks inject additional network traffic and therefore can be detected much easier than passive attacks.

- **Masquerading:** using the identity of another entity for taking part in the network, i.e. an entity pretends to be another entity when sending or receiving messages.

  Spoofing is another term for this kind of attack. It is often used when speaking of attacks at a specific protocol level, e.g. IP Spoofing or TCP Spoofing.
- **Message Tampering:** modification of captured messages. This is often used in combination with a replay attack or in so called *Man-In-the-Middle (MIM)* attacks.[2]

[2] Man in the middle attacks are used to intercept messages from the sender to the recipient, to interpret and may be modify them before sending the possibly altered message to the recipient. Messages from the sender to the receiver might also be dropped and other messages might be sent instead of the original ones.

- **Replaying:** resending of captured network messages.
- **Denial of Service (DoS):** disabling access to a resource. This can be done by overloading a system or trying to bring it otherwise into a non-functional or non-responding state.
- **Exploiting:** using bugs or weaknesses in a system.

Most of these attacks are also defined in [CDK01, Sta98].

3.1.3 The Problem of being wireless

The problem of WLANs considered so far was the one not having control over the distribution medium. While the wires of a traditional LAN are inside the own building and can be controlled the radio waves of WLAN cannot be controlled or kept inside the building.

Passive attacks are therefore much easier to perform compared to wired LANs. The attacker only needs to be within the reach of a WLAN transmitter to capture and possibly analyze the traffic sent by the transmitter.

The situation is similar for active attacks. Here, the victim STA has to be within the reach of the transmitting attacker STA. This is enough for the attacker to launch an attack against the victim. If the victim STA is an AP without any applied security mechanisms, the security situation is even worse because this compromises the wired network too. In this case, the attacker could attack a device on the wired network via the insecure AP without the need to plug into the wired network. Obviously, there is a special need to secure the interface between the wireless and the wired network, i.e. the APs.

These properties mainly addressed violations of the requirements confidentiality and integrity, but there are possible attacks to availability as well. [Fri03] discusses the limitation of the bandwidth of WLAN. Considering this limitation and the discussed reasons, it is much easier to perform a DoS attack on WLAN by consuming bandwidth and processing power. It is also thinkable to use a jammer to disturb WLAN signals and therefore attack the availability of WLAN, but these kind of attacks to availability are not addressed within this thesis.

3.1.4 Cryptographic Terms

- **Plaintext:** A sequence of bytes (e.g. characters) that is readable and interpretable by a human or a computer.

- **Cleartext:** a synonym for Plaintext

- **Encryption:** is the process where an algorithm is used to scramble the plaintext. The algorithm may depend on a key to perform the encryption. The output of the process is non-readable and non-interpretable, neither by human nor by computer, without having knowledge about the algorithm and the appropriate key.

- **Ciphertext:** the output of the encryption process.

- **Decryption:** is the process where an algorithm is used to recover the plaintext out of the ciphertext. The algorithm may depend on a key to perform the decryption.

- **Shared (Secret) Key:** is shared by the sender and the recipient and used for encryption and decryption.

- **Public Key:** is used at the sender for the encryption of a message.

- **Private Key:** is used at the receiver for the decryption of a message. Because it is a secret of the authorized receiver, it is also called *Secret Key*.

- **Asymmetric Cryptography:** is also called *Public Key Cryptography* and states that different keys are used for encryption and decryption. The idea is that the key used for encryption is open to the public, but the key for decryption is a secret of the owner. A further requirement is that the secret key cannot be derived from the public key.

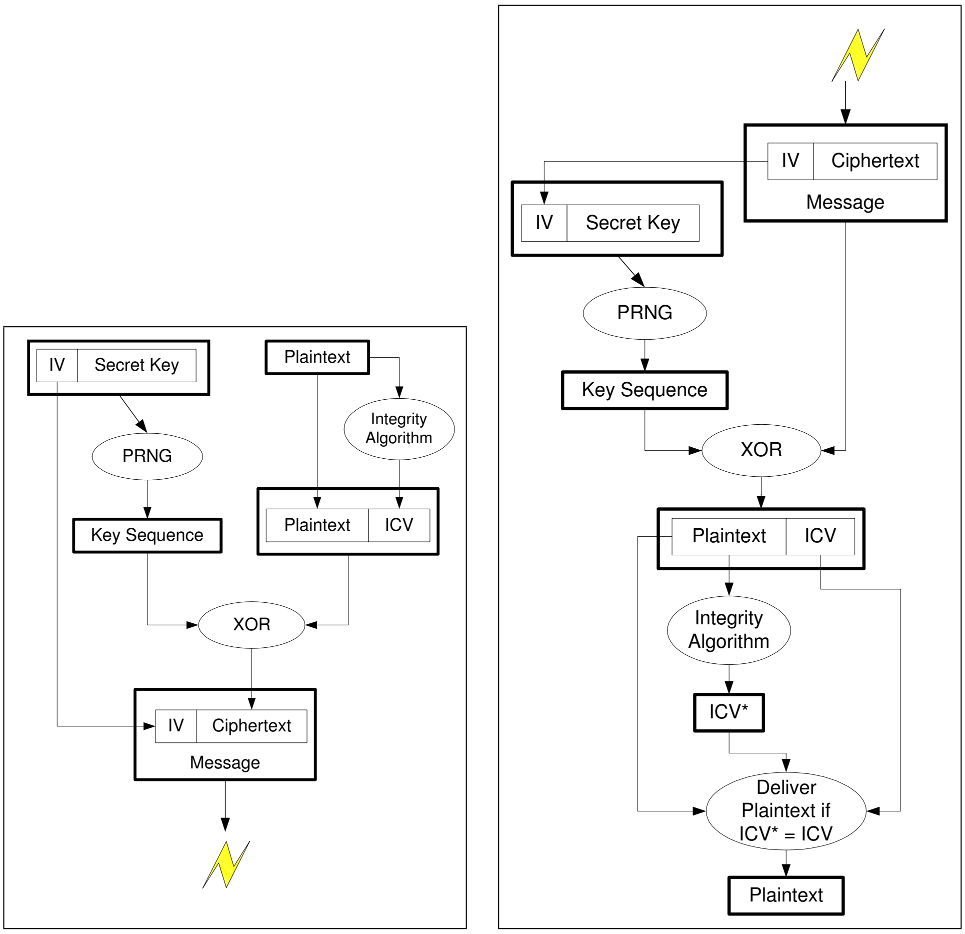
- **Symmetric Cryptography:** states that the same key is used for encryption and decryption. Therefore, the key has to be a secret to the sender and the receiver and must not be open to the public.

- **Credentials:** are attributes of an entity that are used to verify the identity of the corresponding entity. Nowadays, often username and password are used as credentials for that purpose.

3.2 Wired Equivalent Privacy (WEP)

The first security mechanism introduced to WLANs was WEP. WEP stands for *Wired Equivalent Privacy* and as the name already reveals, it was an attempt to make WLANs as secure as traditional wired networks. It was defined in [IEE99] after the initial specification in 1997 had no security mechanisms at all.

3.2.1 The Algorithm



(a) Encryption

(b) Decryption

Figure 3.1: WEP Algorithm

WEP Encryption

WEP relies on a shared secret key of 40 bits length, that is used by the sender for encryption and at the receiver for decryption. The key is appended to a generated 24 bit *Initialization Vector (IV)* to produce the 64 bit seed for the RC4 *Pseudo Random Number Generator (PRNG).*

The PRNG produces a key sequence of same length as the data to be sent plus 4 octets. These additional 4 octets are for the *Integrity Check Vector (ICV)* that is the result of the integrity algorithm used to verify that the message has not been modified during transmission. The key sequence is then XORed with data + ICV to produce the ciphertext. WEP encryption is shown in Figure 3.1(a).

WEP Decryption

As only the 40 bits of the key are shared by sender and receiver but the 24 bits of the IV are chosen by the sender, they have to be included in the header of the message to enable the decryption of the packet by the receiver.

After receiving a message, the receiver extracts the IV out of the header, prepends it to the shared key and uses this 64 bits for the PRNG to produce the key sequence. The key sequence is XORed with the ciphertext to get the plaintext. Afterwards, the ICV is generated for the data of the received message and compared to the ICV contained in the message. If they are equal, the received data has not been modified during transmission and can be delivered. The decryption process is shown in Figure 3.1(b).

3.2.2 Shared Key Authentication

This kind of authentication allows STAs to be authenticated by the fact that they know a shared secret key. For the description of the necessary sequence of messages the STA requesting authentication is called *requester*, the STA authenticating the requester is called *responder*.

1. At first, the requester sends a shared key authentication request to the responder.

2. The responder answers with a message containing *successful* or *not successful* depending on whether it allows shared key authentication. In case of *not successful*, authentication ends with an unauthenticated

state for the requester and no further messages are exchanged - in case of *successful*, the responder supplies a challenge text to the requester.

3. As response to the message with the challenge text, the requester sends an encrypted message (by using the secret key) containing the challenge text to the responder.

4. If the responder is able to decrypt the message and the decrypted message contains the challenge text, this is the proof that the requester knows the secret key and should be authenticated. Finally, the responder replies with the result of the authentication process by sending an authentication message with state *successful* or *unsuccessful* to the client.

3.2.3 Problems of WEP

Several problems of WEP have already been identified. They include, but are not limited to the following:

- The standard does not define how to distribute the secret keys, it only states: *"Data confidentiality depends on an external key management service to distribute data enciphering/deciphering keys"*, [IEE99]. The problem now is that key distribution is often done manually, which is unacceptable for large enterprise scale systems.

- Because of the lack of an integrated key distribution system, keys are not changed as often as would be required to avoid using the same IVs twice, which makes attacks on WEP much easier.

- A serious problem with WEP is that it can be cracked automatically. [SIR01] describes such a successful attack and also two tools that are already available to the public: AirSnort[3] and WEPCrack[4]

- Shared Key Authentication reveals the key sequence used for encryption of the challenge text. This is because the attacker can capture the message from the responder to the requester containing the plaintext challenge text and also the encrypted message from the requester to the responder.

[3] The homepage of the AirSnort project can be found at: `http://airsnort.shmoo.com/` and `http://airsnort.sourceforge.net/`

[4] WEPCrack can be found at: `http://wepcrack.sourceforge.net/`

The used key sequence for encryption of the message can be derived by simply XORing the plaintext message with the encrypted message. This issue is already considered in [IEE99] and it is suggested not to use the key/IV pair for subsequent messages. Nevertheless, sensitive information is revealed by this authentication mechanism.

The problems listed above seem to be the ones that are most understandable without having deeper knowledge of cryptography and the WEP algorithm. A more complete summary is given in [Gas02b].

Further Reading

- [BGW] Nikita Borisov, Ian Goldberg, and David Wagner. *Security of the WEP Algorithm.* `http://www.isaac.cs.berkeley.edu/isaac/wep-faq.html`
- [FMS] Scott Fluhrer, Itsik Mantin, and Adi Shamir. *Weaknesses in the Key Scheduling Algorithm of RC4.* `http://www.crypto.com/papers/others/rc4_ksaproc.ps`
- [SIR01] Adam Stubblefield, John Ioannidis, and Aviel D. Rubin. *Using the Fluhrer, Mantin, and Shamir Attack to break WEP*, AT&T Labs Technical Report TD-4ZCPZZ. `http://www.cs.rice.edu/~astubble/wep/wep_attack.pdf`
- [Wal] Jesse Walker. *Unsafe at any Key Size; An Analysis of the WEP Encapsulation.* `http://grouper.ieee.org/groups/802/11/Documents/DocumentHolder/0-362.zip`

3.3 Extensible Authentication Protocol

The *Extensible Authentication Protocol (EAP)* was originally defined 1998 in RFC 2284 [BV98] as an extension to the authentication mechanisms of the *Point-to-Point Protocol (PPP)*. It does not define an authentication mechanism itself but a carrier protocol for authentication mechanisms to carry the credentials. It is contained here because 802.1X (see 3.4) makes use of EAP.

EAP was designed for PPP and requires the existence of a point-to-point link between the two parts performing the authentication. This might sound as being a problem for WLANs, but it is not. As mentioned in the WLAN overview chapter, a STA has to associate with an AP before any type of data is accepted by the AP. This association between the STA and an AP creates the necessary dedicated link for EAP between a STA and an AP.

3.3.1 Terminology

- **Authenticator:** *"The end of the link requiring the authentication. The authenticator specifies the authentication protocol to be used in the Configure-Request during Link Establishment phase"*, [BV98].
- **Peer:** *"The other end of the point-to-point link; the end which is being authenticated by the authenticator"*, [BV98].

3.3.2 Protocol Overview

The EAP protocol has four types of messages: EAP-Request, EAP-Response, EAP-Success and EAP-Failure. The protocol works as follows:

1. The authenticator sends one or more requests to authenticate the peer. A request contains a type field to indicate the algorithm being requested, e.g. MD5, One-Time-Password, ...
2. The peer replies to each request with a response containing the required information and a type field corresponding to the type field of the request. The type field of the response may only differ from the request to tell the authenticator that the requested authentication mechanism is not supported. In that case, a different authentication mechanism can be suggested by the peer.
3. The authenticator ends the authentication process with a success or failure packet.

An example EAP authentication sequence is given in Figure 3.2. In this example, the authenticator requests an appropriate MD5 hash value to verify the identity of the peer.

Various different authentication mechanisms are used in conjunction with EAP. Examples for authentication mechanisms are:

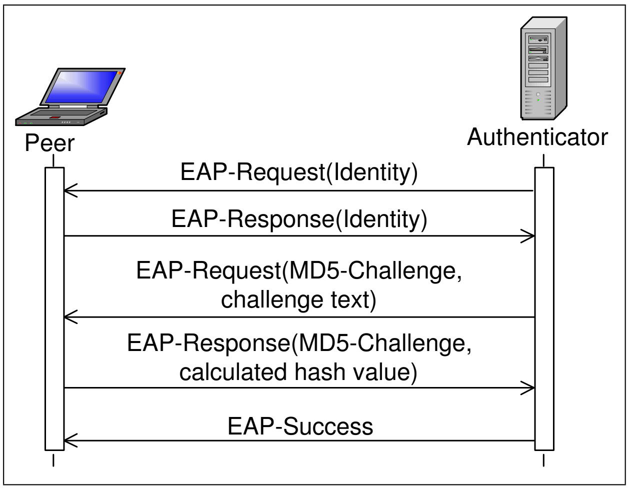

Peer
Authenticator
EAP-Request(Identity)
EAP-Response(Identity)
EAP-Request(MD5-Challenge, challenge text)
EAP-Response(MD5-Challenge, calculated hash value)
EAP-Success

Figure 3.2: EAP Example for EAP-MD5

- **MD5-Challenge:** This is the only authentication mechanism required in an EAP implementation. For implementation of this mechanism [BV98] refers to the specification of the PPP CHAP[5] Protocol in RFC 1994 [Sim96] that can be summarized to the following steps:

  1. The authenticator sends a challenge message containing an identifier and a challenge text to the peer.
  2. The peer responds with a value calculated using the identifier, the challenge text and a secret as input to a one-way hash function.
  3. The authenticator compares the response with its own calculation of the expected hash value. If the values match, the authentication is successful.

- **One Time Password (OTP):** makes use of a secret user passphrase and a challenge text of the authenticator as input to a secure hash function. The hash function is performed N times on this input by the peer to generate a first OTP. This value is stored on the authenticator.

  To generate the next OTP the peer performs the hash function only N-1 times on the input. This is because the authenticator knows the last used OTP, performs the hash function one time on the received OTP and compares it to the stored OTP. If the received and the calculated OTP match, the received OTP is correct and the peer authenticated

[5] CHAP: Challenge-Handshake Authentication Protocol

successfully. Therefore, this new value is stored for further authentication by the authenticator.

Note that OTP needs to be reinitialized from time to time due to the limited times the hash function can be performed on the same input. OTP is defined in RFC 1938 [HM96] as a mechanism to prevent eavesdropping / replay attacks.

- **Generic Token Card:** the authentication mechanism for use with various token card implementations that require user input. Typically, the request and the response contain ASCII text whereas the text in the request contains a message to be displayed to the user and the text for the response is read by the user from a token card device and entered manually.
- **Transport Layer Security (TLS):** defined in RFC 2716 [AS99] and makes use of the TLS Protocol defined in RFC 2246 [DA99]. TLS is based on SSL 3.0[6] and uses asymmetric cryptography for authentication and establishment of a master key for encrypting communication of the TLS session.
- **SIM:** makes use of the GSM *Subscriber Identity Module (SIM)* as used in for example mobile phones. In this authentication mechanism, the peer receives one or more random numbers as challenge from the authenticator and uses this challenge as input to an authentication algorithm that is run on the SIM. Furthermore, a secret key stored in the SIM is used by the algorithm. The output of the algorithm is sent to the authenticator that verifies the output and responds with an EAP-Success or EAP-Failure Message. EAP-SIM is currently an Internet Draft [HS03].

EAP was originally designed for PPP, but now it is also used in IEEE 802 networks (LANs). Therefore, the EAP specification is currently undergoing a redesign phase for not only fitting PPP but IEEE 802 networks too [BVA$^+$03]. The network layers around EAP can now be drawn as shown in Figure 3.3.

Information on EAP-TTLS and PEAP, which are shown in the Authentication Layer of Figure 3.3, can be found in Sections 3.3.4 and 3.3.5. EAPOL stands for EAP over LAN and is defined in [IEE01]. Some information on EAPOL is provided in 3.4.3.

[6] Secure Socket Layer (SSL) 3.0 Protocol Specification was published by Netscape.

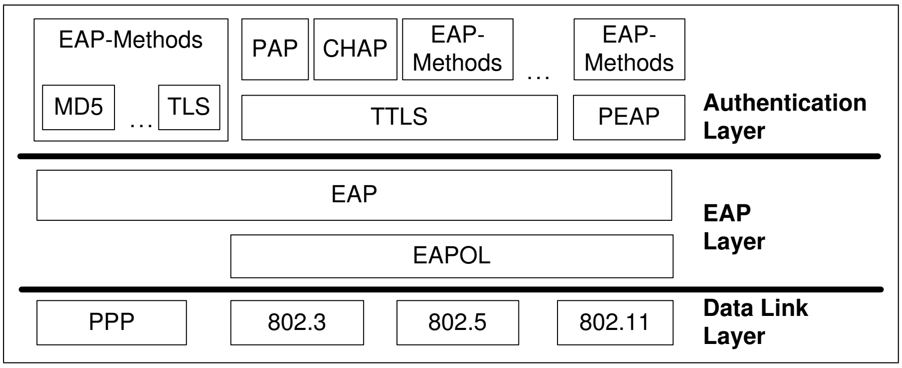


Figure 3.3: EAP Stack

3.3.3 EAP Deficiencies

EAP was designed with a wired world in mind. This imposes some security issues when applying this authentication protocol to a wireless world. The main problem is that EAP provides no encryption by itself and therefore allows an attacker to discover sensitive data such as user IDs or challenge texts and corresponding hashed responses. Eavesdropping of this information is especially easy in a wireless networks and allows the use of brute force[7] or dictionary[8] attacks against authentication mechanisms.

Furthermore, EAP does not provide a per packet protection of the message integrity. This allows an attacker to change the content of the message without the message tampering being detected.

To get a secure authentication mechanism, it is absolutely necessary that the authentication mechanism used in conjunction with EAP is itself secure. An example of such a secure authentication mechanism is EAP-TLS.

Not a security issue but still a drawback of EAP is that it does not support fragmentation. It is up to the individual EAP methods to support fragmentation if required.

Another concern often mentioned when speaking of EAP is that EAP does not provide a service for key exchange. Such a service would support cryptographic algorithms depending on a key and requiring the key to be changed from time to time. An example for such an algorithm is WEP.

[7] **Brute Force Attack:** tries to discover a secret by trying every possible value.

[8] **Dictionary Attack:** similar to a brute force attack, but only tries the values contained in a dictionary, e.g. one that contains all words of the English language

Because of the security issues of EAP and the unavailability of better authentication mechanisms, LEAP, EAP-TTLS and PEAP were developed as improvements of EAP. LEAP is a proprietary solution of Cisco and is not handled here. An overview of EAP-TTLS and PEAP is given in Sections 3.3.4 and 3.3.5.

EAP-TTLS and PEAP are both Internet Drafts of the IETF[9]. Both serve the same purpose, but originate from different companies. While EAP-TTLS was defined before PEAP and, according to [Gas02a], is currently more widely implemented, PEAP was defined by Microsoft, is now implemented in Windows XP and gets the support of Cisco. A technical comparison of the two technologies can be found in [Gas02a].

3.3.4 EAP Tunneled TLS (EAP-TTLS)

EAP-TTLS [FBW02] is an extension of EAP-TLS and allows the use of legacy authentication mechanisms over a secure connection. The advantage now is that an existing authentication infrastructure can be used and authentication of the peer can be done by using nearly any protocol.

For an easy integration into the existing authentication infrastructure, a TTLS server can be used for the translation between EAP-TTLS packets and the legacy authentication mechanism used by the authentication server. This creates a secure tunnel between the peer and the TTLS server. Of course the communication link between the EAP-TTLS server and the authentication server has itself to be secure, e.g. because of having physical control over the wire. Such an authentication environment is shown in Figure 3.4.

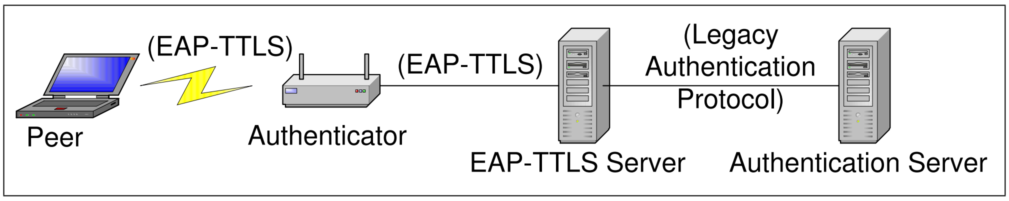


Figure 3.4: Using legacy authentication with EAP-TTLS

To accomplish its task, EAP-TTLS works in two phases. Phase one uses TLS to authenticate the authentication server to the client and optionally the client to the authentication server. Furthermore, a cipher suite is negotiated and activated.

[9] Internet Engineering Task Force

Phase two uses the TLS record layer to exchange authentication information in form of *Attribute-Value Pairs (AVPs)* that are compatible with RADIUS[10] or Diameter[11]. Encryption, decryption, and data transfer is handled by the TLS record layer. The authentication itself may be EAP or a legacy protocol such as PAP, CHAP, MS-CHAP or MS-CHAP-V2. It might not be necessary to perform user authentication in phase two, if the peer has already been authenticated in phase one by use of the mutual authentication option of the TLS handshake protocol.

