Real-life paradigms of wireless network security attacks

I. P. Mavridis, A.-I. E. Androulakis, A. B. Halkias
Dpt. of Computer Science and Biomedical Informatics
University of Central Greece
Lamia, Greece
{hmauridis, gandroulakis, axalkias}@ucg.gr

Ph. Mylonas
Image, Video and Multimedia Laboratory
National Technical University of Athens
Athens, Greece
fmylonas@image.ntua.gr

Abstract— **Wirelesses Local Area Networks (WLANs) have become more prevalent and are widely deployed in many popular places like university campuses, cafés, airports, residences, etc. However, WLAN security is a very important but usually neglected issue. Focusing on three major types of typical wireless security standards: WEP, WPA and WPA2, we aim to explore the current state-of-the-art in security protocols and to present an overview of their real-life vulnerabilities by issuing successful attacks against WEP and WPA-protected WLANs.**

Keywords – WLAN, WEP, WPA, WPA2, attack against WEP, attack against WPA, tools to protect WLAN

I. INTRODUCTION

The IEEE 802.11 [1] is a set of standards for implementing wireless local area network (WLAN) computer communication in the 2.4, 3.6 and 5 GHz frequency bands. Such networks are met frequently in typical home or office facilities. The first IEEE 802.11 standard came with a basic protecting mechanism called Wired Equivalent Privacy (WEP). WEP requires all clients and access points (APs) in the network to share up to four different secret symmetric keys. This makes difficult the implementation of a larger installation, where users change frequently, and that's the reason why most installations use a single secret key named *root key*. WEP has some major drawbacks and was hacked in 2001 by Fluhrer, Mantin, and Shamir [2, 3]. They showed that an attacker may recover the secret key of a network with an average consumer laptop in an average of 1-2 hours. Nowadays, it is possible to recover the secret key in less than 60 seconds [4].

To fix the above insecure behaviour, a new standard was proposed in 2003, named Wi-Fi Protected Access (WPA). WPA came to solve the problems of WEP cryptography method. It uses a 3xtimes longer encryption key, as well as a message integrity key to prevent capturing, altering and/or resending of data packets. A year later, in 2004 came the second version, named WPA2, to replace plain WPA protocol. WPA2 introduces an even better encryption mode incorporating stronger security.

II. WIRELESS SECURITY PROTOCOLS

A. The WEP protocol

The WEP protocol uses the RC4 algorithm [6] for confidentiality and the CRC-32 checksum for integrity. At first, the secret key used is 40-bit long with a 24-bit Initialization Vector (IV) that is incorporated to it for acting as the encryption/decryption key resulting in a 64-bit total key size. Then the resulting key is used as an input, the so called 'seed', for a Pseudo-Random Number Generator (PRNG) that yields a key sequence equal to the length of the plaintext plus the Integrity Check Value (ICV). The result of the key sequence and the ICV will go to RC4 algorithm. A final encrypted message is made by attaching the IV in front of the ciphertext. Also the key may be 128-bit long, the only difference is that the secret key size becomes 104 bits and the IV remains 24 bits. Fig.1 depicts the WEP's encryption processes.

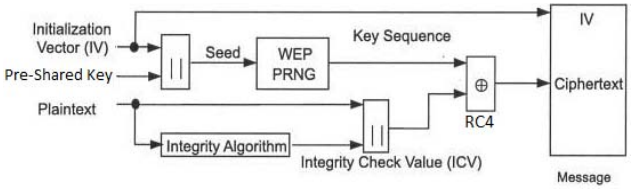

Figure 1. WEP Encryption Algorithm

For the WEP *decryption*, the IV of the incoming message and the Pre-Shared Key is used to generate the key sequence necessary to decrypt the incoming message. Thereafter the ciphertext and Secret key go to RC4 algorithm and a plaintext comes as a result. Next, the plaintext goes to Integrity Algorithm to make a new ICV (ICV') and finally the new ICV (ICV') compares with the original ICV. Fig.2 depicts a schematic of the objects and details [5]:

