Attacks against the WiFi protocols WEP and WPA

Matthieu Caneill Jean-Loup Gilis

October - December 2010

Abstract

Wireless networks are today an entire part of the Internet, and are often used by companies and particularies. Security of information is thus important, and protocols like WEP and WPA can be attacked. We present in this report the existent WLANs protocols, an overview of the most efficient attacks on them and two attacks we have found on WEP.

Contents

Part I
WEP and WPA protocols: an overview

1 WEP

Nowadays, numerous networks are wireless. Either at home or at work, it is necessary to encrypt the data transmited on those networks to prevent eavesdropping. One of the most common protocol aiming at wireless network security and privacy is the Wired Equivalent Privacy (WEP) protocol, created in 1999. It is part of the 802.11 standard for wireless LAN communications. In this part, we are going to describe the WEP protocol.

1.1 How WEP works

The WEP [1] protocol uses the RC4 cipher to ensure privacy and a CRC-32 Checksum to ensure integrity of the data transmitted. It works as follow:
First, a secret key k is shared between the users of the network. The protocol does not specify the way the key must be shared. It is usually a 40 bits key, though improved versions of WEP use a 104 bits key.
To send a message M, one has to compute the integrity checksum $c(M)$ of the message and concatenate it: one has now $M.c(M)$
Then, one encrypts $< M.c(M) >$ by XORing it with a RC4 stream generated by k and a public initialisation vector (IV) of 24 bits, named v. We note it $RC4(v, k)$.
The result $C =< M.c(M) > \oplus RC4(v, k)$ is then sent on the network. An user who knows k can get the message by XORing C with $RC4(v, k)$.
Here is scheme of the encryption process:

1.1.1 RC4

The RC4 [2] stream cipher used by WEP is based upon two algorithms. The first one being RC4-Key Scheduled Algorithm (KSA), which transforms a key of length 1 to 256 bits into a initial permutation S of the numbers 0 to N. The internal state of RC4 consists of two numbers i and j used as pointers to elements of S. The second algorithm is RC4-Pseudo Random Generation Algorithm (PRGA). It generates a single byte of keystream from the current internal state of RC4 and then updates the internal state. Originally, $N = 255$, but the algorithm can work with different values of N.

1.1.2 CRC-32

The Cyclic Redundancy Check (CRC) is a hash function used by the WEP protocol to ensure data integrity. The principle of CRC32 stands in a kind of polynomial division:

Algorithm 1 KSA (Permutation)

```
procedure KSA
    for $i \leftarrow 0...N-1$ do                              ▷ initialisation
        $S[i] \leftarrow i$
    end for
    $j \leftarrow 0$
    for $i \leftarrow 0...N-1$ do                              ▷ permutation
        $j \leftarrow (j + S[i] + K[i \bmod l]) \bmod N$
        $Swap(S[i], S[j])$
    end for
end procedure
```

Algorithm 2 PRGA (Output)

```
procedure PRGA
    $i \leftarrow 0$                                            ▷ initialisation
    $j \leftarrow 0$
    loop                                                        ▷ generation loop
        $i \leftarrow (i + 1) \bmod n$
        $j \leftarrow (j + S[i]) \bmod n$
        $Swap(S[i], S[j])$
        $k \leftarrow (S[i] + S[j]) \bmod n$
        $Print(k)$                                              ▷ output result
    end loop
end procedure
```

the original message is XORed with a constant of 32 bits followed by as many 0 as necessary to reach the length of the message. The result becomes the new "message" and the operation is repeated until the length of the result is under the length of the constant. It is important to note that this hash function is linear and unkeyed.

2 WPA

In 2001, the first attacks against the WEP protocol were found. It was necessary to find a new protocol more secure than WEP, but still driver upgradable from the infrastructures previously running with WEP. WPA standardizes two modes on how payloads can be protected during transmission, Temporal Key Integrity Protocol (TKIP) and Counter Mode with CBC MAC Protocol (CCMP). TKIP is designed to work with old hardware, it's a sort of update of WEP, whereas CCMP is a more secure but completely new protocol for modern wifi-cards.

