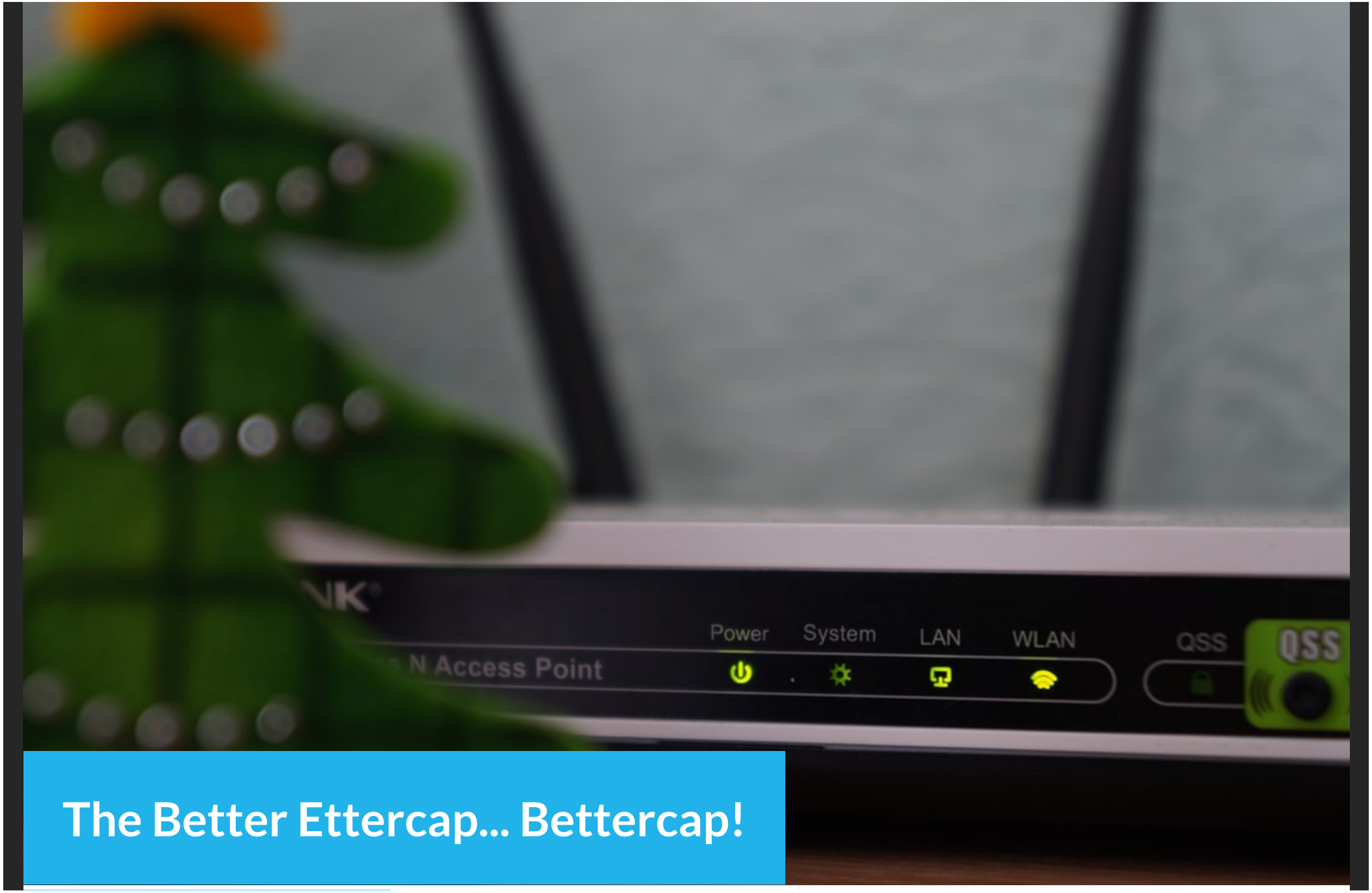