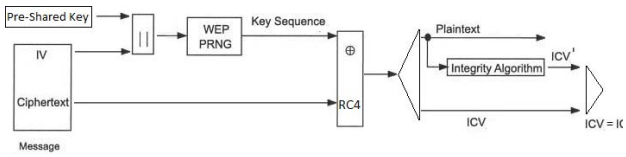

Figure 2. WEP Decryption Algorithm

B. The WPA protocol

As discussed, the WEP protocol had some serious security problems, such as: it does not prevent forgery of packets, it does not prevent replay attacks, it uses RC4 improperly, because the keys used are very weak and can be brute-forced on standard computers in hours to minutes, it allows an attacker to undetectably modify a message without knowing the encryption key, etc.. The WPA came with the purpose of solving the problems in the WEP cryptography method, without the need to change the

978-0-7695-4389-5/11 $26.00 © 2011 IEEE
DOI 10.1109/PCI.2011.25

hardware. The standard WPA specifies two operation manners:

1) Personal WPA or *WPA-PSK (Pre-Shared Key)* which is used for small office home and domestic use, it does not use an authentication server and data cryptography key can go up to 256 bits. Unlike WEP, this can be any alphanumeric string and is used only to negotiate the initial session with the AP. Because both the client and the AP already possess this key, WPA provides mutual authentication, and the key is never transmitted over the air

2) Enterprise or *Commercial WPA* in which the authentication is made by an authentication server 802.1x, generating excellent control and security in the users' wireless network traffic. This WPA uses 802.1X+EAP for authentication, but also replaces WEP with the more advanced TKIP encryption. No preshared key is used here, but you will need a RADIUS server. And you get all the other benefits 802.1X+EAP provides, including integration with the Windows login process and support for EAP-TLS and PEAP authentication methods.

One of the benefits of WPA is that it allows a more complex data encryption on the Temporal Key Integrity Protocol (TKIP) and it is also assisted by MIC (Message Integrity Check), whose function is to avoid bit-flipping type attacks which can easily be applied to WEP by using a hashing technique. Fig. 3 illustrates the overall WPA process in a nutshell.

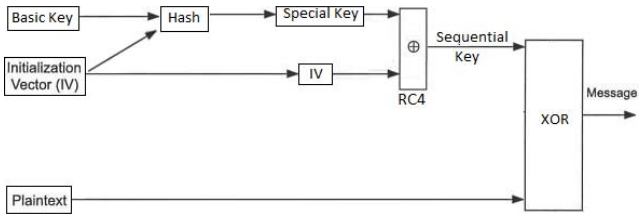

Figure 3. WPA Encryption Algorithm

As you see, TKIP uses the same WEP's RC4 Technique, but makes a hash before using the RC4 algorithm. A duplication of the initialization vector is made. One copy is sent to the next step, and the other is hashed (mixed) with the base key. After performing the hashing, the result generates the key to the package which is going to join the first copy of the initialization vector, following the RC4 algorithm increment. After this step, there is a generation of a sequential key with a XOR from the text that will be encrypted, generating then the cryptography text. Finally, the message is ready for send. Decryption is performed by inverting the above process.

C. The WPA2 protocol

WPA2 (or IEEE 802.11i-2004) in addition to TKIP, supports the AES-CCMP encryption protocol. Based on the very secure AES national standard cipher combined with sophisticated cryptographic techniques, AES-CCMP was specifically designed for wireless networks. AES-CCMP requires more computing power than TKIP. Like WPA, WPA2 supports two modes of security, Personal and Enterprise. In Personal mode a pre-shared secret key is used, much like WEP. APs and clients are all manually configured to use the same secret, of up to 64 ASCII characters, password such as "this_is_only_my_password_." An actual 256-bit randomly generated number may also be used, but this is difficult to enter manually into client configurations. The "Enterprise" security is based on 802.1X, EAP authentication framework, one of the several EAP types, and secure key distribution.

Dealing with security issues, 802.11i provides key enabler for secure and flexible wireless networks, allows client authentication, wireless network authentication, key distribution and the necessary for roaming pre-authentication. Using 802.1X in conjunction with 802.11i, it is strongly suggested to use EAP as a framework for authentication, and use an EAP type for the actual authentication that provides the optimal balance between cost, manageability and risk mitigation. Most often an 802.1X setup uses EAP-TLS for authentication between the wireless client (supplicant) and the AP (authenticator). In theory, several options may replace EAP-TLS, but this is rare in practice.