2.1 How WPA works

TKIP [3] is based upon RC4, namely for hardware compatibility reasons. CCMP is compulsory in WPA2. Both modes are explained below.

2.1.1 TKIP

TKIP is also based on WEP, with new tools of security. The main problem, for reasons of compatibility, is the use of the RC-4 stream cipher, on which several attacks exist. TKIP is a modified WEP:

- CRC-32 is not used anymore, MIC (Message Integrity Check) replaces it. Michael is a keyed cryptographic algorithm, that means every packet is hashed with his content and a general key. Without knowing the key, an attacker cannot forge packets neither modify some;
- Countermeasures are used: if the Access Point receives two fake MICs (a message not matching its hash) in less than one minute, all clients are deconnected and the network isn't available for one minute; it prevents the network from brute-force attacks;
- The IV (initialisation vector) is replaced by a TKIP Sequence Counter (TSC): a 48 bit number increasing for each packet, that prevents IV reuse and doesn't reveale weak keys;
- Many temporal keys are generated from a master key for the different encryptions algorithms used by TKIP, and a important difference with WEP is the *Per-Packet Key Mixing*: a different key for each packet is used to encrypt the packet, it's not just a concatenation between a key and an initalisation vector.

In brief, TKIP is an improvment of WEP, correcting its main security problems, like IV reuse. But, always based on RC-4 stream cipher, many attacks also exist on TKIP.

2.1.2 CCMP

Counter Mode with CBC MAC Protocol is today the most secured algorithm for wireless transmissions. It's part of the WPA-2 standard, and assures — as usual — integrity, authentication and confidentiality of the information. CCMP uses the AES block cipher algorithm, it's why it's completely different from WEP and TKIP algorithms. A packet consists of a packet number (corresponding of the WEP initialisation vector), a header, and, encypted: the data and the MIC (hash of the data).

It's because CCMP is very different from WEP and TKIP and doesn't use the same algorithms, that previous known attacks on either WEP or WPA don't work.

2.1.3 PSK and 802.1X

For both TKIP and CCMP encryptions, 2 modes are proposed: 802.1X and Pre-Shared Key (PSK). 802.1X is a mode with an identification server, distributing keys at users, when needed. PSK mode is less secure: a unique key (the same for all the users) is used to access the network.

Part II
Attacks on WEP

3 FMS Attack

The FMS attack [4] is a statistical attack on WEP released in 2001 by Fluhrer, Mantin and Shamir.

This attack uses weaknesses in RC4. In addition, the attacker knows the IV: this is three bytes of the per packet key. If four conditions hold, he can then perform a manipulation on RC4 that allows him to guess with a five percent probability a byte of the key. Using a system of vote, he can guess a probable key and test it. If the key is not correct, he will try another likely correct key and try again. This attack requires quite a lot of packets to reach a fifty percent success rate: up to $6,000,000$.

To be more specific, the attack works as follow:

As the attacker knows the first l bytes of the per packet key, he can simulate the l first steps of RC4-KSA. So he knows S_l and j_l. In the next step, $j_{l+1} = j_l + K[l] + S_l[l]$ and $S_l[l]$ is swapped with $S_l[j_{l+1}]$. Fluhrer, Mantin and Shamir showed that if:

- $S_l[1] < l$
- $S_l[1] + S_l[S_l[1]] = l$
- $S_l^{-1}[X[0]] \neq 1$
- $S_l^{-1}[X[0]] \neq S_l[1]$

Then the value $S_{l+1}[l]$ will take the value of $S_l[j_{l+1}]$ in the next round of RC4-KSA and this value $S[l]$ will not change through the rest of the process with a probability of approximately five percent. Eventually the first byte of keystream $X[0]$ will be $S[l]$, so we'll be able to calculate our next byte of key K:

$$K = S_l^{-1}[X[0]] - j_l - S_l[l] = S_l^{-1}[S_{l+1}[l]] - j_l - S_l[l]$$

As we are only five percent sure about K we perform this operation on multiple packets and chose the most probable K as our next byte of key. We can now perform this operation incrementaly, getting one more byte of key each time. We can eventually test the key. If it does not work, we switch a byte of the key with another probable value for this byte and perform the operation again. This way, we can perform a full key recovery !

