Practical WLAN Security

Johannes Förstner Niclas Zeising
Email: johfo249,nicze549@student.liu.se
Supervisor: David Byers, david.byers@liu.se
Project Report for Information Security Course
Linköpings universitetet, Sweden

Abstract

This Report concerns itself with security and security weaknesses in wireless networks, WLANs, based on the IEEE 802.11 standard.

The report is based on studies on different attacks on wireless networks and how to prevent them. We will show how some common attacks are done, and what can be done to mitigate them and making the network more secure. We will also show how to protect your traffic on a inherently insecure network, such as a unencrypted hotspot network.

We will look into how to break the encryption on a WEP protected network, dictionary attacks on WPA networks to find bad keys and pass phrases, which is intended to gain access to networks. Then we will look into different methods of redirecting and eaves-dropping (snooping) on traffic to get at the information flowing through the network. Lastly we concern us with how to protect the network, and how to protect your traffic on a insecure network.

1. Introduction

Wireless networks and wireless communication is becoming more and more common. They are convenient to use since no cabling is required and they make it possible to move around in an area without loosing connection, by so-called roaming. But there are not just positive aspects on wireless communication. It introduces new threats and security considerations not present in wired communication. In this article we will focus on wireless local area computer networks, so called WLANs, based on the IEEE 802.11 standard. We will look into some common attacks on wireless networks to gain unauthorized access to the network and the information therein. This will demonstrate why it is important to have a general idea about security issues in wireless networks and what can be done to reduce the possibility of someone gaining unauthorized access, thereby reducing the risk of unwanted information disclosure. We will also look into different methods of redirecting and eavesdropping on traffic in the network to get access to communication between other systems which makes it possible to get unauthorized access to information. In the end, we will discuss some ways to make the network more secure, and some methods on how to secure your communication on a insecure wireless network where you are worried other parties might try to listen in to your traffic.

2. Cracking WEP

This section is about cracking the keys of a WEP encrypted WLAN by listening on traffic and injecting special traffic to collect the data needed to crack the encryption key. First we will discuss some of the principles behind the attack and why it works from a theoretical point of view. The next part will then contain information on how to set up and perform the actual attack.

2.1 Principles

WEP encrypts its packets with a stream cipher: it uses the RC4 algorithm to produce a key stream from a key, and the key stream is XOR'ed with the plaintext to form the ciphertext. The actual key for a packet consists of a 3-bytes “initialization vector”, which is sent in plaintext together with the packet, followed by a 5-bytes (or sometimes 13-bytes) fixed key part which is pre-shared among the clients and should remain secret. The goal of WEP cracking is to recover these 5 or 13 secret key bytes, to be able to use the network. This is done by capturing packets from the network (usually combined with actively inducing traffic) and then cryptanalyzing them.

The ciphertext of a packet is the plaintext XOR'ed with the key stream, so the key stream can be recovered if the plaintext is known. In practice, plaintext can often be guessed; for example, it can be assumed that packets of length 28 bytes are probably request- or response-packets of the ARP (address resolution protocol), which have fixed bytes in their header.

In 2001, Fluhrer, Mantin and Shamir have shown that it is possible to draw conclusions from a known RC4 key stream output back to the used key. Essentially, RC4 is a PRNG (pseudo random number generator) which is seeded with the key (IV+PSK). With certain “weak” IVs, it's possible to make guesses about the PSK bytes with a certain probability. Repeating this with multiple collected packets, and

counting which key bytes are guessed most often, eventually reveals the correct key bytes. This is combined with brute-forcing through different combinations of the bytes with the highest probabilities (for example, if for keybyte 1 there are 50 “votes” for 0xAA and 45 “votes” for 0x35) and also brute-forcing the last one or two keybytes.