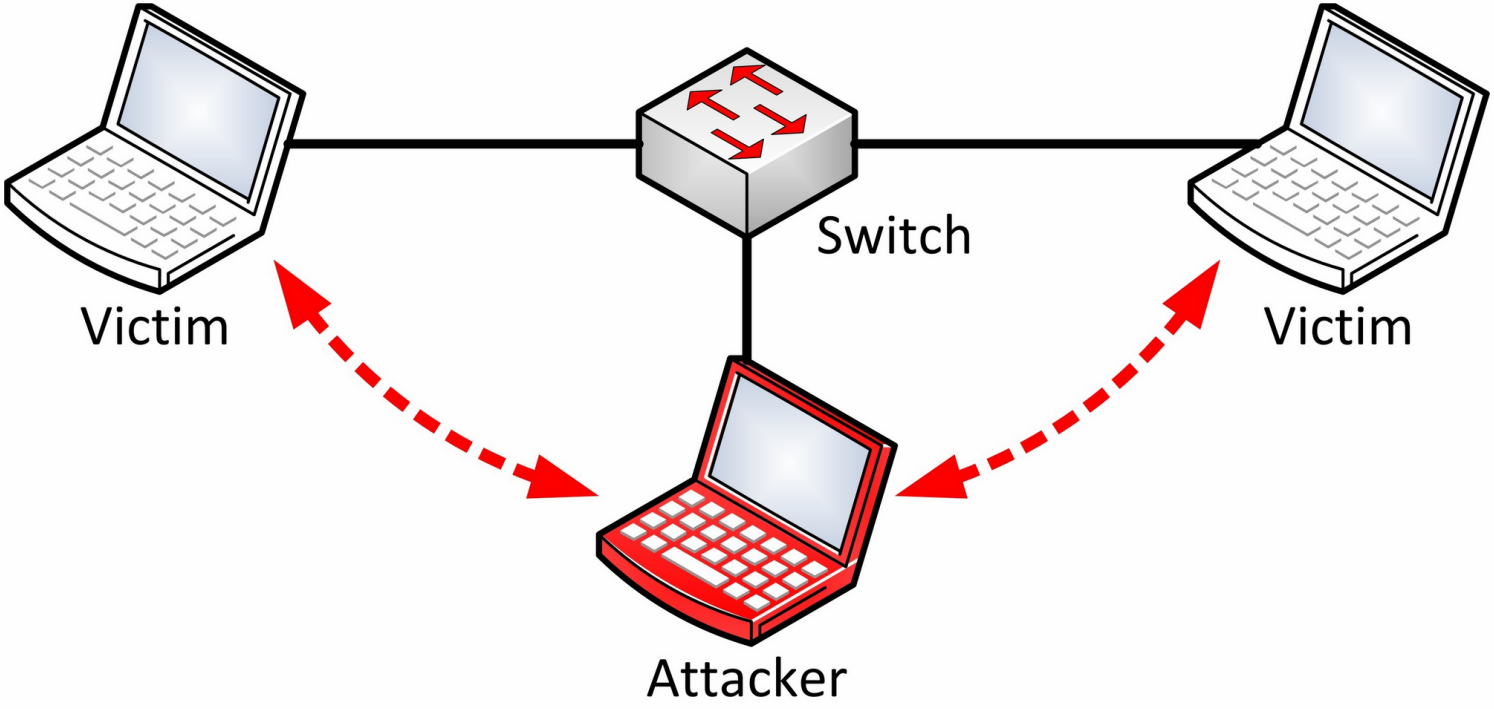


