Bachelor thesis
Computer Science

Radboud University

Security in public Wi-Fi networks

Author:
Richard van Ginkel
s4599047

First supervisor/assessor:
Erik Poll
e.poll@cs.ru.nl

Second assessor:
Thom Wiggers
t.wiggers@ru.nl

August 17, 2019

Abstract

Since open WiFi networks are widely used, it is important to secure such networks. In this research public wireless networks and their security are discussed. With the new Wireless Access Protocol, WPA3, the security of public networks has improved significantly. However, regarding *evil twin attacks* nothing is changed in WPA3, thus this kind of attacks is still one of the possibilities an adversary has. In this thesis we will first review the improvements in WPA3 with respect to WPA2 and introduce the new techniques and protocols in WPA3. After that, multiple approaches are presented to protect clients against *evil twin attacks*. Finally, the approach that is best regarding our criteria, i.e., the use of "trust on first use", is chosen and recommendations are made on how to use this.

Contents

Chapter 1

Introduction

Public Wi-Fi networks are used more and more, and can be found at lots of places nowadays. Whether it is in a local coffee shop, the supermarket nearby or the train we travel with every day, most of us use such open networks from time to time. Unfortunately, these networks are not secure. This thesis focuses on improving security in such networks.

Currently, most Wi-Fi networks use Wireless Protected Access 2 (WPA2), or in the case of public networks, no security is used at all. This means that an attacker can perform multiple attacks. In public networks, an attacker could connect to the network and eavesdrop on all data that is sent in the network, because no encryption is used. When the network is secured with WPA2, someone malicious could eavesdrop all data in the network if he is in the network himself, because all conversations are encrypted with the same key, as discussed in chapter 2. An adversary could also try to crack the password when WPA2 is used. The attacker could automatically try all possible passwords until the correct one is found.

With the new security standard for WiFi, Wireless Protected Access 3 (WPA3), wireless networks became more secure. A passive attack, like the eavesdropping, as was possible before, is no longer something an attacker can do. Because of Opportunistic Wireless Encryption (OWE) in public networks and Simultaneous Authentication of Equals (SAE) in secured networks, every client in the network will, together with the access point, generate his own unique keys to encrypt the data being sent back and forth between client and access point. SAE makes it for an adversary impossible to try a lot of passwords really fast. Next to that, SAE requires the attacker to stay online during every try of a password and SAE makes it possible to set a maximum on the number of tries. This makes it almost impossible for an attacker to crack the password.

However, the introduction of WPA3 did not solve all security issues; someone malicious could still perform an active attack, a so-called *evil twin attack*. That is, an attacker could impersonate a public network and pretend

to be a trustworthy network. In the case of an open network in a train, an attacker could be on the same train and open his own public network with the exact same network name as the network the train company offers. Assume a new client wants to connect, to send some business emails before arriving at the office. The client will see two equally named network listed and randomly select the malicious network. Although everything seems to work just fine, the client does not notice that his credentials used to log in to the email server are available to the attacker and that data might be modified. Such sensitive data should be protected at all cost. To this end approaches to a solution for this problem are investigated further in this thesis. We look for possible security measures against such an attack. The research question we will try to answer is: **How can we defend against evil twin attacks?**

To answer this question, we will discuss multiple possible approaches. The advantages and disadvantages of these approaches are discussed. **Usability**, **feasibility**, **overhead** and **effectiveness** are the criteria on the basis of which the best approach, the use of trust on first use, is presented and recommendations about this best option are made.

Firstly, in chapter 2, we will discuss Wi-Fi in general and possible attacks on public networks. In chapter 3 the improvements in WPA3 compared to WPA2, the deployment of WPA3 and testing WPA3 are discussed. Next to answering our research question in chapter 4, we will have a look at related work in chapter 5. Because WPA3 is a new security standard, very little has been written about it and lots of research has to be done. Official documentation of the release is available, as well as some papers about the functionality of WPA3. Next to the documentation on WPA3 we will have a look at research about *evil twin attacks*, *man in the middle attacks*, *eavesdropping* and *password cracking* in this chapter. In chapter 6 we will summarize our findings and draw a conclusion out of these. This research can be used for new insights and a basis for further research into this area.

Chapter 2

Preliminaries

Since the introduction of the first standard for wireless networks, about twenty years ago [19], wireless networks gained more and more popularity. The downside of this popularity is the fact that attackers found more and easier ways to do malicious things on such networks. To mitigate against these attacks, security measures where introduced.

As from 2004 Wireless Protected Access 2 is the common standard used for WiFi Security. Even before that, Wired Equivalent Privacy (WEP) and Wireless Protected Access (WPA) were used [20]. These two security protocols were only used for a short time, due to major security problems. WEP is based on the RC4 stream cipher. One of the issues is that, when one changes one bit in the plain text, the same bit will be changed in the cipher text. Another weakness is, RC4 uses an initialization vector (IV). The use of IVs is not the real problem, but that about nine thousand out of 16 million options for these IVs are weak, is [12]. This makes it possible to crack the pre-shared key of any network by cryptanalysis, after capturing millions of data packets. WPA was released to offer solutions for the security flaws in WEP, but to make sure that devices only needed a firmware update to use WPA, TKIP was used in WPA. TKIP is a kind of a wrapper around RC4. Now longer IVs are used, but the main problem, RC4, is still present. This resulted in similar possibilities to crack a password [5]. Vanhoef and Piessens have shown that multiple vulnerabilities in TKIP can be used in for attacks in realistic environments [26].

WPA2 solves this issue by getting rid of RC4. Instead it uses AES, which is more safe. WPA2 was secure for a long time. However, in 2016, a Key Re-installation Attack (KRACK) was discovered [29]. We will discuss this attack later. Another attack is the dictionary attack. This is simply trying all possible combinations of characters very fast until the password is found. Due to these attacks, a new security standard was needed.

Wireless Protected Access 3 (WPA3) was announced in January 2018. Despite the fact that we will focus on attacks that are still possible with this

new standard, it is useful to have a short recap of WPA2 and the differences between WPA2 and WPA3. Although the four-way handshake is used in all three WPA standards, it is especially interesting for this thesis since some adjustments are made to it in WPA3. Next to understanding the four-way handshake, it is useful to understand how a connection between client and access point is made. To this end, we will have a look at wireless networks in general and at the four-way handshake firstly.

2.1 Authentication and association phase

Access points advertise their network through beacons. These are frames with information about the access point, like the Medium Access Control (MAC) address and security policy of the access point. Another way for clients to get information about access points nearby is to send a probe request. You could see this as a question to the AP to advertise itself. The AP will response with a probe response, which is basically the same as a beacon. When the client has decided which access point it wants to connect to, it will send an authentication request and the AP will respond that the client is authenticated. These two messages do not really contain anything. These were designed for the old security standard WEP, which is proven to be insecure, thus the two authentication messages have no real function and are left empty in most cases.

After this comes the association phase. This starts with the client sending an association request. In this request, capabilities like encryption types and supported rates are included. The access point will now check the request. That is, the access point will check whether the capabilities included in the request match his own capabilities. If this is the case, an association response will be sent to the client. The response includes the confirmation of the capabilities and an association identifier.

Now the client and the access point are connected successfully, the four-way handshake will be performed and data can be sent if the network is an open or public network. However, if the network is protected with any kind of WPA, before the handshake the present authentication mode will be executed in order to get a Pairwise Master Key. This key will be the input of the four-way handshake that is performed subsequently. After this the client and the access point can exchange data.

2.2 Four-way handshake

Before exchanging the actual data, encryption keys have to be exchanged. This is done in the four-way handshake. Let's assume we have a client (STA, short for station, in Figure 2.1) that wants to connect to an access point (AP, short for access point, in Figure 2.1). WPA and WPA2 have two authenti-

cation modes; Personal and Enterprise. WPA-Personal is the mode that is commonly used in home networks. When setting up a network like this, a passphrase is chosen. When someone wants to connect to this network, this passphrase is entered to authenticate. On the other side, WPA-Enterprise is mostly used in business environments like companies or universities. Enterprise uses one of many Extensible Authentication Protocol (EAP) exchange methods [32]. EAP is an authentication framework used for point-to-point connections. One could use a user name and personal password for example. Every client has his own individual credentials. This way it is possible for an administrator to decide who should be able to connect to the network, because every user in the network has a different key. This way it offers better security.

For the handshake, both parties need the Pairwise Master Key (PMK). When using WPA2-Personal, the PMK is derived directly from a Pre-Shared Key (PSK), the pass phrase. On WPA-Enterprise, the PMK is obtained using the EAP exchange. The goal of the handshake is, besides authentication, to turn this PMK into data encryption keys and integrity keys. Before this can be done, both parties need some information from each other. To this end, the access point sends a pseudo-randomly generated nonce, a number only used once (ANonce in Figure 2.1).

By the time the client receives this nonce, it has already generated its own nonce. Because of the message the access point sent, the client now knows its MAC address. Obviously, the client knows its own MAC address. Now the client has all the information required to calculate a Pairwise Transient Key (PTK), which can be calculated from the two MAC addresses. This is actually a group of five keys, of which four are used during the handshake phase. The client now sends his nonce. Since the client already has the PTK, it can ensure integrity with a Message Integrity Code (MIC). After receiving this message, the access point now has all the inputs to generate the PTK as well. With this PTK the access point can calculate the MIC and check for integrity. The PTK is used to encrypt data in one-to-one conversations between client and access point.

The access point and clients, however, might want to send group messages as well. An example of group messages are ARP messages. ARP is short for Address Resolution Protocol. This protocol is used to learn MAC addresses, associated with IP addresses. A client can share to the whole network, which IP address he is on, so that others can send him data. To encrypt messages like this, the access point will create a Group Temporal Key (GTK). Now this GTK will be sent to the client, again protected with a MIC. Both parties have the two keys needed for encrypted data transfer, so the client sends an acknowledge (ACK) to let the access point know everything succeeded [17]. This last two messages both contain a nonce.

Vanhoef and Piessens wrote an article about these group keys and their vulnerabilities in 2016 [27]. One of there findings is the fact that the random

number generator that is used to generate the keys provides an insufficient amount of entropy. Next to that, they found a way to force the four-way handshake to use RC4 to encrypt the group key when transmitted in the handshake. As discussed earlier RC4 is not safe to use.