3.3.5 Protected EAP (PEAP)

Protected EAP [PSZJ03] is similar to EAP-TTLS. It also makes use of two phases where in phase one a TLS session is established and in phase two EAP authentication is performed. The major difference now is that with PEAP, only EAP authentication is allowed in phase two, but no other authentication mechanisms.

PEAP also requires mutual authentication between the peer and the authentication server, but as with EAP-TTLS, client authentication is not required as part of TLS session establishment and might be done in phase two.

3.4 IEEE 802.1X Port-Based Network Access Control

It is often the case that IEEE 802 LANs are deployed in professional or enterprise environments where unauthorized devices may be physically connected to the LAN. An example of such an environment is a hotspot provider offering Internet connectivity through a WLAN access point. In such cases, it is desirable to limit the access to the LAN and the network resources.

802.1X [IEE01] defines an authentication framework for such environments. It makes use of the following terms:

- **Supplicant**[12]: An entity requesting authentication.
- **Authenticator:** An entity authenticating a supplicant.

[10] **RADIUS:** Remote Authentication Dial In User Service; IETF RFC 2865

[11] **Diameter:** intended as replacement for RADIUS; it is currently an IETF Draft

[12] 802.1X uses *supplicant* instead of another common term *peer* that is used for example in EAP.

- **Authentication Server (AS):** An entity providing the authentication service to the supplicant. It decides, based on the credentials of the supplicant, whether authentication of the supplicant is successful. The AS may reside within the Authenticator or in a different device.
- **Network Access Port:** *"A point of attachment of a system to a LAN. It can be a physical port, for example, a single LAN MAC attached to a physical LAN segment, or a logical port, for example, an IEEE 802.11 association between a station and an access point"*, [IEE01].

  The 802.1X standard uses the term *port* as an abbreviation for network access port.

Figure 3.5 gives a graphical illustration of the roles in 802.1X (supplicant, authenticator and authentication server), where the authenticator and the authentication server reside in different devices.

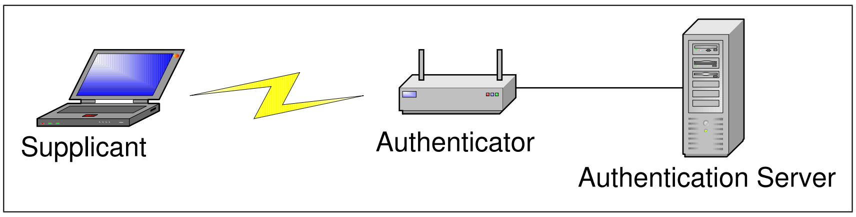


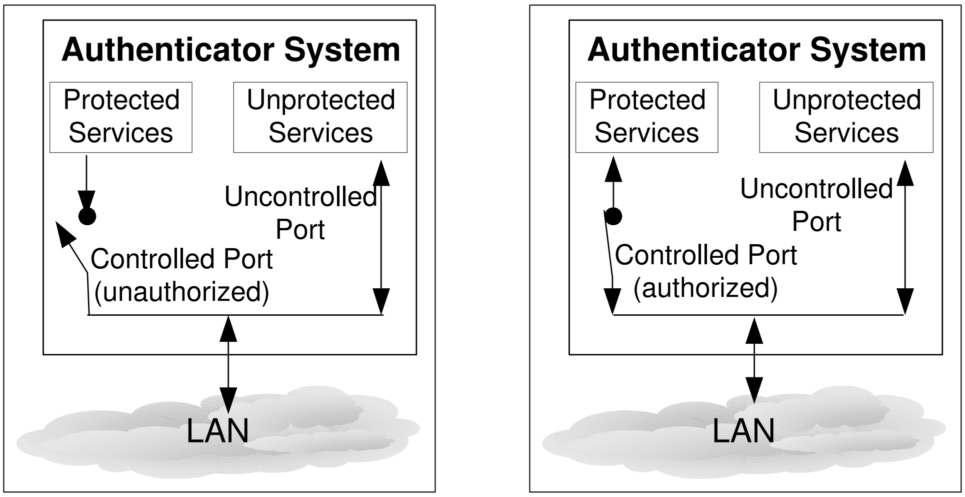
Figure 3.5: Roles of 802.1X as implemented in WLAN

3.4.1 Controlled - Uncontrolled Port

The authentication framework defines the mechanism of a controlled and an uncontrolled port. Both are available at a single point of attachment (physical / logical port) of the authenticator to the LAN. Data received from the LAN is made available on both ports. Data on the controlled port is only allowed to pass through if the port is in authorized state, on the uncontrolled port, data is always allowed to pass through unconditionally.

For an easier understanding, the controlled port can be viewed as a switch that is on if it is in authorized state and off otherwise. Protected services and resources are behind the controlled port. A graphical representation of controlled and uncontrolled port is shown in Figure 3.6.

In 3.6(a), the controlled port is in the unauthorized state. Therefore, traffic from the protected services to the LAN and traffic from the LAN to the



(a) Controlled Port unauthorized

(b) Controlled Port authorized

Figure 3.6: Controlled and Uncontrolled Port; details [IEE01], Section 6.3

protected services ends at the controlled port and is not allowed to pass through.

The idea now is to use the uncontrolled port for authentication and hence to set the state of the controlled port to authorized or unauthorized based on successful or unsuccessful authentication.

Figure 3.6(b) shows the controlled port after the corresponding supplicant has been successfully authenticated by using the uncontrolled port and accessing an unprotected service (the authentication service). Therefore, the controlled port is now in authenticated state (the switch is closed).

3.4.2 Environment Example

Figure 3.7 gives an example of the use of 802.1X. The controlled port of the authenticator system is used to restrict access to the Internet. The state of the controlled port initializes to unauthorized for disallowing access to the Internet.

If the wireless supplicant wants to access the Internet, it has to authenticate at the authentication server first. After successful authentication the authenticator sets the state of the controlled port to authorized and allows the supplicant to access the Internet.

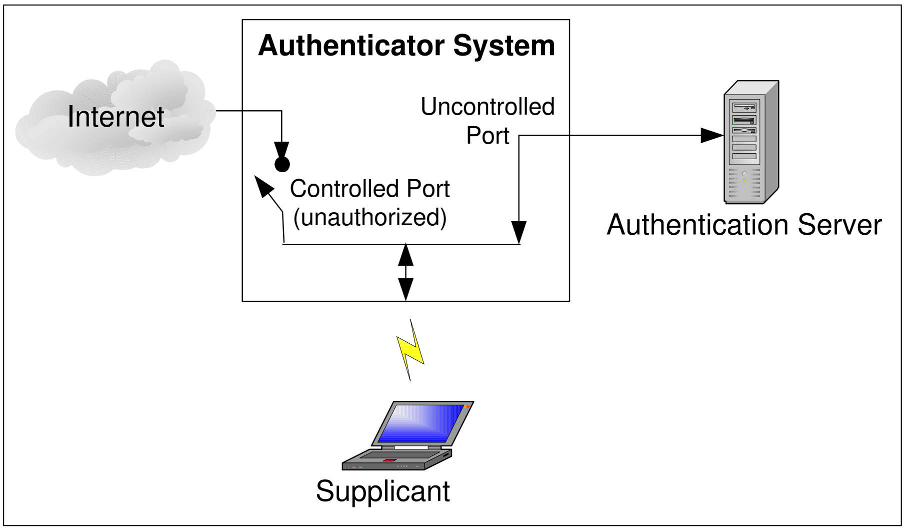


Figure 3.7: 802.1X Environment Example

3.4.3 Authentication Protocol

For an authentication framework it is also necessary to have an authentication protocol. 802.1X does not introduce a new protocol but makes use of the *Extensible Authentication Protocol (EAP)* defined in RFC 2284 [BV98] (see 3.3). To be able to use EAP on LANs, 802.1X defines an extension called *EAP over LAN (EAPOL)* that describes how to encapsulate EAP packets in 802.3/Ethernet MACs and Token Ring/FDDI MACs.

3.4.4 Support for Key Management

Key management has always been a problem so far. 802.1X provides a solution to this by specifying an EAPOL-Key message to support the transmission of global key information. This support is optional and key information can be transmitted from the authenticator to the supplicant or from the supplicant to the authenticator.

If key transmission is enabled, it normally takes place after successful authentication and whenever a new key becomes available. However, the standard does not define when and for what reason key information is transmitted. Such information is defined for example in the 802.11i standard [IEE03b], see Chapter 4.

3.4.5 Security Analysis

802.1X surely represents an improvement to the security situation of WLANs. It enables access control to protected resources on a per user or per device basis by making use of higher layer authentication mechanisms. This also allows the use of an already existing authentication infrastructure.

Furthermore, 802.1X can strengthen WEP because of dynamic key management, that allows the WEP keys to be changed more often. This also provides a solution to the key distribution problem and makes WLAN more applicable in large scale enterprises.

Using the controlled port for access restriction to protected services has also one advantage: The DHCP service can be specified to be a protected service only accessible via the controlled port. This allows DHCP traffic only to pass through the controlled port after successful EAP authentication. The advantage is now that up to this time the client does not have an IP address.

This technique prevents an attacker from getting an IP address because of using a legitimate MAC address of the company (or the company does not apply MAC restrictions at all!!).

Besides these advantages, [MA02] identifies two major design flaws in 802.1X:

- The absence of mutual authentication that allows a MIM attack to be performed.
- The possibility to successfully perform session hijacking.

Furthermore, the appendix of [MA02] lists several DoS attacks possible to perform against a specific STA or the whole network. These attacks include the spoofing of EAPOL-Logoff, EAPOL-Start, EAP-Failure or 802.11 management frames.

The described attacks are not possible with all authentication methods. EAP-TLS for example provides mutual authentication. This prevents the MIM attack. Session hijacking is only possible if encryption is not enforced by the authenticator or the encryption key is known by the attacker that hijacks the session of the original supplicant.

Besides this, [MA02] describes a MIM attack independent of the used higher layer authentication method. The attack is performed by a malicious AP that sends an EAP-Success message to the supplicant with the result that the controlled port of the supplicant switches to the authorized state and all network traffic of the supplicant flows through the attacker AP.

This is only of interest in the case that the supplicant is associated with the attacker AP and that the attacker AP is able to provide the same services like a true AP of the desired network (or at least some of them). Achieving this goal is difficult for the attacker AP because normally it is not part of the network in concern. If the attacker AP cannot manage to provide at least a simulation of the environment, the supplicant is not able to use any services and therefore does not produce much network traffic, resulting in rendering the attack of the attacker AP relatively useless. It seems to be a smaller issue than having an attacker sniffing the whole network traffic.

Finally, it has to be considered that 802.1X provides an authentication framework for using various kinds of different authentication mechanisms. In the current solution, it depends on the authentication mechanism whether authentication is secure or insecure.

Certainly, it would be possible to provide a secure authentication framework. But in cases where this additional security is not used, the overhead for security would have to be paid even if not needed. Such cases might be the result of the underlying hardware infrastructure or the use of a secure authentication mechanism such as EAP-TLS.

3.5 Evaluation

All of the security mechanisms of this chapter besides WEP are only concerned about authentication and are not especially designed for wireless networks. Therefore, confidentiality in WLAN is only concerned by WEP that is already known to be insecure.

Furthermore, WLANs suffer from a key management and key distribution problem as described in 3.2.3 that prevents the keys from being updated often. This imposes a security issue because WEP keys can be cracked if enough data is available. An improvement can be achieved by using 802.1X, but this is still an issue, especially for small and private networks not capable of using 802.1X.

Because of these reasons, it soon became clear that a better solution for wireless network security has to be found. The Task Group i[13] of the IEEE 802.11 Working Group is working on this topic since May 2001 where the security part was moved out of Task Group e.

[13] A Task Group (TG) is a committee responsible for a specific part of an IEEE Working Group. 802.11i specifies TGi of the Working Group 802.11. An overview of the IEEE 802.11 Task Groups can be found at `http://grouper.ieee.org/groups/802/11/QuickGuide_IEEE_802_WG_and_Activities.htm`

There is a growing demand for mobility and hence wireless technology. Companies are able to make much profit with WLANs. However, this requires WLAN to be reasonable secure. Therefore, after the break of WEP in 2001, many authentication mechanisms found their way into WLAN for improving at least the authentication process. Most of them have been described in this chapter.

With these measures, a reasonable improvement in the authentication process could be achieved, but it only slightly improved the situation regarding the confidentiality aspect of WLANs. 802.11i takes care of the confidentiality of WLAN but it is still a draft.

The confidentiality situation is one of the main reasons why a subset of 802.11i resulted in a security standard defined by the Wireless Fidelity (Wi-Fi) Alliance[14] . This security standard is called *Wi-Fi Protected Access (WPA)* [All03] and is the latest security enhancement of WLANs. Further information on WPA is found in Section 4.7.

[14] `http://www.wi-fi.org/`

Chapter 4

IEEE 802.11i
MAC Security Enhancements

This chapter deals with the current state of the 802.11i specification for *MAC Security Enhancements* of WLANs, [IEE03b]. It includes, but is not limited to an introduction of the new cryptographic algorithms, the used keys and key management, and WPA as the already available subset of 802.11i.

4.1 Security Framework

[IEE03b] defines a so called *Robust Security Network (RSN)* - a network that only allows the creation of Robust Security Network Associations.

The definition of a *Robust Security Network Association (RSNA)* is as follows:

"A pair of STAs is said to be using a Robust Security Network Association if the procedure to establish Authentication/Association between them includes the 4-way handshake.", [IEE03b] (see Section 4.3.1)

802.11i distinguishes between two kinds of devices: Devices that are able to create Robust Security Network Associations are called *RSNA Capable Equipment*, other devices are called *Pre-RSNA Equipment.*

The standard also supports combinations of RSNA-capable and pre-RSNA equipment in the same network, called a *Transition Security Network (TSN).* In such a network, it is essential to be able to distinguish pre-RSNA equipment from RSNA-capable equipment.

Identifying RSNA Capable Equipment

802.11i defines a so called *RSN Information Element (RSN IE)* that contains information about supported security features (cipher suite, ...) of a device. RSNA capable equipment includes this RSN IE in some specific network packets (beacons, probe response, association request/response, ...) to advertise the sender as RSNA-capable. As pre-RSNA equipment does not include such an RSN IE, RSNA-capable equipment can be detected by the existence of an RSN IE.

4.1.1 Overview of the 802.11i Authentication Process

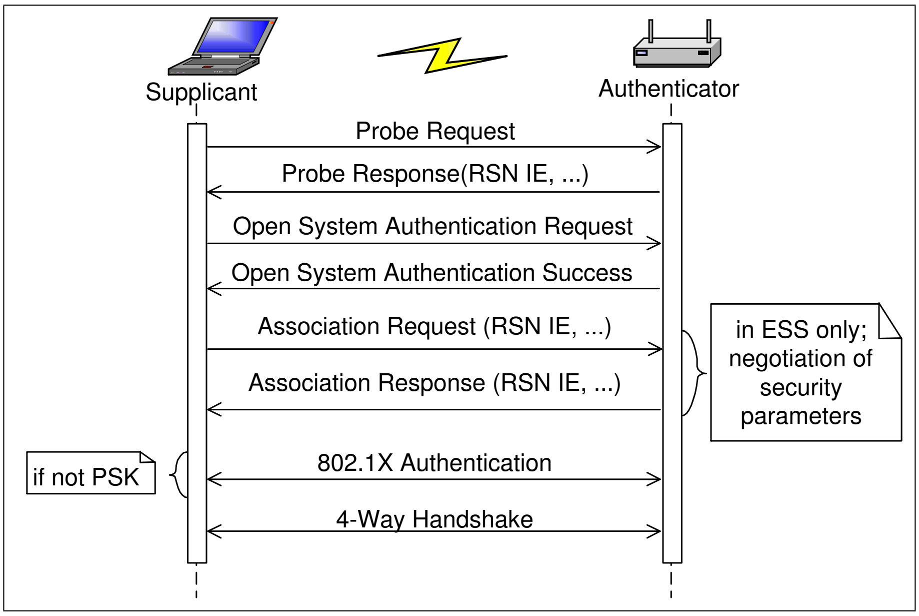

Supplicant
Authenticator
Probe Request
Probe Response(RSN IE, ...)
Open System Authentication Request
Open System Authentication Success
Association Request (RSN IE, ...)
Association Response (RSN IE, ...)
in ESS only; negotiation of security parameters
if not PSK
802.1X Authentication
4-Way Handshake

Figure 4.1: 802.11i Authentication Overview

Note for the following that in an ESS, STAs are supplicants and APs are authenticators. In an IBSS, STAs are supplicants and authenticators.

If a supplicant wants to authenticate at an authenticator, it probes for available authenticators. Each authenticator that receives a probe request sends a probe response as answer. This probe response includes the aforementioned RSN IE.

The supplicant then decides on one of the available authenticators based on criteria such as signal strength, security parameters contained in the RSN IE, etc. and performs open system authentication at the desired authenticator.

Only if the supplicant is in an ESS, it associates with the authenticator (the AP in that case). Negotiation of security parameters takes place during this process.

Now there are two ways to continue: using 802.1X authentication or a *Pre-Shared Key (PSK)*. The PSK is a key known in advance by the participants of the authentication process.

802.1X authentication results in having available a master key that is specific to the used EAP method. The *Pairwise Master Key (PMK)* is derived from this master key and used for deriving further keying information. In the case of a PSK, no further authentication has to be performed and the PSK is used in place of the PMK.

After having available the PMK, further temporal keys are derived from it. The temporal keys are afterwards used for securing the unicast communication. To install the keys, the *4-way handshake protocol* (4.3.1) is used. This protocol further installs the key used for broadcast/multicast communication.

With the completion of the 4-way handshake protocol, an RSNA is successfully established.

4.1.2 RSNA Assumptions and Constraints

802.11i lists several assumptions and constraints that have to be fulfilled for providing the desired security (see [IEE03b], Section 8.1.4). One requirement that is related to other parts of this thesis is presented here for clarification.

In the security analysis of 802.1X in Section 3.4.5, possible attacks to 802.1X are described. 802.11i makes use of 802.1X and therefore the question arises whether these attacks apply to 802.11i networks as well.

As the possibility of the MIM attack depends on the specific EAP method in use, 802.11i requires that:

"When IEEE 802.1X authentication is used, the specific EAP method used performs mutual authentication." [IEE03b]

This means that the described MIM attack is not possible in networks that apply 802.11i. It further states that EAP methods such as EAP-MD5 are not allowed to be used. Examples for allowed EAP methods are EAP-TLS, EAP-TTLS and PEAP.

Furthermore, 802.11i provides confidentiality and data origin authentication by introducing two new cryptographic algorithms. Therefore, the other attacks of 3.4.5 cannot be applied to 802.11i. This means that using 802.1X does not compromise the overall security of 802.11i.

Finally, there is the only open issue of DoS attacks. Here, the EAPOL specific DoS attacks, described in the appendix of [MA02], do not work for 802.11i, because of encryption and data origin authentication. However, the issue with DoS attacks is that they can never be eliminated completely. Overloading the network or using a jammer to perform a DoS attack on the physical layer will always be possible.

4.2 Cryptographic Keys

The cryptographic keys used in 802.11i are defined in two different key hierarchies: a *pairwise key hierarchy* to protect unicast traffic and a *group key hierarchy* to protect multicast traffic.

4.2.1 Pairwise Key Hierarchy

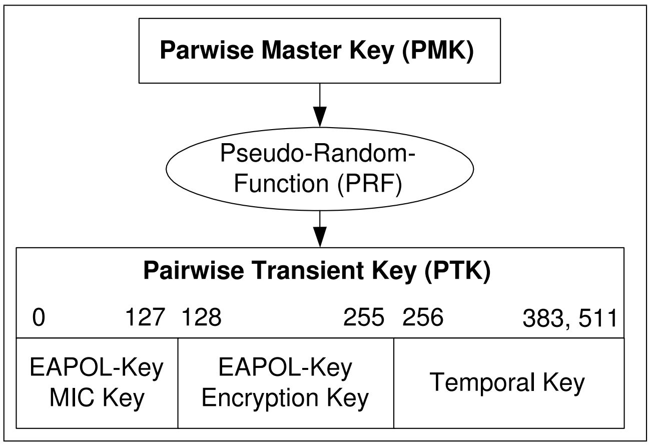


Figure 4.2: Pairwise Key Hierarchy; details [IEE03b], Section 8.5.1.2

At the root of the pairwise key hierarchy stands the *Pairwise Master Key (PMK)*. This key is derived from the master key that is generated during EAP authentication. Key generation is normally performed independently of each other at the supplicant and the authentication server. This process uses information that is sent between them during the authentication.

The length of the PMK is defined to be 256 bits. It is kept secret and only used to derive the *Pairwise Transient Key (PTK)* with a length of 384 or 512 bits. The PTK is thereafter the basis for several other keys:

- *EAPOL-Key MIC Key*: represented by bits 0-127 of the PTK, provides *"data origin authenticity in the 4-way handshake and group key distribution messages"*, [IEE03b]
- *EAPOL-Key Encryption Key*: represented by bits 128-255 of the PTK, provides *"confidentiality in the 4-way handshake and group key distribution messages"*, [IEE03b]
- *Temporal Key (TK)*: represented by bits 256-383 or 256-511, depending on the length of the PTK and further on the pairwise key cipher suite in use, TKIP or CCMP.

4.2.2 Group Key Hierarchy

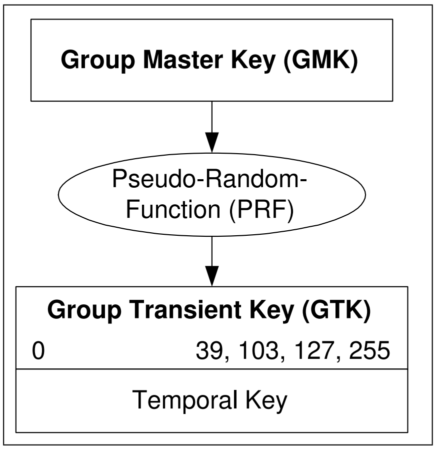


Figure 4.3: Group Key Hierarchy; details [IEE03b], Section 8.5.1.3

Three key levels are shown in Figure 4.3. This is the same as in the pairwise key hierarchy. The root of the hierarchy is the *Group Master Key (GMK)*, derived from it follows the *Group Transient Key (GTK)*, that is the basis for the Temporal Keys (TKs).

The length of the GTK depends on the used cipher suite: for WEP 40 or 104 bits are used, CCMP requires 128 bits, and TKIP needs 256 bits. Interpretation and length of the TKs is also cipher suite specific. The TK has the same length as the GTK.

Note that the GTK is the same for all supplicants of an authenticator.

4.3 Key Distribution

Key distribution depends on 802.1X, more specifically on the EAPOL-Key messages defined in 802.1X. 802.11i provides two mechanisms to distribute keying information: the *4-way handshake* for pairwise keys and the *group key handshake* for group keys.