The Better Ettercap... Bettercap!

RYAN VILLARREAL / JULY 20, 2019

What is an Ettercap and how can it be better? To explain I will speak briefly about what Ettercap is, and why it's useful. Taken directly from the Ettercap home page: "Ettercap is a comprehensive suite for man in the middle attacks. It features sniffing of live connections, content filtering on the fly and many other interesting tricks. It supports active and passive dissection of many protocols and includes many features for network and host analysis."



Ettercap is a free and open source network security tool that helps penetration testers or attackers to perform network protocol analysis or active those same network protocols. Ettercap was originally released in March 14, 2015 and is written in the C language. Unfortunately, Ettercap has not advanced enough to stay relevant with some of the newer protocols. As well, there is quite a few stability problems, and the worst issue is the extensibility is very difficult. More about the pitfalls of Ettercap can be described here.

Therefore, it would be nice to have a "better" Ettercap, and thus spawned the project Bettercap by @evilsocket. Bettercap is a fully extensible and portable framework written in Go which hopes to be a direct replacement for penetration testers and attackers to have an all-in-one solution. Bettercap also aims to add different protocols such as WiFi, Bluetooth Low Energy, HID devices, and Ethernet networks.

Bettercap has more features than would be possible to discuss in a single blog post, but for today I will mostly be focusing on using Bettercap to perform different wireless attacks. In order to be consistent I will be using version 2.4 as a pre-compiled binary downloaded from GitHub. Source install instructions are also available, but this will allow the blog to be somewhat shorter.

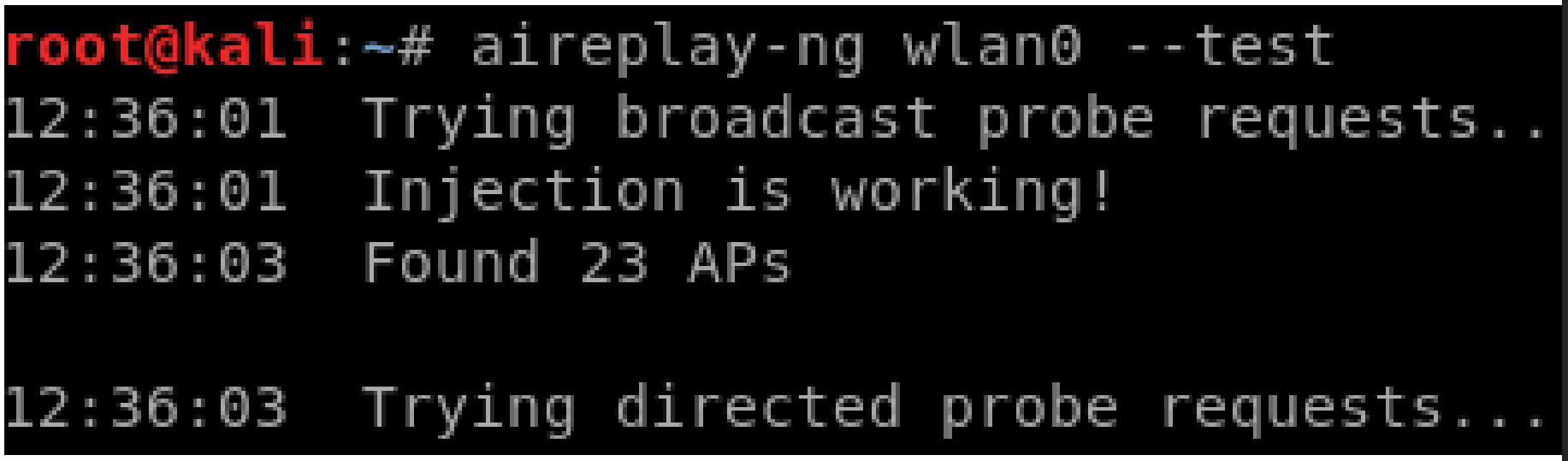
In order to follow along with this blog post you will need a wireless card that is able to inject packets. The following code snippet is a quick and easy way to grab a pre-compiled binary of Bettercap.

```
root@kali: wget
https://github.com/bettercap/bettercap/releases/download/v2.24/betterc
ap_linux_amd64_2.24.zip
root@kali: unzip bettercap_linux_amd64_2.24.zip -d /opt/bettercap/
root@kali: cd /opt/bettercap
```

The screenshot below shows the wireless interface that will be used for the remainder of this blog post.

```
root@kali:~# iwconfig
wlan0     IEEE 802.11  ESSID:off/any
          Mode:Managed  Access Point: Not-Associated   Tx-Power=18 dBm
          Retry short limit:7   RTS thr:off   Fragment thr:off
          Encryption key:off
          Power Management:off

eth0      no wireless extensions.

lo        no wireless extensions.
```