Before, we mentioned the KRACK attack. The main idea behind this attack is to repeatedly reset the nonce that is transmitted in the third message of the four-way handshake, as explained by Vanhoef and Piessens [28]. Doing this, an attacker is able to learn the key used to encrypt and decrypt the data traffic. Vanhoef, Schepers and Piessens published an article in 2017 which revealed multiple logical vulnerabilities in the handshake [30]. These vulnerabilities, which lead to for example Denial of Service, were discovered using a model-based testing technique which generated tests that cover all states that the handshake can be in. In these states the edge cases were explored. These vulnerabilities are no longer a problem, since they have been patched now.

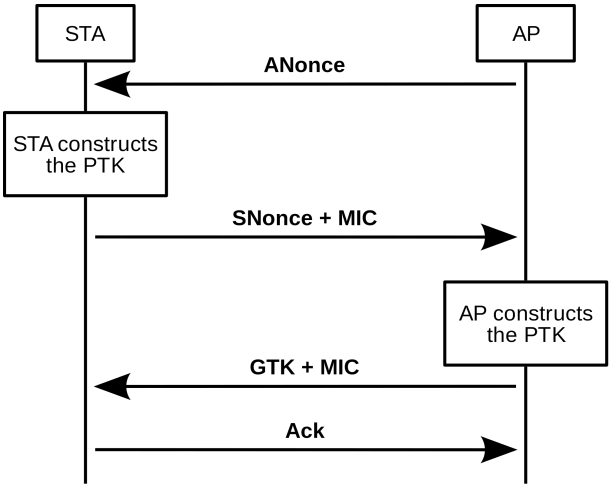

Figure 2.1: Four-way Handshake [35]

2.3 Data encryption

Now all the keys are generated and known by both parties, encrypted data can be sent. Data is encrypted using a block cipher counter mode (see Figure 2.2). This cipher uses a nonce concatenated with a counter as input. Together with the Temporal Key (TK), one of the five keys in the PTK, this is the input for the block cipher encryption used inside the block cipher. The result this gives us, is then used to do a bit-wise XOR with the plain text, the data we want to encrypt. This results in the encrypted data that is sent via the network.

The block cipher encryption, as discussed above, is based on Advanced Encryption Standard (AES) [24]. This is a way of encryption that is currently assumed secure and unbreakable.

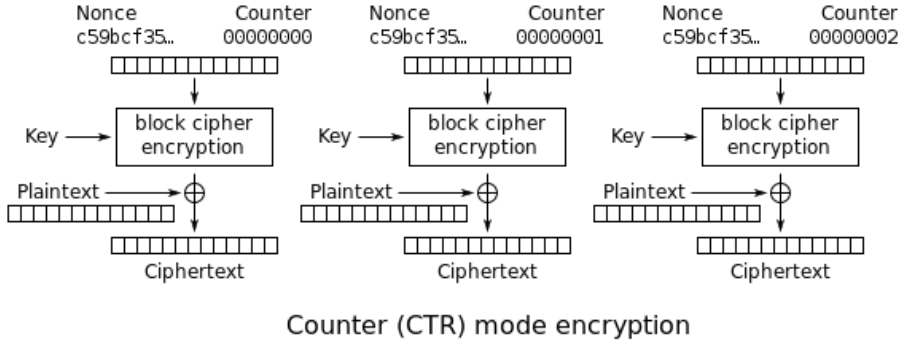

Figure 2.2: Counter mode block cipher [34]

2.4 Attacks on wireless networks

Now that we have seen how wireless networks in general work, we can look at possible attacks against these networks. We look at the following most important attacks:

- **Eavesdropping**: When a wireless network offers no security, an attacker can eavesdrop on the network. The attacker can connect to the network, and then use a tool like Wireshark to capture data that is sent on the network. Since the network uses no security, this data can easily be inspected. When a network used WPA2 security, an attacker could do the same if he is a member of the network. WPA2 makes all clients in the network use the same encryption keys, which makes it possible for someone malicious to decrypt all data.

- **Password cracking**: When a network is protected with a password, an attacker can try to find this password. Someone malicious will simply try all possible combinations of characters, until the password is found. This can be done smarter by firstly trying all standard passwords and passwords that are used often. These passwords are put into a so-called dictionary. The attacker then automatically tries all the entries of this dictionary until the password is found. This is called a dictionary attack. When WPA2 is used, an attacker can capture a small number of handshakes. When the attacker has all four messages of the handshake captured successfully, the attacker can go offline and try passwords. A dictionary is used for this. If the password is somewhere in the dictionary file, the attacker will find the password.

- **Evil twin attack**: An attacker could impersonate a public network and pretend to be a trustworthy network. In the case of an open network in a train, an attacker could be on the same train and open his own public network with the exact same network name as the network the train company offers. Assume a new client wants to connect, to send some business emails before arriving at the office. The client will see two equally named network listed and randomly select the malicious network. Although everything seems to work just fine, the client does not notice that his credentials used to log in to the email server are available to the attacker and that data might be modified.[1]

2.5 Discrete logarithm problem

The discrete logarithm problem is used in new protocols that will be discussed later. To this end we explain it now. The discrete logarithm problem can be seen as a one-way function. It is easy to calculate the result, but going back from the result to the values we started with is very hard and infeasible. For this one-way function we use modulo calculations, using a big prime p. The numbers 1 to p-1 are then a so-called group, or a finite field. A group also has a group operation, in this case and mostly, multiplication. Now we can choose an element g out of this group, which is a generator of the group. That is, an element that when raised to all powers (the number of times the group operation is performed on this element itself) of the group, gives a different result when the modulo p is taken for each power. This way every element of the group will be a result of one of these calculations, so all elements are generated. Hence we call it a generator. When we raise g to a power a, and take the outcome of this modulo p, we get a result between 1 and p-1. An example: Let p be 17, let g be 5 and let a be 13. Then we can calculate our result as follows:

$$5^{13} = 1220703125 \equiv 3 \mod 17$$

This is a computationally easy problem. However, when given the result and p, it is computationally very hard to find g and a. In this case, when we know that 3 is the result and we used 17 as the prime number, it is difficult to find 5 and 13 as our starting values. Even if g is known, to find a, one would have to try every possible a to see whether it is correct. Now in this example with the prime 17, this is easy, but when we take a huge prime number, this is infeasible.

[1] This definition can be found in chapter 1 as well.

Chapter 3

Wireless Protected Access 3

Now we have seen how wireless network in general work and what kind of attacks are possible against such networks, we will discuss WPA3 and see what attacks are no longer possible and, most importantly, what attacks are still possible for an attacker. Firstly we will see what is changed in WPA3 and how this improves the security. We will also discuss the deployment of this new standard and make predictions on when we can expect this to be widely used. The concepts of authentication and association phase, the four-way handshake and data encryption before are described as used in WPA2. However, these are used with some small changes and additions in WPA3 as well, as we will see now when discussing the two main subjects of WPA3: Opportunistic Wireless Encryption and Simultaneous Authentication of Equals.

In both of these, the discrete logarithm problem, as discussed in the previous chapter, is important.

3.1 Simultaneous Authentication of Equals

Now that we have discussed the discrete logarithm problem, we can see how this is used in SAE, short for Simultaneous Authentication of Equals [15]. The goal of the protocol is to generate a cryptographically strong shared secret for securing other data like network communication. The protocol, between client and access point, starts when one of the parties discovers the other and wants to connect. We assume a group and a prime number r are known. The protocol can be initiated by both sides. Before sending any messages, both the access point and the client select a password element (PWE). This is done based on the identities of both parties and the password. This can be done in multiple ways, depending on the type of group that is used. Now the PWE is known, the access point selects two random numbers, randA and maskA. The client does the same and chooses randS and maskS. After this the client and the access point produce

a commitScalar and a commitElement, respectively:

$$commitScalarS = (randS + maskS) \mod r$$
$$commitElementS = PWE^{-maskS}$$

and

$$commitScalarA = (randA + maskA) \mod r$$
$$commitElementA = PWE^{-maskA}$$

These scalars and elements are exchanged and are then used by the client and the access point to compute the shared secret K, respectively:

$$K = (PWE^{commitScalarA} * commitElementA)^{randS}$$

and

$$K = (PWE^{commitScalarS} * commitElementS)^{randA}$$

From this K, a Key Confirmation Key (KCK) and a Pairwise Master Key (PMK) are derived. The KCK is used to verify the key they agreed on. To do this, the client sends the following to the access point:

$$HMAC_{\text{KCK}}(commitScalarS, commitElementS, commitScalarA, commitElementA)$$

and the access point sends to the client:

$$HMAC_{\text{KCK}}(commitScalarA, commitElementA, commitScalarS, commitElementS)$$

The HMAC function computes a message authentication code (MAC) involving a hash function and a secret cryptographic key, in this case the KCK. When both parties have verified that the keys are generated and shared succesfully, the authentication is successful and the four-way handshake as discussed before is performed with the PMK calculated in this SAE.

Because of the fact that the PWE, randS, maskS, randA and maskA are never sent directly to one another, an attacker could never calculate the commit messages and thus never compute the K to derive the keys. An attacker could capture the commit messages itself, but due to the discrete

logarithm problem, he is not able to recover the initial password element or any randomly chosen values.

The old dictionary attack, simply trying all passwords, will be way more time consuming to perform, compared to when the attack is executed on a network with WPA2. When an attacker tries a password, the whole protocol has to be executed before the attackers gets any feedback whether the password was correct. This way only one password can be tried at a time, which makes guessing a lot of passwords inefficient. Next to that, the attacker can now no longer capture the four-way handshake and after that try passwords online. Since the SAE protocol needs to be executed for every try, the attacker has to remain online.