Distribution of keying information follows successful authentication, but is also performed after a security association has already been established. Reasons for transmitting keying information after a security association has been established include, for example, the TKIP countermeasures (see 4.5.1) after an attack has been detected.

4.3.1 4-Way Handshake

The 4-way handshake is performed after 802.1X authentication completes. 802.11i specifies the purposes of the 4-way handshake as follows:

1. *Confirm the existence of the PMK at the peer;*
2. *Insure that the security association keys are fresh,*
3. *Synchronize the installation of session keys into the MAC,*
4. *Transfer the GTK from the Authenticator to the Supplicant, and*
5. *Confirm the selection of cipher suites.* [IEE03b]

Implementation of the 4-way handshake protocol is based on transmitting EAPOL-Key messages. The basic message exchange during 4-way handshake is shown in Figure 4.4. The "P" within the braces of an EAPOL-Key message states that pairwise key information is transmitted.

1. 4-way handshake starts with an EAPOL-Key message from the authenticator to the supplicant. This message includes a nonce called ANonce but no *Message Integrity Code (MIC)* as a PTK is required for computing the MIC. A MIC is used for verifying that the message has not been modified during transmission.
2. After receiving the message from the authenticator, the supplicant:
   - generates a new nonce SNonce
   - derives the PTK by using ANonce and SNonce

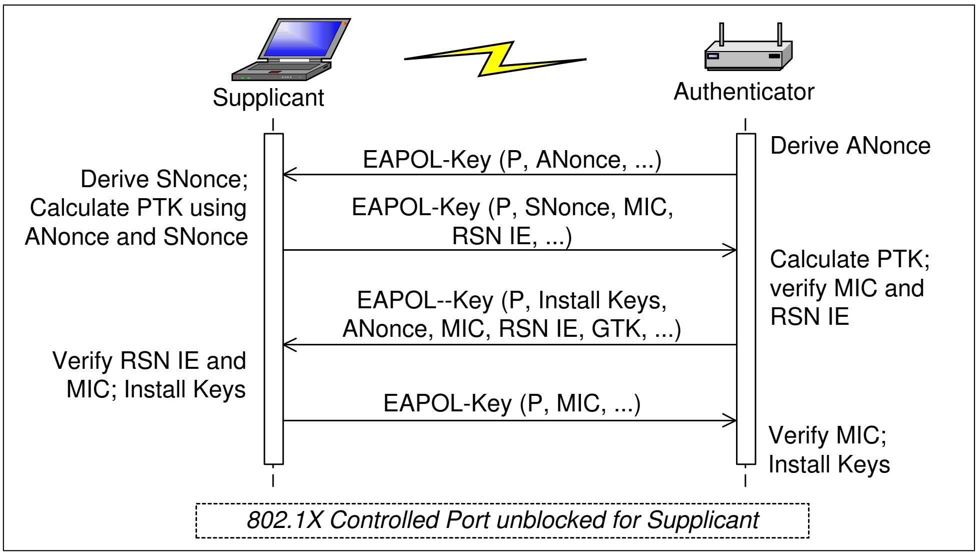


Figure 4.4: 4-Way Handshake

- constructs the message for the authenticator
- computes a MIC for the message and
- sends an EAPOL-Key message including SNonce, MIC, and RSN IE to the authenticator.

3. Now the authenticator:
   - derives the PTK by using ANonce and SNonce
   - verifies the MIC and only continues if verification is successful
   - checks that the RSN IE is bitwise the same as the one included in the previous (re)association request
   - sends an EAPOL-Key message to the supplicant with ANonce, MIC, RSN IE, encrypted GTK, and a bit that tells the supplicant to install the keys.

4. The final step is now that the supplicant:
   - verifies that the RSN IE of the message is the same as the RSN IE contained in the beacon or probe response of the AP
   - checks the MIC
   - installs the keys and
   - sends the last message to the authenticator

> Upon reception of the last message the authenticator checks the MIC and if successful, installs the keys and opens the 802.1X controlled port for the supplicant. With this the 4-way handshake is completed.

Note that no pairwise key information, especially not the PTK, is transmitted during this process.

4.3.2 Group Key Handshake

The group key handshake is performed by an authenticator to send a new Group Transient Key (GTK) to a supplicant. This process is much simpler than the 4-way handshake because it is performed after the 4-way handshake has already been performed. Obviously, the GTK can be safely transmitted because encryption of network packets is already the case.

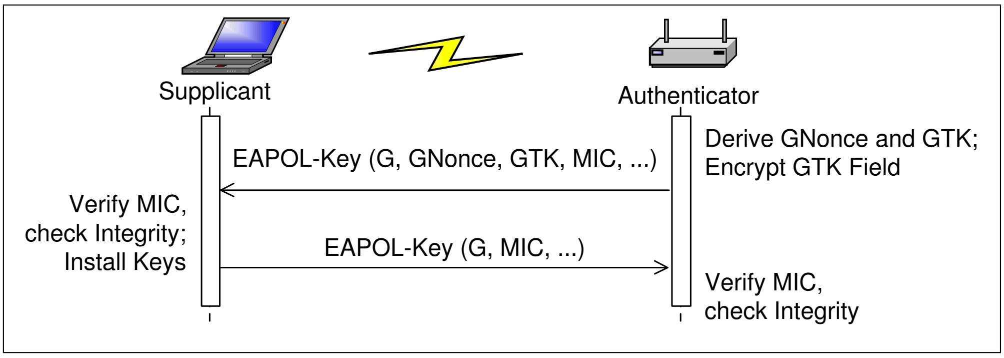


Figure 4.5: Group Key Handshake

As depicted in Figure 4.5, the group key handshake consists of only two messages: one containing the GTK from the authenticator to the supplicant and one from the supplicant to the authenticator. The ”G” within the braces of an EAPOL-Key message states that group key information is transmitted.

Upon reception of the first message, the supplicant verifies the MIC of the message and if successful, installs the new GTK. The second message is a confirmation of the message reception by the supplicant. After receiving the confirmation message, the authenticator also verifies the MIC.

The result of the group key handshake is that the new GTK is known by the supplicant.

4.4 RSNA Security Association Management

802.11i introduces the concept of a security association to 802.11 networks. Secure communication only takes place within the context of such a security association which provides everything (keys, counters, etc.) that is necessary for correct operation of the cipher suites.

The life cycle of a security association is different in ESSs and IBSSs. Nevertheless, RSNA establishment can be based on a pre-shared key (PSK) or on 802.1X authentication for both network types.

If a PSK is used, 802.1X authentication is skipped. Establishing the RSNA continues in this case after association with an AP (ESS) or open system authentication (IBSS) with the 4-way handshake protocol. See Figure 4.1 on page 37 for an overview of RSNA establishment.

4.4.1 Security Association in ESS

In an ESS there are two further cases besides the distinction between 802.1X and using a PSK: an initial establishment of the security association and the establishment of a security association during roaming of the STA within the ESS.

The following description of RSNA establishment is for the case of using 802.1X authentication. If a PSK is used, the necessary steps can be derived by skipping 802.1X authentication and using the PSK as the PMK.

Initial Establishment of the Security Association

In case of an initial establishment of the security association, the STA selects an authorized ESS by selecting an AP that broadcasts the right SSID. Thereafter, the STA authenticates at the AP via open system authentication and associates with the AP. Negotiation of security parameters takes place during this phase by including an appropriate RSN IE in the association frames.

After association has completed, STA and AP only allow 802.1X authentication frames (EAPOL messages) and the AP's authenticator initiates 802.1X authentication. This authentication is performed mutually to ensure that the AP is not a rogue.

The authentication process creates a PMK that is shared between the STA and the authentication server (AS). Now the AS transfers the PMK to the authenticator, the AP. After that, the authenticator performs a 4-way-handshake with the STA. Thereafter, the establishment of the security association is complete and normal data traffic is allowed.

Roaming and Preauthentication

A STA moving between two BSSs of an ESS may get a new security association by one of the following two ways:

- The STA (re)associates at the AP of the new BSS, followed by 802.1X authentication. In this case, it performs the same steps as for initial establishment of a security association, but ends the security association with the AP of the former BSS.
- For some applications, it is absolutely necessary to keep the time between disassociation at the old AP and completed security association at the new AP as small as possible, e.g. when viewing a movie. To minimize the time of not having a network connection due to not being authenticated, 802.11i provides a mechanism called *Preauthentication*.

  This mechanism can be used by a STA that already has a security association with an AP of the ESS. It is therefore able to use the network, especially the distribution system (DS). If the STA decides to switch the AP (and therefore the BSS of the ESS), it uses its established association with the old AP to perform 802.1X authentication with the new AP via the DS.

  After 802.1X authentication with the new AP completes, the STA disassociates from the old AP and (re)associates at the new AP, where it is already authenticated. Hence, the time needed for the 802.1X authentication is saved. The new AP and the STA are now able to proceed directly with the 4-way handshake to install the keys.

Ending the Security Association

A security association can be ended by the STA by issuing a request to delete the cryptographic keys. This destroys the cryptographic keys used in the security association so that they cannot be used to further protect network traffic. Such a request is issued by a STA if it disassociates or deauthenticates at an AP and when it associates to a new AP. Note that it is also possible for an AP to deauthenticate or disassociate a STA.

4.4.2 Security Association in IBSS

Establishment of the Security Association

An RSNA is established with every other STA found in the IBSS with which the STA has no security association. Furthermore, a STA establishes an RSNA whenever it encounters a new STA with which it has no security association. Establishment of an RSNA can be based on 802.1X or on a pre-shared key (PSK):

- **802.1X with 4-way handshake.** If full 802.1X authentication is used in an IBSS network, each STA needs to include an authenticator and an authentication server for being able to authenticate other STAs. In that case, establishing a security association between two STAs is similar to the process of initial establishment of the security association in an ESS.

- **Using a pre-shared key.** In this case, a STA immediately continues after open system authentication with the 4-way handshake by using the PSK as PMK.

In an IBSS, each STA defines its own group key for securing its broadcast/multicast traffic. This group key is transmitted to other STAs during the 4-way handshake. For subsequent group key updates, the group key handshake protocol is used.

Ending the Security Association

Termination of a security association in an IBSS happens the same way as in ESS networks.

4.5 RSNA Data Confidentiality Protocols

The confidentiality aspect of WLANs has always been an issue so far. To ensure the confidentiality of data being transmitted over WLANs, 802.11i defines two new confidentiality protocols:

- *CCMP: Counter Mode / Cipher Block Chaining Message Authentication Protocol*

- *TKIP: Temporal Key Integrity Protocol*

CCMP is the stronger protocol and all devices that claim to be RSNA-capable have to implement CCMP. It is also the default protocol to be used in RSNs.

Implementation of TKIP is optional and only recommended to be used with pre-RSNA equipment that may be patched to implement TKIP for migration purposes. The reason is that TKIP is the weaker protocol and based on WEP.

4.5.1 Temporal Key Integrity Protocol (TKIP)

The *Temporal Key Integrity Protocol* is used to enhance the WEP protocol (Section 3.2) for pre-RSN hardware. Enhancement is achieved by adding the following functionality:

- A transmitter calculates a keyed cryptographic message integrity code (MIC) over source and destination address, priority and the plaintext data of an MSDU. The MIC is appended to the MSDU and sent to the receiver that verifies the MIC and further the ICV that is generated by WEP. The receiver discards all packets where MIC and ICV cannot be verified successfully. This is used as a defense mechanism against forgery attacks.

- TKIP provides countermeasures against a compromise of the TKIP MIC. These countermeasures are used to rate limit the key updates. They further bound the probability of a successful forgery and the amount of information an attacker can learn about a key.

- TKIP introduces a *TKIP Sequence Counter (TSC)* to sequence the MPDUs it sends. This provides a simple form of replay protection. A receiver drops all packets that are received out of order. The TSC is encoded as a WEP IV to transmit the TSC value from the sender to the receiver.

- To defeat weak-key attacks against the WEP key, a cryptographic mixing function is provided. This function combines a temporal key and the TSC into a WEP seed. A receiver recovers the TSC out of the packet and uses the same mixing function to rebuild the WEP seed needed for correct decryption of the packet.

TKIP Encapsulation

Before encryption of a network packet, TKIP calculates a MIC by using the source and destination address, the priority and the plaintext data of the MSDU and further a key. The calculated MIC is appended to the MSDU.

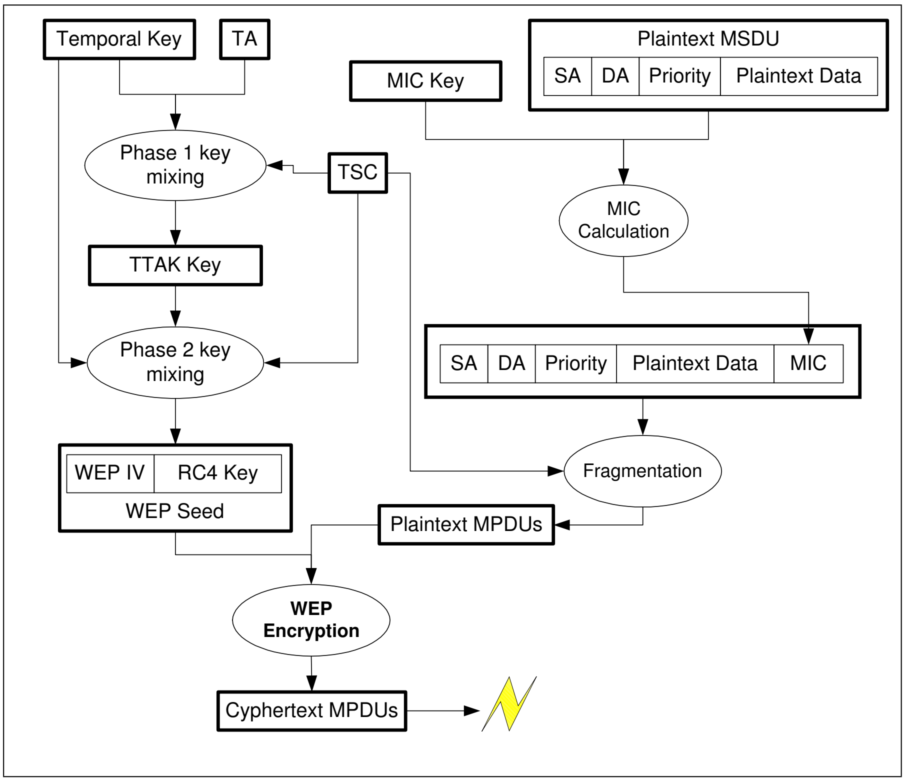


Figure 4.6: TKIP Encapsulation

This MSDU with the appended MIC is possibly fragmented resulting in one or more MPDUs that are encrypted afterwards by using WEP encryption.

For WEP encryption, a WEP seed is needed. This WEP seed is derived in to steps: phase 1 key mixing and phase 2 key mixing. Phase 1 key mixing uses the transmitter address (TA), a temporal key and the TKIP sequence counter (TSC) to produce the intermediate result, the TKIP mixed Transmit Address and Key (TTAK).

The TTAK is used as input to phase 2, together with the TSC and the temporal key. Phase 2 results in the output of a WEP seed of 128 bits, also called the per packet key, represented as WEP IV and WEP key. This WEP seed is different for each MPDU that needs to be encrypted. Furthermore, the WEP seed is used as WEP default key that is identified by a key ID associated with the temporal key.

TKIP encryption results in one or more ciphertext MPDUs that are transmitted afterwards to the receiver. A graphical illustration of the TKIP encapsulation process is found in Figure 4.6.

TKIP Decapsulation

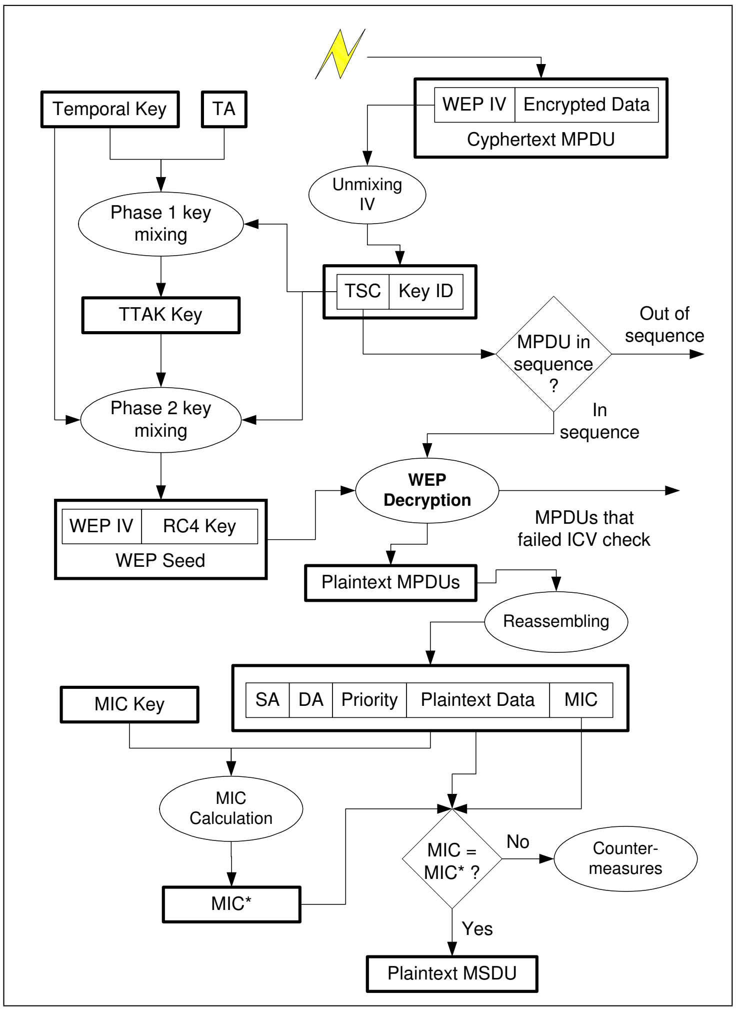


Figure 4.7: TKIP Decapsulation

Before decryption of a received MPDU, TKIP extracts the TSC and key ID out of the WEP IV. Any MPDU that is received out of order is discarded. If the MPDU is received in order, it is decrypted by using the WEP decryption algorithm. The seed for the WEP algorithm is constructed the same way as the seed in the TKIP encryption process.

During WEP decryption, all MPDUs that fail the integrity check are discarded. After decryption of the MPDUs, they are reassembled to an MSDU-with-MIC. Now the MIC value for the packet is calculated and compared to the MIC contained in the packet. If they are the same, the MSDU is eventually delivered - if they differ, countermeasures are taken. The TKIP decapsulation process is shown in Figure 4.7.

TKIP Countermeasures

If a TKIP implementation detects an active attack, indicated by a MIC failure of a received MSDU, it takes countermeasures to accomplish the following goals:

- The currently used authentication and encryption keys are destroyed to prevent an attacker from learning anything about these keys from the MIC failure.
- The incident is logged as a security relevant matter.
- The rate of MIC failures is kept below one per minute to prevent an attacker from performing a large number of forgery attempts in short time. This also requires that keys are not distributed more than once per minute in this case.

4.5.2 Counter-Mode/CBC-MAC Protocol (CCMP)

CCMP is based on the *Advanced Encryption Standard (AES)* [NIS01] and *Counter with CBC-MAC (CCM)* [WHF02] as the mode of operation. CCM combines *Counter (CTR)* Mode for confidentiality and *Cipher Block Chaining Message Authentication Code (CBC-MAC)* for authentication and integrity.

Counter with CBC-MAC has been submitted to the National Institute of Standards and Technology (NIST) as a proposed mode of operation. It is further an IETF Internet Draft. A security analysis of CCM can be found in [Jon02] and a specification of requirements for the use of CCM is given in [CCM].

The properties of CCM can be summarized as follows:

- CCM operates on whole packets and does not support partial or stream processing.
- CCM requires a unique nonce per packet. The size of the nonce limits the amount of packets that can be encrypted with a single block cipher key. Reuse of a nonce with a given block cipher key must not occur.

- CCM expands the packet size because of appending authentication information. This authentication information is used to verify the source and the integrity of the packet.
- CCM provides confidentiality by encryption of the packet.

CCMP Encapsulation

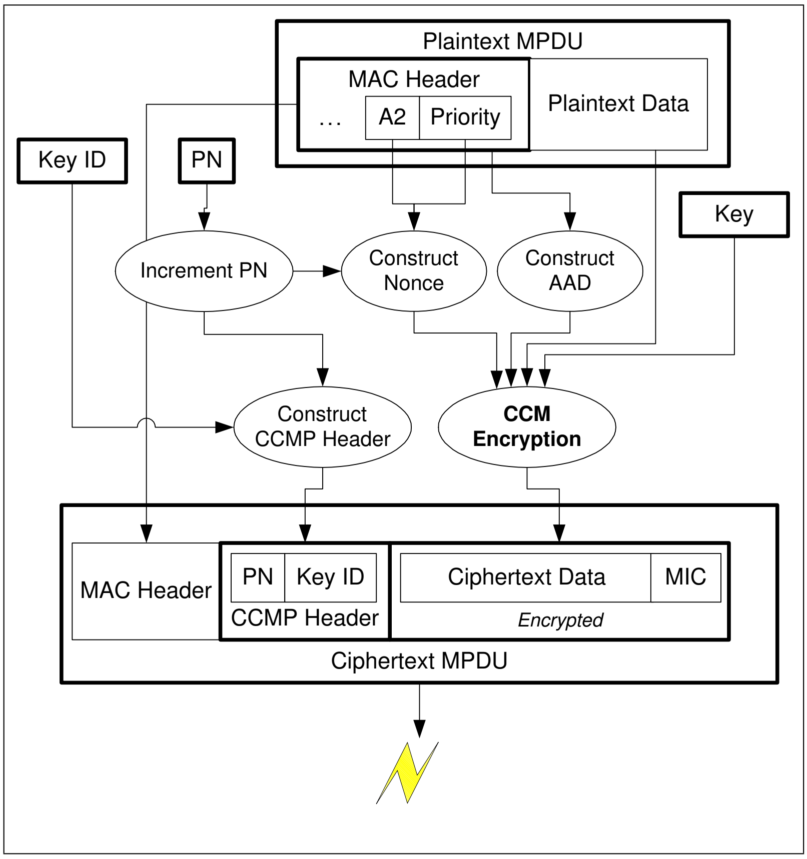


Figure 4.8: CCMP Encapsulation

In contrast to TKIP, CCMP starts at the MPDU level and not with MSDUs. At first, a unique nonce has to be created for each MPDU. CCMP constructs this nonce by using a 48-bit packet number (PN), the address 2 field (A2)[1] and the priority of the MPDU. The PN is incremented with each MPDU.