4 KoreK Attack

KoreK, an anonymous participant of the security forums of NetStumbler.org, has found many different attacks on WEP. His first attack [5] is based on the FMS-Attack, and let the attacker find the key faster. In addition, he published an attack, $A - neg$, which allows the attacker to reduce the key space, thus enabling him to find the key faster.

5 Chopchop Attack

The Chopchop attack [6] (found by KoreK), rather than exploiting a weakness in the RC4 algorithm, exploits design flaws in the WEP protocol itself: the weakness of the CRC32 checksum and the lack of replay protection. It was first released by KoreK around the same time as the eponymous attack.

The Chopchop attack aims at giving an attacker the ability to decrypt a packet without knowing the key. Nevertheless, due to its lack speed, its practical use is limited to eavesdrop a packet, decrypt it, modify it and inject it back into the network to generate more traffic and therefore give more usefull information to perform a full key recovery attack (i.e. a PTW attack).

The Chopchop attack is based upon the fact that one can flip a bit in the cipher text and then calculate which bit in the encrypted CRC32 value must be flipped so that the packet is still valid. The attack works by taking away the last byte of a packet and trying to guess its value.

It is actually possible to do so by injecting the truncated packet back into the network. This packet is invalid because of a incorrect ICV, but it is possible to render this packet valid again by XORing it with a value depending only on the truncated byte. This value ranges from 0 to 255. Therefore, the attacker (thanks to the lack of replay protection) can bruteforce the value: when the correct value is tested, the AP will return the packet back into the network. Knowing this value, the attacker can calculate the byte of plaintext (and therefore the keystream). By repeating this operation, the attacker is able to decrypt a packet. He can get both the plaintext and the keystream without even knowing the master key.

6 Fragmentation Attack

Though a "fragmentation issue" had already been mentioned before, the first practical Fragmentation attack [1] was released in 2005 by Bittau et al. in a paper called "The final nail in WEP coffin". The attack works as follow:

At first, the attacker needs to eavesdrop a packet. As all packet send in a 802.11 network have similar headers, the attacker can know/guess the first 8 bytes of clear text. By XORing these 8 bytes with the 8 corresponding bytes of cipher text we obtain 8 bytes of keystream for a specific IV. Those 8 bytes of keystream cannot be used to send a whole packet on the network (it would be ridiculously small). But the WEP protocol allows to send a single packet in up to 16 fragments. Therefore, we can use the 8 bytes of keystream we know to broadcast a packet containing 64 bytes of know text in 16 fragments. (We can only have 64 bytes of known text because each fragment needs its 4 bytes long CRC32 checksum). When the AP receives those 16 fragments, it will decipher them, combine them into a single packet, encrypt it and send it back on the network. This packet is 68 bytes long (64 bytes of known text and 4 bytes ICV). With a XOR, the attacker has now 68 bytes of keystream for a given IV. By repeating this process, the attacker can get up to 1500 bytes of keystream for a IV.

When knowing 1500 bytes of keystream for a given IV, it is easy to get 1500 bytes of keystream for other IVs by simply sending a broadcast packet of 1500 bytes to the AP. The AP will then relay this packet, but encrypted with a new

IV. As $C \oplus M = K$ the attacker can get the keystream for other IVs and build a dictionnary, allowing him to decipher every single packet on the network, and also to create traffic.

7 PTW Attack

The Pyshkin Tews Weinmann (PTW) attack [7] was released in 2007. It introduced two new concepts.