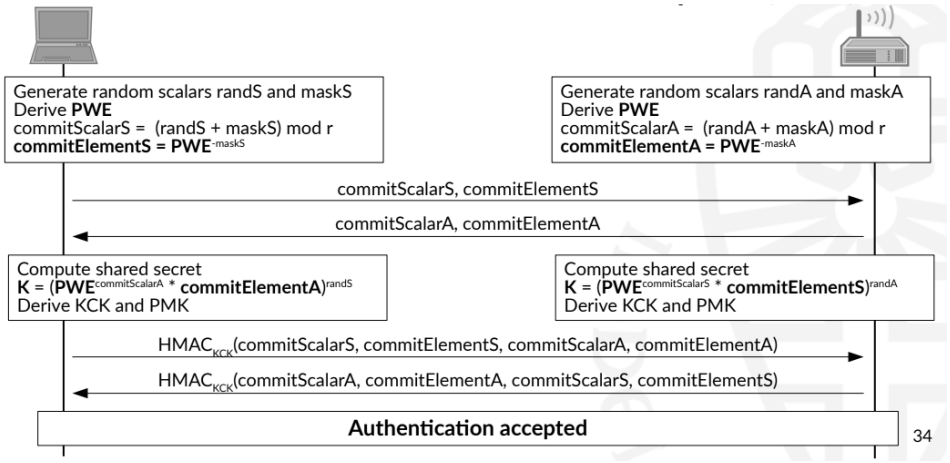

Figure 3.1: Simultaneous Authentication of Equals [9]

3.2 Opportunistic Wireless Encryption

Opportunistic Wireless Encryption, abbreviated as OWE, is based the discrete logarithm problem as well, like SAE. The goal of OWE is equal to the goal of SAE, to generate a shared secret used as input to the four-way handshake, rather than a publicly shared pre-shared key. However, OWE uses a well known protocol to create this computable infeasibility, namely the Diffie–Hellman key exchange protocol. This protocol is, in this case, used to exchange the Pairwise Master Key (PMK). Both the access point and the client have a private 'key' a. This key is actually just a number that is member of a group, with size p-1, with p prime. This group is public and thus known by both parties. They will also decide on a public prime base element g out of this group. The AP as well as the client will now compute an intermediate value:

$$v = g^a \mod p$$

The result of both parties will be sent to one another. Now both will compute the following:

$$k = v^a \mod p$$

This will be the result of the Diffie-Hellman key exchange. The following example might make this more clear: Let's say Alice and Bob want to exchange a key. We have the following (p and g are public, both parties have a private a):

$$p = 19$$
$$g = 13$$
$$a_{\text{Alice}} = 8$$
$$a_{\text{Bob}} = 11$$

Then Alice and Bob would compute and send to each other the following, respectively:

$$13^8 = 815730721 \equiv 16 \mod 19$$
$$13^{11} = 1792160394037 \equiv 2 \mod 19$$

Now Alice and Bob can calculate the shared key, in this case 9, respectively:

$$k = 2^8 = 256 \equiv 9 \mod 19$$
$$k = 16^{11} = 17592186044416 \equiv 9 \mod 19$$

Because of the discrete logarithm problem, it is impossible for an attacker to find out the a's or the key.
We will now look at how OWE is arranged in case of an open or public network. The key exchange happens in the association phase. The following element will be included in both requests as the responses [16]:

Element ID	Length	Element ID Extension	element-specific data	
255	variable	32	group	public key

The three most interesting fields are the variable field, the group field and the public key field. The variable field is the public prime base element out

of the group, g in our example above. The group field contains an integer which stands for the prime group modulus, such as p above. The last of the three, the public key field contains the result of raising the variable g to the power of the private key of the client (C) or the access point (AP) and taking the modulo p of this, respectively:

$$publickey = variable^{privatekey_{\mathrm{C}}} \mod group$$
$$publickey = variable^{privatekey_{\mathrm{AP}}} \mod group$$

This way a Pairwise Master Key is exchanged using the Diffie-Hellman key exchange protocol and this PMK can be used as input for the four-way handshake to establish the session keys.

3.3 Attacks in public networks continued

Now that the changes in WPA3 are discussed, we can see how this impacts the possible attacks we discussed in the previous chapter:

- **Eavesdropping**: This attack could be performed in open network without protection and in networks that are secured with WPA2. In WPA3 OWE is used in the first scenario. This means that all data is encrypted with a shared secret generated with a Diffie-Hellman key exchange. Because of this encryption, an attacker can no longer see the content of messages that are sent in the network. When WPA2 is used, eavesdropping is also possible when the attacker is in the network, and thus knows the pass phrase. This is because in WPA2, the same keys for encryption are used by all clients in the network. With WPA3, and more specifically SAE, every conversation has its own shared secret to encrypt the data, thus eavesdropping is no longer possible.

- **Password cracking**: Because SAE in WPA3, this attack is no longer possible. Every time a attacker tries a password, the whole SAE protocol has to be executed which takes significantly more time than trying a password took before WPA3. Because of this delay, trying lots of passwords will take a huge amount of time and thus will be no longer feasible. With WPA2, an attacker could capture the packets of the four-way handshake and if this was done successfully, the attacker could try to crack the password offline. With SAE this is no longer possible. Every password guess has to be sent to the access point in order to check if it is correct. The attacker is now forced to stay online during the attack, and the network could limit the amount of guesses the attacker can make.

- **Evil twin attack**: We have seen that WPA3 solves the attacks above, but it does not solve evil twin attacks. Actually, nothing has changed regarding this kind of attacks. Someone malicious could still set up a network and perform to be the original network. Because this problem is not fixed in WPA3, we will propose a solution to this problem in this thesis.

3.4 Deployment of WPA3

In this section we will look into the deployment of WPA3. To this end we first look back at the deployment of WPA and WPA2, which will help us to understand how such a deployment takes place in practice. After that we will review the documentation on WPA3 and discuss the future plans for this standard.

As discussed in the preliminaries section, WPA was released after WEP was proven to be very insecure. As mentioned, because TKIP, a wrapper around RC4, was used in WPA, only a firmware update was needed for the devices that used the wireless standard. This was the reason WPA could be released and deployed quickly. Most of the old devices used for WEP would work with WPA and thus did not have to be replaced. However, this holds only for all devices with a wireless network interface card. The firmware on this wireless network interface card could be updated to work with WPA. Access points and routers from before 2003 could mostly not be updated to support WPA. However, it is more feasible to replace the access points than to replace all the client devices. The idea behind the release of WPA was to give a temporary solution to the problems found in WEP. In the meantime the Wi-Fi Alliance would remain to work on a whole new standard, WPA2.

In 2004, Wi-Fi Alliance released a new version of the *IEEE Standard for Information technology—Telecommunications and information exchange between systems Local and metropolitan area networks—Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications* [20], better known as IEEE 802.11. This was, from this moment on, the new desired standard for wireless networks, which could be implemented as WPA2. This was also the moment Wi-Fi Alliance started to certify devices with the WPA2 certificate if the devices met the conditions. From March 2006, de WPA2 certificate was mandatory to get the Wi-Fi certificate. This certificate means the devices where tested, and when the new standard was implemented correctly, it would be granted the right to use the Wi-Fi CERTIFIED logo. On the website of Wi-Fi Alliance we can see that *"Wi-Fi CERTIFIED^{TM}*

is an internationally-recognized seal of approval for products indicating that they have met industry-agreed standards for interoperability, security, and a range of application specific protocols" [1]. So we can see that it took around 2 years from the release of the WPA2 standard until the moment it became one of the requirements for all new wireless devices. The most important reason for this delay is the fact that for this new standard the devices all needed to replaced, instead of getting a firmware update.

In 2016, Wi-Fi Alliance released a new version of the standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. In this version we now see the new protocols Simultaneous Authentication of Equals and Opportunistic Wireless Encryption as discussed in chapter 2. On one side, these two new techniques are great improvements for the wireless security, on the other side these are the main reason that the deployment of WPA3 takes a lot of time. At the moment of writing this, it has already been three years since the release of the of the new standard by Wi-Fi Alliance and there are still no to very few devices that support WPA3 out there. For this reason, we had to experiment with WPA_Supplicant environment. In the official Wireless Protected Access 3 Specification Version 1.0 [2] is stated that new devices that do support WPA3, will be able to get a Wi-Fi CERTIFIED WPA3^{TM}, just like the WPA2 certificate. Although Wi-Fi Alliance did announce that such a certificate will be available, they did not specify any moment in the future when it will be mandatory for the overall Wi-Fi CERTIFIED^{TM}. This together with the fact that we are now already waiting longer for this to happen than when WPA2 was introduced, we can not make any accurate estimate.

3.5 Playing with WPA3

Because WPA3 is not used widely yet, but we still wanted to test it and play around with it, we had to test in different ways. Here is described how this can be done. We used WPA Supplicant [33]. This is a test environment for wireless networks. It offers lots of tests for all kind of protocols, including OWE and SAE. The environment can be found on `https://w1.fi/wpa_supplicant/`. Cloning the git repository that is on this website will give a folder named `hostap`. In `hostap/tests/hwsim` a file named `example-setup.txt` can be found. Following this guide will lead to the correct setup. Once everything is setup correctly we can try to run the tests and capture the corresponding packets. The script will setup a virtual access point and a virtual client. These two then connect using the chosen

protocol, OWE or SAE in our case. In the virtual environment created during the setup, we can switch to another terminal. There we can run tshark and capture on the `hwsim0` interface, as illustrated in Appendices: figure 2 and figure 4. Tshark is the terminal oriented version of Wireshark. While this is running we run the tests for OWE and SAE in the first terminal, as can be seen in Appendices: figure 1 and figure 3.
After we successfully captured the tests, we send the capture files to our own live environment. This can be done in multiple ways, in this case git was used. Now we can open the files with Wireshark to analyze the data. However, the standard downloads that can be found on their site, do not yet support SAE and OWE. We have to download a newer version, that can be found on their website as well. `2.9.0` is the version that we used during this project. Following the guide on how to build the application from source that is on their website, this version of Wireshark can be installed. We now opened the capture files in there and analyzed the data packets, as illustrated in Appendices: figure 5 and figure 6. This was helpful for a better understanding of the WPA3 protocols and finding out exactly where the additional information for these protocols were added. We could see that the key exchange in SAE can be found in the authentication phase (Figure 5: the two blue highlighted lines) and that the Diffie-Hellman key exchange in OWE was added in the association phase (Figure 6: the two blue highlighted lines).