Furthermore, the *Additional Authentication Data (AAD)* for CCM is constructed by using some fields of the MAC header. AAD provides integrity

[1] A2 is most likely the source address of the sender.

protection for the MAC header fields contained in the AAD. Because some fields of the MAC header might change during transmission, e.g. TA or RA, they have to be excluded from the AAD.

CCM encryption can be performed by using the constructed nonce, the AAD, the plaintext data, and the temporal key (TK) as input. The output of this encryption process is the encrypted data with MIC.

CCMP requires its own header, called the CCMP header. This header is constructed by using the incremented PN and the key ID. Finally, the CCMP header is appended to the MAC header and the output of the CCM encryption process is appended to the CCMP header.

A graphical representation of the CCMP encapsulation process is given in Figure 4.8.

CCMP Decapsulation

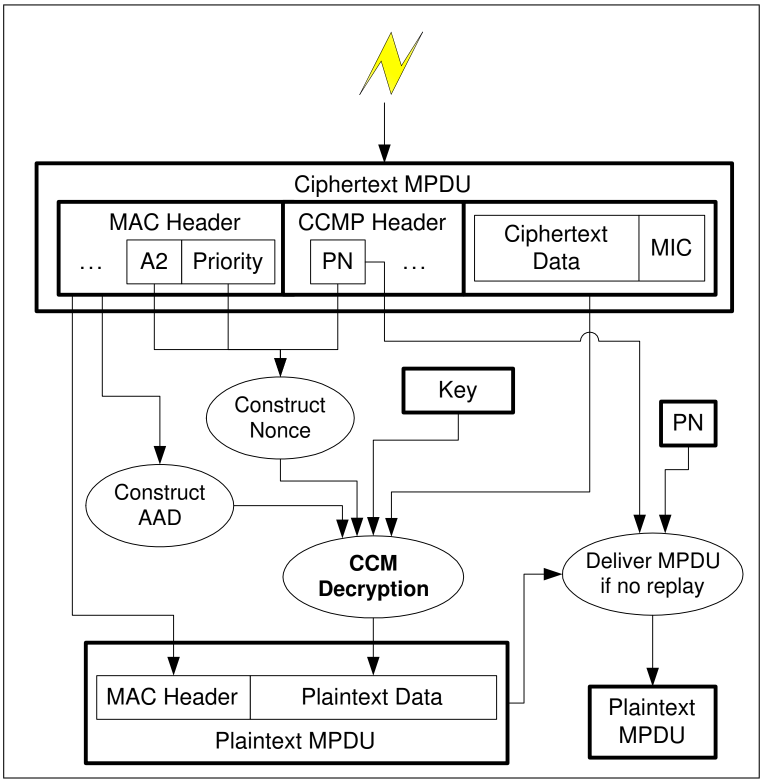


Figure 4.9: CCMP Decapsulation

To start with the decapsulation process, CCMP constructs the nonce by using PN, A2 and the priority of the received packet. It furthermore recalculates the AAD and proceeds with CCM decryption by using nonce, AAD, TK, MIC and the ciphertext data as input. The decapsulation process is only able to succeed if the MIC and a recalculated MIC* at the receiver are the same.

After successful decryption, a replay check is performed. For this check, the receiver has to maintain its own PN for each MAC address from which it receives CCMP traffic. If the packet has not been replayed, it is eventually delivered.

The CCMP decapsulation process is depicted in Figure 4.9.

4.6 Summary

4.6.1 Differences between ESS and IBSS

The differences between ESS and IBSS networks provided in Table 4.1 are the ones that are related to the security enhancement of 802.11i.

ESS	IBSS
• STA initiates all connections	• STA must be prepared for other STAs to initiate communication
• AP enforces uniform security model	• STAs define and implement their own security model
• AP always chooses the security suite in use	• STAs negotiate security algorithms to use
• AP offloads authentication decision to authentication server	• STA must make its own authentication decision

Table 4.1: Differences between ESS and IBSS Networks

4.6.2 Benefits of 802.11i

802.11i solves several problems that exist in 802.11 networks today. First of all, there is the provision of confidentiality due to two new cryptographic algorithms, TKIP and CCMP. They do not only provide confidentiality, but also solutions to other problems such as data origin authentication.

When considering authentication, 802.11i does not reinvent the wheel but includes 802.1X as authentication mechanism. This provides a standardized mechanism for authentication of STAs on a per-user or per-device basis.

Furthermore, 802.11i does not render legacy hardware devices unusable, but allows for a smart migration to a so called robust security network (RSN) by the intermediate form of a transition security network (TSN), where RSNA capable and pre-RSNA equipment can be used in one and the same network.

Key Distribution is also one of the major benefits of 802.11i. Until 802.11i, there was no standardized mechanism for distributing keying information (except for WPA, see 4.7).

The use of replay counters enables STAs to perform replay detection of already sent networks packets. This allows detection of active (replay) attacks and to take appropriate countermeasures such as changing the used keys.

Also MAC spoofing attacks can be detected due to the use of a different pairwise key per individual STA.

4.7 Wi-Fi Protected Access (WPA)

Wi-Fi Protected Access (WPA) can be considered to be a subset of the 802.11i standard. Because the 802.11i standard is still a draft and because of the urgent need for improving security of WLANs, WPA was defined.

The WPA standard as defined in [All03] gives many references to the 802.11i standard and only provides information where it is necessary to receive a whole integrated solution. To make sure the references to the 802.11i standard do not change, the referred version is fixed to the 802.11i draft version 3.0, [IEE02].

Almost all of the previous sections of this chapter are valid for WPA too. As WPA is so closely related to 802.11i, it should be sufficient to provide a short comparison of WPA and 802.11i and to list aspects that are different. A main summary of the different goals of WPA and 802.11i can be stated as follows:

802.11i adds an overall security concept to WLANs. WPA adds a security concept to WLANs too, but with the additional requirements that it was to be available as soon as possible and was to be implementable for legacy devices.

The latter requirement specifies that WPA is destined more or less for legacy devices, i.e. it should be possible to upgrade legacy devices to WPA by a simple firmware update. CCMP for example, as a resource intensive encryption algorithm, is only optional for WPA. The main reason for this is that new hardware is required to provide sufficient performance.

4.7.1 Comparison of WPA and 802.11i

The following provides short comparison of WPA and 802.11i. It mainly deals with differences of the two standards and with features provided by 802.11i that are not included in WPA.

- WPA does not support IBSSs. It only includes a description for a simple approach to IBSSs as reference. This approach is based on a PSK configured as group key and used for encryption of messages in the IBSS.
- The RSN IE of 802.11i is modified for WPA to mark it as the WPA IE (see [All03], Sect. 2.1, p. 9-10).
- CCMP is the default cipher suite for 802.11i, TKIP is the default for WPA.
- WPA does not support preauthentication for fast handoff. Note that here the preauthentication in combination with 802.1X is meant, not the MAC layer preauthentication.
- The TKIP countermeasures defined in 8.3.2.4.2 of [IEE02] are replaced in WPA, Sect. 3.1.
- Latest versions of 802.11i (at least version 7.0) distribute the first GTK during the 4-way handshake and use the group key handshake only to change the GTK. Version 3.0, on which WPA is based, does not specify to distribute the GTK during the 4-way handshake. The GTK is always distributed by using the group key handshake.

For the rest of this thesis, the differences between 802.11i and WPA are not of further importance. It is sufficient to keep in mind the information provided for 802.11i.

Chapter 5

Security in HotSpot and Enterprise Environments

After having studied the available security mechanisms for WLAN and 802.11i, the focus of the thesis now turns to the practical application of WLAN. This is the area where often incompatibility issues arise because of having to apply several standards in one integrated environment. Furthermore, the migration from an infrastructure with a given set of standards to a new infrastructure, using new or different standards, is often a significant issue.

Small Office and Home Office (SOHO) users are often not much concerned about migration considerations because they likely have the option of simply replacing their old hardware with new one that is capable of the latest (security) features. Furthermore, the security needs and hardware infrastructure of SOHO users is far from the requirements of large scale enterprises or hotspot operators.

This chapter focuses on the issues that arise when requiring security and additionally the support of legacy devices that are not capable of the latest security mechanisms. The environment in mind is the one of a *Wireless Internet Service Provider (WISP).*

5.1 State of the Art

5.1.1 HotSpot Environments

The present situation is not very satisfying for users that require confidentiality because wireless hotspots do not provide encryption at all. One of the reasons is that WEP is still the only widely available possibility, but it is not applied due to several reasons.

First, each customer that wants to use a hotspot would have to know the pre-shared WEP key for a specific location. Second, key distribution and management of WEP keys would be a real problem for the hotspot operator. Finally, the STAs of all customers connected to a single AP of the WISP would have the same WEP key. This provides only a limited form of confidentiality. Additionally, there are still the already known weaknesses of the WEP algorithm itself.

Mainly because of these reasons WEP is currently not applied in hotspot environments. Hence, users requiring security for their applications have to use other (upper layer) security mechanisms such as *Virtual Private Networking (VPN)* or HTTPS[1] in case of only using the web.

Virtual Private Networking

VPN is the term used for a technology that connects private networks together by using encrypted tunnels over a public network. Each endpoint of the tunnel is responsible for encryption and decryption of data that is transmitted to, or received from, the other endpoint. An example of a VPN, created by connecting two private networks together via an encrypted tunnel, is provided in Figure 5.1.

Due to the encryption of network packets that are sent over the public network, the data is kept private. This has the effect that the connected private networks appear as one private network without the need for an extra private leased line between them. Because VPN applies encryption of transmitted data by itself, it does not require the connection between the two tunnel endpoints to be secure. Therefore, if the tunnel endpoint resides in the WLAN device, it does not matter if WLAN traffic is not encrypted at the MAC layer.

[1] HyperText Transfer Protocol (HTTP) over Secure Socket Layer (SSL)

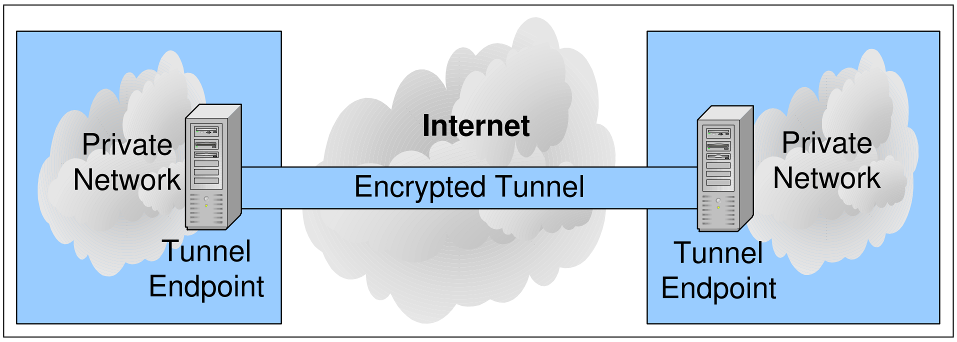


Figure 5.1: VPN consisting of two private networks

A further advantage of VPN is that it saves the costs of an extra link by using public shared resources. This is especially useful if the connected private networks are far from each other and the costs for a leased line would be high. Furthermore, VPN is more flexible because a device can establish a secure tunnel with another device by simply using the Internet. Due to this, the users do not need to care for a special leased private line between the devices beforehand.

A more thorough overview of VPN is found in [Mic03].

HotSpot Infrastructure

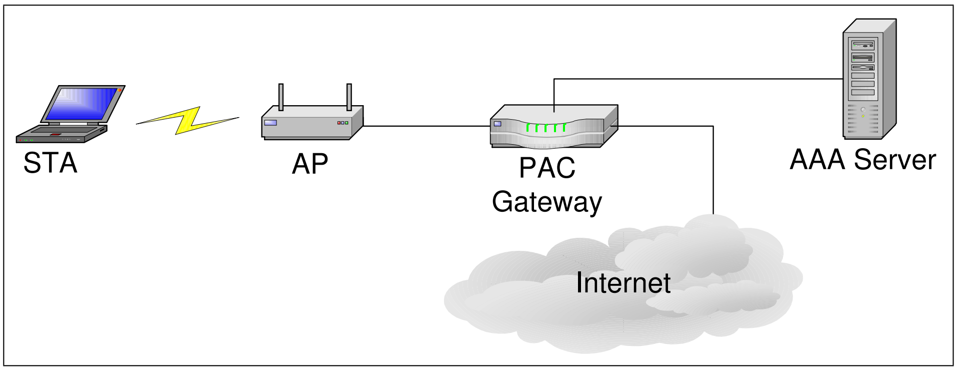


Figure 5.2: Hardware Infrastructure for HotSpot Environments

A general hardware infrastructure of hotspot environments is depicted in Figure 5.2. In the figure, the wireless client first associates with an access point that forwards client traffic to the *Public Access Control (PAC)* Gateway. The PAC gateway has to decide, based on the authentication state of the wireless client, whether to allow the traffic of the client to pass through to the Internet or not.

The authentication state is changed due to successful authentication or deauthentication. Currently, a web-based solution is used for authentication that works as follows:

If a user tries to browse the web, but is unauthenticated, an authentication page is provided to the user's web browser - regardless of the user's desired web site. This authentication portal allows the user to enter his/her credentials. For security reasons, SSL is used to encrypt the authentication traffic. If the provided credentials can be successfully verified, the user is authenticated.

After successful authentication, user traffic is allowed to pass through the PAC gateway to the Internet. Furthermore, a possibility is provided to the user, to end the authenticated session. Such a possibility might be a button or a link provided in a browser window after the authentication process successfully completes. This kind of authentication is called the *Universal Access Method (UAM).*

Finally, there is the *Authentication, Authorization and Accounting (AAA)* server that is responsible for authenticating users, based on the provided credentials. Furthermore, it cares for authorization of users and is responsible for accounting to provide a basis for billing. Note that the functionality of AP, PAC gateway and AAA server need not necessarily be separated into three different devices.

Further information on WISP and roaming is found in *Best Current Practices for Wireless Internet Service Provider (WISP) Roaming* [ABS03] provided by the Wi-Fi Alliance.

5.1.2 Enterprise Environments

Regarding the use of WLAN, enterprises can be classified into three categories:

1. Enterprises that apply WLAN in an insecure manner.
2. Enterprises using WLAN with WEP and may be 802.1X.
3. Enterprises with a "No WLAN" policy.

The first class of enterprises are the ones that are responsible for headlines such as *War Drive Survey: 57% of Enterprises Wireless LANs Not Encrypted*, [Air03]. Some of the reasons for insecure WLAN installations are that the enterprises are often not aware of the risks of WLANs compared to traditional LANs, may not have sufficient knowledge, or do not care much about security at all.

Enterprises with an urgent need for WLANs and with a security policy that allows the existence of the current risks of WLANs, after applying available security mechanisms, reside in the second category. Using WEP and MAC filtering is often considered to provide sufficient security for smaller companies. Security measures taken by enterprises to improve security of their networks when using WLAN are described later in this section.

"No WLAN" is the current policy of enterprises with highest security needs. They do not allow WLAN because it is still considered to be too insecure. This situation might slightly change now with the availability of WPA and definitely when 802.11i will be available.

Security Measures

This section describes some security measures taken by enterprises for improving the overall security of their network when applying WLAN.

MAC Filtering. MAC Filtering is a mechanism where network access is only allowed if the MAC address of a network device is found in a configured list of privileged MAC addresses. Such a list might contain the MAC addresses of all network devices of a company to deny network access to foreign equipment. The issue with MAC filtering is that the MAC address of a network device may be configured manually, allowing an attacker to use an official MAC address of the company.

WEP. As other algorithms are still to be more widely implemented, WEP is the most common encryption mechanism used for ensuring confidentiality of transmitted data. The problems of WEP have already been discussed in 3.2.3.

Treat WLANs as external. This means that WLANs should not be trusted and a firewall has to be placed between the wired and the wireless network. Thereafter, only some specific traffic is allowed to pass through the firewall. Mechanisms such as VPN are used to allow privileged users to get full access to the internal wired network of the company.

802.1X. 802.1X provides additional security by providing port based network access control. Only secure authentication mechanisms such as EAP-TLS, EAP-TTLS or PEAP should be used to avoid additional security risks.

Remove unauthorized APs. It is not sufficient to only secure the own hardware equipment of the company. Employees might buy APs for their workplace to enjoy the advantages of WLAN, regardless of the fact that installing private APs is likely to be forbidden. Therefore, the security staff of a company has to actively search for such unofficial APs and remove them.

Using secure Services. A further possibility to improve overall security is to allow only secure services to be accessed via WLAN devices. Such services could be HTTPS or SSH[2] because they apply their own encryption at the application layer and do not require further encryption for example at the network or MAC layer.

Further advice for hardening the network is found in [Wea02].

5.1.3 Guests and non-privileged Users

For enterprises and hotspot operators it is often not sufficient to only provide full access to a network to authorized users and keep others out. Often a further requirement is to provide some limited network access to guests and non-privileged users. Services offered to this user group might include the access to public resources or the provision of some local content for e.g. advertising purposes or instructions on how to use the service.

It would be possible to provide different physical hardware infrastructure and therefore different APs for privileged and non-privileged users, but as this causes an explosion of costs it is not desirable. Therefore, *Virtual LAN (VLAN)* [IEE98] technology is used for the purpose of user differentiation. VLAN allows various different virtual LANs to be implemented on top of a single physical LAN. Different VLANs are distinguished by a different VLAN ID and traffic from one VLAN to another VLAN must be routed.

[2] SSH: Secure SHell

If different user groups are assigned to different VLANs, the needs of the various user groups can be handled individually due to different configuration of the VLANs. The advantage now is that the different user groups do not have to be physically separated, but might even be physically connected to the same device.

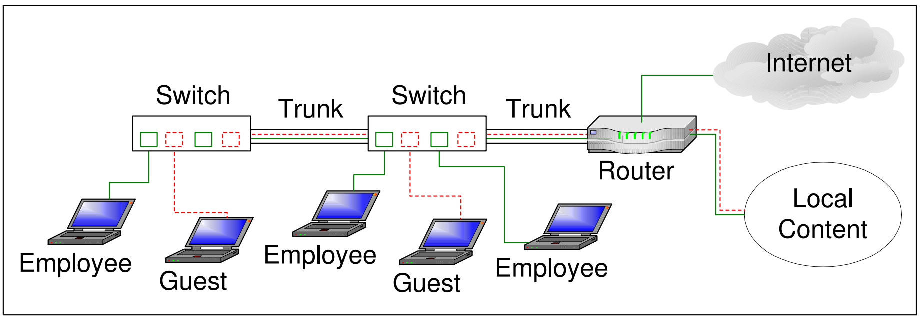


Figure 5.3: VLAN Infrastructure Example

Figure 5.3 shows an example VLAN setup. Two VLAN-enabled switches provide access to two different VLANs, called VLAN-Priv and VLAN-Guest. VLAN-Priv is for privileged users, e.g. employees of a company, and VLAN-Guest is for non-privileged users, e.g. guests of a company. The ports of the switches are assigned to either the first or the second VLAN. Graphically, the different VLANs are distinguished by different colors and solid or dotted lines.

Traffic of both VLANs has to be transmitted between the switches. Such a connection where traffic of all VLANs passes through is called a *trunk*. Physically, a trunk is a simple connection like any other network link, but the network fragments transmitted on this trunk are tagged to be able to distinguish between the different VLANs. Furthermore, a router is connected via a trunk to one of the switches and is responsible for routing traffic between the two VLANs and the Internet or some local content.

The routing decision is based on the VLAN the sender or recipient of a packet (further called user) resides in. Users of VLAN-Priv are allowed to access the Internet and the local content area, users of VLAN-Guest are only allowed to access local content. Note that although there are shown two lines from the router to the local content, it is only one physical connection. This connection is not a trunk - the two lines are only provided here to illustrate the paths of the different VLANs.

To extend VLANs to wireless networks and therefore provide the possibility of an AP serving several VLANs, a unique VLAN ID is mapped to a unique

SSID of the wireless network. [Cis03] states that different wireless configuration is possible for different VLANs. Configuration options include for example different encryption keys for different VLANs.

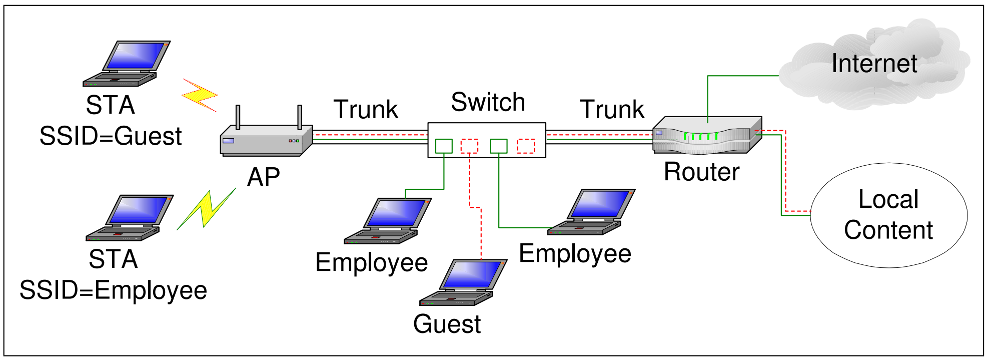


Figure 5.4: VLAN with AP Infrastructure Example

An infrastructure example for using VLAN with WLAN, including an AP that serves two VLANs, is drawn in Figure 5.4. The AP is configured to accept two different SSIDs, *Employee* and *Guest*. SSID Employee is mapped to VLAN ID VLAN-Priv of the example above and SSID Guest is mapped to VLAN-Guest. Different authentication mechanisms and restrictions are applied to wireless clients depending on their used SSID.

Due to the mapping of VLAN ID to SSID, STAs using Employee as SSID are in VLAN VLAN-Priv and are allowed to access Internet and local content. STAs using SSID Guest are in VLAN VLAN-Guest and are only allowed to access local content.

Further information on wireless VLAN is found in [Cis03].

5.1.4 Deficiencies and Requirements for the Future

First of all, for the case of hotspots, there is the issue of not using any encryption at the MAC layer. This leaves the responsibility for confidentiality of transmitted data in the user's hand. It is clear that the situation regarding confidentiality has to be improved.

Furthermore, current network infrastructures lack key management and key distribution services. This prevents hotspot operators and large scale enterprises from using WEP because of not being able to dynamically update the keys. A solution to this issue is under way with 802.11i and with WPA already available but not sufficiently widespread.

VPN is a suitable solution for ensuring security of transmitted data, but the issue is that it cannot always be applied. Users can only apply VPN if they have another tunnel endpoint where they can connect to. This might be OK for business users connecting to their company via VPN, but private users are generally not able to use VPN.