The first one is based upon the Jenkins correlation. In 2005, Klein showed that $l - X[l-1]$ takes the value of $S[l]$ with a probability of 2/256. And if $S[l]$ does not change until $X[l-1]$ is produced, then

$$S_l^{-1}[l - X[l-1]] - (S_l[l] + j_l) : (1)$$

takes the value of $K[l]$ with a probability of 2/256. But if $S[l]$ is changed during the process, then (1) takes a more or less random value. So (1) can take the value of $K[l]$ with a probability of approximately 1.37/256.

As there is no condition on the key, every packet can be used.

The second new concept consists in a new structure for the attack. Instead of trying to guess the key byte per byte, the attack works on a multibyte correlation: if the attacker knows the first l bytes of a key and recovers $k = S_{l+2}[l+1]$ (instead of $S_{l+1}[l]$ in FMS), he will be able to use $S_{l+1}^{-1}[k] - S_{l+1}[l+1] - S_l[l] - j_l = K[l] + K[l+1]$

Knowing that, Pyshkin Tews and Weinmann modified (1) not to vote for a byte of the key but for the sum of the next m bytes of the key with m taking every value between 1 and 13. we note $\sigma_i = \sum_{k=0}^{i} Rk[k]$ and we have:

$$S_l^{-1}[l + m - 1 - X[l + m - 2]] - (\sum_{a=l}^{l+m-1} S_l[a]) : (2)$$

depending only on the IV and voting for σ_i.

So, to perform a PTW attack, the attacker needs to capture packets, recover their keystream and then, knowing the first 3 bytes of all perpacket keys, he can calculate (2) for every packet and every m and gets votes for all σ_i. Then, he can calculate and try a root key (with $Rk[0] = \sigma_0$ and $Rk[i] = \sigma_i - \sigma_{i-1}$. If the test is not successful, the attacker, the attacker can test another probable key by updating σ_i and using 12 substractions.

The tests showed that only 35,000 to 40,000 packets were required to get a fifty percent succes probability.

Part III

Our contribution: Attacks on WEP

8 Google Replay Attack

The Google Replay Attack is based on the fact that any lambda user with an access to the internet will do a Google search. A lot of web users have the URL `http://www.google.com` as home page. It means the Google logo, a 7330-byte-image, is downloaded every time. With this data, an attacker can easily recover a part of the keystream, knowing plain- and encrypted text.

The main difficulty for the attacker is to know exactly at which moment the client will download the Google logo. He has to study the structure of TCP/IP packets, and to wait for a packet which size is exactly 7330 bytes.

NB : this attack is not implemented yet !

9 Coolface Attack

The Coolface attack uses the second mode of WEP-authentication: Shared key. As opposed to the Open System authentication, the client has to resolve a challenge to be connected. The AP sends to the client a challenge, and the client will send back the encrypted challenge. If it is correct, the AP accepts the connection. It is a new opportunity for the attacker, who can get both a plain- and an encrypted text. To switch from Open System to Shared key mode, the attacker can begin a denial of service against the AP and then usurpate the AP's identity, thus enabling the Shared key authentification.

Repeating this operation, the attacker builds an IV-dictionary very fast !

NB : this attack is not implemented yet !

Algorithm 3 Coolface protocol

```
procedure DENIAL OF SERVICE ON THE REAL ACCESS POINT
    request connections
end procedure
procedure CONNECTION WITH CLIENT                              ▷ in parallel
    set the adversary MAC adress
    set mode Shared − key authentication
    send plaintext
    receive ciphertext
    get keystream
end procedure
```

10 Summary of WEP attacks

10.1 Key-recovery attacks

Name	**Type**	**Year**	**Packets**	**Ratio**
FMS	statistical	2001	6,000,000 (64 bit WEP)	86
KoreK	statistical	2004	200,000 (64 bit WEP)	3
PTW	statistical	2007	70,000 (64 bit WEP)	1