Chapter 4

Defending against evil twin attacks

In this chapter, we will first have a closer look at the problem we are trying to solve. We have seen that OWE offers better security for public networks, but that it does not solve all problems. A remaining problem is the so-called *evil twin attack*. That is, one could set up a network with the exact same name as a public network, at the same place. Users will be unable to tell the difference. When the attacker is lucky, a victim will connect to his malicious network. Session keys are then exchanged between attacker and victim, so the attacker is able to decrypt all the data that is sent back and forth.

Luckily, applications can use TLS to encrypt the data that is being sent back and forth between the application and the server. However, not all applications and websites use this technique, so sometimes user names and/or passwords will pass the access point of the attacker, which is extremely bad practice. All he has to do is decrypt the data like a trustworthy access point would do, inspect the decrypted data and look for interesting data. Even if the application hashes the password instead of sending it in plain text, the attacker could save the hash value and sent it to the server when the password is required. The fact that some users use the same password for multiple, or even all purposes, makes this problem even more serious. Finding credentials for a small application without any sensitive data, might lead an attacker to credentials for other applications or websites, that do handle sensitive data.

Unfortunately, using TLS is not a perfect solution. Multiple kinds of attacks on TLS are known, such as downgrade attacks [25]. HTTP Strict Transport Security (HSTS) [18] is a security mechanism that protects against these downgrade attacks. However, defending against evil twin attacks is still important. TLS and HSTS are not always used and even if these security measures are used, an attacker can still gather information about clients.

To this end, we will discuss four possible approaches to this evil twin problem and describe how these could be realized. For each approach we will the discuss four criteria;

- **Usability for the client**: This criteria describes how convenient using the approach is for the users. If the user has to perform an action for the approach, the approach will score negative on this criteria. The more technical knowledge that is required to perform this action, the lower the score will be. A positive score is achieved when users will not experience any problems using the approach.
- **Feasibility to implement the approach as an actual solution**: While a solution can be very suitable in an ideal world, it can be nearly impossible to implement it. This criteria describes whether implementing the approach is feasible.
- **Overhead**: Ideally, the user should not experience any delay when connection to a public wireless network with our suggested solution implemented. This criteria is about the presence of a delay and the length of the delay the approach will cause.
- **Effectiveness**: This criteria describes how effective the approach will be. In other words, to what extent does the approach provide the desired security.

4.1 Types of approaches

Now we know the theory of what is new in WPA3 and what is still lacking and causing security issues, we can discuss approaches for our problem. In this section we will discuss the following three approaches and compare them using our four criteria:

- Certificates
- Trust on first use
- Static network names

4.1.1 Certificates

A possible solution that may come to mind is the use of certificates. Certificates are for example used for secured web browsing. The goal of certificates is to prove the identity of a web server, or more specifically, prove the ownership of the private key that is in the certificate. When a client wants to connect to a web server, a Transport Layer Security (TLS) handshake is performed [21]. First the client sends a *hello message* to the server, telling the

server it wants to connect. This message includes information about what encryption modes the client supports and what version of TLS it uses. The server then checks whether the TLS version and encryption modes match with its own capabilities. If this is the case, the server will respond with a message which includes the certificate. For this handshake, the most important field in a certificate is the public key of the server. This public key is used by the client to send a shared secret to the server, which the server can decrypt with his private key. This shared secret can then be used for encrypted data transfer.

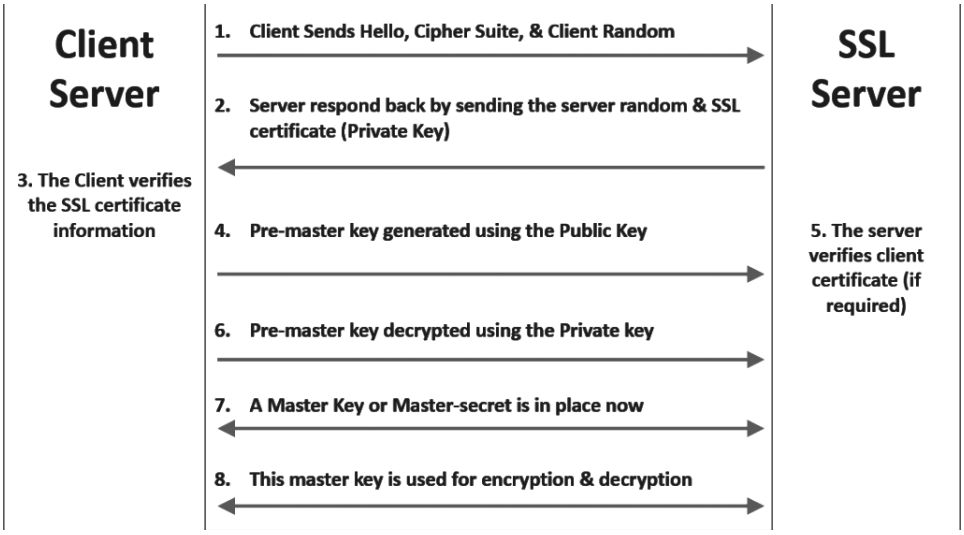

Figure 4.1: Transport Layer Security handshake [6]

Next to the public key field, other information can be found in the certificate. The most important fields are the issuer and subject fields. The subject of a certificate, is the entity the certificate belongs to. In the case of a client connecting to a web server, the subject is the web server. The issuer is the entity that has verified the information in the certificate and signed it, using its private key. In practice, certificates are signed by a Certificate Authority (CA). A subject can request a certificate. The CA will then check the data about the subject, and if everything is correct, the subject will receive a signed certificate. In this way, CAs are used as a trusted third party. This is the main downside of certificates. Third parties are undesired because it needs a complex infrastructure between subjects, issuers and clients. Furthermore, when a CA is compromised, certificates are no longer secure. This means there is a single point of failure. Another field in certificates is the validity period, in which the certificate is valid. Information about the different uses of the certificate and the public key are also in the certificate. Next to web browsing, for example it can be used for email protection or signing other certificates. The last two fields contain the information about the signature of the CA. The signature itself can be

found in the certificate, as well as the algorithm used to sign the certificate [4].

This technique, as used in web browsing, could also be used in our case about public wireless networks. For example, a coffee shop that wants to set up a public network for its guests, will ask a CA for a certificate. CAs in this case could be the same as in the case of web browsing. This would mean CAs would have to widen their scope. Alternatively, new CAs could be introduced, specially for public networking. In both cases, a client would need a list of trusted CAs and their public keys before connection to a network. Applications used for browsing the web, like Chrome and Firefox, have such a list built in. Such a list should be built in the device software or firmware in our case. This would require companies to update their software or firmware regularly. Then clients, with a CA list at hand, could check the certificate of a network by checking the signature with the public key in the list.

Why certificates do not work

However, there are reasons why certificates is not a workable solution. Firstly,there is a identification problem, i.e. it is difficult to determine what should be identified. For web browsing the domain name is used, because a domain name can only be used by one person or organization. Wireless networking does not have such a restriction to the network name. In web browsing the party that hosts the website can prove that it is the only that uses that domain name and that it is the one that has the administrator rights the the website. In our case, it is impossible to prove to the CA that a party is the only one that uses the network name and that it is the only one with administrator rights. This way multiple certificates for the same network name could be distributed, which makes it possible for an attacker to get a certificate for his evil twin network as well. Hypothetically, this could be solved by showing clients somewhere clearly visible in the room which certificate the trustworthy network uses. However, the identification problem remains. It is very difficult to prove to the CA that a network belongs to a host.

Secondly, this will not work for consumer networks. Someone setting up a open network at home for his guests will have problems getting a valid certificate for his network. This might be costly, which is undesirable for small consumers. There exist free options to get a certificate, but these are based on automated distribution of certificates. Because of our identification problem, it is very hard to automate the distribution of certificates in the case of Wi-Fi networks. Next to that, consumers do not have a organization that they can prove the network belongs to. Then again we see the same problem as stated above; there is no way to identify the party that hosts the network.

In the thesis written by Matthias Ghering [13], we find a third problem with this approach. In this thesis evil twin attacks on Enterprise networks are discussed. As an example the Eduroam network is used. In such a network, certificates are already used. However, on the client side these certificates need to be installed. Because of the fact that some devices do not support the correct settings and that other devices make it very hard to set this up correctly, the certificates are often not installed correctly. This way the client device will accept any certificate that is presented.

- **Usability for the client**: Most people are already used to this approach from web browsing. This works exactly the same for the client. When the access point has a valid certificate, the client will not even notice this approach, because the device performs the check. When the certificate is not valid, the user should be informed about this. This can be done by using the same green lock icons that are used by most web browsers. This icons could be shown next to the access point's name in the list of available networks on the device. When a client decides to connect to an access point without a valid certificate, or the certificate of a known access point is expired and the client wants to connect, the device should inform the user that the network might not be safe. Like in web browsers, clients can still connect to the possibly unsafe network in an advanced options section. However, as we discussed above, setting up the use of certificates at the client side is very difficult or even impossible. Using the correct settings is necessary for this approach to work. Next to that, this approach might not be ideal for consumers. Getting a certificate might be costly. Because of these two issues, the approach does not score very well on usability.

- **Feasibility to implement the approach as an actual solution**: As discussed above it is hard to implement this as a solution because there is no way to identify the host of the network, because of the identification problem. Next to that, it will be difficult or impossible for users to set up their device correctly.

- **Overhead**: This approach can be implemented in such a way that there will be little overhead. Just like in web browsing, checking the certificate does not cost much time. The client will barely notice that the device performs a check on the certificate.

- **Effectiveness**: Assuming there was a way to identify the host of a network, certificates would be an effective approach. However, as we already discussed this is very hard to do, which makes this approach not that effective. Because certificates are based on a Public Key Infrastructure (PKI), it would be effective if the identification problem

could be solved. CAs sign the certificates they distribute with their private key. Users can, with the public key of the CA, check the signature. Attacking certificates would require an adversary to generate fake certificates. However, since certificates are only accepted by clients when the certificate is signed correctly, an adversary would need to generate a fake signature. To do this, one would need to crack the private key of the CA. However, because of the key lengths that CAs use, this is infeasible. For example, if we have a look at *Let's Encrypt*, which is a popular CA, their documentation tells us that private keys are at least 2048 bits long [11]. Such key lengths are unbreakable. For this approach we have to assume that CAs are trustworthy and do their work well. Then adversaries will not be able to get a valid certificate. The risk of this approach is the fact that users might make wrong decisions. A client can choose to connect to an access point without a valid certificate. In this case, security can not be guaranteed.