Here's a listing of the RC4 PRNG algorithm:

(initialization:)

```
begin ksa(with int keylength, with byte key[keylength])
      for i from 0 to 255
          S[i] := i
      endfor
      j := 0
      for i from 0 to 255
              j := (j + S[i] + key[i mod keylength]) mod 256
          swap(S[i],S[j])
      endfor
end
```

(PRNG:)

```
begin prga(with byte S[256])
      i := 0
      j := 0
      while GeneratingOutput:
          i := (i + 1) mod 256
          j := (j + S[i]) mod 256
          swap(S[i],S[j])
          output S[(S[i] + S[j]) mod 256]
      endwhile
end
```

Today, aircrack-ng uses the PTW (Pychkine, Tews, Weinmann) crack method by default, which is a refinement of the FMS method and needs far less captured packets (approximate 20000 instead of 250000 for a 5-byte PSK), because it's not dependent on weak IVs, and also runs a lot faster.

Another often exploited weakness of WEP is that it has no protection against replays, so an attacker can capture an (encrypted) packet on the network and then replay it. Thereby, an attacker who wants to capture a lot of traffic on an idle network, can capture a single ARP request packet (guessing by its length) and then replay it; the access point will respond with a new ARP reply packet with new IV every time.

There are a lot more weaknesses in WEP. For example, the integrity protection doesn't stop you from altering an encrypted packet with known plaintext (it's possible to calculate the required bit flips in the encrypted integrity check value for a bit flip in the ciphertext). However, these weaknesses are not essential for cracking the key.

2.2 Software and setup

Here we describe our setup for the WEP crack attack, which is the basis setup for most of the PoC (proof of concept) attacks of this project. Subsequent “Software and setup” sections will refer to this one, describing additions and differences.

We were using:

- A La Fonera 2100/2200 WLAN access point with OpenWRT installed on it. OpenWRT is a FOSS firmware for WLAN routers.
- An EeePC 1001 laptop, which has an Atheros chipset on its Wi-Fi card and therefore is capable of packet injection. A Wi-Fi card with injection capabilities is essential to carry out the described attacks. A list of capable cards can be found here[1].
- A HP 6910p laptop with an Intel Wireless WiFi link 4965 a/g/n card, also capable of injection. Both laptops were running Backtrack 4.2[2]. Backtrack is a Linux distribution composed specifically for security-auditing and similar purposes, and comes with all the tools we needed for the project except for Open Office.

Tools used: the aircrack-ng[3] suite, specifically:

- airmon-ng – to put the wireless card into monitor mode
- airodump-ng – first to find available WEP networks around, then to capture packets of the chosen network
- aireplay-ng – to wait for an ARP request packet and continuously replaying it when found.
- aircrack-ng – to perform the PTW cryptanalysis to find the PSK.

Performing the actual attack is somewhat straightforward once you have the network and laptops set up and know which tools to use. There are several detailed step-by-step, script-kiddie-usable tutorials on the web, amongst others by the aircrack-ng community itself[4]. The tools are simple command-line tools with a human-friendly ncurses interface.

Steps in short:

- Put wireless card in monitor mode.
- Look for available networks with airodump-ng (optional in this case, since we know the MAC addresses of all participants.)
- Start capturing the chosen network's traffic.
- At the same time, start aireplay-ng, so it can induce ARP traffic as soon as an ARP request packet was captured.
- Wait for sufficient captured data (~10.000 packets for PTW attack on a 5-byte PSK.)
- Start aircrack-ng and watch it recover the PSK
- Stop all the tools, put the wireless card into managed mode again, and use the key to connect to the network.

Disclaimer: Remember to only try this on your own network, since gaining unauthorized access to a network might be illegal.

3. Dictionary attacks

This section will discuss principles of dictionary attacks and how to use them to discover bad pass phrases used in creating keys for wireless networks. We will focus on WPA networks, since it's much

harder (to not say impossible) to crack the key than for WEP networks.

Dictionary attacks are a form of brute-force method where an attacker tries to guess a password by trying a number of different passwords to see which works. Usually the different passwords tried come from a list, or dictionary, hence the name. This is why it is important to mix letters, numbers and other characters when making up a password instead of just using common words. There are a number of different variants of this scheme, but for our purposes, a quite simple method was enough.