In 802.1X, no such port exists until the client connects and associates to the wireless AP. This immediately poses a problem, since beacon packets and probe request/response packets cannot be protected or authenticated. Fortunately, access to this data is not very useful for attackers, other than potentially causing denial - of - service attacks, and identifying wireless clients and APs by their hardware MAC addresses.

An 802.1X wireless setup consists of three main components:

- Supplicant (the wireless client).
- Authenticator (the AP).
- Authentication server (usually a RADIUS server).

The supplicant initially connects to the authenticator, as it would to a WEP or WPA protected network. Once this connection is established, the supplicant has in effect a network link to the authenticator (AP). The supplicant can then use this link to authenticate and gain further network access. The supplicant and authenticator first negotiate capabilities. These consist of three items:

- The pairwise cipher suite, used to encrypt unicast (point-to-point) traffic.
- The group cipher suite, used to encrypt multicast and broadcast (point-to-multiple-points) traffic.
- The use of either a pre-shared key (PSK, or "home user" security, using a shared secret) or 802.1X authentication.

So, the main problem of WPA is solved by dividing the type of security to three categories where just one of them uses pairwise and the two others use group cipher and pre-shared key.

III. ATTACKING A WEP NETWORK

According to the above discussion, in order to sufficiently crack a real-life WEP key of a wireless AP, we need to gather lots of initialization vectors (IVs). Normal network traffic does not typically generate these IVs very quickly. Theoretically, if you are patient, you can gather sufficient IVs to crack the WEP key by simply listening to the network traffic and saving them. However, in this work, we use a technique called *injection* to speed up the process. Injection involves having the AP resend selected packets over and over again very rapidly. This allows us to capture a large number of IVs in a short period of time. Once we have captured a large number of IVs, we can use them to determine the WEP key.

In practice WEP cracking can easily be demonstrated using tools such as Aircrack[1]. Aircrack contains three main utilities, used in the three attack phases that take place to recover the key:

- *airodump*: wireless sniffing tool used to discover WEP-enabled networks,
- *aireplay*: injection tool to increase traffic,
- *aircrack*: WEP key cracker using collected unique IVs.

The main goal of the attack is to generate traffic in order to capture unique IVs used between a legitimate client and an AP. Some encrypted data is easily recognizable because it has a fixed length, fixed destination address etc. This is the case where ARP [9] request packets with a fixed length of 68 octets (see Inset *ARP request*), are sent to the broadcast address (FF: FF:FF:FF:FF:FF). ARP requests can be replayed to generate new ARP responses from a legitimate host, resulting to the same wireless messages being encrypted with new IVs.

Now we will attempt an attack against the following network that uses WEP encryption Fig.4:

```
Cell 03 - Address: 00:05:59:05:FD:2F
          Channel:6
          Frequency:2.437 GHz (Channel 6)
          Quality=28/70  Signal level=-82 dBm
          Encryption key:on
          ESSID:"NetFasteR IAD (PSTN)"
          Bit Rates:1 Mb/s; 2 Mb/s; 5.5 Mb/s; 11 Mb/s; 22 Mb/s
          Bit Rates:6 Mb/s; 9 Mb/s; 12 Mb/s; 18 Mb/s; 24 Mb/s
                    36 Mb/s; 48 Mb/s; 54 Mb/s
          Mode:Master
```

Figure 4. *Target network*

After setting the wireless interface in monitor mode, so as to listen to the AP channel (Fig.5), we use aireplay-ng to implement a fake authentication (Fig.6) with the AP. In order for an AP to accept a packet, the source MAC address must be already associated. If the source MAC address you are injecting is not associated then the AP ignores the packet and sends out a "DeAuthentication" packet in cleartext. In this state, no new IVs are created because the AP is ignoring all the injected packets.

[1] http://www.aircrack-ng.org/

```
root@andreas-desktop:~# aireplay-ng -1 0  -a 00:05:59:05:FD:2F -h 00:1f:1f:30:33:1a mon0
00:37:54  Waiting for beacon frame (BSSID: 00:05:59:05:FD:2F) on channel 6

00:37:54  Sending Authentication Request (Open System) [ACK]
00:37:54  Authentication successful
00:37:54  Sending Association Request [ACK]
00:37:54  Association successful :-) (AID: 1)
```

Figure 5. Fake authentication

In Fig.5 the parameters are:

- -1 means fake authentication
- 0 is the reassociation timing in seconds
- -a 00:05:59:05:FD:2F is the AP MAC address
- -h 00:1f:lf:30:33:la is our card MAC address
- mon0 is the wireless interface name

We then start aireplay-ng in ARP request replay mode Fig.6. The purpose of this step is to start aireplay-ng in a mode which listens for ARP requests then re-injects them back into the network. The reason we select ARP request packets is because the AP will normally rebroadcast them and generate a new IV. Again, this is our objective, to obtain a large number of IVs in a short period of time.

```
root@andreas-desktop:~# aireplay-ng -3 -b 00:05:59:05:FD:2F -h 00:1f:1f:30:33:1a mon0
00:54:13  Waiting for beacon frame (BSSID: 00:05:59:05:FD:2F) on channel 6
Saving ARP requests in replay_arp-0531-005413.cap
You should also start airodump-ng to capture replies.
Read 77628 packets (got 33994 ARP requests and 18219 ACKs), sent 20193 packets...(499 pps)
```

Figure 6. ARP reinjection

Then we start the airodump-ng to capture the IVs. The purpose of this step is to capture the IVs generated. This step starts airodump-ng to capture the IVs Fig.7 from the specific AP.

```
airodump-ng -c 6 --bssid 00:05:59:05:FD:2F -w output mon0
```

Figure 7. Start airodump

In Fig.7 the parameters are:

- -c 6 is the channel for the wireless network
- --bssid 00:05:59:05:FD:2F is the AP MAC address.
- -w output is the file name prefix for the file which will contain the IVs.

The results are shown in Fig.8 where "data" is the IV's captured:

```
CH  6 ][ Elapsed: 27 mins ][ 2011-05-31 00:58

BSSID              PWR RXQ  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

00:05:59:05:FD:2F  -73  75    14874    54670  327   6  54 . WEP  WEP    OPN  NetFasteR IAD (PSTN)

BSSID              STATION            PWR   Rate    Lost  Packets  Probes

00:05:59:05:FD:2F  00:1F:1F:30:33:1A    0    0 - 1   1527   117709
00:05:59:05:FD:2F  00:22:FA:4C:23:5A  -83    2 - 5   1082     1563
```

Figure 8. Capture the IVs

The final step is to run the "aircrack-ng" command in order to obtain the WEP key from the IVs gathered in the previous steps. We type: *Aircrack-ng –b 00:05:59:05:FD:2F output*.cap*

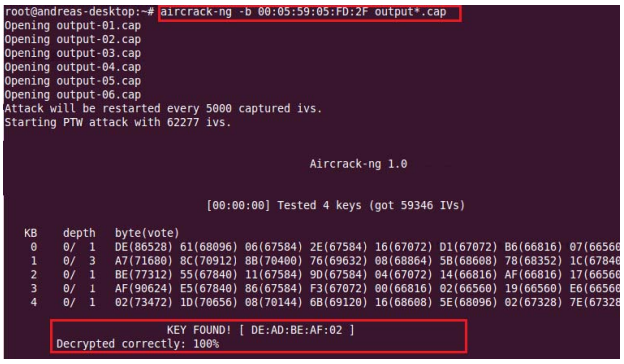

Figure 9. Run aircrack command

In Fig.9 the parameters are:

- -b 00:05:59:05:FD:2F selects the one AP we are interested in. This is optional since when we originally captured the data, we applied a filter to only capture data for this one AP.
- output*.cap selects all files starting with "output" and ending in ".cap".

As we can see the key has been successfully identified as: ***DEADBEAF02***.

The amount of gathered data required to fulfill the aircrack process depends on the key size. In our example 59 346 IVs were sufficient to crack the key. In case the IVs are insufficient we just wait a few minutes to gather some more data packets and then we run the "aircrack-ng "command again. For example 256-bit keys may require up to 500 000 data packets for a successful cracking attempt)

IV. ATTACK AGAINST A WPA2 NETWORK

Unlike WEP, where statistical methods can be used to speed up the cracking process, usually only plain brute force dictionary techniques may be used against WPA/WPA2 in an attempt to determine the shared passphrase. That is, because the key is not static, so collecting IVs like when cracking WEP encryption does not speed up the attack. This means that the passphrase must be contained in the dictionary you are using to break WPA/WPA2. The only thing that does give the information to start an attack is the handshake between client and AP. Handshaking is done when the client connects to the network.