A further problem of VPN is that it is not compatible with *Network Address Translation (NAT)*. This is because VPN ensures that the source IP address of the network packet has not been changed during transmission of the packet. As NAT changes the source address, it leads to VPN verification of the source address being unsuccessful. Solutions to this problem exist, but have to be supported by the devices on the network path. A solution to this issue is described in [Cis].

Authentication is not always an easy task for an end user, especially for an unexperienced user. The best solution for the authentication process would be to perform authentication automatically, without too much user interaction. One might require that a user has only to turn on his/her notebook and after booting the operating system (OS) he / she is already authenticated.

5.2 Infrastructure Improvements

One current issue is that the communication between APs and STAs is not encrypted due to lack of key management for WEP. Technically, an improvement of this situation could already be implemented because of the availability of WPA. The only problem is that evolution is not only a matter of technical feasibility, but of several other reasons such as costs, profitability, customer requirements, compatibility etc.

Compatibility is one requirement that is of special interest for a hotspot provider, desiring to migrate to a WLAN environment with enhanced security. The considered case is a migration to WPA or later 802.11i. Denying access to users with devices that are not capable of WPA or 802.11i is not an option for a hotspot provider.

The challenge now is to find a solution to allow customers with WPA- or 802.11i-capable devices to make use of the new security features as well as providing service to users with legacy devices.

A possible solution is to roll out the infrastructure twice, covering the desired area with APs with and without WPA or 802.11i. This is not a very promising and surely not the cheapest solution, especially if the APs without security do not already exist. Furthermore, covering an area with too many APs will lead into interference problems, decreasing available performance and bandwidth.

Another solution is to use VLAN technology as is already implemented in e. g. APs of Cisco. In this solution, two SSIDs are used with a single AP, regardless of the fact that the AP is only able to advertise a single SSID in its beacon frames.

Each SSID maps to a different VLAN ID. Therefore, network traffic of the two ESSs, distinguished by the SSIDs, can be handled individually. This solution already removes the need for doubling, but requires VLAN enabled APs and a VLAN infrastructure behind them. Since the VLAN infrastructure might not be already available, it also increases the costs for migration to a secure environment.

This thesis provides a solution to the migration problem where a single AP can be used to serve both devices capable of not capable of WPA. Devices not capable of WPA do not need to support 802.1X. The presented solution does not need VLAN technology, prevents doubling the hardware and is also applicable to devices that implement 802.11i in the future.

Chapter 6

Intersil AP Development Kit

This chapter gives an introduction to the Intersil *Access Point Development Kit (APDK)*, ISL36356A. It describes the basic structure and principles of operation of the AP. Furthermore, this chapter provides an introduction to the development environment that was used for customization of the AP.

6.1 Terminology

- **User Space:** is the area for individual user applications.
- **Kernel Space:** is the area where core operating system functionality is implemented.
- **System Call:** happens if a user space application accesses functionality implemented in the operating system kernel.

6.2 Access Point Overview

Figure 6.1 gives a rough overview of the structure of the AP. It consists of mainly three components: the hardware, the *MAC Virtual Coprocessor (MVC)*, and the main system.

At the bottom is the underlying hardware: CPU, memory, network interface, etc. This layer cannot be changed and is not of much interest for this thesis. Therefore, the reader requires no further understanding of this layer.

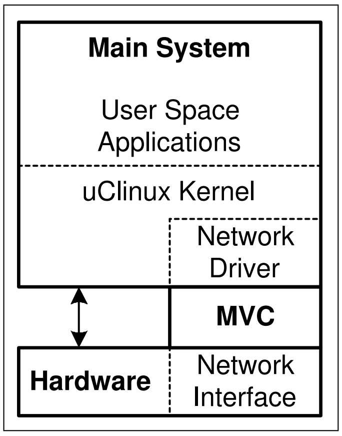


Figure 6.1: AP Structure Overview

Above the hardware comes the MVC that is already implemented in software. The MVC is a proprietary module by Intersil only available as binary image. It provides an interface to the main system to handle and configure the network devices as well as for reading information. Most of the functionality is for the wireless network interface. Almost the whole functionality of WLAN specific communication is already implemented in this layer.

The next layer is the main system that can be further divided into two sub-components: the uClinux[1] kernel including the drivers for the hardware, especially the MVC, and the user space applications.

Additionally, Figure 6.1 shows the two components that are directly interacting with the MVC, the network interface below the MVC and the network driver above the MVC. For the rest of this thesis, keep in mind that the network interface is only accessed via the MVC by the network driver.

Further information on the AP structure is found in the reference manual *Architecture of a Linux based Intersil AP* [Int02a].

6.2.1 Main Software Components

Flash memory is used for storing the main software components of the AP. Two components were already mentioned above - the MVC and the main system. Besides the images of the MVC and the main system, two further images are stored in the flash memory of the AP: the bootloader and a rescue image.

[1] uClinux is a port of Linux to systems without a Memory Management Unit (MMU), see `http://www.uclinux.org/`.

The bootloader is a life time component of the AP and cannot be updated. It is responsible for starting the boot process.

Finally, there is the rescue image. It contains software necessary to bring up the AP to a state where the main system can be restored via the MVC. This might be necessary if the main system has been corrupted and is not bootable any more.

6.2.2 Boot Process

The bootloader is the software that is started first. It searches the flash memory for valid images of rescue system, MVC and main system. Dependent on the presence of the different images, the bootloader boots up in either rescue or normal mode.

To boot the AP, the bootloader first starts the MVC. After the MVC has been loaded, the bootloader looks for a valid image of the main system. If a valid image can be found, the main system is loaded and the AP started in normal mode. If no valid image of the main system is found, or the user pressed the reset button while plugging in the power supply, the AP is booted in rescue mode.

6.2.3 Firmware Update

Each main software component can be updated individually, except the bootloader that is the same during the whole life of the AP. Images for rescue system and MVC are provided as binaries by Intersil, only the image for the main system can be developed individually with the development kit.

Two ways are provided for firmware update:

1. via TFTP (Trivial File Transfer Protocol) by using a TFTP client
2. via HTTP (HyperText Transfer Protocol) by using an HTTP client, better known as (Internet) browser

After a new image has been uploaded to the AP, it gets installed and afterwards the AP restarts for the changes to take effect.

6.3 Network Functionality of the AP

6.3.1 Overview

This section provides an overview of the flow of network data through the AP. For this we start with an incoming packet at the wireless interface and follow the path until the response for the packet leaves the AP.

After a network packet is received by the hardware, processing starts in the MVC. Necessary steps include decryption (if enabled) of an encrypted wireless packet, integrity checking and more. If the packet is received successfully by the MVC, i.e. no failures for integrity check etc., it is passed to the network driver.

From here on processing of the packet is continued by the uClinux kernel. This happens the same way as in the usual Linux kernel, e.g. Ethernet, IP, TCP, UDP processing. Furthermore, firewalling functionality, called netfilter[2], is implemented in the kernel. The firewall rules, configured with help of the iptables userspace application, are applied to each packet.

If the packet is received successfully by the uClinux kernel, passes all checks of the different network layers, and is not filtered due to some firewall settings, it is passed to the appropriate user space application. After that, the application further processes the packet and produces a response.

The response of the application is processed by the different network layers in the kernel and - if not filtered by the firewall rules - is passed by the network driver to the MVC. The MVC now takes all the steps needed to send a valid wireless packet to the intended wireless recipient. Necessary steps include encryption (if enabled), calculating integrity check values and of course managing the physical transmission of the packet.

Note that the described network path is only valid for a packet with the AP as destination. For many network packets the path already ends or returns in the kernel or the MVC.

Forwarding between Wireless and Wired Interface

One of the central features of an AP is to forward network traffic between the wireless and the wired interface. To accomplish this goal, the Intersil AP makes use of the bridging mechanism provided by the uClinux kernel.

[2] `http://www.netfilter.org/`

With help of the *brctl* userspace utility, a bridge interface is created. Both network interfaces - the wireless and the wired one - are added to this bridge. The result is that traffic received on one interface with a destination other than the AP is made available on the other interface as well.

Further details on how to use bridges in the Linux kernel are found in [Rad02].

6.3.2 MVC Software Interface

Now that a rough overview of the Intersil AP architecture has been given, the MVC as one of the central components is described in more detail in this section.

As already stated, the MVC is the layer that handles most aspects of WLAN specific communication autonomously. It only provides an interface to the software layers above to transmit and receive frames, set and read configuration options and to obtain statistics.

Further functionality is provided, but not mentioned here because it is not required for the rest of this thesis. Only the configuration options have to be explained in more detail.

Configuration of the MVC is achieved by accessing so called MVC device objects. These device objects are identified by an *Object Identifier (OID)* and can be accessed in three ways:

- **Read:** Information of the device object can be read.
- **Write:** Information can be written to the device object.
- **Trap:** A device object with this access type is some kind of event. Applications can subscribe to the event and if the event occurs, the application is informed.

In theory, any combination of the different access methods is possible. The allowed kinds of access are specified on a per object basis in the *MVC Software Interface Manual* [Int03]. For further details on the MVC this manual should be studied.

Accessing the MVC

The only software that directly interacts with the MVC is the network driver in the uClinux kernel. Like other network drivers, it implements the device dependent network functionality. Again, the only aspect of special interest for this thesis is the configuration of the MVC device objects. Necessary routines needed for configuration of the MVC are implemented in the network driver.

Some user space applications might need to change or read configuration settings of the MVC. Further they might be interested in some of the events generated by the MVC. For this purpose, the *ioctl* system call is used for interaction of the user space application with the kernel space network driver. The network driver then directly accesses the interface provided by the MVC.

A short summary of the necessary steps to configure an MVC device object is provided here. For further details, example source code for accessing MVC device objects via user space application is provided in Section A.2.

First, the user space application needs to set the right value for the device object depending on the datatype and the allowed values for the object. Furthermore, the desired object has to be selected by specifying the right OID. If configuration options should be read, no values besides the OID have to be set, but the memory for the desired values has to be allocated.

After having set up the configuration options, the application creates a socket and receives a file descriptor for this socket. The received file descriptor is then used for the ioctl system call. Further information necessary for this call is the kind of the call and a variable containing some information about the call and the configuration settings for the MVC device object.

If all the necessary information has been provided, the network driver receives the configuration request and performs the necessary steps to configure the MVC device object, read configuration settings or subscribe to a trap.

MVC device objects that represent an event, i.e. trap-able objects, are handled similarly to a write request. First, the application installs a signal handler for a specific signal. Next, it provides the signal number of the selected signal in the configuration request. Each time the event occurs, the application gets signaled with the specified signal.

6.4 WPA Implementation

Most of WPA specific functionality is already implemented in the MVC. The software layers above the MVC only have to configure the MVC appropriately. Configuration options affected by WPA include:

- Enable / Disable encryption
- Set encryption keys: group key, individual unicast key per STA
- Enable / Disable 802.1X
- Authenticate / Deauthenticate a STA: changes the state of the 802.1X controlled port

and some more. Of further interest is, how the authentication of a STA is handled.

6.4.1 802.1X

Figure 6.2 provides some details about the implementation of the 802.1X controlled and uncontrolled port in the MVC.

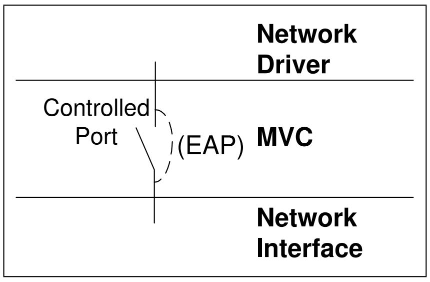


Figure 6.2: 802.1X Implementation in MVC

If a STA is unauthenticated, i.e. the controlled port is open, only EAP messages are allowed to be exchanged between the network interface and the network driver by using the bypass of the controlled port. This bypass can be considered to be the uncontrolled port of 802.1X.

In contrast to the common view of 802.1X, in this case, the paths for controlled and uncontrolled port are united behind the controlled port. Therefore, the network driver is not able to distinguish between network packets received via the controlled port and the uncontrolled port.

After a STA gets authenticated, the controlled port is closed and everything is allowed to be passed between the network interface and the network driver.

6.4.2 Authentication Process

The implementation of WPA functionality in the MVC is limited to MAC level features such as sending, receiving, encryption, decryption, etc. What further needs to be implemented is the logic of the WPA authentication process. Additionally, someone has to be responsible for configuring the MVC appropriately. This is where the *Port Access Entity Daemon (paed)* comes to play.

This daemon subscribes to various traps of the MVC to be notified of STAs that want to associate, disassociate, etc. with the AP. At the time the paed receives an association notification for a STA, it starts the WPA authentication process by sending an *EAP request identity* packet to the STA. Note that the controlled port is in the unauthenticated state for the whole authentication process.

After successful completion of the authentication process, the paed brings the controlled port to the authenticated state and installs the unicast keys for the STA. This is done by configuring the MVC device objects appropriately. After this process, the STA is allowed to access the network resources behind the AP.

6.5 Development Environment

As already stated, the only image that can be developed individually is the one for the main system. The platform of the development host is Linux because of the tools needed for building the image. Several utilities (including a GNU[3] gcc compiler, GNU gdb debugger, ...) for the target platform are included. The target platform is ARM because the AP uses an ARM processor.

The build process is based on the make utility and therefore on makefiles. Hence, building an image of the main system reduces to a simple make command such as ”make image”. If make finishes successfully, i.e. without compilation errors, an image of the main system is created out of the sources.

Basis for the build process are the firmware sources provided by Intersil. They can be divided into three main categories:

[3] GNU is the abbreviation of *GNU's Not Unix*, see `http://www.gnu.org/`

- uClinux Kernel
- GNU GPL[4] applications, e.g. DHCP client and server
- Non-GNU-GPL applications provided by Intersil and available as binaries only

As no source code for the binaries provided by Intersil is available, they are simply included in the image to provide the desired functionality, but cannot be changed.

The main purpose of the APDK is to have an AP that can be adapted individually and enriched with further functionality by adding various applications. Of course, this possibility is limited by the physical resource constraints of the AP, e.g. almost the whole memory is already used.

Details on how to setup the development environment is found in *The uClinux Development Environment* [Int02c].

[4] GNU GPL: GNU General Public License

Chapter 7

New Migration Solution: The Dual-Mode Access Point

Having the Intersil Access Point Development Kit (APDK) available, the goal now was to implement a solution for the issue described in 5.2. The presented solution allows a single AP to authenticate and serve legacy clients as well as clients capable of WPA or 802.11i.

7.1 Problem Description

Before having a look at the final solution, recall the infrastructure and roles used for 802.1X authentication (described in Section 3.4). The entities specific to the implementation are written in braces after the corresponding role.

At first, we have the supplicant (the wireless STA), desiring to access the network. Next, there is the Authenticator (the AP) that authenticates the supplicant. And finally, there is the authentication server (a RADIUS server) that provides the authentication service to the supplicant.

Recall further that 802.1X makes use of a controlled port to protect services and an uncontrolled port on which authentication is performed. Successful authentication allows traffic to pass through the controlled port.

Figure 7.1 shows the setup as used by 802.1X. The controlled and uncontrolled port are shown within the authenticator. Again, the entities specific to the implementation are written in braces after the corresponding role.

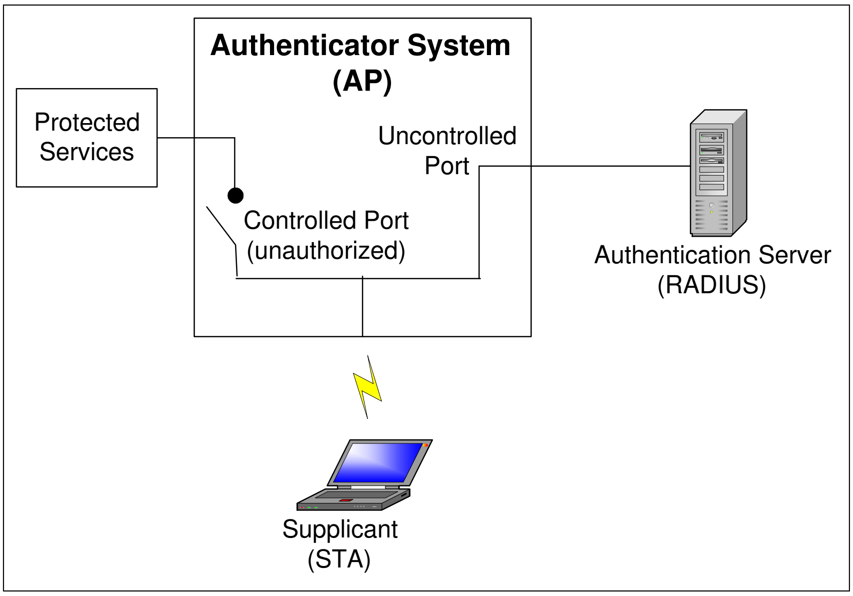


Figure 7.1: 802.1X Revisited

For authentication, the supplicant uses EAPOL to communicate with the authenticator. In our case, the authenticator encapsulates received EAP messages within the RADIUS protocol and transmits them to the RADIUS server.

The only issue with migration to such an infrastructure is the one of legacy supplicants, not capable of 802.1X. These supplicants are not able to authenticate themselves to the authentication server. Therefore, the controlled port is always in the unauthorized state for legacy clients. Hence, they cannot use the protected services.

Note that for the following sections, this thesis will only distinguish between legacy clients and WPA-capable clients. This is a reasonable approach because WPA makes use of 802.1X and is the precursor of 802.11i. Furthermore, WPA implementations are already available, 802.1X will not be successful on its own, and 802.11i is still a draft. Nevertheless, the solution can be used to support all four kinds of clients: legacy devices, devices capable of 802.1X, WPA, and 802.11i.

7.2 Theoretical Solution

This section provides the idea and the theoretical aspects of the dual-mode solution for the issue with legacy devices described in 7.1.

Supporting legacy devices and WPA-capable devices at once can be done by combining the new authentication method (via 802.1X and EAP) with a legacy authentication method, like UAM.

To allow this combination, the authenticator system has to be extended with a switch, further called the *Extension Switch*. This switch is added to the uncontrolled port and used to distinguish between network traffic of legacy devices and WPA-capable devices.

In Figure 7.2, the extension switch is connected to the left contact. Obviously, the supplicant is a legacy device. For WPA-capable supplicants, the extension switch is connected to the right contact, allowing direct access to the authentication server via the EAP protocol.

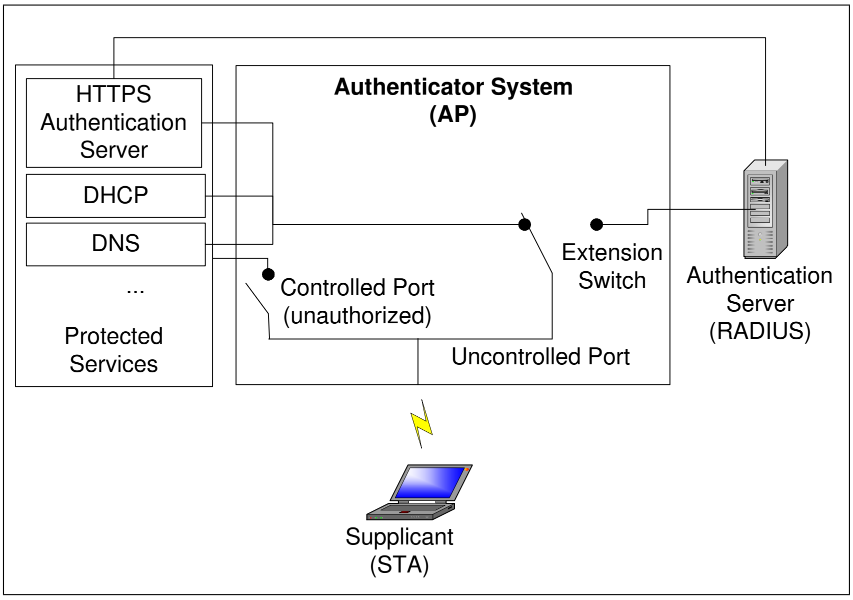

HTTPS Authentication Server
DHCP
DNS
...
Protected Services
Authenticator System (AP)
Controlled Port (unauthorized)
Extension Switch
Uncontrolled Port
Authentication Server (RADIUS)
Supplicant (STA)

Figure 7.2: New Authentication Solution

Furthermore, an HTTPS authentication server has to be introduced to allow UAM. As UAM is the authentication mechanism of today, the HTTPS authentication server might already exist in the target infrastructure.

For WPA-capable devices, the authentication procedure is as usual. The authenticator repackages EAP messages transferred between the supplicant and the authentication server. The state of the controlled port is set depending on the success of the authentication process.

Legacy devices are allowed to use UAM for authentication purposes. For this, the extension switch is changed to allow legacy devices to access the DHCP server, DNS server, and HTTPS authentication server.

DHCP is necessary to provide an IP address to the legacy device. Note that no IP address is necessary for authentication of WPA-capable devices. Furthermore, the DNS service has to be allowed to provide UAM to the legacy client.

The HTTPS authentication server presents an authentication portal to the user of the legacy device. There, the user enters his/her credentials, usually username and password, and submits them. Afterwards, the HTTPS authentication server uses the same authentication server as is used for WPA capable devices for checking the credentials. This allows centralized user management because only one authentication server is responsible for all accounts.

Upon successful authentication via the HTTPS authentication server, the controlled port is set to the authenticated state for the legacy device. Now the legacy device is able to access the protected services.

Note that WPA-capable clients are not allowed to use UAM for authentication.

7.2.1 Example of Legacy Authentication

For an in-depth view of the legacy authentication used in the dual-mode AP, the following example is provided. It is the use case scenario of a user of a legacy device, desiring Internet access via a public hotspot of a WISP (Wireless Internet Service Provider).

The user powers on his/her notebook and the system starts to boot. At the time the legacy wireless network card becomes operable, it tries to detect BSSs in the area. Fortunately, it detects the AP of the WISP and tries to associate with it. As the association request of the legacy device does not contain a WPA IE (Information Element), the AP knows that this is a legacy device. Therefore, it sets the extension switch appropriately and allows association of the STA.

Thereafter, the legacy device tries to get an IP address by using DHCP. This is allowed for the legacy device. Further information such as gateway address or DNS servers is specified in the DHCP response. Hence, the device has all the information that is necessary to successfully reach the Internet. Note that accessing the Internet is one of the protected services behind the controlled port. Note further that the controlled port is still in unauthorized state for this client.

Now the user tries to access the Internet by pointing his/her browser to a specific URL (Universal Resource Locator) , let us assume `http://www.google.com/`. Now the domain name "www.google.com" has to be resolved into an IP address. Therefore, the client sends a DNS request to the DNS server that has been specified in the DHCP response.

It is essential that the client receives the right IP address for "www.google.com" and not the IP address of the HTTPS authentication server. This is necessary because the client might cache the resolved IP address and further access the HTTPS authentication server whenever the user wants to access "www.google.com".