3.1 Performing against WLANs

A wireless network using WPA (both versions) encryption and protection is very hard, to not say impossible, to break cryptographically like was done with WEP as demonstrated. This does not mean that WPA is entirely safe at all times. When using WPA in pre-shared key (PSK) mode, which is the common method for home use, the same cryptographic key has to be shared between all hosts using the network. This key is 256 bits long, meaning that it requires 64 hexadecimal digits to print, something which is both cumbersome and error-prone to enter into a new host. Instead of entering the key directly, it can be derived from a pass phrase. This pass phrase can be any string between 8 and 63 characters in length which makes it possible to use words or phrases that's easy to remember, but also easy to guess. In a dictionary attack against WPA protected network this is what is attempted.

3.2 Software and setup

This section will demonstrate the tools needed to perform a dictionary attack against a wireless network and go through the necessary steps to perform the attack.

The equipment is mostly the same in this attack as in the previous one. We had to change the AP to a D-Link DIR-615 using the standard firmware when for unknown reasons the other AP failed to use WPA encryption.

From the aircrack-ng software suite the following programs were used:

- airmon-ng – to put the wireless card into monitor mode to capture the WPA handshake and also to be able to inject packets
- airodump-ng – to look for suitable networks and also to collect the WPA handshake
- aireplay-ng – to optionally disassociate a client from the network in order to get the handshake when the client reassociates
- aircrack-ng – to do the actual dictionary attack.

Performing this attack is straightforward once you have set everything up. Unless you know the network you want to attack you have to listen on the air to find potential networks. Once a network has been found, you need to capture the four-way handshake that is completed every time a client associates with the AP. If it takes too much time to wait for a new client to associate, it is possible to inject traffic into the network to make an already connected client disassociate. This client will then automatically reassociate with the AP, and it is possible to capture the handshake when it does.

3.3 Speed and feasibility

This question has to be regarded from two points of view, since the attack is carried out in two very different steps: collecting the data and analyzing it.

For collecting data, all that's needed is to record one single WPA handshake. This can be sped up by actively deauthenticating a client. On the other hand, if the attack is carried out with a Wi-Fi rifle from far away, the attacker has more time to wait unobtrusively for an eventual handshake.

The actual dictionary attack is carried out off-line, and is in many regards the more problematic point. The algorithm iterates through the dictionary, generating the WPA key for every word and trying to apply it to the recorded handshake (if it decrypts successfully, the current word is the network's pass phrase). The involved cryptographic algorithms are computing-intensive, and so only a few hundred words can be tried per second. In practice, this had been ~270 keys/s on the EeePC 1001 (Intel Atom N450 processor) and about 1100 keys/s on the HP 6910p (Core2 Duo 2GHz processor), which illustrates the CPU-dependence of the attack speed. Since this can be done off-line, this is not much of a problem though, except in cases where you want to crack into a WPA network as fast as possible. The far bigger problem is: if the correct pass phrase is not in the dictionary file, no algorithm will be able to crack it. This can be helped a bit for example by trying several permutations of appended/prepended numbers or signs to each tried dictionary word (word lists can be generated by another program and piped into aircrack-ng), but of course this multiplies the time needed to iterate through the whole dictionary by the number of combinations used.

Bottom line: with standard computing equipment, the WPA dictionary attack is feasible if you have time to crack off-line and the pass phrase is weak.

3.4 On WPA(2) security

This section provides some further details about the WPA and RSN, more commonly known as WPA2, security standards for wireless networks, to give a deeper understanding about why the WEP cryptanalysis attack can't be applied here.

WPA (Wi-Fi Protected Access) was designed to provide a second security layer on top of the existing WEP and also running on the same hardware (so that

WEP could be replaced with a simple and free firmware upgrade).

By using a master key, WPA produces a new RC4 key for every package; additionally, this key mixing function is designed to avoid weak RC4 keys. So this is why cracking the WEP key of a WPA network doesn't make any sense.