One way to test your card for wireless injection is if you are currently using Kali Linux you can use the pre-built Aircrack-ng Suite to check for packet injection. The command `aireplay-ng wlan0 --test`. If your wireless card is currently able to inject packets into wireless you will see an output like the one below.



```
root@kali:~# aireplay-ng wlan0 --test
12:36:01  Trying broadcast probe requests...
12:36:01  Injection is working!
12:36:03  Found 23 APs

12:36:03  Trying directed probe requests...
```

Now we can begin looking further into Bettercap and it's abilities. Of course one of the first things to do with any new tool is to run `--help` or find the man page in order to learn the different flags that are available.

```
root@kali:/opt/bettercap# ls
bettercap  LICENSE.md  README.md
root@kali:/opt/bettercap# ./bettercap --help
Usage of ./bettercap:
  -autostart string
        Comma separated list of modules to auto start. (default "events.stream")
  -caplet string
        Read commands from this file and execute them in the interactive session.
  -cpu-profile file
        Write cpu profile file.
  -debug
        Print debug messages.
  -env-file string
        Load environment variables from this file if found, set to empty to disable environment persistence.
  -eval string
        Run one or more commands separated by ; in the interactive session, used to set variables via command line.
  -gateway-override string
        Use the provided IP address instead of the default gateway. If not specified or invalid, the default gateway will be used.
  -iface string
        Network interface to bind to, if empty the default interface will be auto selected.
  -mem-profile file
        Write memory profile to file.
  -no-colors
        Disable output color effects.
  -no-history
        Disable interactive session history file.
  -silent
        Suppress all logs which are not errors.
  -version
        Print the version and exit.
```

To begin Bettercap I will simply run `./bettercap -iface wlan0` this will tell Bettercap to start using our wireless interface. If you do not specify an interface Bettercap will attempt to

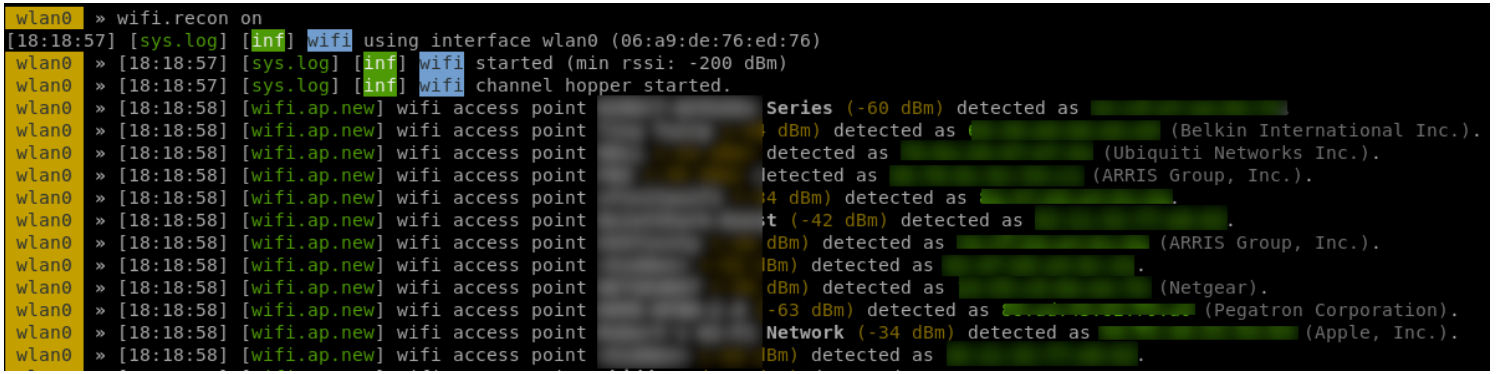
find the primary, however in our instance we do not want to use the Ethernet interface but instead the wireless.

```
root@kali:/opt/bettercap# ./bettercap -iface wlan0
bettercap v2.24 (built for linux amd64 with go1.10.4) [type 'help' for a list of commands]

 wlan0  »
```