10.2 Packets-building attacks

Name	Type	Year	Packets
Chopchop	fake ARP	2004	1 at begin (later: injection-capture)
Fragmentation	fragmentation	2005	1 at begin (later: injection-capture)
Google replay	replay	2010	1 at begin (later: injection-capture)
Coolface	man-in-the-middle	2010	0 at begin (later: injection-capture)

Part IV
Attacks on WPA

11 Beck and Tews' Improved Attack on RC4

In 2008 Beck and Tews released an attack on WPA [8]. This is not a key recovery attack, but still exploits weaknesses in TKIP to allow the attacker to decrypt ARP packets and to inject traffic into a network, even allowing him to perform a DoS (Denial of Service) or an ARP poisoning.

In order to be practical, the attack requires the Quality of Service (QoS) to be enabled. The QoS is a feature of WPA standard that allows several channels to be used. Each channel has its own TSC. As channel 0 is used for most of the traffic, it will be possible to inject valid packets in other channels were the TSC will likely be lower. The attack also requires the Key Renewal Interval to be longer than 15 minutes (the time needed to decrypt an ARP packet with this attack).

The attack works as follow:

First, the attacker de-authenticate a station (STA). Then, the attacker can capture an ARP packet. He will then perform a modified Chopchop attack to recover the Integrity Check Value (ICV) and MIC of the packet. When this is done, the attacker will have to guess the last part of the packet: the IP adresses. Eventually, he reverses the MICHAEL algorithm and gets the MIC key. Knowing the keystream and the MIC key, the attacker can now inject custom packets into the network, but only on channels with a lower TSC.

The reason why a modified version of Chopchop must be used is that the attack has to bypass the MIC countermeasure. The modified Chopchop attack works as follow:

The attack works as an AP sending data to a STA. It chopps off the last byte of a packet as the regular Chopchop attack does. When the correct byte is guessed, the ICV of the truncated packet is correct, but the MIC is not. This will cause the STA to send a MIC failure report. So when the attacker receives a MIC failure report, he knows that his guess was correct. Then, he has to wait for a minute in order to prevent the MIC countermeasure from triggering. Eventually, he can chopp off a new byte, etc.

Nevertheless, this attack has limitations: the TSC limits the number of packets that can be injected into the network from 3 to 15 per QoS channel.

Moreover, a key renewal implies that a new attack has to be performed. As the attack takes about 15 minutes, a key renewal interval of a few minutes simply prevents the attack.

12 Ohigashi-Morii Attack (Beck-Tews + Man-in-the-middle)

The Ohigashi-Morii Attack [9] (2009) is an improvement of the Beck-Tews attack on WPA-TKIP. Indeed, this new attack is efficient for all modes of WPA and not just those with QoS features. The time to inject a fake packet is reduced to approximately 15 minutes to 1 minute at the best. For this attack, a man-in-the-middle attack is superposed to the Beck-Tews attack, with tips to reduce the execution time of the attack.

13 Michael Attacks

We wrote earlier that the Michael algorithm is expected to produce a hash of some plaintext. Nevertheless, in 2008, Beck and Tews found a way of reversing the Michael algorithm [10].

And in 2010, Beck found a way to perform an attack based upon the flaws in Michael. Actually, he found that if the internal state of Michael reaches a certain point, the Michael algorithm resets. Therefore, we can inject some text of our choice in a packet, add a string that resets the Michael algorithm, then the packet is changed but the Michael's result remains correct. A complete protocol allowing to perform a Michael Reset Attack is described in the Beck's paper, but one has to notice that the requirements of this attack are even tighter than the requirements of a classic Beck and Tews attack. Moreover, the simple fact of disabling QoS renders this attack impossible.

14 The Hole196 Vulnerability

The Hole 196 vulnerability [11], found by Sohail Ahmad (Airtight Networks) in 2010, comes from the page 196 of the standard paper about 802.11 protocols, where there's a hole.