4.1.2 Trust on first use

A better approach is trust on first use. This security model is used by clients to establish a connection with an unknown endpoint. Generally, a client will try to look up the endpoint in his local database of trusted endpoints. If it is in there, this means the endpoint is known and trusted. If there is no entry for this endpoint, one is added.

In our case a combination of the network name and the public key belonging to this network will be saved. This public key is generated the first time the network is set up. A private key is then generated as well. When connecting to the network, the access point will send the client a message that is signed with the private key of the network, together with the public key of the network. The client can now decrypt this message with the public key. If this is done successfully, the client knows that a valid key pair is used and the public key can be saved together with the network name, which is also called the SSID of the network.

Using trust on first use, only new clients are possible victims. Clients who have already connected to the trustworthy access point in the past, will see that the key pair of malicious access point does not match the entry in the trust database, and a connection will not be made. For new clients, there is a chance they pick the attacker's network instead of the original, trustworthy one. Now the client will add an entry into the trust database. Usually, attackers can perform their attack only a limited amount of time. So when a victim tries to connect to the network later, only the trustworthy network is available. The client will then notice that the public key does not match the public key of the saved network and the client will notice an attack has been performed or is being performed at the moment. This way the client will find out something went wrong. Even though the harm

is already done in the first case, the client will be informed that something suspicious happened and the client will be able to take action. This is already an improvement to the current situation.

- **Usability for the client**: This approach can be implemented in such a way that the user will not notice it. The device of the user does all of the work, and the client can use it the same way as without trust on first use. Only when the public key that is saved locally does not match the public key of the network that is presented, the user will notice that this approach is used.
- **Feasibility to implement the approach as an actual solution**: Trust on first use is used in other scenarios, such as SSH. A similar implementation is possible for wireless networking. A downside is the fact that a database of trusted access points should be saved locally, which requires some space in the memory of the device. This may be a problem for Internet of Things (IoT) devices. IoT devices do not have memory to maintain such a database, which means trust on first use is not a option. Because of this, IoT devices should always use TLS when communicating with other devices.
- **Overhead**: This approach has little overhead. We can consider two cases; the case where the access point is known and in the database and the case where the access point is unknown. In the first case, the device will find the access point in the database, check the public key and the device will establish the connection. This can be done in a short time. In the latter case, the device will notice the access point is not in the database and a new entry is added, which can be done in little time as well.
- **Effectiveness**: This approach is not perfectly effective. In the case where the user has already connected to the network sometimes before an attack is performed, the attack itself will be noticed by the device. When the client wants to connect to a network for the first time and an attack is performed at that time, the user might not notice that the network should not be trusted. Because the latter case will only very rarely, this approach still can be seen as effective and as an improvement.

4.1.3 Network names

In this section we will discuss the following two approaches that use the network name, the SSID:

- Static network names
- Public key as network name

Static network names

A simple approach is making it impossible to change network names. Like the unchangeable Media Access Control (MAC) addresses used for devices in the data link layer, routers and access point could have a static network name. When setting up a network, the name is in the current situation one of the fields that can be changed. This is no longer possible with this approach. When the network name can not be changed, it could be written somewhere visible in the room, for example in the coffee shop. It is impossible for an adversary to use the same network name, thus different networks can easily be distinguished by the client user. The downside of this approach is that is requires human activity. The user has to look for the correct network name somewhere in the room and compare it to the network it connects to. To make this more convenient, the client device can show a pop-up with the question whether the network names are equal. The user can then confirm to establish the connection, or reject the network if the network name is not correct. This technique can be combined with trust on first use. In that case a user has to perform this action only once, when connecting for the first time.

Although this approach looks simple and secure, it has a big security flaw. Like MAC addresses, it should be impossible for users to change the network name. However, making sure this is really impossible is hard. Changing MAC addresses, which is called MAC address spoofing, is an easy thing to do for an adversary. It appears it is hard to make MAC addresses really static, which could be the case as well with static network names. A possibility for an adversary to change the network name, will make this approach completely useless.

- **Usability for the client**: The usability of this approach depends on how things are organized. If network names are chosen completely at random, it will require some effort from the user to compare this to the correct network name which can be found somewhere nearby. However, when it is possible for the owners of access points to choose a network name, it will require less effort, because it will consist of known names or number combinations, which make sense to the user.

- **Feasibility to implement the approach as an actual solution**: Implementing this solution is very challenging when the network name should be unchangeable. When we look at MAC addresses, we see that it is almost impossible to make a MAC address, or in our case a network name, really static. MAC addresses are used as a unique ID for a Network Interface Card (NIC). The manufacturer gives the NIC its MAC address, and the MAC address is linked to the hardware part of the NIC. This is, as desired, unchangeable. Similarly, this could be done with access points as well. The network name could be linked

to the hardware and thus be unchangeable. However, although MAC addresses are static in the hardware, it is possible to fake the address in the software. This is called MAC spoofing. In our case, network name spoofing could be performed similarly. This attack is very hard to defend against.

- **Overhead**: This approach has little overhead, because the way of connecting to an access point does not really change. The only thing that changes is the fact that a user now needs to perform a check. This will cost some time, but the actual connecting to the access point will barely be slowed down.

- **Effectiveness**: As mentioned above, clients have to perform a check. This is the reason this solution is not completely effective. Clients can make mistakes. When we assume that users are able to perform the check perfectly every time and the network name is really static, we can say this solution is perfectly effective. An adversary will not be able to convince clients that his network is trustworthy because the difference in network names will reveal the attack. When comparing this approach to trust on first use, static network names is less effective. This is because it depends on human beings performing a check, where trust on first use can be fully automated. The chances the device makes a mistake is lower than the chances a user makes a mistake.

Public key as network name

Another way of using the network name to defend against evil twin attacks is to use the public key of the network as network name. This approach fixes the main problem of our trust on first use approach, namely the fact that the first time connecting to a network is not secure. Now the network should generate a combination of a public and private key when setting up the network the first time. The public key then becomes the network name. Now a client that wants to connect can use the public key in the handshake to determine that the network is the original network with that network name. The fact that only the original network has access to the private key, makes it impossible for an attacker to pretend to be the same network. This approach is similar to hidden services in Tor, the onion router [10]. These hidden services use a domain name in which the public key of the host of this hidden service is used. A downside to this way of using the network name is the fact that the user can no longer see to who or what the network belongs. This can be solved by showing the network name somewhere in the room. However, the user then has to check for the correct public key which is less user friendly. Next to that, public keys are strings that make no sense to the user. Users might not be able to distinguish the public keys correctly.

- **Usability for the client**: The user will now have to compare the public key with the public key that can be found somewhere nearby. Public keys are random strings which makes this harder to do for the user. This means this approach is less user friendly.

- **Feasibility to implement the approach as an actual solution**: When the public key is used as network name, implementing this approach is not hard. The access point should generate a public and private key the first time the network is set up and the public key should be used as network name. Then the access point can send a message encrypted with the private key and the client will decrypt this with the public key. If this happens correctly, the network can be trusted.

- **Overhead**: This approach has little overhead, because the way of connecting to an access point does not really change. The access point has to encrypt one of the messages in the handshake with its private key and the client has to decrypt this message with the public key. This is simple cryptography which can be done in little time. The only thing that changes is the fact that a user now needs to perform a check. This will cause some inconvenience, but the actual connecting to the access point will barely be slowed down.

- **Effectiveness**: This approach can be effective. However, users can make mistakes performing the network name check, which might cause problems. If a user decides to connect to another network than the one that is visible somewhere in the room, security can not be guaranteed. Here we see the same problem as with static network names. Because this approach depends on human interaction, it is less effective than trust on first use.

4.1.4 Evaluation of approaches

Regarding the analysis of the approaches in this section, trust on first use is chosen to mitigate against evil twin attacks. Using the public key as network name might be more effective if the user is able to verify network name successfully every time. However, it is not user friendly. We decided usability is more important than the the risk trust on first use has when connecting for the first time. In section 4.2 we will discuss how trust on first use could be implemented in more detail and we will consider different scenarios in which this approach can be used.

4.2 Implementing trust on first use

As stated in section 4.1.2, the access point will generate a public key and private key the first time it is set up. Actually, every time the network name is changed by the administrator, a new key pair should be generated. Access points and routers that are currently used, should get a firmware update to be able to generate such a key pair. However, such changes will require the whole security protocol to be updated. This means a new WPA version is needed where trust on first use is a part of. After the key pair is generated, clients can connect to the network. As described in chapter 2, the authentication and association phase are firstly performed. The access point then computes a hash value of these two phases, signs this hash with its private key and sends the following message to the client during the four-way handshake:

$$publickey, (Hash(authenticationphase + associationphase))_{privatekey}$$

The client computes also the hash value of the authentication and association phase. When the client receives the signed hash value as described above, the client uses the public key to verify the signed hash value. Now the client compares the hash value with the hash value that it computed itself. If a valid key pair is used, these two values will be equal. If these values are not equal, the key pair that is used was not valid and the connection should not be made. In case the client connects to the network for the first time, the public key, together with the network name, will be locally saved. If the network name is already locally saved, together with a public key, then the saved key should be used to decrypt the hash value, instead of the public key that is sent alongside the signed hash value.

When this approach is used, and a adversary sets up an evil twin network, the client will see that the network name is already known. When the client then used the saved public key to decrypt the signed hash value, the client will notice that the public key does not belong to this network and thus that the network is not the original trustworthy network. In this case a connection should not be made.

4.2.1 Scenarios

Wireless networks, and thus trust on first use as an approach, are widely usable, so it can be used in different scenarios. We will now discuss the different scenarios and how trust on first use as a solution works in these scenarios.