After receiving the IP address for "www.google.com" from the DNS server, the client tries to make a connection to this IP address. Because it is still unauthenticated, the authenticator redirects this request to the HTTPS authentication server. The authentication server in turn presents the authentication portal to the user and remembers the requested web page.

The user enters username and password in his/her browser and submits this information. The HTTPS authentication server takes username and password and uses the authentication server (RADIUS) to check the credentials. If they are valid, the HTTPS authentication server informs the AP of the successful authentication and redirects the browser of the user to the initial requested web page, `http://www.google.com/`. Furthermore, it opens another browser window to allow the user to deauthenticate at the authenticator.

Upon reception of the successful authentication message of the HTTPS authentication server, the AP brings the controlled port to the authenticated state. This allows the following redirection of the browser to the initial URL to be successful. Therefore, the web page of the original request (`http://www.google.com/`) is displayed in the user's browser.

7.2.2 Requirements to the AP MAC Layer

So far, a global view of the dual-mode solution has been discussed, but there are some further requirements to the MAC layer of the wireless network interface of the AP that are inevitable for a complete solution.

The AP needs to support encrypted and unencrypted communication at once - encrypted communication with WPA-capable devices and unencrypted communication with legacy devices. This is no problem for unicast traffic because unicast traffic is encrypted with a different unicast key for each STA. The MAC layer only needs to allow unencrypted unicast traffic with legacy clients.

For broadcast communication, the situation is different. When using WPA, the AP has a group key for encryption of broadcast messages. As it is not allowed to specify a null-key, which results in data being transmitted in the clear, the only option for the AP is to send broadcast messages twice - one time encrypted and one time unencrypted. See 8.1.2 for further discussion on some issues of this approach.

7.3 Infrastructure

This section now provides the information about the hardware, software and network infrastructure that was used for development.

7.3.1 Terminology

- **Authenticated Traffic:** Network traffic produced by a wireless device that has successfully authenticated itself to the authenticator. Furthermore, traffic needed for authentication itself is considered to be authenticated traffic. This includes EAPOL messages from the supplicant to the authenticator for the case of WPA. In the case of legacy devices, DHCP and DNS requests as well as HTTPS authentication traffic is considered to be authenticated.

- **Unauthenticated Traffic:** Network traffic produced by a wireless device before having authenticated itself to the authenticator. An exception to this rule is the traffic needed for the authentication process itself.

7.3.2 Hardware and Network Infrastructure

Development

Figure 7.3 depicts the infrastructure that was set up for development of the dual-mode solution.

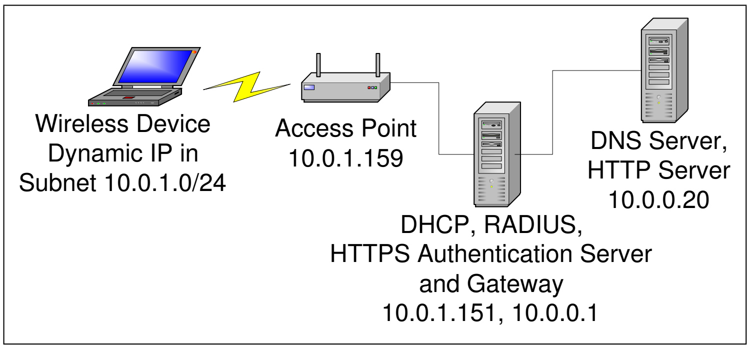


Figure 7.3: Development Infrastructure

The wireless device (the supplicant) was a notebook equipped with a 3Com CRPAG175 WLAN PC card. This card supports legacy mode as well as WPA mode of operation. An access point of the Intersil ISL36356A APDK was used as authenticator system. Two additional Linux machines were used for performing the authentication process.

As depicted in Figure 7.3, the wireless device, the access point, and one Linux machine are in the subnet 10.0.1.0/24. In the subnet 10.0.0.0/24 are only the two Linux computers. The Linux box with IP 10.0.1.151 has also an IP in the subnet 10.0.0.0/24 (10.0.0.1) and acts as a gateway between these two subnets.

For development purposes, the computer with IP address 10.0.0.20 represents the Internet zone, i.e. any traffic to this machine has to come from an authenticated device. The only exception to this rule is DNS traffic that is also allowed from unauthenticated legacy devices.

Although this infrastructure might seem to be nearly the same as in Figure 5.2, page 58, it is pretty different because network traffic from unauthenticated STAs is already dropped by the APs. In the solution in Figure 5.2, the AP forwards everything to the PAC gateway. The next example is provided to point out the difference.

New HotSpot Infrastructure

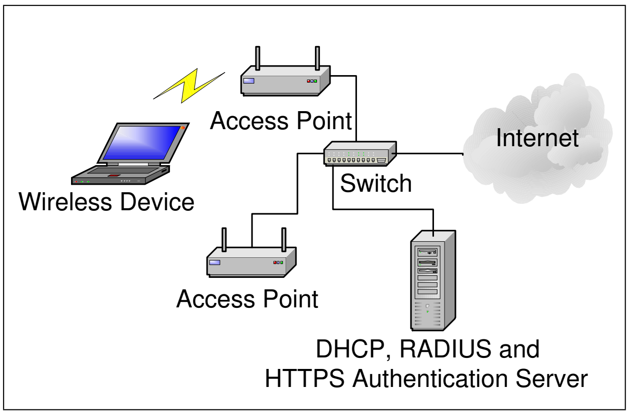


Figure 7.4: New HotSpot Infrastructure Example

The infrastructure in Figure 7.4 is not recommended to be applied one-to-one in real life. It is only used for explaining the differences to the infrastructure provided in Figure 5.2.

In Figure 5.2, network traffic produced by wireless STAs is forwarded by the AP to the PAC gateway unconditionally. The PAC gateway has to decide whether to allow or deny Internet access to the specific STA. This implies that network traffic of unauthenticated STAs also reaches the wired network.

The situation is different for the infrastructure in Figure 7.4. Unauthenticated traffic is already dropped by the access points and never reaches the wired network. Therefore, it is sufficient to directly connect the APs to the Internet. However, it would be wise to place at least a firewall in between - but this has been left out for better comparison with Figure 5.2.

Real World Example

The example in Figure 7.5 provides a network infrastructure that can be used for enterprise environments as well as for hotspot operators. Note that for hotspots, the Intranet of course should not be the Intranet of the WISP but might be considered as the part of a network that connects several hotspots together.

We already know that for the new solution, no unauthenticated traffic ever reaches the wired network. Therefore, it is sufficient to directly connect the APs to the Intranet. No firewall is needed between the APs and the

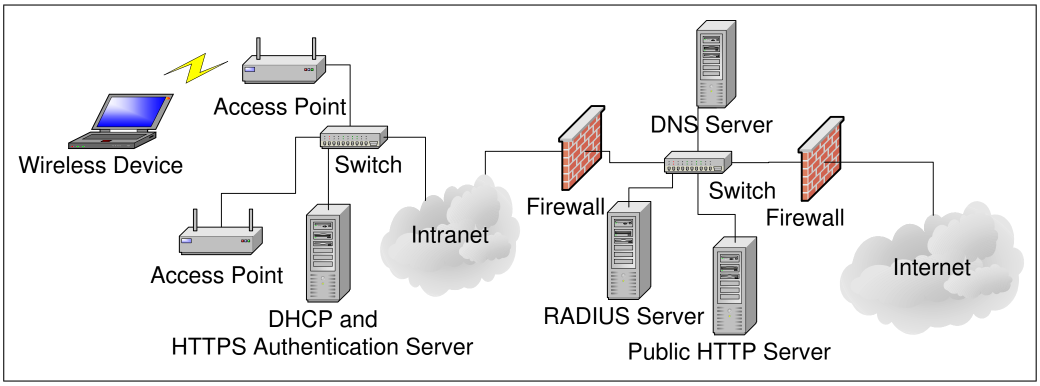


Figure 7.5: Real World Infrastructure Example

Intranet. This is a contrast to the recommendation before 802.1X or WPA was integrated into WLAN.

This might only be partly true if the Intranet contains services that cannot be allowed to be accessed by legacy devices. In that case, either place a firewall between the APs and the Intranet or configure the APs themselves appropriately.

Besides the APs, there is also a server for the DHCP service and HTTPS authentication in the same Ethernet broadcast domain. This allows centralized management of IP addresses at this special server without having to configure a separate IP range for each access point. If each AP contains a DHCP server, serving a separate IP subnet, this removes the link layer mobility feature of WLAN. In this case, routing has to be done between STAs associated with different APs.

The other infrastructure, including the DeMilitarized Zone (DMZ) between the two firewalls, should be already existent. Shortly explained, all the services reachable by both, Intranet and Internet, as well as services only reachable via the Internet should be placed in the DMZ. The DMZ itself is protected by two firewalls, one at each side, against Intranet and Internet. Furthermore, the Intranet can be considered to be protected by two firewalls against the Internet.

7.3.3 Software Infrastructure

Now follows the description of the software infrastructure for the development environment provided in Figure 7.3.

Wireless Device

The wireless device runs Windows XP. For management of wireless interface related issues, 3com includes tool for the WLAN card. This tool allows the selection of the authentication mechanism. Two different mechanisms were tested successfully for the case of WPA: EAP-TLS and EAP-TTLS. For testing legacy authentication, the authentication mechanism of the 3Com WLAN tool was set to ”no authentication”. No further software besides an Internet browser is needed.

Access Point

For the initial setup, there is nothing to install at the access point.

Gateway

Routing between the two subnets, 10.0.0.0/24 and 10.0.1.0/24, is one of the responsibilities of the gateway. This is achieved by configuring the routing tables appropriately. The gateway further provides a DHCP service to the wireless devices and is the RADIUS server with which the access point communicates for authentication of WPA-capable clients. Furthermore, it operates as the HTTPS authentication server.

Software installed at the gateway to provide necessary services for authentication and network operation:

- GNU/Linux as operating system
- DHCP server of the Internet Software Consortium
- Apache HTTP server with mod_ssl and mod_php for implementing the the HTTPS authentication portal.
- FreeRADIUS as the RADIUS server

Almost all the required software is already provided by many popular Linux distributions. If FreeRADIUS is not included, see `http://www.freeradius.org/` for the sources.

DNS / HTTP Server

The whole software needed for this server is already provided by Linux distributions. Any DNS server and HTTP server is sufficient.

7.3.4 Certificates

To be able to use EAP-TLS and EAP-TTLS as authentication mechanisms, certificates have to be created and installed for the RADIUS server and in the case of EAP-TLS for the supplicant as well. For development purposes, fully functional certificates can be created with the OpenSSL[1] toolkit. It is not necessary to get an official certificate from a *Certification Authority (CA)* like VeriSign.

The process of creating and installing certificates is only summarized here at a basic level. For in-depth information on how to setup EAP-TLS with FreeRADIUS and Windows XP as supplicant see [Ros02].

First of all, a self-signed certificate needs to be created that is used for the root CA. Next, two other certificates are generated - one for the supplicant (Windows XP) and one for the authentication server (FreeRADIUS). Both certificates have to be signed by the root CA.

The server certificate must contain the server authentication certificate purpose in the *Enhanced Key Usage (EKU)* extension of the certificate[2]. Similarly, the client certificate needs to contain the client authentication certificate purpose[3] in the EKU extension of the certificate.

Finally, the certificates need to be installed in the FreeRADIUS server (root CA and server certificate) and the Windows XP supplicant (root CA and client certificate).

Note that for the case of EAP-TTLS, the client certificate is not necessary.

7.4 Implementation of the Dual-Mode AP

Chapter 6 already provided an overview of the Intersil APDK, primarily about aspects related to this thesis. Therefore, this section only deals with

[1] `http://www.openssl.org/`

[2] Server authentication certificate purpose is specified by using Object Identifier (OID) 1.3.6.1.5.5.7.3.1

[3] Client authentication certificate purpose is specified by OID 1.3.6.1.5.5.7.3.2

the enhancements that needed to be implemented for the dual-mode access point. Furthermore, the implementation of the HTTPS authentication portal is described.

7.4.1 Distinguishing WPA and Legacy Equipment

To successfully apply the dual-mode solution, it is absolutely necessary to distinguish between WPA-capable devices and legacy devices. This is accomplished by examining the association request of a specific STA. If such an association request contains a WPA IE, the requesting STA is WPA-capable. Therefore, legacy authentication is forbidden for this STA. If no WPA IE can be found, a legacy device tries to associate and UAM has to be enabled for this STA.

Two different scenarios follow, one for each kind of authentication. Both scenarios start at the time a STA tries to associate with the AP. Recall that before association, open system authentication is necessary. Note that open system authentication only happens at the MAC layer and has nothing to do with the enhanced authentication of WPA.

WPA Authentication. At the time the AP receives the association request of the STA, the paed is notified of this event by the MVC. After that, the paed examines the association request and - as the STA is WPA-capable - is able to find the WPA IE. This implies that legacy authentication is forbidden for this specific STA and WPA authentication has to be used. Therefore, WPA authentication, as already described in 4.1.1, follows.

Legacy Authentication. After notification of the association request, the paed is not able to find the WPA IE in the request. Hence, it takes the necessary steps (see 7.4.2) to allow UAM for the legacy STA.

As the AP already supports WPA, no further implementation was needed for this authentication type. For the case of legacy authentication, the changes needed are described in the following example authentication scenario. The required changes were implemented as an extension to the existing paed.

7.4.2 Legacy Authentication Sequence

After the paed detects the association of a legacy device, it sets the following restriction rules for the STA by configuring the kernel netfilter via iptables:

1. Allow the DHCP service.
2. Allow the DNS service.
3. Redirect any HTTP and HTTPS requests to the HTTPS authentication server.
4. Allow network traffic to the HTTP and HTTPS port of the HTTPS authentication server.
5. Drop any network packets received from the STA that are not explicitly allowed by one of the previous rules.

For performance considerations, it is recommended to implement the following policy:

1. Explicitly allow for all STAs:
    - the DHCP service
    - the DNS service
    - network traffic to the HTTP and HTTPS port of the HTTPS authentication server.

   at the startup of the paed.
2. For legacy STAs that are associated but not authenticated:
    - Redirect any HTTP and HTTPS requests to the HTTPS authentication server.
    - Drop any network packets received from the legacy STAs that are not explicitly allowed by one of the previous rules.

Note that no special restriction measures have to be taken for WPA-capable STAs. Unauthenticated traffic of these STAs is already dropped due to the 802.1X controlled port mechanism in the MVC. For legacy STAs, these restriction rules have to be set because of the following issues:

After applying the aforementioned restriction, the paed brings the 802.1X port of the legacy STA to the authenticated state. Therefore, it allows any network traffic between the network interface and the network driver. Because of the applied restrictions, the STA is not able to use anything else than the secure authentication service provided by the HTTPS authentication server.

Now the user of the legacy STA authenticates by using the authentication service provided by the HTTPS authentication server. After successful authentication, the AP needs to be informed by the HTTPS authentication server of the success.

Upon notification of the HTTPS authentication server, the AP removes the following restrictions for the authenticated STA:

1. Redirection of HTTP and HTTPS requests to the HTTPS authentication server.
2. Dropping of network traffic received from the legacy STA that is not needed for authentication purposes.

After these restrictions are removed, the STA is allowed to fully access the network behind the AP.

7.4.3 AP specific Implementation

PAED Extension

The paed was already able to detect the WPA IE and therefore to distinguish between WPA-capable devices and legacy devices. This mechanism only had to be extended to support the restriction of legacy devices.

Restriction of the client was achieved by creating a child process and calling the iptables user space tool with appropriate arguments for configuring the netfilter. The used iptables firewall rules are provided in B.1.

Furthermore, the AP needs to listen for incoming notifications of the HTTPS authentication server about the authentication status of individual STAs. It was decided to further enhance the paed by this functionality to have one single application for supporting the whole authentication process.

Finally, the restrictions of STAs have to be removed at the time the user of the STA authenticates successfully. Implementation of this functionality was also achieved by calling iptables with appropriate arguments.

Kernel Upgrades

No proprietary extensions were needed for the uClinux kernel, but a patch had to be applied. Iptables - as the name implies - is used for configuring firewall rules at the IP level, and therefore at the network layer. This works fine for most applications, but is not sufficient for bridges because they are located one level below at the data link level. Therefore, the netfilter does not recognize traffic that is bridged between the wired and the wireless interface.

Therefore, a mechanism is required to let the netfilter "see" the traffic of bridges. A Linux kernel patch that provides this functionality is already available. It is called bridge-nf and is found at `http://ebtables.sourceforge.net/`.

Unfortunately, the aforementioned patch is for the usual Linux kernel and not for the uClinux kernel. The Intersil AP comes with a uClinux kernel version 2.1.17 but the earliest bridge-nf patch provided is for the usual Linux kernel version 2.4.18. Fortunately, the patch can be applied to the uClinux kernel without major problems. Small incompatibilities can be removed quickly.

After applying the patch, network traffic that would otherwise be forwarded between the interfaces of a bridge, can be dropped due to some iptables firewall rules.

Configuration Options

Some new configuration options had to be introduced for the legacy authentication solution:

- The IP address of the HTTPS authentication server.
- Specification of two DNS servers to which access should be allowed to an associated but unauthenticated legacy STA.
- A TCP port to listen for incoming messages of the HTTPS authentication server.

7.4.4 HTTPS Authentication Portal

PHP[4] was used to implement the HTTPS Authentication Portal. It provides a simple interface to the user for entering username and password. For val-

[4] `http://www.php.net/`

idation of these credentials at the FreeRADIUS server, a radius extension[5] for PHP has to be installed.

The HTTP server is configured to provide the default authentication page for any request not matching a file on the HTTP server. Furthermore, only the files needed for the authentication portal are stored on the HTTP server. Therefore, if the request for a specific file on a specific webserver is redirected by the AP to the HTTPS authentication server, the authentication portal is provided.

For incoming requests the to the main page of the PHP authentication portal, the following steps are performed:

1. If username and password are not provided:
   (a) If the desired host of an HTTP/1.1 request is different to the authentication portal → Remember the following information:
      - Host of the HTTP/1.1 request.
      - File on the host.
      - Whether the host should be accessed via HTTP or HTTPS.

   (b) If the HTTPS authentication server is accessed via HTTP:
      - Redirect to the HTTPS port of the HTTPS AS.

   (c) Provide an input form for username and password.
2. If username and password are provided → Check the validity of these credentials at the FreeRADIUS server.
   (a) If the credentials are valid:
      i. Inform the AP of the success (the AP removes the STA restrictions).
      ii. Redirect the user to the web page requested in 1a, or show a success page in case the information about the initial request is not present (due to having an HTTP/1.0 client or for other reasons).
      iii. Open a separate deauthentication page, showing a *Logout* button to enable the user to deauthenticate his/her STA at the AP.

   (b) If the credentials are invalid:

[5] see `http://pecl.php.net/package/Auth_RADIUS`

i. Display an error page and again the form for input of the credentials.

The deauthentication mechanism provided by 2(a)iii above prevents malicious users from continuing other user's sessions. Session continuation could be achieved by simply setting the MAC address of the attacker's network device to the MAC address of the previous (legitimate) user's network device. Furthermore, the right network settings have to be known by the attacker. These might be discovered via a network sniffer, or by simply using DHCP. See Section 8.2.2 for further discussion.

For incoming requests to the deauthentication page of the PHP authentication portal, the STA that produced the request is deauthenticated. To accomplish this task, the deauthentication page of the portal informs the AP of the deauthentication. The AP in turn deauthenticates the STA. See 8.3.1 for further considerations about the deauthentication mechanism.

IP to MAC Address Resolution

Security related functionality of the AP only operates at the MAC level. Therefore, the legacy authentication implementation provided by this thesis operates at the MAC level too.

The web application of the HTTPS authentication server is only able to get the IP address of the remote client. This introduces the necessity to resolve the IP address seen by the authentication web portal into the MAC address of the STA accessing the portal.

As only one DHCP server is allowed per subnet, it is placed on a separate computer, and not within an AP. Therefore, the APs of this subnet do not have the knowledge of which IP addresses are assigned to which MAC addresses.

Hence, the decision was to do the IP to MAC address resolution in the HTTPS authentication portal and to place it on the same computer that serves as DHCP server. Thereafter, resolution of the MAC address is done by looking at the leases file of the DHCP server. For a given IP address, the MAC address of the latest valid lease is returned.

This MAC address is provided to the AP in the authentication and deauthentication messages. Given this MAC address, the AP is able to authenticate or deauthenticate the appropriate STA.

7.4.5 Not achievable Goals

There is one special issue with the MVC of the Intersil AP that prevented the implementation of a fully functional dual-mode access point. PrivacyInvoked is the configuration parameter that specifies whether the supported confidentiality mechanisms should be used. Supported confidentiality mechanisms are WEP, 802.1X and WPA.

Enabling the confidentiality mechanisms results in every network data traffic besides EAPOL messages to be encrypted. Disabling them results in no encryption at all. Using no encryption excludes the aforementioned confidentiality mechanisms, whereas using encryption excludes legacy devices. This is the case because the MVC does not support to specify individual STAs with which communication should be unencrypted.

A workaround to this problem is not possible because the whole encryption related stuff is implemented in the MVC and can only be influenced via configuration parameters. Further, it is not possible to bring the MVC into a dummy mode and reimplement the functionality in the network driver.

This constraint violates one of the requirements mentioned in 7.2.2. Further, the requirement for broadcast messages cannot be fulfilled but is of no further interest because already the unicast communication fails.

As WPA-capable STAs can be distinguished from legacy STAs by the AP, it is still possible to use different authentication mechanisms for each of them. The only constraint is that only one authentication mechanism is supported at one time. Enabling PrivacyInvoked means WPA is used while disabling PrivacyInvoked allows UAM.

7.5 Configuration of the Environment

This section sums up the needed configuration settings for the authentication solution within the described development environment. The concrete configuration files or configuration settings are not provided here. Only the necessary steps are described.

7.5.1 FreeRADIUS

For the FreeRADIUS server, the appropriate authentication mechanisms have to be enabled. These were EAP-TLS and EAP-TTLS for the case of this thesis. In order to enable these two, the root and server certificate as created in 7.3.4 need to be installed by configuring the settings for EAP-TLS appropriately. These settings are found in the file `radiusd.conf` of the FreeRADIUS configuration directory.

Furthermore, the RADIUS clients have to be defined. At least two clients are needed: the AP and the HTTPS authentication server. RADIUS client configuration is found in `clients.conf`.

At least one user account has to be created for being able to perform authentication. These changes have to be made in the file `users.conf`.

7.5.2 HTTPS Authentication Server

Two configuration settings are required: to enable SSL and to set the HTTP 404 error document to the main page of the authentication web portal. Note that for a production environment, the certificate used for HTTPS needs to be issued by a known CA. A self-created certificate is sufficient for development.