This attack isn't a key-recovering attack, the attacker has to be an authorized user of the network. First, he sends an ARP request with his MAC-address and the IP-address of the AP. So the other clients of the AP will update their ARP tables, and will send their packets to the MAC-address of the attacker. So the attacker will receive the packets decrypted by the AP and re-encrypted with his key: he is also able to read them. It's a man-in-the-middle attack, and it works because everyone can build and broadcast fake packets with the GTK (shared group key).

15 Dictionary attack against the handshake

It exists a key-recovery attack on WPA (Pre-Shared Key version), when the key is a word from a dictionary.

Eavesdropping the network, the goal of the attacker is to get a handshake; the hash of the key swapped between the client and the AP when the client begins the connection. The attacker can wait, or launch a deauthenticate-attack against the client.

When he gets the hash, he can try to find the key with a dictionary-attack, a rainbow-attack or one of the multiple attacks that exist on hashed keys in general.

16 Summary of WPA attacks

Name	**Year**	**Utility**	**Ratio**
Beck and Tews	2008	inject traffic (QoS features)	24
Ohigashi-Morii	2009	inject traffic (in all modes)	2
Michael	2010	inject traffic (in all modes)	1
Hole196	2010	man-in-the-middle, inject traffic, DoS attack	-
Dictionary attack		key-recovery	-

Part V
Tools

17 Aircrack-ng

Aircrack-ng[12] is a Wireless 802.11 WEP and WPA-PSK key cracking program. Usually used in a terminal, it is able to perform the PTW attack, the FMS attack, and various replay attacks, including the Fragmentation attack. It is currently in its version 1.1.

Aircrack-ng is composed of many utilities: airmon-ng (sets the wifi card to the monitor mode), airodump-ng (eavedropes the network and saves the IVs), aireplay-ng (launches the attacks against the AP), aircrack-ng (reads the IVs and calculates the key with the statistical attacks) and other ones.

Aircrack-ng is an open source program.

18 Wireshark

Wireshark [13] is a free and open source packet analyzer. It is similar to tcpdump but has a graphical interface and advanced filtering options.

Wireshark is usefull to understand the structure of both 802.11 and TCP/IP packets.

References

[1] Andrea Bittau, Mark Handley, and Joshua Lackey. The Final Nail in WEP's Coffin. 2006.

[2] Andreas Klein. Attacks on the RC4 stream cipher. 2006.

[3] Finn Michael Halvorsen and Olav Haugen. Cryptanalysis of IEEE 802.11i TKIP. 2009.

[4] Scott Fluhrer, Itsik Mantin, and Adi Shamir. Weaknesses in the Key Scheduling Algorithm of RC4. 2001.

[5] KoreK Attack, 2004. `http://www.netstumbler.org/f49/need-security-pointers-11869/`.

[6] KoreK, Chopchop Attack, 2004. `http://www.netstumbler.org/f50/chopchop-experimental-wep-attacks-12489/`.

[7] Andrei Pyshkin, Erik Tews, and Ralf-Philipp Weinmann. Breaking 104 bit WEP in less than 60 seconds. 2007.

[8] Martin Beck and Erik Tews. Practical attacks against WEP and WPA. 2008.

[9] Toshihiro Ohigashi and Masakatu Morii. A Practical Message Falsification Attack on WPA. 2009.

[10] Martin Beck. Enhanced TKIP Michael Attacks. 2010.

[11] Sohail Ahmad (Airtight Networks). WPA 2 Hole196 Vulnerability. 2010.

[12] Aircrack-ng. `http://www.aircrack-ng.org/`.

[13] Wireshark. `http://www.wireshark.org/`.

[14] Nikita Borisov, Ian Goldberg, and David Wagner. Intercepting Mobile Communications: The Insecurity of 802.11. 2001.

[15] KoreK, Next generation of WEP attacks, 2004. `http://www.netstumbler.org/f18/next-generation-wep-attacks-12277/index3.html#post93942`.

[16] Andrea Bittau. The Fragmentation Attack in Practice. 2005.