Consumer networks

Firstly, we discuss consumer networks. Our proposed solution works for consumer networks. When a consumer sets up a home network, the access point

or router can generate a key combination without any human interaction. In this case the approach can be exactly used as described above. Small-scale public Wi-Fi networks, such as the networks used in a coffeeshop, are in this category as well.

Public networks

Secondly, we look at large-scale public networks. As an example we take the well known *WiFi in de trein* network from the Nederlandse Spoorwegen. This network is present in all trains from this organization. A client has to accept the terms and conditions, and after that is done the client can use the free public network. This network is present at a lot of places under the same network name. This means that all these networks should use the same combination of private and public key. So the Nederlandse Spoorwegen should distribute the same keys to all the trains that they use. For an attacker to break the security trust on first use offers, an attacker would have to get the private key. Because this key is saved in all access point, an attacker could try to find an access point and physically connect it to his own device to get to the private key. In case of WiFi in de trein, the access points are hidden in such a way that they can not be easily found. This makes it harder to perform this attack. Next to that, such networks are mostly located at busy places. Thus social control makes it less likely for an attacker to get access physically to an access point.

Enterprise networks

Lastly, trust on first use should be suitable for networks with WPA Enterprise. As an example we look at *Eduroam*, the network used in most educational environments. This network lets users log in with their own credentials, i.e. a email address and a password. When logged in successfully, the user has access to the internet via this network at schools and universities all over the world. Because all this educational institutes use Eduroam as a network and thus have the same network name, all this institutes should use the same combination of public and private key. Where this was not a big problem for public networks like WiFi in de trein, because it is only one organization, it is more of a problem for a network like Eduroam. Sharing the same key pair over these numbers of organizations is practically hard, but also a security risk. All organizations that use eduroam would have access to the public and private key combination. This means the risk that the private key will fall into wrong hands is higher. When the private key would be leaked, our proposed solution would be fully compromised. Thus, for enterprise networks, the distribution of the public and private key requires special care and policies.

Chapter 5

Related Work

Ever since the introduction of wireless networks, adversaries have been looking for new ways to attack such networks. On the other side, people have been trying to mitigate against these attacks and making wireless networks more secure. Man in the middle attacks, eavesdropping, password cracking attacks and evil twin attacks are the most important attacks. For these attacks, security measures have been proposed. Related work on these attacks is discussed in this chapter. Section 5.1 covers eavesdropping, password cracking is discussed in section 5.2, evil twin attacks are addressed in section 5.3 and in section 5.4 man in the middle are discussed. Because of the changes in WPA3, eavesdropping and password cracking attacks are no longer possible. Evil twin attacks and man in the middle attacks are still things someone malicious could do.

5.1 Eavesdropping

The possibility to eavesdrop on public networks and countermeasures against it are discussed in a research by Cheng et al. [7]. This research is about eavesdropping in public networks. Cheng et al. came to the conclusion that privacy leakage can be up to 68 percent as a result of such an attack. This result is based on data traffic they analyzed at twenty airports in four different countries. However, with WPA3, and more specifically Opportunistic Wireless Encryption, eavesdropping as in this research is no longer possible.

5.2 Password cracking

Networks protected with a password can be attacked with password cracking. The idea behind this attack is simply trying all possible passwords until the correct one is found. Brute forcing passwords like this is analyzed by Visan [31]. They calculated that it would take 351 years to crack a password consisting of eight alphanumeric characters, both uppercase and lowercase.

To speed this process up, they propose to use a graphics processing unit instead of a central processing unit. This would be significantly faster, with an expected fourteen months of time needed to crack a similar password. They propose that passwords should be longer, at least twelve to fourteen characters and that passwords should be as random as possible with special characters included.

With the introduction of WPA3, and more specifically SAE, password cracking in this way is no longer possible. The SAE protocol has to be executed completely before an attacker gets the feedback whether the password was correct. Because of the relatively long execution time, guessing passwords will take much more time, making it infeasible to perform such an attack. Next to that the attacker can no longer capture the four-way handshake and try to crack the password offline. Every guess has to be sent to the access point and thus the attacker has to stay online during the whole attack. Also, the network could limit the amount of guesses the attacker is allowed to make, which makes it even harder for the attacker to crack the password.

5.3 Evil twin attacks

On evil twin attacks, research already has been done as well. We could divide this into two categories; on one side, solutions for which users actively have to perform an action, and detection of evil twins on the other side.

Roth et al. published a solution of the first category, where the user actively has to perform an action [23]. In their case, the access point is placed such that it is clearly visible for all clients. A light with multiple colors is added to the device. When a client wishes to connect, the access point will show a sequence of colors, which the client has to enter on their device they want to connect. When the sequence is correct, the client knows the access point is legitimate and the connection is established.

In the other category, a research has been done by Gonzales et al. [14] Their research was focused on how to detect an evil twin. The first approach discussed works with comparing the set of SSIDs at a location. That is, the first time someone connects to a public network at a given location, that client makes up a set of all available access points at that location. The assumption here is that the access point to which the client connects the first time, is trustworthy. When the client returns to that location later, to connect to an access point again, the set of all available access points is compared to the set the client gathered the first time. When these sets are equal, there is no evil twin present. When the sets are not equal, and all names of the network are the same, then there is probably an evil twin at work. A downside to this technique is when a router or access point is changed. This will result in a false positive.

In the second category we also see a research done by Mustafa and Xu [22]. They propose to look at the time it takes to send and receive packets. They state that genuine access points are mostly faster and especially more constant. The reason behind the delay and differences in packet time is the fact that evil twins most of the time change or analyze the data that is sent. Other reason might be that the evil twin performs as a *man in the middle*. In this case all data has to be forwarded to the legitimate access point and back to the client. By analyzing the time that it takes to send and receive packets Mustafa and Xu are able to get a hundred percent detection accuracy, when the adversary is using a mobile phone to set up an evil twin. When the attacker is using a laptop with software designed to launch evil twin attacks, the accuracy is about 98 percent.

In this category we also see the work of Bahl et al. [3]. They present a cheap framework that can detect evil twin attacks. This framework is called DIAR.

5.4 Man in the middle attacks

A special kind of evil twin attacks is man in the middle attacks. An attacker can set up an evil twin network and provide the internet to the clients itself, or the attacker forwards all data to a legit access point to connect to the internet. In the latter case, this attack is actually a man in the middle attack. A man in the middle attack is performed by an adversary, who is forwarding all data between two clients, or a client and an accesss point, and the attacker makes them think they are communication with each other directly. Because all data passes via the attacker, he is able to inspect and even alter the data. The solution that Choi et al. proposed, was to add strong encryption and strong authentication of both devices and users [8]. Strong authentication is probably the most effective way to prevent man in the middle attacks.

Chapter 6

Conclusion

From chapter 2 we can conclude that Wireless Protected Access 3, and more specifically Simultaneous Authentication of Equals and Opportunistic Wireless Encryption, will increase the level of security in Wi-Fi networks significantly. However, we have also seen that the level of security is still far from perfect, and focused on the evil twin attack in this thesis, since there are no improvements in WPA3 regarding evil twin attacks. We have seen that multiple countermeasures exist against this kind of attack, such as the use of certificates, static network names, public key as network name and trust of first use. Even though the use of certificates may seem an obvious solution because we already use it in web browsing, it does not work in the case of wireless networking. Instead we have concluded that the use of "trust on first use" is the best option regarding the four criteria we used. We have elaborated on how this could be implemented and how the approach would work in different scenarios. Some additional information has to be sent during the four-way handshake, next to the regular packets transmitted with the current implementation. This can be used to implement our provided solution in combination with WPA3, or one of the standards that is still to come.

However, our proposed solution alone is not enough to offer the desired security. Next to our solution to evil twin attacks in the link layer, security could be also offered in other layers. In the transport layer TLS, short for Transport Layer Security, is an important protocol that should always be used. Trust on first use combined with TLS offers the best security. When TLS is already used, trust on first use is still a useful improvement. An attacker can still gather information about clients connected to his network when TLS is used.

A downside of our proposed solution is that all devices need a firmware update. Access points and routers will need a update that makes it possible to generate a pair of a public and private key, calculate a hash value and sign this hash value. Client devices such as mobile phones and laptops will

have to be updated to enable the device to compute the same hash value, decrypt the signature and locally save the public key. An update this big will mean a new security protocol is needed and thus a new WPA version has to be released. As discussed in section 3.4, deploying a new security protocol takes time.

Because trust on first use does not guarantee security when connecting to a network for the first time, we want evil twin attacks to be only performed for a limited amount of time. Bahl et al. presented a cheap framework called DAIR that can detect evil twin attacks [3]. This framework can be used by administrators so that they can detect evil twins. As discussed in chapter 5, Gonzales et al. [14] proposed a way to detect evil twin attacks. When combined with this techniques, our proposed solution will offer better security.

Bibliography

[1] Wi-Fi Alliance. Certification, 2019. [Online; accessed May 23, 2019]. URL: https://www.wi-fi.org/certification.

[2] Wi-Fi Alliance. Wpa3 specification version 1.0, 2019. URL: https://www.wi-fi.org/download.php?file=/sites/default/files/private/WPA3_Specification_v1.0.pdf.

[3] Paramvir Bahl, Ranveer Chandra, Jitendra Padhye, Lenin Ravindranath, Manpreet Singh, Alec Wolman, and Brian Zill. Enhancing the security of corporate wi-fi networks using dair. In *Proceedings of the 4th international conference on Mobile systems, applications and services*, pages 1–14. ACM, 2006.

[4] Sharon Boeyen, Stefan Santesson, Tim Polk, Russ Housley, Stephen Farrell, and Dave Cooper. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 5280, May 2008. URL: https://rfc-editor.org/rfc/rfc5280.txt, doi:10.17487/RFC5280.

[5] Nancy Cam-Winget, Russ Housley, David Wagner, and Jesse Walker. Security flaws in 802.11 data link protocols. *Communications of the ACM*, 46(5):35–39, 2003.

[6] CheapSSLSecurity. What is ssl tls handshake understand the process in just 3 minutes, 2019. [Online; accessed May 23, 2019]. URL: https://cheapsslsecurity.com/blog/what-is-ssl-tls-handshake-understand-the-process-in-just-3-minutes/.