During the handshake the AP and each station need an individual so-called Pairwise Transient Key (PTK) to protect unicast communication between them. The PTK is derived from the PMK (Pairwise Master Key), a fixed string, the MAC address of the AP, the MAC address of the client and two random numbers. The weakness of WPA-PSK is based on the pairwise master key (PMK) that is derived from the concatenation of the passphrase, SSID, length of the SSID and nonces (a number or bit string used only once in each session). This is the algorithm: PMK = PBKDF2(password, SSID, SSID length, 4096, 256). The result string is hashed 4,096 times to generate a 256-bit value and then combined with nonce values. As already mentioned, the PTK is derived from the PMK using the 4-Way Handshake and all information used to calculate its value is transmitted in plain text. By capturing the 4-Way Handshake, we have the data required to subject the passphrase into a dictionary attack.

In the following we shall perform an attack against a NetFaster router, which is used by Hellas On Line (HOL) Greek ISP customers. As it appears in Fig.10 our attack target uses WPA2 security.

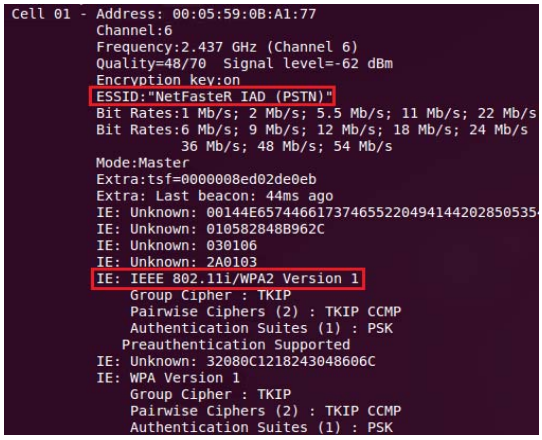

Figure 10. Our target network

A. *Four-Way handshake capture*

The first step is to start the wireless interface in monitor mode on channel 6, the one the AP uses. Then we start airodump-ng Fig. 11 to collect the four-way authentication handshake.

Figure 11. Start airodump-ng

In Fig.15 the parameters are:

- -c 6 is the channel for the wireless network
- bssid 00:05:59:0B:A1:77 is the AP MAC address
- -w psk is the file name prefix for the file which will contain the 4-way handshake.

The results are shown in Fig.12

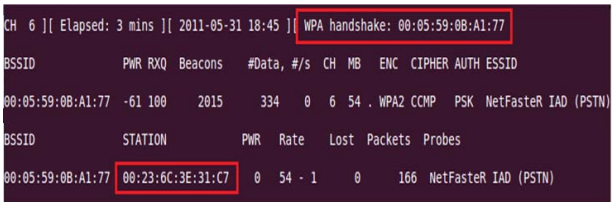

Figure 12. Four-Way handshake capture

In Fig.12 we can see that one client, identified by the "00:23:6C:3E:31:C7" MAC address is associated and authenticated on this wireless network (meaning that the Four-Way Handshake has already been done for this client). This client will subsequently be disassociated, forcing him this way to initiate a new association and allowing us to

capture Four-Way Handshake messages. The "aireplay" command will be used for this attack and this will disassociate the selected client with the specified BSSID by sending a fake disassociation request Fig.13.

If there was no wireless client currently associated with the AP, then we had to be patient and wait for one to connect to the AP so that a handshake can be captured.

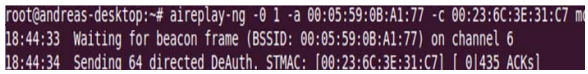

Figure 13. Client deauthentication

In Fig.13 the parameters are:

- -0 means deauthentication
- 1 is the number of deauths (deauthentications) to send
- -a 00:05:59:0B:A1:77 is the MAC address of the AP
- -c 00:23:6C:3E:31:C7 is the MAC address of the client we are deauthing (deauthenticating)

In Fig.12 we notice the "WPA handshake: 00:05:59:0B:A1:77" in the upper-right corner of our monitor. This means airodump-ng has successfully captured the four-way handshake.

B. Dictionary Attack

Now we can launch the dictionary attack using a wordlist. Based on our observations, the factory set, default password of NetFaster routers has the following format: "[MAC address]-[a four-digit random number]", e.g.: "AABBCCDDEEFF-1234". So we can generate the suitable wordlist using a program called Crunch Fig.14. Knowing the router's MAC address, which allows us to use a wordlist containing only 10.000 words, i.e. equal to the amount of all different four-digit number combinations.