Bettercap has auto-complete functionality on the command line which means if you are not sure of the command you need to run, or if you want to see options you can simply hit the tab key twice and it will list the options. As you can see from the screenshot below I ran the auto-complete for the WiFi modules built into Bettercap. The first command we are going to run is the `wifi.recon on` command. This will start Bettercap in a reconnaissance mode where it will listen for all beacons and probes while jumping through the different channels.

```
root@kali:/opt/bettercap# ./bettercap -iface wlan0
bettercap v2.24 (built for linux amd64 with go1.10.4) [type 'help' for a list of commands]

 wlan0  » wifi.
wifi.deauth                wifi.assoc              wifi.recon              wifi.clear
```

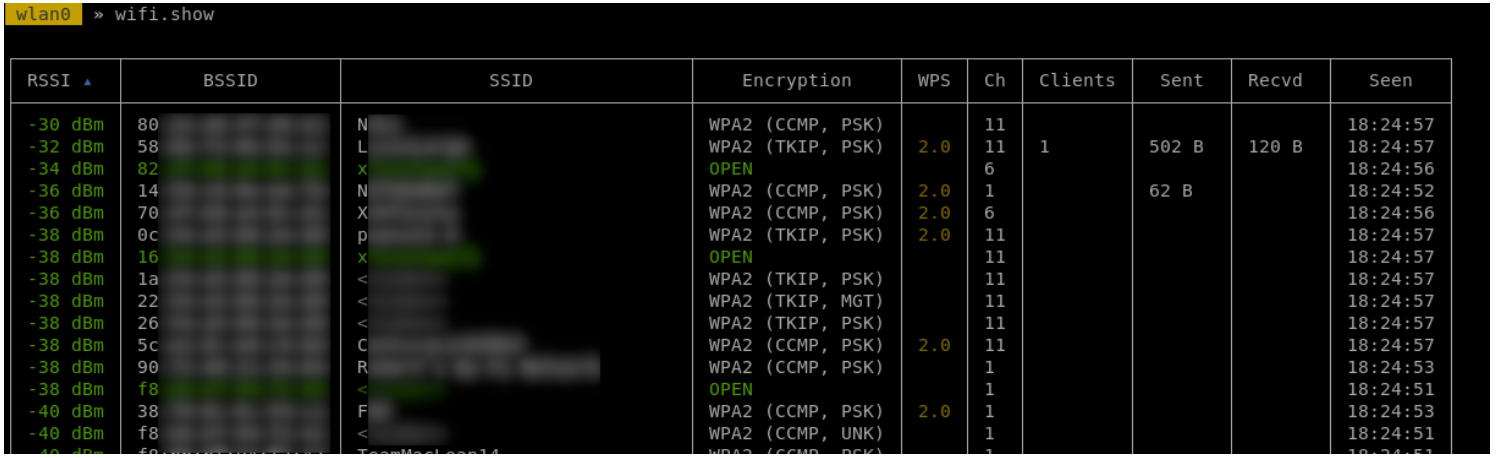
The screenshot below shows the output, and as you can see nearly immediately wifi access points become visible to the wireless interface. These wifi access points are found by sniffing for beacons.



If you are performing a wireless engagement or already know the Access Point you want to attack, you can simply set the channel to not hop channel to channel, but instead be on a static channel by running `wifi.recon.channel <number>`.

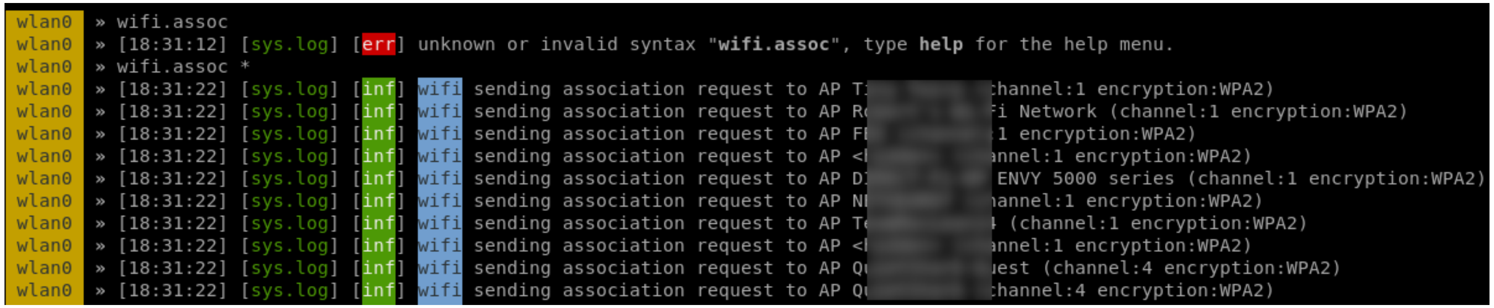
```
wlan0 » wifi.recon.channel 11
wlan0 »
```