Authentication Portal

Configuration options of the authentication portal:

- IP address of the AP.
- TCP Port of the AP to which to send the authentication and deauthentication messages.
- IP address and TCP port of the RADIUS server.
- Secret for the RADIUS server.
- Path to the leases file of the DHCP server.

7.5.3 Access Point

The implementation of the legacy authentication mechanism at the AP introduces three new configuration parameters:

- IP addresses of two DNS servers.
- IP address of the HTTPS authentication server.
- Port on which to listen for messages of the authentication portal.

Furthermore, a timeout parameter for deauthenticating STAs at the MAC level, that was already implemented, was made configurable. This parameter is interesting for fine tuning between keeping legal sessions open while preventing malicious people from continuing other people's session. Further discussion on this topic is found in 8.2.2.

7.5.4 Wireless Device

Configuration of the wireless device reduces to selecting the right authentication mechanism. Used mechanisms were EAP-TLS, EAP-TTLS and "No Security" for UAM.

For EAP-TLS the root and client certificate as created in 7.3.4 need to be installed. [Ros02] describes the necessary steps for installation of the certificates. EAP-TTLS only needs the installation of the root certificate in order to avoid warnings about the unknown CA.

Chapter 8

Evaluation and Further Work

8.1 Common Considerations

The presented solution is achieved by combining authentication methods for WPA-capable devices with UAM as the authentication method for legacy devices. All the necessary changes are possible to do in software. Therefore, a migration to dual-mode operation does not need additional hardware but can be achieved via firmware updates of access points.

Saving costs for hardware upgrade is of especial interest for companies owning many access points. Nevertheless, the necessary firmware has to be provided by the access point manufacturers.

Cases might exist where a simple firmware upgrade is not sufficient. Still an integrated solution for WPA and legacy authentication is cheaper than to provide different access points for different authentication methods or using VLAN technology. As already mentioned above, doubling the APs might also lead into interference problems and performance decrease.

Furthermore, the only new component in the infrastructure is an HTTPS authentication server. The other components (RADIUS server, DHCP server, etc.) will likely exist in enterprise scale networks. Another advantage of the new dual-mode access point solution is that it allows to use the same network paths for legacy devices and devices with newest security features.

Besides these advantages, some issues arise because the described dual-mode solution is not fully supported by the standards. Incompatibilities are introduced by the requirements to the AP MAC layer described in 7.2.2. An evaluation of these requirements is provided in the two following subsections.

8.1.1 Unicast Communication

The requirement for unicast communication is that the AP is able to encrypt communication with some STAs while doing unencrypted unicast communication with other STAs. With the introduction of 802.1X, an AP can use different unicast keys for each associated STA.

Setting the encryption key for a STA to the null-key would result in no encryption of the packet but results in an error as specified in [IEE99]. For dual-mode operation, the AP should either allow this null-key or provide another mechanism to transmit frames unencrypted for legacy devices.

Providing this possibility does not violate any security guarantees to users of devices capable of 802.1X, WPA or 802.11i. These devices are further called *enhanced security devices*. Unicast traffic with STAs of this type is still decrypted with a different per-STA key. Users of legacy devices have had no encryption in the past and will have no encryption in the future. They have to be aware and live with the security restrictions of their devices.

8.1.2 Broadcast Communication

The situation is different for broadcast communication because only one group key is used for communication. STAs, that know the group key, are able to receive and send broadcast messages.

To take part in network communication, it is absolutely necessary to be able to receive and transmit broadcast messages. An example for this requirement is the *Address Resolution Protocol (ARP)* that resolves the MAC address for a given IP address. Therefore, the requirement specified in 7.2.2 for broadcast messages is to send broadcast messages twice, one time encrypted and one time unencrypted.

The issue with this approach is that STAs capable of 802.1X, WPA or 802.11i cannot be sure that their broadcast communication is not received by an unauthorized STA. To limit the amount of information that can be gathered by an unauthorized STA, only network packets of some essential protocols should be broadcasted in the clear. Limiting the cleartext broadcast traffic to the ARP protocol should be sufficient for correct network operation.

While this constraint improves the confidentiality of enhanced security devices, it might limit the kinds of applications that are able to run on legacy devices.

Further investigation could be done to provide a list of applications that do not run on legacy devices, whose broadcast communication is limited to ARP.

8.2 Design and Development Decisions

8.2.1 Placing the HTTPS Authentication Server

Principally, there are two possibilities where to place the HTTPS authentication server: externally or internally to the authenticator system. If the HTTPS authentication server already exists, the external solution that has been described in this thesis might be preferred.

Figure 8.1 shows the architecture if an internal HTTPS authentication server is used. See Figure 7.2 on page 77 for the external case.

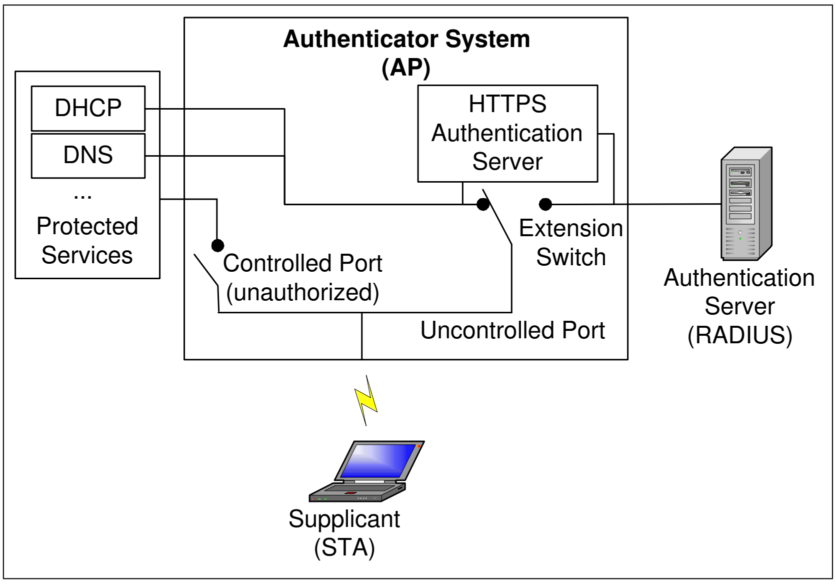


Figure 8.1: Architecture with internal HTTPS Authentication Server

If it is still to be decided whether to include the HTTPS authentication server within the AP, the following considerations have to be taken.

- If the DHCP server is placed outside the AP but the HTTPS authentication server is included in the AP, the AP does not know about the IP address assignments but needs to resolve IP addresses to MAC addresses. This can be handled in three ways:

1. The HTTPS server is implemented manually (or at least an extension) and allows the web application to receive the MAC address of the remote client.
2. The web portal or the AP uses the ARP protocol to resolve IP addresses by itself.
3. The AP listens for DHCP requests and responses and keeps track of the IP assignments by itself. This approach needs much processing power of the AP and an individual implementation of this feature.

- AP legacy authentication part and HTTPS authentication web portal closely interact with each other. As the messages from the HTTPS authentication portal authenticate or deauthenticate STAs, it is crucial that the AP is able to verify the sender of the message. Messages with a different sender than the HTTPS authentication web portal must not be accepted.

  How the authentication portal and the AP establish a trusted and secure relationship to achieve the aforementioned goal is not provided in this thesis. Various possible solutions already exist. One possibility is to use certificates and TLS[1] to establish this relationship. See 8.3.2 for further discussion.

  If the HTTPS authentication server is placed within the AP, a secure communication channel between AP and HTTPS authentication server is neither needed nor is useful.

- The resources on an AP are limited. Therefore, an authentication portal with many graphics or other memory intensive additions is either not possible or means to place these resources externally on a separate server.

- Including the HTTPS authentication server in the AP allows to have only one common interface to the main authentication server.

- Having only one external HTTPS authentication portal for many APs means changing only one portal in contrast to updating the portals of all the different APs.

[1] TLS (Transport Layer Security): is the successor of SSL

8.2.2 Session Continuation

The AP makes use of an idle timeout after which inactive STAs are deauthenticated. There is only one issue with the idle timeout. After a user stops networking, e.g. leaves the hotspot area, the possibility exists that other users make use of the authenticated state for the previous user's STA. This process is further called *session continuation*.

Session continuation can be achieved by setting the MAC address of the attacker STA to the MAC address of an authenticated STA that just left the BSA of the AP without deauthentication. This might happen due to a user who, for example, just moved out of the hotspot area. For the attack to be successful, the MAC address has to be changed within the time the idle timeout has not elapsed for an authenticated STA. After a successful attack, the attacker is able to use the network as long as desired.

Two mechanisms can be used to reduce the probability or prevent this attack:

1. The idle timeout of the AP has to be fine tuned between deauthentication of legitimate users before they actually have stopped networking and leaving sessions open for too long, which increases the probability of a successful attack.

   Note that WLAN devices do not only transmit data frames but also control and management frames. These frames should also reset the idle timeout counter of the AP. Therefore, it should be possible to set the idle timeout to only a few seconds. However, one might have to experiment with this parameter for getting the best results.

2. The deauthentication mechanism described in 7.4.4 allows the legitimate user to deauthenticate his/her STA at the AP. With this mechanism, a STA is deauthenticated immediately, i.e. the session is closed and no further use can be made of it.

The deauthentication mechanism should never be omitted, because it is possible for an attacker to set the MAC address of his/her STA to the MAC address of the victim STA before the victim STA has actually finished networking.

Undefined results occur if the attacker does this MAC address setting in combination with assigning his/her device an IP address and further activates his/her device. It is likely that both, the attacker and the legitimate user, are not able to use the network any further.

Note that this attack is not a WLAN specific attack but can be applied to wired networks too. If WPA or 802.11i are in use, this MAC spoofing attack against the victim STA is not possible because it can be detected by the AP due to the attacker having no appropriate key for encryption. In this case, the network behind the AP is not affected by the attack. The frames of the attacker are already dropped by the AP.

8.2.3 HTTPS Authentication Portal

Some features of the HTTPS authentication portal need special considerations because of issues that arise due to different browser behavior or the redirection mechanism.

Redirection

Principally, two mechanisms exist to achieve the redirection of a browser to the secure HTTPS Authentication portal without using a scripting language such as javascript:

1. Redirection via HTTP by sending an HTTP response with error code 302. This indicates ”Resource moved temporarily” and provides the new location in the `Location:` entry of the HTTP header of the response.
2. Redirection via HTML by specifying `<meta http-equiv="refresh" content="[seconds]; URL=[url]" />` in the `<head>` section of the HTML document.

Both mechanisms achieve their goal quite well. Nevertheless, a link to the desired location should always be provided in the body of the HTML document, just for the case a browser does not work as expected.

Redirection via HTTP is faster because browsers do neither display nor interpret the content that is received in an HTTP 302 response. They only take the provided location of the `Location:` entry in the HTTP header and immediately redirect to this URL.

This mechanism leads to two problems:

1. As the provided content of the HTTP 302 response does not get interpreted, scripting code for the client provided in the response does not become effective.

2. Redirecting the initial request of the user to the HTTPS authentication portal via HTTP 302 leads to a problem with the Opera browser. Opera takes the location of the HTTP 302 response and connects to the authentication portal. After authentication it sends a conditional HTTP GET request to the server of the initial request by specifying `If-Modified-Since:`.

   As most of the time the web page on the desired server will not have changed in the short time of authentication, Opera provides the cached content. This is the authentication portal because of the first redirection.

Due to the aforementioned issues for redirection via HTTP, the second mechanism via HTML was used in the prototype for redirection.

Redirection via HTML is slightly different. The provided content of an HTTP response gets interpreted and displayed for as many seconds as are specified in the content attribute of the `<meta>` element in the HTML `<head>`.

Because this kind of redirection is the only one where the content of the HTTP response gets interpreted, redirection after successful authentication has to be done by this mechanism. This is required because a script needs to be executed in the browser to open the deauthentication window that shows the logout button for deauthentication purposes.

Deauthentication Window

In order for the deauthentication window to pop up after successful authentication, javascript needs to be enabled by the browser and no popup blocker should be installed. A popup blocker such as integrated in the Mozilla web browser prevents the deauthentication window from appearing.

To handle this issue, in the prototype implementation, a link to the deauthentication window is already provided in the main page of the authentication portal. This link allows the user to open the deauthentication window in advance.

Another solution to the popup issue is to return the content of the deauthentication window as HTTP response after successful authentication. Furthermore, a window with the URL of the initial request is opened by using a scripting language.

In this second approach the content of the existing window and the popup window are simply exchanged. Therefore, the deauthentication page is always shown, regardless of whether popup windows are allowed or javascript is enabled.

Server Certificate

The HTTPS authentication server needs to have a valid server certificate. Nevertheless, if the initial request of the user of a legacy device is not an HTTP request but an HTTPS request, a browser warning about the certificate will be the case.

This browser warning cannot be prevented because when sending the initial request, the STA is still unauthenticated at the AP. Therefore, every HTTP- and HTTPS-request gets redirected to the HTTPS authentication server. The browser now thinks that it is accessing the server of the initial request while the certificate shows that this is not the case. Due to this situation, a browser warning will appear.

To avoid this situation, redirection of unauthenticated legacy STAs should only be done for HTTP-requests, but not for HTTPS-requests. Thereafter, HTTPS requests of unauthenticated legacy STAs with a destination other than the HTTPS authentication server are unsuccessful.

HTTP Proxy Requests

Special care has been taken to be able to appropriately handle HTTP proxy requests received via the HTTP port. This works as long as the received proxy request tries to access a remote resource via HTTP by using the default HTTP port 80 for the proxy server. In case of the HTTPS protocol, the request cannot be handled by the HTTPS authentication portal.

If the client has configured a different port for the proxy server than the default HTTP port, nothing will be shown to this client. This is because of the restrictions that apply to legacy clients. Recall that only accessing the HTTP and HTTPS port is possible.

Generally, no HTTP proxies should be used in hotspot environments. Therefore, the issue with proxy requests should not affect WISPs. But enterprises may want to use proxy servers and may want to configure the proxy server to a different port than the default HTTP port.

Therefore, as a further enhancement of the prototype implementation, an additional port for proxy requests could be allowed to be accessed on the

HTTPS AS. Alternatively, the port for HTTP could be made configurable. A different port needs to be supported by the HTTPS AS as well as the AP.

8.3 Attacks against the Dual-Mode Solution

This section only deals with attacks that are specific to the presented dual-mode authentication solution and its prototype implementation. Other common attacks, no matter whether they are WLAN specific, are not provided here.

8.3.1 DoS Attacks

In order to prevent simple DoS attacks against legitimate users, deauthentication of an authenticated association is only allowed via HTTPS for legacy devices.

If deauthentication via HTTP would be allowed, malicious users could deauthenticate other users by sending a deauthentication request to the HTTPS authentication server. This deauthentication request would need to have the source IP address of the legitimate user. Such a request could be performed by implementing a TCP spoofing attack. Implementation of this attack is far more complicated in case of using HTTPS.

A TCP spoofing attack is possible for HTTPS in theory because the attacker and the victim are in the same subnet. Therefore, the attacker is able to receive responses from the HTTPS authentication server for forged packets with faked IP addresses. Furthermore, HTTPS does not use mutual authentication - only the server is authenticated to the client.

Allowing only HTTPS for deauthentication makes an attack harder to be performed, but not impossible. To prevent deauthentication of a legitimate user by a malicious attacker, further identification of the client is needed during the deauthentication process. This identification could be achieved by using the same credentials as are used for the authentication process, i. e. username and password. Only if the provided credentials are valid, deauthentication is allowed.

Whether to apply this additional measure is a trade-off between additional security, usability and customer satisfaction. As the probability for such an attack is low and the result of an attack is only the deauthentication of a user, this additional identification has been omitted in the prototype implementation. The effect of this decision is that the user only needs to press a Logout button for deauthentication but does not have to enter username and password again.

8.3.2 HTTPS AS and AP Communication

The prototype implementation of the dual-mode solution uses TCP/IP for transmission of the (de)authentication messages from the HTTPS authentication server to the access point. This violates the requirement mentioned in 8.2.1 that the AP and the HTTPS AS have to establish a trusted and secure relationship, so that the AP is able to detect whether the received message has really been sent by the HTTPS AS.

Establishment of such a relationship is not a matter of research but of implementation. Therefore, it was considered sufficient for the prototype to limit the security mechanism at the AP to verifying the IP address of the sender. Any packets with another source IP than the HTTPS authentication server are dropped.

As TCP is used in the prototype implementation to transmit the (de)authentication messages between the HTTPS AS and the AP, simple IP spoofing attacks are not possible with the implementation. Nevertheless, TCP spoofing attacks are possible. Therefore, the establishment of a secure communication channel between the HTTPS authentication server and the AP cannot be omitted in a production environment.

8.4 Prototype Enhancements

As written in 7.4.5, it was not possible to allow 802.1X (and authentication mechanisms based on it) and UAM to be performed at the same time due to some MVC restrictions. To remove this restriction, the MVC has to be adapted to satisfy the requirements provided in 7.2.2. As no sources for the MVC are available, this means to get Intersil to implement the extension.

For some services, it might be desired that they are only accessed via WPA capable or RSNA-capable STAs. Therefore, future enhancements could include additional configuration parameters for disallowing access to some services for legacy devices.

Currently, the configuration parameters for the prototype implementation can only be set by changing the configuration file directly. This can be done before the image of the main system of the AP is created, i.e. before compile time, or after compile time by remote login to the AP. In order to deliver a full, customer usable solution, configuration should be possible via web interface or SNMP (Simple Network Management Protocol).

Furthermore, the aforementioned trusted and secure communication channel between the AP and the HTTPS authentication server should be implemented.

After implementation of these additional features, a fully functional dual-mode access point solution exists that is ready to be deployed in a production environment.

Chapter 9

Summary and Conclusion

802.11 network technology (WLAN) may be considered as one of the key technologies for the present and the future. Although useful, it still suffers some problems and is still undergoing active development. One of the issues of WLAN is its security.

An analysis of the current security situation of WLAN shows that although WEP is considered inherently insecure, reasonable security can be achieved by using additional security mechanisms such as 802.1X or WPA. While 802.1X allows secure authentication and dynamic update of WEP keys, WPA further introduces TKIP as a new cryptographic algorithm for providing better confidentiality.

With WPA, a security solution that should satisfy most needs is already available. Nevertheless, it is only the forerunner of 802.11i that is considered to be the final security solution for 802.11 networks.

Migration to the new security mechanisms WPA or 802.11i is a special challenge for WISPs and larger enterprises. The network infrastructure of these environments needs to support both, pre-RSNA and RSNA-capable equipment, at least for a certain period of time. A sudden upgrade to 802.11i is not possible for WISPs as they have to meet their customers needs and cannot force their customers to upgrade their devices.

A new solution to this challenge without doubling hardware or having to use VLAN technology was presented in this thesis. The solution introduces a dual-mode access point that is able to serve pre-RSNA and RSNA-capable equipment at the same time. This goal is achieved by combining the new authentication methods of WPA and later 802.11i with UAM as the authentication method for pre-RSNA devices.

Although the presented dual-mode authentication solution removes the need for further hardware, it imposes some new requirements to the AP MAC layer. The requirement that broadcast messages for legacy devices need to be sent in the clear is the only one that indirectly also affects RSNA-capable devices.

As the possibility exists to limit cleartext broadcast network traffic of the AP to the ARP protocol, this should be a minor disadvantage than having to buy additional hardware, worrying about interference problems, or to change network infrastructure.

Appendix A

Intersil APDK

This chapter deals with information about the Intersil APDK that might not be found somewhere else in the user manuals of the APDK.

A.1 Build Configuration

A.1.1 Reconfiguration of the Main System Image

During development of the main system image it is sometimes necessary to change the configuration of the main system. Most of the time this is done to reconfigure the applications that are included in the main system.

If the image of the main system has already been built before, it is necessary to keep to the following steps to reconfigure and rebuild the image:

```
make clean
make menuconfig
make image
```

It is absolutely necessary that *make clean* is executed before *make menuconfig*. Executing *make clean* after *make menuconfig* or skipping *make clean* at all will lead to a corrupt main system image.

Note that if only changes to the source code of an application have been made, no *make clean* is necessary. Executing *make image* is sufficient to produce a correct image. Skipping *make clean* in this case is recommended as the time needed for building the image decreases significantly after a previous build.

A.1.2 Adding individual Applications

Sometimes it is required to add new applications and include them in the main system image. The steps needed to achieve this goal are:

1. Create a new subdirectory in the $ROCKBASE[1]/apps directory, e.g. $ROCKBASE/apps/my_application.

2. Put the sources of your application in the directory created above.

3. Provide an appropriate Makefile in this directory for compiling your sources. How this Makefile should be written can be found out by looking on Makefiles of other applications. A simple example is provided below.

4. Add an appropriate entry to the file $ROCKBASE/config.in. This file is read by *make menuconfig* for displaying the configuration options. The configured parameters are written to $ROCKBASE/.config when leaving the configuration tool.

The following subsections provide some examples for the aforementioned steps to add an individual application. These examples are taken from an application called *lfutil* I have written during development for testing various functions of the AP.

Makefile Example for an Application

```
# Specify the name of the executable application
EXEC = lfutil

# Specify the source files needed to compile the application
SOURCES = lfutil.c mvcfuncs.c

# Include generic definitions needed by the build process
include $(ROCKBASE)/make_include/app.mk

# default configuration file
.PHONY: etc
```

[1] $ROCKBASE: is the environment variable that should be set up to point to the firmware/OS/uClinux/ directory of the extracted sources of the APDK.

```
etc ::
# @echo "No default config for $(EXEC)" >&2
```

The *etc* target of the makefile is not used in this example but necessary steps for installing needed configuration files should be provided here. For further information on configuration files of applications see A.1.3.

A variable that might be of interest is *EXTRA_INCLUDES* where with an entry like *-I./includes* additional directories for header files can be specified.

Another variable of use for larger applications is *FLTFLAGS*. This variable can be used to specify flags for the elf2flt tool. Using *FLTFLAGS = -s 8196* specifies a stack size of 8196 bytes. Further information on this topic is found in [Int02b], Sect. 3.1.

Various other variables exist but are not mentioned here because they have not been further investigated.

Input of the Configuration Tool

The file *$ROCKBASE/config.in* is used as input to the configuration tool that is started by executing *make menuconfig*. To make an individual application known by the configuration tool a simple line needs to be added to config.in with the syntax:

> bool '[Text to display]' CONFIG_APPS_GENERIC_[apps_dir]

[Text to display] specifies the text that is provided after the corresponding checkbox in the configuration tool. It should contain necessary information to identify the application.

[apps_directory] specifies the subdirectory of $ROCKBASE/apps that contains the source of the application.