Figure 14. Wordlist generation

In Fig.14 the parameters are:

- [17 17] min and max word length, i.e. fixed length: 17
- -o wordlist.txt is the output wordlist file
- -s 0005590BA177-0000 is the starting word
- -t 0005590BA177-%%%% is the word format.
- % represents a number

At this point, we use the "Pyrit"[2] utility to implement a dictionary offline attack. It takes as input the .cap file with the Four-Way Handshake, we have captured and the wordlist (dictionary) file we have generated Fig.15. Pyrit takes the capture-file "psk-01.cap" and attacks the key-negotiation using the dictionary-file "wordlist.txt. It computes 4096 hashes for each password attempt, producing one Pairwise Master Key. Every PMK is 'worth' exactly one megabyte of data getting pushed through PBKDF2-HMAC-SHA1. In turn, computing 10.000 PMKs per second is equivalent to hashing 9,8GB of data with SHA1 in one second.

[2] http://code.google.com/p/pyrit/

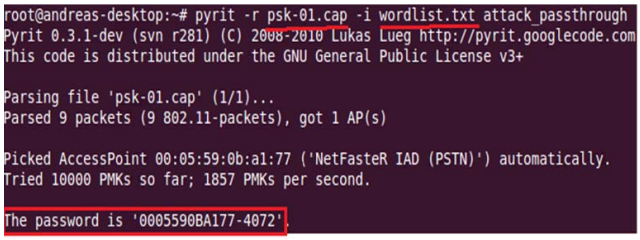

Figure 15. Dictionary attack with Pyrit.

The advantage of Pyrit is that it can exploit the computational power of GPUs. As it appears in Fig.16 Pyrit uses my cuda GPU (Nvidia GT9500) to compute PMKs. As we can see the GPU is much more effective than plain CPU (e.g. Intel Core 2 Duo 2.93GHz). That happens because GPU has more cores than CPU.

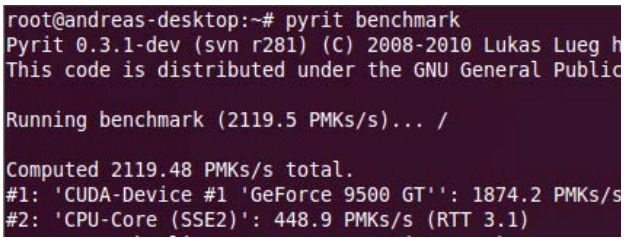

Figure 16. Pyrit benchmark

V. CONCLUSIONS

Wireless networks are becoming the most rapidly spread technology over the world; thus, they should be well protected, in order to prevent exploitation of confidential data. In this paper we presented a brief overview of them, focusing on three main security protocols WEP, WPA and WPA2. We discussed and presented in detail an analytical procedure towards WEP and WPA2 cracking, derived from real-life situations. Our motivation was the need for increased wireless security and the common feel that nowadays WPA/WPA2 security protocols are difficult for a stranger to hack; however, our study depicted that any wireless network may be suffering from successful hacking attempts, if it is not carefully setup and protected.

REFERENCES

[1] IEEE-SA Standards Board, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, IEEE Communications Magazine, 2007

[2] S. R. Fluhrer, I. Mantin, A. Shamir, Weaknesses in the key scheduling algorithm of RC4, in S. Vaudenay, A. M. Youssef (eds.), Selected Areas in Cryptography 2001, LNCS Vol. 2259

[3] A. Stubbleeld, J. Ioannidis, A. D. Rubin, A key recovery attack on the 802.11b wired equivalent privacy protocol (WEP), ACM Transactions on Information and System Security, May 2004

[4] E. P. Weinmann, A. Pyshkin. In S. Kim, M. Yung, H.-W. Lee (eds.), WISA, LNCS Vol. 4867, Springer, 2007.

[5] A. H. Lashkari, F. Towhidi, R. S. Hoseini, "Wired Equivalent Privacy(WEP)", ICFCC Kuala Lumpur Conference, 2009

[6] D. C. Plummer, RFC 826: Ethernet Address Resolution Protocol: Or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware, November 1982