WPA also has a second integrity check value on the network layer (meaning when the SDU is put together again from the possibly fragmented packets), and a sequence number that prevents blind replay attacks.

RSN, or WPA2, is designed from scratch and doesn't run on all legacy hardware. It doesn't use the RC4 key stream generator anymore at all, but uses AES instead (in "counter mode", to use the block-cipher as a stream-cipher).

Security weaknesses still in place with these new standards are the distribution of pre-shared keys (resulting in people using simple dictionary words as pass phrases), and the unprotected management frames (which makes denial-of-service attacks easier).

4. ARP redirection on wireless networks

This section will demonstrate how to redirect traffic using a common method known as ARP redirection or ARP poisoning. This makes it possible to get access to all or selected parts of the traffic in a network once the network is broken into.

4.1 What is ARP redirection

ARP redirection is a common way to redirect traffic on an Ethernet network to a computer of the attacker's choice. It uses the inherited insecurities, mostly lack of authentication, in the address resolution protocol used on Ethernet networks. ARP is used to map between network addresses, such as IP addresses, and hardware MAC addresses. By faking unsolicited ARP replies it is possible to make one or more computers on the network send traffic to a specified machine instead of the genuine one.

This is done by inserting ARP messages that contains information that a certain network address is at a specific MAC address instead of the actual one, making the victim send traffic to the wrong computer. Since ARP traffic is not authenticated in any way the victim has no possibility to verify if an ARP reply is genuine or not.

What's more, ARP is a stateless protocol, to make simple and fast implementations possible. That means that a computer doesn't remember what IP addresses it asked the MAC addresses for, but instead it simply stores whatever it hears on the network into its ARP cache, so that a poisoner doesn't have to wait for ARP requests and then has to be faster than the genuine computer to do a fake reply, but instead can send unsolicited replies. For detailed information about how ARP works, see[5].

4.2 Performing ARP redirection

A common tool used for this is ettercap[6]. It comes with a CLI as well as a GUI, and lets you perform ARP redirections by first finding hosts on the network, then defining two target groups and finally launching the attack, which will poison the two groups' ARP caches about the respective other group. It is also possible to redirect all traffic to a specified host. In this state, ettercap continuously sends out "poisonous" ARP replies with some seconds interval. When the "poisoned" computers want to send IP traffic to the other group, they will now in fact send it to the attacker's MAC address; ettercap by default forwards the packets to the genuine MAC addresses, allowing the attacker to sniff into the traffic between the groups, often without the victims noticing anything wrong.

To actually capture the traffic for analysis, a program called wireshark[7] was used. Wireshark can capture some or all the traffic going to a network interface and also do some decoding, filtering and other similar things.

Setup:

- For our proof-of-concept attack, we had a third person run a web service in our network, so one of us could connect to that site and the other one could perform the attack with ettercap.
- We had wireshark running on the attacker's laptop as well, to see if the attack actually was performed properly.

In wireshark, you can see the following:

- When ettercap scans the network for hosts, it sends out a lot of ARP requests (for the whole subnet address space).
- When ettercap is performing the attack, it sends out unsolicited ARP responses with some seconds interval.
- A communication attempt of T1 (someone in the target group 1) to T2 (in group 2) is sent as a packet with T2's IP address, but the attacker's MAC address, and then re-sent from the attacker to T2 with the proper MAC address; same goes for communications in the other direction.
- You can "follow TCP stream" to examine the communication between two clients closer and see if there's maybe a password sent over in the clear. Although you only can "follow TCP stream" for tcp connections, wireshark will still capture all traffic for analysis.

5. Eavesdropping

This section demonstrates how to eavesdrop on radio traffic while it's in the air. This makes it possible to get to traffic in a wireless network unobtrusively, without connecting to it. Connecting could be logged by the AP and noticed by the admin, and it would also be needed to circumvent an eventual MAC filter. Using a big antenna and signal amplifiers even makes it possible to listen from a distance. The only caveat is that if the traffic is encrypted, you have to be able to decrypt it to get to the information.