The screenshot below shows Bettercap's ability to produce a nice well organized table of the different wifi access points as well as the current information gathered about them.
Information such as, the bssid, ssid, encryption method, if WPS is supported and if so what version, the current channel of the wifi access point, the amount of data sent and received, as well as the last beacon received.



Now that we have information on the nearby wifi access points we can use Bettercap to go on the offensive. I should take the time at this point to note that attacking a wireless network that is not yours is illegal in many countries. The wireless access point being used for demonstration in this blog post is owned and operated by me. The screenshot below shows the next command we will use: `wifi.deauth <mac address>`. This will tell Bettercap to begin sending deauth packets to the mac address specified. You might note that in the screenshot below that Bettercap returns with an error message saying it doesn't have detected clients. That is because at the time of attack I had not waited long enough for a client to appear. If you are having the same issue it might be helpful to set the wireless interface card to only listen on the channel of the wireless access point you are attacking. This will help listen for client beacons as well.

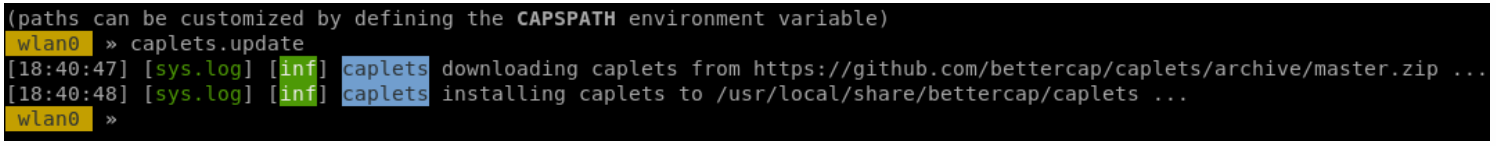
If all goes well in the screenshot above Bettercap should capture a `.pcap` file that will have the four-way handshake from the wifi access point. The next type of attack that Bettercap can perform is known as an association attack. Also known as a clientless attack, because it was discovered in 2018 by a user on the [Hashcat forums](#) that many modern routers append an optional field at the end of the first EAPOL frame sent by the AP itself when someone is associating. This optional field is the Robust Security Network or RSN which includes the PMKID. Using this PMKID, we can use already pre-obtained data to generate a four-way handshake that Hashcat can crack. Well luckily for us, Bettercap allows for wildcard commands that will look for any access point beacons and try to associate with them. Looking specifically for the PMKID. The command output can be shown in the screenshot below.



Once an access point responds with a valid RSN PMKID Bettercap will save the four-way handshake to a `.pcap` file. As you can see in the screenshot below the four-way handshake will be highlighted with red text.

Up to this point we have demonstrated two attack types that Bettercap can do using wireless only. Additionally, Bettercap comes with the ability to automate the start up and attack vectors using what is known as `caplets`. By running `caplets.paths` inside of Bettercap it will inform the user where the caplets are stored, and the order in which it looks for import.

The command `caplets.update` will update all the caplets from the github repo. Another pro-tip, keep in mind if you have a custom caplet that you have in the directory it will remove it when updating. Keep your caplets in a separate location for backup.