[7] Ningning Cheng, Xinlei Oscar Wang, Wei Cheng, Prasant Mohapatra, and Aruna Seneviratne. Characterizing privacy leakage of public wifi networks for users on travel. In *INFOCOM, 2013 Proceedings IEEE*, pages 2769–2777. IEEE, 2013.

[8] Min-kyu Choi, Rosslin John Robles, Chang-hwa Hong, and Tai-hoon Kim. Wireless network security: Vulnerabilities, threats and countermeasures. *International Journal of Multimedia and Ubiquitous Engineering*, 3(3):77–86, 2008.

[9] Joeri de Ruiter. Advanced network security: Wifi security, 2018. [Lecture slides Master course Advanced Network Security from the master Cyber Security].

[10] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. Technical report, Naval Research Lab Washington DC, 2004. URL: https://svn.torproject.org/svn/projects/design-paper/tor-design.pdf.

[11] Let's Encrypt. Integration guide, 2019. [Online; accessed May 23, 2019]. URL: https://letsencrypt.org/docs/integration-guide/.

[12] Scott Fluhrer, Itsik Mantin, and Adi Shamir. Weaknesses in the key scheduling algorithm of rc4. In *International Workshop on Selected Areas in Cryptography*, pages 1–24. Springer, 2001.

[13] Matthias Ghering. Evil Twin vulnerabilities in Wi-Fi networks. 2016. URL: https://www.cs.ru.nl/bachelors-theses/2016/Matthias_Ghering___4395727___Evil_Twin_Vulnerabilities_in_Wi-Fi_Networks.pdf.

[14] Harold Gonzales, Kevin Bauer, Janne Lindqvist, Damon McCoy, and Douglas Sicker. Practical defenses for evil twin attacks in 802.11. In *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*, pages 1–6. IEEE, 2010.

[15] Dan Harkins. Simultaneous authentication of equals: A secure, password-based key exchange for mesh networks. In *Sensor Technologies and Applications, 2008. SENSORCOMM'08. Second International Conference on*, pages 839–844. IEEE, 2008.

[16] Dan Harkins and Warren "Ace" Kumari. Opportunistic Wireless Encryption. Number 8110 in Request for Comments. RFC Editor, March 2017. URL: https://rfc-editor.org/rfc/rfc8110.txt, doi:10.17487/RFC8110.

[17] Changhua He and John C Mitchell. Analysis of the 802.11 i 4-way handshake. In *Proceedings of the 3rd ACM workshop on Wireless security*, pages 43–50. ACM, 2004.

[18] Jeff Hodges, Collin Jackson, and Adam Barth. HTTP Strict Transport Security (HSTS). RFC 6797, November 2012. URL: https://rfc-editor.org/rfc/rfc6797.txt, doi:10.17487/RFC6797.

[19] IEEE Computer Society LAN/MAN Standards Committee and others. 802.11-1997 - IEEE Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications. *IEEE Std 802.11*, 1997.

[20] IEEE Computer Society LAN/MAN Standards Committee and others. 802.11i-2004 - IEEE Standard for information technology-Telecommunications and information exchange between systems-Local and metropolitan area networks-Specific requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Amendment 6: Medium Access Control (MAC) Security Enhancements. *IEEE Std 802.11*, 2004.

[21] Paul Morrissey, Nigel P Smart, and Bogdan Warinschi. A modular security analysis of the tls handshake protocol. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 55–73. Springer, 2008.

[22] Hossen Mustafa and Wenyuan Xu. Cetad: Detecting evil twin access point attacks in wireless hotspots. In *Communications and Network Security (CNS), 2014 IEEE Conference on*, pages 238–246. IEEE, 2014.

[23] Volker Roth, Wolfgang Polak, Eleanor Rieffel, and Thea Turner. Simple and effective defense against evil twin access points. In *Proceedings of the first ACM conference on Wireless network security*, pages 220–235. ACM, 2008.

[24] Douglas Selent. Advanced encryption standard. *Rivier Academic Journal*, 6(2):1–14, 2010.

[25] Yaron Sheffer, Ralph Holz, and Peter Saint-Andre. Summarizing Known Attacks on Transport Layer Security (TLS) and Datagram TLS (DTLS). RFC 7457, February 2015. URL: `https://rfc-editor.org/rfc/rfc7457.txt`, `doi:10.17487/RFC7457`.

[26] Mathy Vanhoef and Frank Piessens. Practical verification of wpa-tkip vulnerabilities. In *Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security*, ASIA CCS '13, pages 427–436, New York, NY, USA, 2013. ACM. `doi:10.1145/2484313.2484368`.

[27] Mathy Vanhoef and Frank Piessens. Predicting, decrypting, and abusing wpa2/802.11 group keys. In *25th USENIX Security Symposium, USENIX Security 16*, pages 673–688, 2016.

[28] Mathy Vanhoef and Frank Piessens. Key reinstallation attacks: Forcing nonce reuse in wpa2. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1313–1328. ACM, 2017.

[29] Mathy Vanhoef and Frank Piessens. Release the kraken: new KRACKs in the 802.11 standard. In *Proceedings of the 25th ACM Conference on Computer and Communications Security (CCS)*. ACM, 2018.

[30] Mathy Vanhoef, Domien Schepers, and Frank Piessens. Discovering logical vulnerabilities in the wi-fi handshake using model-based testing. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, pages 360–371. ACM, 2017.

[31] Sorin Andrei Visan. Wpa/wpa2 password security testing using graphics processing units. *Journal of Mobile, Embedded and Distributed Systems*, 5(4):167–174, 2013.

[32] John Vollbrecht, James D. Carlson, Larry Blunk, Dr. Bernard D. Aboba Ph.D., and Henrik Levkowetz. Extensible Authentication Protocol (EAP). (3748), June 2004. URL: `https://rfc-editor.org/rfc/rfc3748.txt`, `doi:10.17487/RFC3748`.

[33] w1.fi. Wpa_supplicant, 2019. [Online; accessed May 23, 2019]. URL: `http://www.w1.fi`.

[34] The Free Encyclopedia Wikipedia. Block cipher mode of operation, 2019. [Online; accessed May 23, 2019]. URL: `https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation#/media/File:CTR_encryption_2.svg`.

[35] The Free Encyclopedia Wikipedia. Ieee 802.11i-2004, 2019. [Online; accessed May 23, 2019]. URL: `https://en.wikipedia.org/wiki/IEEE_802.11i-2004#/media/File:4-way-handshake.svg`.

Appendices

Playing around with WPA3

These are the screen shots from playing around with WPA3 as described in section 3.5.