5.1 Software and setup

This section will demonstrate the tools needed to perform eavesdropping, both on unencrypted and encrypted networks. As for all other attacks demonstrated, it is necessary to put the network interface into monitor mode. This can be achieved with airmon-ng as previously explained, but it is also possible to use iwconfig. Iwconfig is a tool for managing wireless network cards that usually comes with the Linux operating system installation.

A software that captures packets is also necessary. For the capturing of traffic the command line tool tcpdump is sufficient. This can be used to capture packets to a file for later analysis. For more advanced analysis, including decryption, the software package wireshark is needed. This also gives the benefit of a graphical user interface, which can be convenient.

5.2 Unencrypted eavesdropping

To eavesdrop (or sniff as it is also called) unencrypted traffic, the first step is to put the interface in monitor mode. This can be achieved with airmon-ng as previously explained, or using iwconfig to set the mode of the wireless network card to monitor. It's not needed to associate to the access point where the interesting traffic is originating. By putting the network interface in monitor mode, all traffic in the air can be captured (as long as the signal is strong enough).

The next step is to start wireshark and select the interface on which to capture the traffic. The traffic should then show up. With wireshark it is possible to do all kinds of analysis. One possibility is to follow traffic streams to see traffic between two communicating hosts. When using this options, data in multiple TCP packets will be reassembled and the data between hosts are displayed for further analysis. This can in turn be used to find unencrypted passwords used to log on to websites, e-mail servers and other applications and also complete e-mails or data posted in a web forum. It is also possible to use wireshark to save the files transferred as part of an HTTP request. Wireshark has numerous other filters and tools to filter, decode and otherwise manipulate captured data.

One caveat is that it might be needed to fiddle with the protocol options for IEEE802.11 in the preferences menu in wireshark. Especially the option “Assume packets have FCS”, which was needed to make our capture work.

5.3 Encrypted eavesdropping with key

When the computer and wireshark is set up to capture unencrypted traffic, it is easy to take the next step and also decrypt encrypted traffic. For WEP, all that's needed is the 40 or 104 bit WEP key, which we have previously explained how to obtain. Then it is just to select “Enable decryption” and enter the key in the protocol options for IEEE 802.11 in wireshark. This will decrypt traffic using this key, and present the decrypted information in the same way as unencrypted packets. This can then be analyzed as described above.

To decrypt WPA/WPA2 traffic, a little more involvement is needed. WPA/WPA2 can be decrypted by either using the pass phrase or the pre-shared key, which is entered in the same place as the WEP key. This is not all that is needed though. It is also necessary to capture the complete 4-way eapol handshake to be able to decrypt the traffic. This is the same information as that needed when doing dictionary attacks against WPA/WPA2 networks. Wireshark can capture this handshake, but it might be needed to disassociate clients connected to the access point to get this in a timely manner, as explained in 3.2 Software and setup. It might be needed to do this more than once, especially if there is a lot of other traffic in the air, as not all packets might get captured if there is too much traffic.

6. Securing WLANs and traffic

Now that we showed some common security weaknesses and you have a vague idea of how and why attackers might attack you or your network, here's a section about how to close these holes and what to keep in mind when using a wireless network.

6.1 Securing WLANs

If you are setting up your own home WLAN, here are some important checkpoints to configure it in a secure way:

- Use the newest security standard RSN, more commonly known as WPA2. It is not possible to use WEP in a secure way, as demonstrated in section 2 Cracking WEP. See also section 3.4 *On WPA(2) security*.
- You can use a MAC filter as an additional layer to protect from opportunistic crackers; though for people who know how to circumvent them, it's merely a little annoyance. This should never be used as the only means of security, since it is usually trivial to fake a MAC address.
- In simple home WLANs, the most common authentication technique is PSK (pre-shared keys), generated from a pass phrase. The usual precautions about secure passwords apply: use a secure password (at least 8 characters, not to be found in any dictionary, with random numbers and special characters etc.) and don't write it down and leave the note lying around in your room. Many operating systems today have the possibility to save the key so you only have to enter it once, meaning you only need to have the password in case the saved password gets lost or you want to grant access to new computers.
- Always make sure to configure your AP properly in the first place. Many APs come unencrypted by default or have a default admin password that attackers can derive from the default SSID.