Once inside the Bettercap shell you can run `caplets.show` to see what caplets have been loaded as well as the path.

```
root@kali:/opt/bettercap# ./bettercap -iface wlan0
bettercap v2.24 (built for linux amd64 with go1.10.4) [type 'help' for a list of commands]

 wlan0  » caplets.show
```

Name	Path	Size
ap	/usr/local/share/bettercap/caplets/ap.cap	307 B
auto	/usr/local/share/bettercap/caplets/auto.cap	437 B
crypto-miner/crypto-miner	/usr/local/share/bettercap/caplets/crypto-miner/crypto-miner.cap	666 B
download-autopwn/download-autopwn	/usr/local/share/bettercap/caplets/download-autopwn/download-autopwn.cap	2.6 kB
fb-phish/fb-phish	/usr/local/share/bettercap/caplets/fb-phish/fb-phish.cap	140 B
gitspoof/gitspoof	/usr/local/share/bettercap/caplets/gitspoof/gitspoof.cap	216 B
gps	/usr/local/share/bettercap/caplets/gps.cap	109 B
hstshijack/hstshijack	/usr/local/share/bettercap/caplets/hstshijack/hstshijack.cap	1.1 kB
http-req-dump/http-req-dump	/usr/local/share/bettercap/caplets/http-req-dump/http-req-dump.cap	591 B
http-ui	/usr/local/share/bettercap/caplets/http-ui.cap	382 B
https-ui	/usr/local/share/bettercap/caplets/https-ui.cap	661 B
jsinject/jsinject	/usr/local/share/bettercap/caplets/jsinject/jsinject.cap	210 B
local-sniffer	/usr/local/share/bettercap/caplets/local-sniffer.cap	244 B
login-manager-abuse/login-man-abuse	/usr/local/share/bettercap/caplets/login-manager-abuse/login-man-abuse.cap	236 B
mana	/usr/local/share/bettercap/caplets/mana.cap	61 B
massdeauth	/usr/local/share/bettercap/caplets/massdeauth.cap	302 B
mitm6	/usr/local/share/bettercap/caplets/mitm6.cap	551 B
netmon	/usr/local/share/bettercap/caplets/netmon.cap	42 B
pita	/usr/local/share/bettercap/caplets/pita.cap	900 B
proxy-script-test/proxy-script-test	/usr/local/share/bettercap/caplets/proxy-script-test/proxy-script-test.cap	57 B
rogue-mysql-server	/usr/local/share/bettercap/caplets/rogue-mysql-server.cap	501 B
rtfm/rtfm	/usr/local/share/bettercap/caplets/rtfm/rtfm.cap	210 B
simple-passwords-sniffer	/usr/local/share/bettercap/caplets/simple-passwords-sniffer.cap	131 B
tcp-req-dump/tcp-req-dump	/usr/local/share/bettercap/caplets/tcp-req-dump/tcp-req-dump.cap	413 B
web-override/web-override	/usr/local/share/bettercap/caplets/web-override/web-override.cap	254 B

Quickly I will show a brief example of creating a custom caplet. So first we will start `vim test.cap` and enter `wifi.recon on`. In the screenshot below it shows the basic example of what the caplet will look like.

```
root@kali:/opt/bettercap# vim test.cap
root@kali:/opt/bettercap# cat test.cap
wifi.recon on
root@kali:/opt/bettercap#
```

You can even call the caplet straight from the command line with the flag `-caplet <filename>`. As you can see from the screenshot below it automatically starts the WiFi recon.

```
root@kali:/opt/bettercap# ./bettercap -iface wlan0 -caplet test.cap
bettercap v2.24 (built for linux amd64 with go1.10.4) [type 'help' for a list of commands]

[18:42:26] [sys.log] [inf] wifi using interface wlan0 (76:4f:ab:54:fb:31)
 wlan0  » [18:42:26] [sys.log] [inf] wifi started (min rssi: -200 dBm)
 wlan0  » [18:42:26] [sys.log] [inf] wifi channel hopper started.
```

As a final note, I will also show that you can start a command directly from the command line by using `-eval "<command>"`. This can be helpful if you are trying to start Bettercap with a list of commands on device boot.

```
root@kali:/opt/bettercap# ./bettercap -iface wlan0 -eval "wifi.recon on;"
bettercap v2.24 (built for linux amd64 with go1.10.4) [type 'help' for a list of commands]

[18:43:25] [sys.log] [inf] wifi using interface wlan0 (76:4f:ab:54:fb:31)
[18:43:26] [sys.log] [inf] wifi started (min rssi: -200 dBm)
 wlan0  » [18:43:26] [sys.log] [inf] wifi channel hopper started.
```

In coming posts I will be working on creating a single-board computer solution that will attempt to leverage Bettercap's caplets in order to automate wireless pentesting from a covert position while performing physical or red teaming exercises. Until next time, keep on injecting packets!