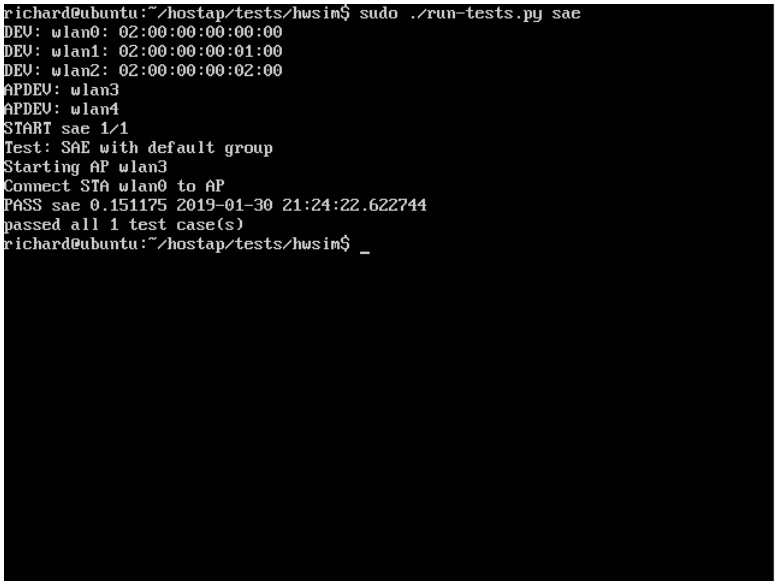

Figure 1: Testing Simultaneous Authentication of Equals with WPA Supplicant

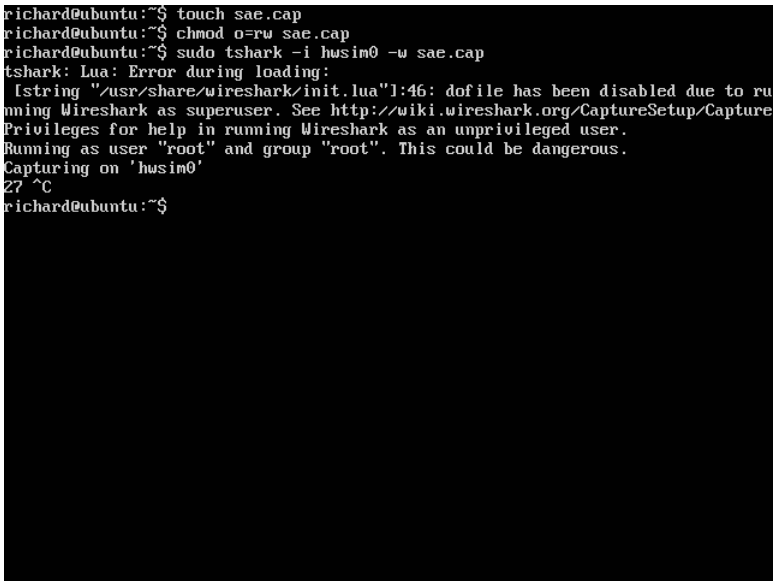

Figure 2: Capturing Simultaneous Authentication of Equals with tshark

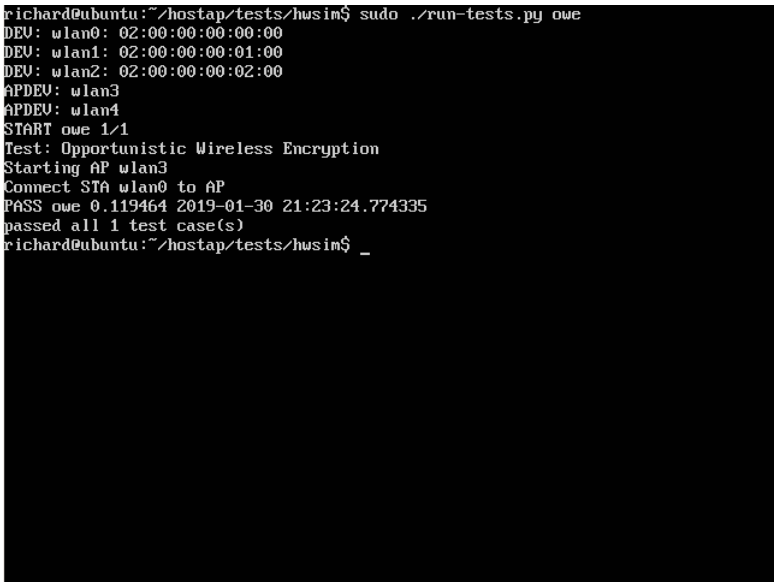

Figure 3: Testing Opportunistic Wireless Encryption with WPA Supplicant

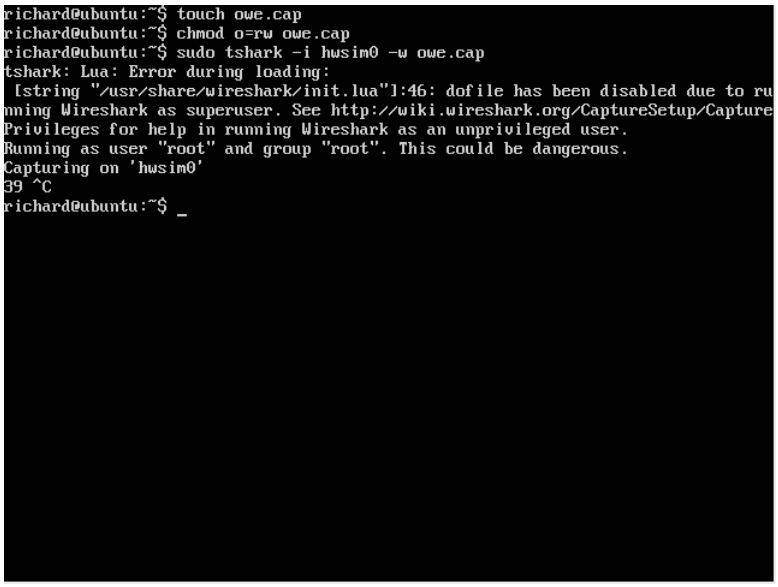

Figure 4: Capturing Opportunistic Wireless Encryption with tshark

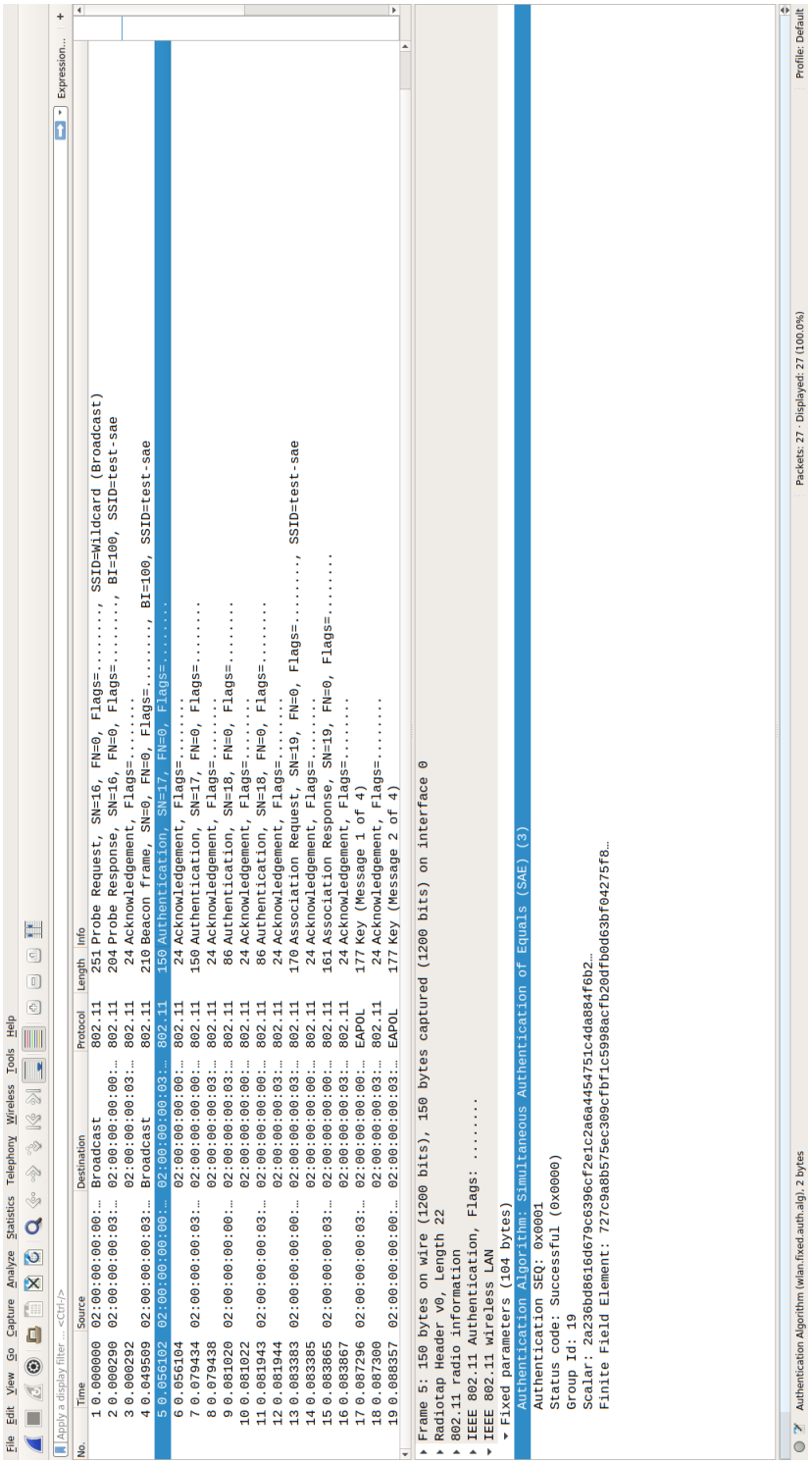

Figure 5: Simultaneous Authentication of Equals captured in Wireshark

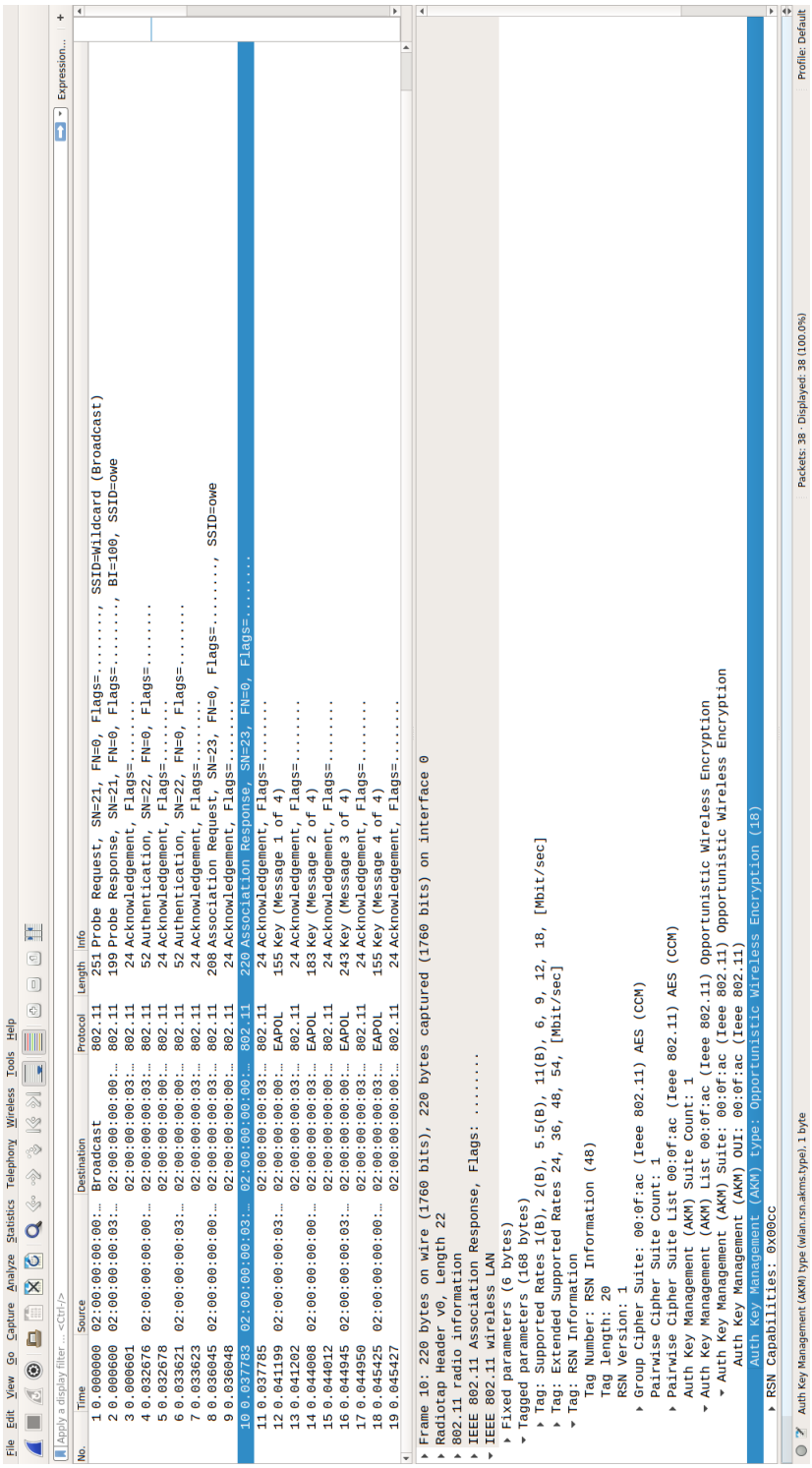

Figure 6: Opportunistic Wireless Encryption captured in Wireshark