6.2 Securing your traffic

There are other situations where you're not the admin of the WLAN you want to use, and you don't trust the security

of the network (or the admin). This can be the case when you're using a public hotspot; they are often unencrypted, and you never know whether someone might be eavesdropping on your traffic. In this case, here are some possibilities to protect yourself and your traffic from attackers:

- Use an additional layer of encryption. You can use SSL connections to write mails, surf on the web (https) and for chatting (make sure this is enabled in your client). Using SSL encryption is especially important when logging in to web sites such as facebook, gmail or your online bank, or e-mail servers where you authenticate with a password. Without SSL, this password is sent in clear text, meaning that anyone who can listen in on the traffic can get to it. This is always important, even when using wired networks.
- Since SSL uses cryptographic certificates to authenticate the remote host, you can also detect ARP redirection attacks performed by fellow WLAN users. Only the real host can produce the known certificate. Thus, never trust an unknown certificate on an untrusted network.
- Attackers can still see what hosts you are communicating with, and what protocols you use (by looking at the TCP/UDP ports). If you want to prevent that, you need a tunnel connection to a proxy at a secure location (outside of the WLAN). For this purpose, you can (A) use the services of a privacy proxy company, (B) setup your own proxy server at home, or (C) use an open-source anonymity network like TOR[8], which also provides anonymity at the cost of high latency.

It is also possible to tunnel traffic over an SSH connection or set up your own VPN to a network you trust. The VPN might give the additional benefit of making it possible to access resources inside the trusted network, which aren't available from the internet.

7. Conclusions and summary

In this report we have studied some common attacks on WLAN networks, and possibilities to protect your networks and your traffic. We have studied how to crack the encryption in WEP protected networks, how to perform dictionary attacks on WPA protected networks, and how to perform different types of traffic redirection and snooping to get to the information passing through. Lastly we have looked into some methods of securing your wireless network, and some methods to protect your traffic in a unsecured network, such as a public hotspot.

With all the shown proof-of-concept attacks in this report, one might conclude that security in wireless networks is a big issue. Technically, it isn't – all you have to do is to configure your home AP to use WPA2 and give it a proper password; and be aware of the possibilities of eavesdropping and redirection when using a public network. Practically, though, there are plenty of security holes out there that put people at risk. The reason for this is that many people don't care to think about their WLAN being attacked, because they're not aware of the dangers. It's one thing to read that it's "easy to break into a WEP secured network", but a totally different thing to actually have an idea about how such a breach is performed. Only when you know how the attacks work, you can reason about how good your protection actually is, and it also enables you to audit your home WLAN security from an attacker's point of view.

Another thing that can be learned, if not from this report, then from further studies: The design of secure communication protocols is a non-trivial matter. At first sight, WEP as a system looks reasonable and secure; it sure did for its designers. After all, they didn't construct it as lecturing material for students. Still, they managed to produce an inherently broken standard which many people are still using today.

[1] List of supported Wi-Fi cards: http://aircrack-ng.org/doku.php?id=compatibility_drivers#which_is_the_best_card_to_buy
[2] Information on Backtrack Linux and where to download it: http://www.backtrack-linux.org/
[3] Information about aircrack-ng: http://www.aircrack-ng.org
[4] Examples of some aircrack-ng tutorials: http://www.aircrack-ng.org/doku.php?id=tutorial
[5] An Ethernet Address Resolution Protocol, RFC 826: http://tools.ietf.org/html/rfc826
[6] Information about ettercap: http://ettercap.sourceforge.org
[7] Information about wireshark: http://www.wireshark.org
[8] The Onion Router, see http://torproject.org