Sources and Inspiration:

- https://www.bettercap.org/intro/
- @evilsocket - https://twitter.com/evilsocket
- @bettercap - https://twitter.com/bettercap
- https://www.ettercap-project.org/
- https://www.bettercap.org/legacy/

SHARE

TAGS: 802.1X 802.11 BETTERCAP COMMAND LINE HASHCAT PINEAPPLE WIFI WIRELESS WPA

— ABOUT RYAN VILLARREAL

DENVER, COLORADO TWITTER

NEXT

Atomic Red Team

JULY 30, 2019

PREVIOUS

PlexTrac for Faster Report Writing!

JULY 15, 2019

— ABOUT —

Two cybersecurity professionals trying to get better at all things security.

— LATEST POSTS —

Information Gathering With Cobalt Strike
AUGUST 16, 2019

Navigating To A Web Site Step By Step
AUGUST 01, 2019

Atomic Red Team
JULY 30, 2019

— AUTHORS —

- Ryan Smith
- Bestest RedTeam
- Ryan Villarreal

— TAGS —

802.11 802.1X ACTIVE DIRECTORY ANTI-CSRF AUTOMATE AUTOMATION AWS BETA BETTERCAP BGP BITCOIN BLOODHOUND BLUE TEAM BURPSUITE BYPASS BYT3BL33D3R C2 CA CAPTURE THE FLAG CERTIFICATES CLOUD CLUSTER CME COBALT STRIKE COMMAND AND CONTROL

COMMAND LINE CONTAINER CORS CRACKMAPEXEC CSRF CTF CYBERSECURITY DETECTION DOCKER DOMAIN ADMIN DOMAIN CONTROLLER

DVWA ELEARNSECURITY ELK ELKSTACK ENUMERATION EWPT EXECUTIONPOLICY FREERADIUS GHOST GNU RADIO GOOGLE CLOUD GOPHISH

GRAPH THEORY HACKING HACKRF HASHCAT HIJACKING HTTP HTTP/2 IMPACKET INFORMATION GATHERING INTERNAL NETWORK

INTERNET OF THINGS JAVASCRIPT JUICESHOP JWT KALI LINUX KALI TOOLS KERBEROS LATERAL MOVEMENT LINUX MERLIN MICROSOFT

MICROSOFT OFFICE MINING NE0ND0G NEO4J NETWORKING NULL SESSION OFFENSIVE SECURITY OFFSEC OPEN REDIRECT OSWP OWASP

PASSWORD CRACKING PENETRATION TEST PENTEST PHISHING PHP PINEAPPLE PIXEL TRACKING PORTAINER POST EXPLOITATION POWERSHELL

PROTOCOLS PYTHON RADIO FREQUENCY RECON RED TEAMING RED-BARON REDTEAMING REPORTING REVIEW RF RFC RTL-SDR S3 SAMBA

SCANS SCAPY SCRIPTING SERVICE PRINCIPAL NAME SERVICES SHODAN SMB SMBCLIENT SOCIAL ENGINEERING SOFTWARE DEFINED RADIO SPN

SWARM SYSADMIN TERRAFORM TERRAFORMFUN TRAINING UUID VULNERABILITY SCANNING WARDRIVING WEB APP WEB APPLICATION

WEB TOKEN WEBAPP WIFI WIFU WIGLE WINDOWS WIRELESS WPA XSS

OPINIONS EXPRESSED ARE SOLELY OUR OWN AND DO NOT EXPRESS THE VIEWS OR OPINIONS OF OUR EMPLOYERS.