For the application *lfutil* the entry is as follows:

> bool ' Include lfutil' CONFIG_APPS_GENERIC_lfutil

A.1.3 Configuration Files of Applications

The $ROCKBASE/romfs directory is the root (/) directory on the AP after the compiled image of the main system is installed. $ROCKBASE/romfs contains the main structure of a Linux filesystem. Among others, configuration and startup files that do not get replaced during the build process are located in various subdirectories. Compiled applications are copied to the bin subdirectory before the image of the main system is built.

$ROCKBASE/romfs/etc is the main directory for configuration files. As $ROCKBASE/romfs is the root directory of the AP, $ROCKBASE/romfs/etc is the /etc directory of the running AP. Besides main configuration files and startup scripts it contains the directory *defaults/ucd-snmp* that is for files containing the configurable parameters of the AP. These parameters can be changed mainly via web-interface or SNMP.

Recall that the image of the main system is stored in a read-only area of system memory. As writing to the configuration files is absolutely necessary, the files in *defaults/ucd-snmp* are copied to another writable directory */usr/etc* on the AP.

This copying occurs if a configuration file found in defaults/ucd-snmp is not present or newer than the one in /usr/etc. The files are also copied if a user resets the AP to the default settings by pressing the reset button for more than five seconds. Other reasons for updating a configuration file in the /usr/etc directory with parameters provided in the corresponding file in defaults/ucd-snmp might exist.

If a read-only configuration file is sufficient for an application, the configuration file can be put in the $ROCKBASE/romfs/etc directory. In case the configuration file should be writable too, it has to be located in the /usr/etc directory.

As /usr/etc is mounted during startup of the AP, the configuration files need to be copied to /usr/etc after startup of the AP, i.e. after compilation of the image and the upgrade of the AP. This means that it is not possible to simply copy the configuration file to the $ROCKBASE/romfs/usr/etc directory before compile time or during the build process.

For simple applications it might be sufficient to add the configuration file to the $ROCKBASE/romfs/etc/defaults/ucd-snmp/ directory. Files in this directory are already handled appropriately.

Note that configuration files located in /usr/etc on the AP are not replaced during a firmware upgrade. Therefore, after an upgrade of the main system, the AP boots with the same configuration as was used before the upgrade.

A.1.4 Starting Services at Startup

If the individual application is a service that should be started during the AP boot process, two additional steps need to be done:

1. Add a start-stop-script for the service to the *$ROCKBASE/romfs/etc/init.d/* directory. This script needs to support the arguments *start, stop* and *restart*.
2. Add the name of the script on a new line in the file *$ROCKBASE/romfs/etc/startup.list*.

For further information on developing applications for the Intersil AP see the user manual *Porting Guide for the Intersil Access Point* [Int02b].

A.2 Accessing the MVC

This section provides source code examples for accessing MVC device objects in three different kinds of way. First, a device object of type long is read, next a device object of type long is written and the last example subscribes to a trap.

A.2.1 Common Code

This section contains code that is the same for all of the three examples.

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <unistd.h>
#include <errno.h>
#include <signal.h>
#include <sys/socket.h>
#include <sys/ioctl.h>
#include <net/if.h>

#include "blob.h"
#include "../../kernel/drivers/net/prismeth.h"

#define IF_WIRELESS "eth1"
```

```
/*--- prismeth_ioctl------------------------------------------
 * Helper function for ioctls to the prism ethernet driver.
 * Returns -1 in case of error, 0 else.
 *-----------------------------------------------------------*/
int prismeth_ioctl (struct ifreq *ifr) {
  int sockfd = -1;

  if ((sockfd = socket (PF_INET, SOCKDGRAM, 0)) < 0) {
    fprintf(stderr, "Could not open socket: %s",
      strerror(errno));
    return -1;
  }

  strncpy (ifr->ifr_name, IF_WIRELESS, sizeof(IF_WIRELESS));

  if (ioctl(sockfd, PRISMIOCTL, ifr) <0 ) {
    close(sockfd);

    fprintf(stderr, "prismeth_ioctl not successful: %s\n",
      strerror(errno));

    return -1;
  }
  close(sockfd);
  return 0;
}

/*--- signalhandler--------------------------------------------
 * Used for common signals.
 *------------------------------------------------------------*/
void signalhandler(int signal) {
  switch (signal) {
    case SIGINT  :
      printf("Received SIGINT, terminating...\n");
      break;
    case SIGQUIT :
      printf("Received SIGQUIT, terminating...\n");
      break;
    case SIGTERM :
      printf("Received SIGTERM, terminating...\n");
      break;
  }
  exit (0);
}
```

A.2.2 Reading an MVC Device Object

```
/*--- printClientsAssociated------------------------------------
 * Prints the number of associated clients at the
 * access point.
 *--------------------------------------------------------------*/
void printClientsAssociated(void) {
  struct prism_ioctl ioc;
  struct ifreq ifr;
  long assoc_clients = 0;

  ioc.cmd           = OPGET;
  ioc.oid           = DOT11_OID_CLIENTSASSOCIATED;
  ioc.data          = (void *)&assoc_clients;
  ioc.len           = sizeof(assoc_clients);
  ifr.ifr_data      = (caddr_t) &ioc;

  if (prismeth_ioctl(&ifr) < 0) return;

  fprintf(stdout, "Nr. of associated clients: %ld\n",
    assoc_clients);
}
```

A.2.3 Writing to an MVC Device Object

```
/*--- acceptUnencrypted-----------------------------------------
 * Tell the MVC to accept unencrypted network frames, even
 * if encryption is enabled.
 *--------------------------------------------------------------*/
void acceptUnencrypted (void) {
  struct prism_ioctl ioc;
  struct ifreq ifr;
  long value = 0;

  ioc.cmd           = OPSET;
  ioc.oid           = DOT11_OID_EXUNENCRYPTED;
  ioc.data        = (void *)&value;
  ioc.len         = sizeof(value);
  ifr.ifr_data    = (caddr_t) &ioc;

  if (prismeth_ioctl(&ifr)>=0)
    fprintf(stdout,"Now accepting unencrypted frames.\n");
}
```

A.2.4 Subscribing to a Trap

```
/*--- trap_handler-------------------------------------------------
 * Handles the trap to which this process has subscribed
 * via the subscribe_trap method.
 *----------------------------------------------------------------*/
void trap_handler(int signal) {
  switch(signal) {
    case SIGUSR1 :
      printf("Client associated.\n");
      break;
    default:
      fprintf(stderr, "Unknown signal.\n");
  }
}

/*--- subscribe_associated-----------------------------------------
 * Subscribes to the DOT11_OID_ASSOCIATEEX trap by using
 * the signal SIGUSR1. Therefore, the signal SIGUSR1 is
 * provided to the process whenever a STA associates with
 * the access point.
 *----------------------------------------------------------------*/
void subscribe_associateex (void) {
  struct prism_ioctl ioc;
  struct ifreq ifr;
  long signum = SIGUSR1;

  signal(signum, trap_handler);

  ioc.oid         = DOT11_OID_ASSOCIATEEX;
  ioc.cmd         = OPTRAPADD;
  ioc.data        = (void *)signum;
  ioc.len         = sizeof(signum);
  ifr.ifr_data   = (caddr_t) &ioc;

  if (prismeth_ioctl(&ifr) <0) return;

  printf("Subscribed to trap, waiting for events...\n");
  while (1) {
    pause();
  }
}
```

A.2.5 Executing the Example Code

To execute the example code, copy the examples provided above and the main function provided below into one source file. Follow the steps provided in A.1.2 to include this new application in the main system image. Upon *make menuconfig* do not forget to include the *shelld* for telnet access. Build the image and upgrade the access point with the new image. Telnet to the AP and execute the application by calling the executable you specified in the *EXEC* variable of the Makefile.

Main Function

```
int main (int argc, char **argv) {
  signal(SIGINT, signalhandler);
  signal(SIGTERM, signalhandler);
  signal(SIGQUIT, signalhandler);

  printClientsAssociated();
  acceptUnencrypted();
  subscribe_associateex();

  exit(0);
}
```

Appendix B

Legacy Authentication

B.1 IPtables Firewall Rules in the AP

B.1.1 Rules added at Startup

Enabling the DNS Service

Two different approaches exist:

1. Enabling the DNS service without specific servers. This can be used if the DNS servers are not known in advance or are simply not configured:

```
iptables -t nat -I PREROUTING -p udp --dport 53 -j ACCEPT
iptables -t filter -I FORWARD -p udp --dport 53 -j ACCEPT
```

2. Enabling the access to two specific DNS servers:

```
iptables -t nat -I PREROUTING -d dnsserver1 \
    -p udp --dport 53 -j ACCEPT
iptables -t filter -I FORWARD -d dnsserver1 \
    -p udp --dport 53 -j ACCEPT
iptables -t nat -I PREROUTING -d dnsserver2 \
    -p udp --dport 53 -j ACCEPT
iptables -t filter -I FORWARD -d dnsserver2 \
    -p udp --dport 53 -j ACCEPT
```

Note that dnsserver1 and dnsserver2 above need to be replaced by the names or IP addresses of the actual DNS servers. Generally, it is better to specify the IP addresses as this prevents the possibility of a DNS attack.

Enabling Access to the HTTPS Authentication Server

```
iptables -t filter -A FORWARD -d HTTPSAuthServ  \
    -p tcp --dport 80 -j ACCEPT
iptables -t filter -A FORWARD -d HTTPSAuthServ  \
    -p tcp --dport 443 -j ACCEPT
```

HTTPSAuthServ above needs to be replaced by the name or IP address of the HTTPS authentication server.

Enabling the DHCP Service

```
iptables -t filter -A FORWARD -p udp --dport 67 -j ACCEPT
```

Disallowing Access to the Legacy Authentication Port of the AP

Only the HTTPS authentication server is allowed to access the TCP port, where the AP is listening for incoming messages of the HTTPS authentication portal.

```
iptables -t filter -A INPUT -s ! HTTPSAuthServ \
    -p tcp --dport LegacyAuthPort -j DROP
```

Note that HTTPSAuthServ above needs to be replaced by the name or IP address of the actual HTTPS authentication server. LegacyAuthPort needs to be replaced by the port number where the AP is listening for messages of the HTTPS authentication portal.

B.1.2 Rules added for individual STAs

These rules are added whenever a legacy STA associates with the AP. They are removed after successful authentication of the corresponding user at the HTTPS authentication portal.

```
iptables -t filter -A FORWARD \
    -m mac --mac-source STA_MAC_Addr -j DROP
iptables -t nat -A PREROUTING -m mac --mac-source \
    STA_MAC_Addr -j DNAT --to-destination HTTPSAuthServ
```

The first rule drops any traffic received by the STA with MAC address STA_MAC_Addr. The second rule redirects any traffic received from the STA with STA_MAC_Addr to the HTTPS AS.

B.1.3 Some Notes on IPtables

The list of iptables rules is processed from the beginning to the end. Therefore, each rules takes precedence over following rules. The rules that were added during boot time of the AP are at the beginning of the list of iptables rules. Rules for individual STAs are added at the end.

Due to this approach the services that are explicitly allowed by rules added during startup are accessible by STAs, even if a later rule specifies to drop any network packets of them.

Bibliography

[ABS03] B. Anton, B. Bullock, and J. Short. *Best Current Practices for Wireless Internet Service Provider (WISP) Roaming*. Wi-Fi Alliance, February 2003. Version 1.0, http://www.wi-fi.org/OpenSection/downloads/WISPr_V1.0.pdf.

[Air03] AirDefence. *War Drive Survey: 57% of Enterprises Wirelss LANs Not Encrypted*, September 2003. http://www.airdefense.net/newsandpress/09_24_03.shtm.

[All03] WiFi Alliance. *Wi-Fi Protected Access (WPA)*, April 2003. Version 2.0.

[AS99] Bernard Aboba and Dan Simon. *PPP EAP TLS Authentication Protocol*. IETF, October 1999. RFC 2716, http://www.ietf.org/rfc/rfc2716.txt.

[BGW] Nikita Borisov, Ian Goldberg, and David Wagner. *Security of the WEP Algorithm*. http://www.isaac.cs.berkeley.edu/isaac/wep-faq.html.

[BV98] Larry J. Blunk and John R. Vollbrecht. *PPP Extensible Authentication Protocol (EAP)*. IETF, March 1998. RFC 2284, http://www.ietf.org/rfc/rfc2284.txt.

[BVA$^+$03] Larry J. Blunk, John R. Vollbrecht, Bernard Aboba, James Carlson, and Henrik Levkowetz. *Extensible Authentication Protocol (EAP)*. IETF, June 2003. Internet Draft, http://www.ietf.org/internet-drafts/draft-ietf-eap-rfc2284bis-05.txt.

[CCM] *CCM Use Requirements Specification*. http://csrc.nist.gov/CryptoToolkit/modes/proposedmodes/ccm/ccm-ad2.pdf.

[CDK01] George Coulouris, Jean Dollimore, and Tim Kindberg. *Distributed Systems - Concepts and Design.* Addison-Wesley, third edition, 2001.

[Cis] Cisco. *IPSec NAT Transparency.* `http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122newft/122t/122t13/ftipsnat.htm`.

[Cis03] Cisco. *Wireless Virtual LAN Deployment Guide*, 2003. `http://www.cisco.com/en/US/products/hw/wireless/ps430/prod_technical_reference09186a00801444a1.html`.

[DA99] Tim Dierks and Christopher Allen. *The TLS Protocol, Version 1.0.* IETF, January 1999. RFC 2246, `http://www.ietf.org/rfc/rfc2246.txt`.

[FBW02] Paul Funk and Simon Blake-Wilson. *EAP Tunneled TLS Authentication Protocol (EAP-TTLS).* IETF, November 2002. Internet Draft, `http://www.ietf.org/internet-drafts/draft-ietf-pppext-eap-ttls-02.txt`.

[FMS] Scott Fluhrer, Itsik Mantin, and Adi Shamir. *Weaknesses in the Key Scheduling Algorithm of RC4.* `http://www.crypto.com/papers/others/rc4_ksaproc.ps`.

[Fri03] Bob Friday. *Will wireless work with heavy traffic?* Network World Fusion, June 2003. `http://www.nwfusion.com/columnists/2003/0630wizards.html`.

[Gas02a] Matthew Gast. *A Technical Comparison of TTLS and PEAP*, October 2002. `http://www.oreillynet.com/pub/a/wireless/2002/10/17/peap.html`.

[Gas02b] Matthew S. Gast. *802.11® Wireless Networks: The Definitive Guide.* O'Reilly, April 2002.

[HM96] Neil Haller and Craig Metz. *A One-Time Password System.* IETF, May 1996. RFC 1938, `http://www.ietf.org/rfc/rfc1938.txt`.

[HS03] Henry Haverinen and Joseph Salowey. *EAP SIM Authentication.* IETF, June 2003. Internet Draft, `http://www.ietf.org/internet-drafts/draft-haverinen-pppext-eap-sim-11.txt`.

[IEE90] IEEE. *802 - Overview and Architecture*, 1990.

[IEE98] IEEE. *Virtual Bridged Local Area Networks*, December 1998. 802.1Q, http://standards.ieee.org/getieee802/download/802.1Q-1998.pdf.

[IEE99] IEEE. *802.11 - Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, August 1999. http://standards.ieee.org/getieee802/802.11.html.

[IEE01] IEEE. *802.1X - Port-Based Network Access Control*, June 2001. http://standards.ieee.org/getieee802/download/802.1X-2001.pdf.

[IEE02] IEEE. *802.11i - Specification for Enhanced Security* , November 2002. Draft Version 3.0.

[IEE03a] IEEE. *802.11f - Inter Access Point Protocol*, January 2003. Draft Version 5.

[IEE03b] IEEE. *802.11i - Medium Access Control (MAC) Security Enhancements* , October 2003. Draft Version 7.0.

[Int02a] Intersil. *Architecture of a Linux based Intersil AP*, 2002. June.

[Int02b] Intersil. *Porting Guide for the Intersil Access Point*, 2002. August.

[Int02c] Intersil. *The uClinux Development Environment*, June 2002.

[Int03] Intersil. *MVC Software Interface Manual*, February 2003.

[Jon02] Jakob Jonsson. *On the Security of CTR + CBC-MAC*, 2002. http://csrc.nist.gov/CryptoToolkit/modes/proposedmodes/ccm/ccm-ad1.pdf.

[MA02] Arunesh Mishra and William A. Arbaugh. *An Initial Security Analysis of the IEEE 802.1X Standard*, February 2002. http://www.cs.umd.edu/~waa/pubs/1x.pdf.

[Mic03] Microsoft. *Virtual Private Networking with Windows Server 2003: Overview*, March 2003. http://www.microsoft.com/windowsserver2003/docs/vpnoverview.doc.

[NIS01] NIST. *Advanced Encryption Standard (AES)*, November 2001. FIPS PUB 197, http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf.

[OMG03] OMG. *Unified Modeling Language*, March 2003. Version 1.5, http://www.uml.org/.

[PSZJ03] Ashwin Palekar, Dan Simon, Glen Zorn, and Simon Josefsson. *Protected EAP Protocol (PEAP)*. IETF, March 2003. Internet Draft, http://www.ietf.org/internet-drafts/draft-josefsson-pppext-eap-tls-eap-06.txt.

[Rad02] Nils Radtke. *Ethernet Bridge + netfilter Howto*, October 2002. http://www.tldp.org/HOWTO/Ethernet-Bridge-netfilter-HOWTO.html.

[Ros02] Ken Roser. *HOWTO: Setup EAP/TLS with FreeRADIUS and WindowsXP*, April 2002. http://www.freeradius.org/doc/EAPTLS.pdf.

[Sim96] William Allen Simpson. *PPP Challenge Handshake Authentication Protocol (CHAP)*. IETF, August 1996. RFC 1994, http://www.ietf.org/.

[SIR01] Adam Stubblefield, John Ioannidis, and Aviel D. Rubin. *Using the Fluhrer, Mantin, and Shamir Attack to break WEP*. AT&T Labs Technical Report TD-4ZCPZZ, August 2001. http://www.cs.rice.edu/~astubble/wep/wep_attack.pdf.

[Sta98] William Stallings. *Operating Systems - Internals and Design Principles*. Prentice Hall, third edition, 1998.

[Wal] Jesse Walker. *Unsafe at any Key Size; An Analysis of the WEP Encapsulation*. http://grouper.ieee.org/groups/802/11/Documents/DocumentHolder/0-362.zip.

[Wea02] Graeme Wearden. *It's not good to chalk*, September 2002. http://news.zdnet.co.uk/business/0,39020645,2122999,00.htm.

[WHF02] Doug Whiting, Russ Housley, and Niels Ferguson. *Counter with CBC-MAC (CCM) - AES Mode of Operation* , June 2002. http://csrc.nist.gov/CryptoToolkit/modes/proposedmodes/ccm/ccm.pdf, http://www.ietf.org/internet-drafts/draft-housley-ccm-mode-02.txt.

Glossary

AAA	Authentication, Authorization and Accounting, page 59
AAD	Additional Authentication Data, page 51
AES	Advanced Encryption Standard, page 50
AP	Access Point, page 4
APDK	Access Point Development Kit, page 66
ARP	Address Resolution Protocol, page 96
AS	Authentication Server, page 30
AVP	Attribute-Value Pair, page 29
BSA	Basic Service Area, page 5
BSS	Basic Service Set, page 5
BSSID	Basic Service Set Identifier, page 14
CA	Certification Authority, page 85
CBC-MAC	Cipher Block Chaining Message Authentication Code, page 50
CCM	Counter-Mode/CBC-MAC, page 50
CCMP	Counter-Mode/CBC-MAC Protocol, page 50
CHAP	Challenge-Handshake Authentication Protocol, page 25
CTR	Counter, page 50
DA	Destination Address, page 14
DMZ	DeMilitarized Zone, page 83

DoS Denial of Service, page 18

DS Distribution System, page 5

DSS Distribution System Service, page 9

EAP Extensible Authentication Protocol, page 23

EAPOL EAP over LAN, page 32

EKU Enhanced Key Usage, page 85

ESS Extended Service Set, page 7

GMK Group Master Key, page 40

GNU GNU's Not Unix, page 73

GNU GPL GNU General Public License, page 74

GSM Global System for Mobile Communications, page 26

GTK Group Transient Key, page 40

HTTP Hyper Text Transfer Protocol, page 68

HTTPS HyperText Transfer Protocol (HTTP) over Secure Socket Layer (SSL), page 57

IAPP Inter Access Point Protocol, page 5

IBSS Independent Basic Service Set, page 6

ICV Integrity Check Vector, page 21

IEEE Institute of Electrical and Electronics Engineers, page 4

IETF Internet Engineering Task Force, page 28

IV Initialization Vector, page 21

LAN Local Area Network, page 1

MAC Medium Access Control, page 13

MIC Message Integrity Code, page 41

MIM Man In the Middle, page 17

MPDU	MAC Protocol Data Unit, page 5
MSDU	MAC Service Data Unit, page 5
MVC	MAC Virtual Coprocessor, page 66
NIST	National Institute of Standards and Technology, page 50
OID	Object Identifier, page 70
OTP	One Time Password, page 25
PAC	Public Access Control, page 58
paed	Port Access Entity Daemon, page 73
PMK	Pairwise Master Key, page 39
PPP	Point-to-Point Protocol, page 23
PRNG	Pseudo Random Number Generator, page 21
PSK	Pre-Shared Key, page 38
PTK	Pairwise Transient Key, page 40
RA	Receiver Address, page 14
RADIUS	Remote Authentication Dial In User Service, page 29
RSN	Robust Security Network, page 36
RSN IE	RSN Information Element, page 37
RSNA	Robust Security Network Association, page 36
SA	Source Address, page 14
SIM	Subscriber Identity Module, page 26
SNMP	Simple Network Management Protocol, page 105
SOHO	Small Office / Home Office, page 56
SS	Station Service, page 10
SSID	Service Set Identifier, page 6
SSL	Secure Socket Layer, page 26

STA	Station, page 4
TA	Transmitter Address, page 14
TFTP	Trivial File Transfer Protocol, page 68
TK	Temporal Key, page 40
TKIP	Temporal Key Integrity Protocol, page 47
TLS	Transport Layer Security, page 26
TLS	Transport Layer Security, page 98
TSC	TKIP Sequence Counter, page 47
TSN	Transition Security Network, page 36
TTAK	TKIP mixed Transmit Address and Key, page 48
UAM	Universal Access Method, page 59
URL	Universal Resource Locator, page 79
VLAN	Virtual LAN, page 61
VPN	Virtual Private Networking, page 57
WDS	Wireless Distribution System, page 5
WEP	Wired Equivalent Privacy, page 20
Wi-Fi	Wireless Fidelity, page 35
WISP	Wireless Internet Service Provider, page 56
WLAN	Wireless Local Area Network, page 1
WPA	Wi-Fi Protected Access, page 54

Index